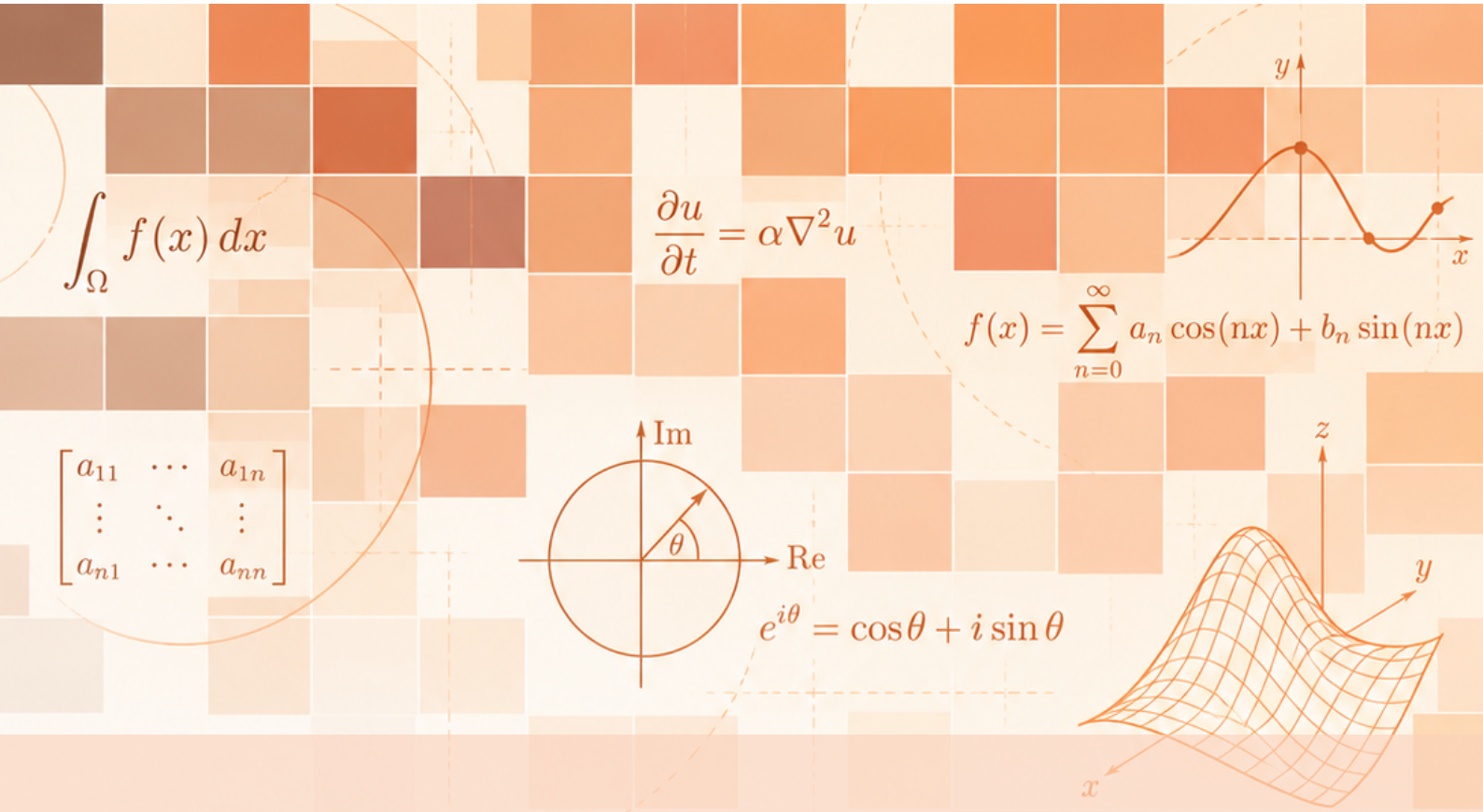
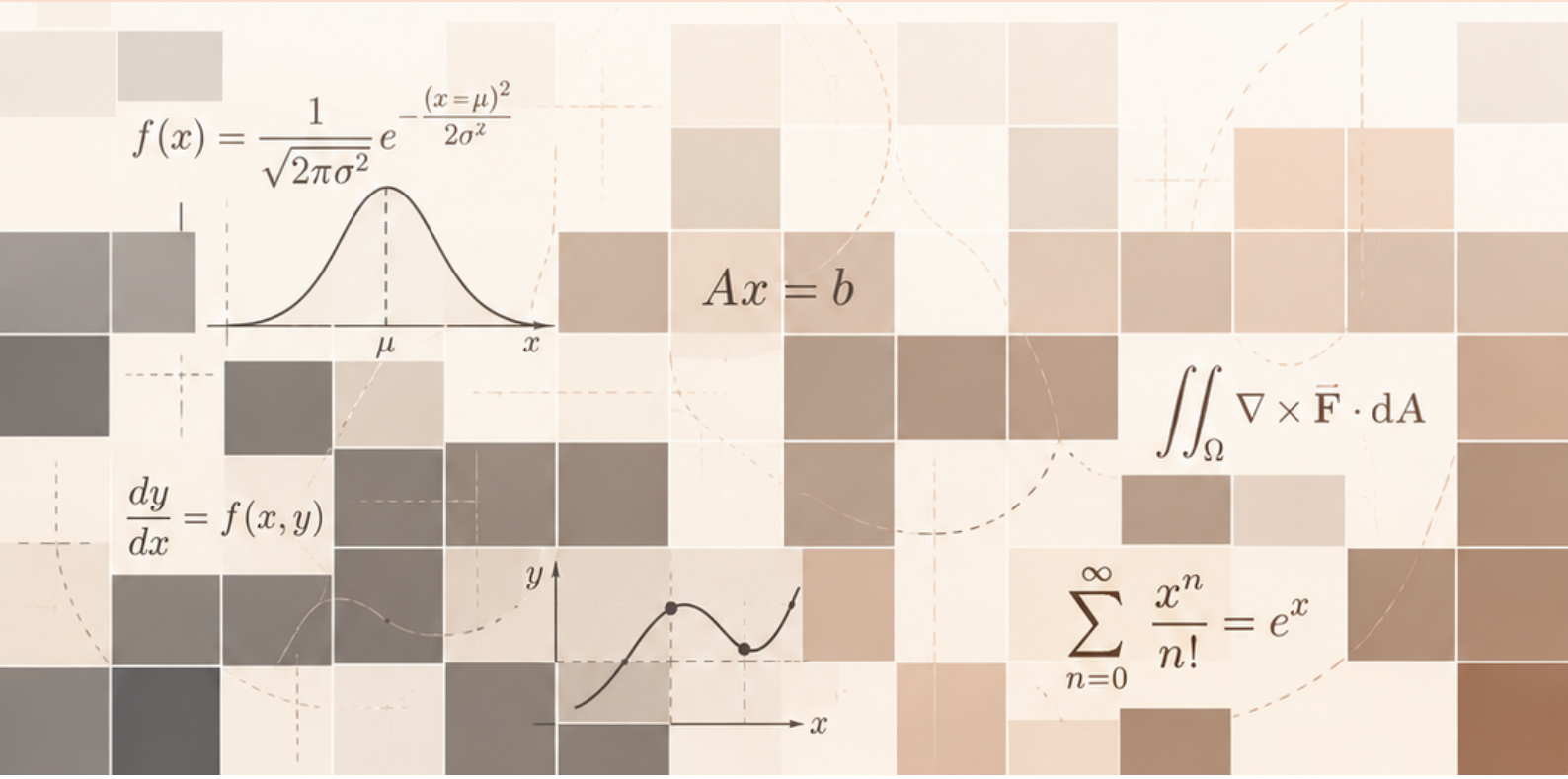


Analisi dei dati per l'ingegneria (146172) (module 1)

Note del corso

Enrico Bertolazzi



Copyright © 2025 Enrico Bertolazzi

Published by Publisher

book-website.com

Licensed under the Creative Commons Attribution-NonCommercial 4.0 License (the "License"). You may not use this file except in compliance with the License. You may obtain a copy of the License at https://creativecommons.org/licenses/by-nc-sa/4.0. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "as is" basis, without warranties or conditions of any kind, either express or implied. See the License for the specific language governing permissions and limitations under the License.

First printing, May 2025



Contents

I Sistemi lineari: metodi diretti

1 Metodo di eliminazione di Gauss . . . 19
1.1 Introduzione . . . 19
1.2 Forma Matriciale del Metodo di Gauss . . . 24
1.3 Proprietà delle matrici di Frobenius . . . 25
1.4 Primo passo del metodo di Gauss . . . 28
1.5 $k-$esimo passo del metodo di Gauss . . . 29
1.6 Algoritmo di Gauss in presenza di pivoting . . . 31
1.6.1 Esempio . . . 34

2 Fattorizzazione di Cholesky . . . 37
2.1 Come si riconosce una matrice SPD: il criterio di Sylvester . . . 40
2.2 Algoritmo di calcolo . . . 42
2.3 Connessioni con la fattorizzazione di Gauss . . . 44

II Fattorizzazione QR

3 Metodo QR per sistemi lineari . . . 49
3.1 Problema dei minimi quadrati . . . 49
3.1.1 Regressione lineare . . . 50
3.1.2 Fitting polinomiale . . . 52
3.1.3 Forma matriciale del problema . . . 53

4 Costruzione della fattorizzazione QR . . . 57
4.1 Matrici Ortogonali . . . 57
4.1.1 Proprietà . . . 57

4.2 Rotazioni di Givens . . . 58
4.2.1 Forma della Matrice di Rotazione di Givens . . . 58
4.2.2 Calcolo di $\cos\theta$ e $\sin\theta$ . . . 59
4.3 Matrici di Householder . . . 59
4.3.1 Proprietà . . . 59
4.3.2 Esempio di Matrice di Householder . . . 60
4.4 Fattorizzazione QR . . . 61
4.4.1 Algoritmo di Givens . . . 61
4.5 Algoritmo di Householder . . . 65
4.5.1 Costruzione del Vettore di Householder . . . 65
4.5.2 Passaggi per la Fattorizzazione QR . . . 65

5 La pseudo inversa di Moore-Penrose . . . 69
5.1 Pseudo-inversa definizione . . . 69
5.2 Proprietà della pseudo-inversa . . . 69
5.2.1 La pseudo-inversa di una matrice è unica . . . 69
5.2.2 La pseudo-inversa di una matrice non singolare è l'inversa "classica" . . . 70
5.2.3 La pseudo-inversa della pseudo-inversa è la matrice originale . . . 70
5.2.4 La pseudo-inverse di un vettore $\boldsymbol{a}$ è $\boldsymbol{a}^+ = (\boldsymbol{a}\cdot\boldsymbol{a})^{-1}\boldsymbol{a}^\top$ . . . 70
5.2.5 Pseudo-inversa di una matrice diagonale . . . 70
5.2.6 Pseudo-inversa ed SVD . . . 70
5.3 Connessioni con la fattorizzazione QR . . . 71
5.3.1 Proprietà 1: $\boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{A}$ . . . 71
5.3.2 Proprietà 2: $\boldsymbol{A}^+\boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}^+$ . . . 72
5.3.3 Proprietà 3: $\boldsymbol{A}^+\boldsymbol{A}$ simmetrica . . . 72
5.3.4 Proprietà 4: $\boldsymbol{A}\boldsymbol{A}^+$ simmetrica . . . 72
5.4 Pseudo inversa e fattorizzazione LQ . . . 72
5.4.1 Proprietà 1: $\boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{A}$ . . . 73
5.4.2 Proprietà 2: $\boldsymbol{A}^+\boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}^+$ . . . 73
5.4.3 Proprietà 3: $\boldsymbol{A}^+\boldsymbol{A}$ simmetrica . . . 73
5.4.4 Proprietà 4: $\boldsymbol{A}\boldsymbol{A}^+$ simmetrica . . . 73

III Interpolazione polinomiale

6 Interpolazione Polinomiale . . . 77
6.1 Unicità del polinomio interpolante . . . 81
6.2 Generalizzazione . . . 84
6.3 Condizione di Haar (o unisolvenza) . . . 84
6.4 Errore di interpolazione . . . 84

7 Interpolazione di Lagrange . . . 89
7.1 Derivazione pratica della interpolazione di Lagrange . . . 91
7.1.1 Stabilità . . . 92
7.2 Polinomio di miglior approssimazione . . . 93

8 Interpolazione di Newton . . . 95
8.1 Proprietà delle differenze divise . . . 98
8.2 Invarianza per permutazioni dei nodi . . . 100
8.3 Formula ricorrente . . . 100
8.3.1 Riassunto sulle differenze divise . . . 102
8.3.2 Costi computazionali . . . 103
8.4 L'algoritmo di Aitken-Neville . . . 104
8.4.1 Osservazioni finali sull'algoritmo di Aitken-Neville . . . 105

9 Esempio di Runge . . . 107
9.1 Approssimazione di funzioni continue . . . 107
9.2 Controesempio di Runge . . . 109
9.3 Conclusioni . . . 114

IV Integrazione numerica

10 Integrazione numerica . . . 117
10.1 Problema dell'integrazione numerica . . . 117
10.2 Strategia *interpolatoria* . . . 118
10.3 Classificazione (largamente incompleta) . . . 118
10.4 Formule di Newton-Cotes . . . 119
10.5 Accuratezza . . . 120
10.6 Metodo dei Coefficienti Indeterminati . . . 121
10.7 Stima dell'errore di integrazione . . . 122
10.7.1 Richiamo: errore della regola dei rettangoli . . . 122
10.7.2 Stime dirette per trapezi e Simpson mediante Rolle . . . 123
10.7.3 Teorema generale per le formule di Newton-Cotes . . . 125
10.7.4 Verifica esplicita del teorema per trapezi e Simpson . . . 126

11 Integrazione composita . . . 129
11.1 Stima a priori degli intervalli . . . 130
11.2 Trapezi e Simpson composite . . . 131
11.3 Scelta a priori del numero di sottointervalli . . . 133
11.3.1 Esercizio — Confronto tra trapezi, Simpson e Gauss-Legendre . . . 133

12 Formule di quadratura . . . 137
12.1 Principali formule di quadratura . . . 137
12.1.1 Newton-Cotes chiuse . . . 138
12.1.2 Newton-Cotes aperte . . . 140
12.1.3 Gauss-Legendre . . . 140
12.1.4 Gauss-Lobatto . . . 141
12.1.5 Gauss-Radau . . . 142

13 Stime a posteriori . . . 145
13.1 Dalla stima a posteriori a un algoritmo adattativo . . . 146
13.1.1 Algoritmo ricorsivo con budget locale . . . 147
13.2 Perché cercare di equidistribuire l'errore . . . 147
13.3 Esperimenti numerici: stima ed errore effettivo . . . 148
13.3.1 Esempio 1: picco localizzato . . . 148
13.3.2 Esempio 2: funzione regolare . . . 149
13.3.3 Esempio 3: oscillazioni localizzate . . . 150

V Polinomi ortogonali

14 Da Gram–Schmidt ai polinomi di Legendre . . . 157
14.1 Il prodotto scalare pesato sui polinomi . . . 157
14.2 Esistenza della famiglia ortogonale . . . 158
14.2.1 Ortogonalizzazione di Gram–Schmidt . . . 158
14.2.2 La matrice di Gram e la matrice di Hankel . . . 158
14.2.3 Il condizionamento della base monomiale . . . 160
14.2.4 Esempio svolto: costruire P_2 a mano . . . 160
14.3 I polinomi di Legendre . . . 161
14.3.1 Definizione e primi polinomi . . . 161
14.3.2 La ricorrenza a tre termini . . . 162
14.3.3 Rodrigues e la dimostrazione dell'ortogonalità . . . 162
14.3.4 Equazione differenziale e funzione generatrice . . . 165
14.3.5 La famiglia in un grafico . . . 165
14.4 Esercizi . . . 167

15 Peso di Jacobi e famiglie classiche . . . 169
15.1 Il peso di Jacobi . . . 169
15.1.1 Definizione e condizione sui parametri . . . 169
15.1.2 La funzione Gamma . . . 169
15.1.3 La norma dei polinomi di Jacobi . . . 170
15.2 Le famiglie classiche come casi particolari . . . 170
15.2.1 La tabella dei casi speciali . . . 170
15.2.2 Chebyshev: coseni travestiti da polinomi . . . 171
15.2.3 La proprietà di minimo dei polinomi di Chebyshev . . . 171
15.2.4 Laguerre e Hermite: domini illimitati . . . 174
15.3 Il peso decide dove si concentra l'errore . . . 175
15.4 La ricorrenza dei polinomi di Jacobi . . . 176
15.4.1 I coefficienti in forma compatta . . . 176
15.4.2 Due casi particolari . . . 177
15.5 Perché l'ortogonalità forza tre termini . . . 177
15.6 Esercizi . . . 178

16 Ricorrenze, zeri e stabilità . . . 179
16.1 La ricorrenza a tre termini . . . 179
16.1.1 Dall'ortogonalità alla ricorrenza . . . 179

16.1.2 Il teorema di Favard . . . 180
16.1.3 La matrice di Jacobi . . . 180
16.2 Gli zeri dei polinomi ortogonali . . . 182
16.2.1 Zeri reali, interni e semplici . . . 182
16.2.2 Separazione degli zeri di gradi consecutivi . . . 183
16.3 Polinomi del nucleo e minimo vincolato . . . 185
16.3.1 Il nucleo e la proprietà riproducente . . . 185
16.3.2 Il polinomio di norma minima vincolata . . . 185
16.3.3 La funzione di Christoffel . . . 187
16.4 Stabilità della ricorrenza nel calcolo . . . 187
16.5 Esercizi . . . 188

17 Minimi quadrati con basi ortogonali . . . 189
17.1 Il problema, e tre domande . . . 189
17.2 Impostazione del problema . . . 189
17.2.1 Che cosa si vuole minimizzare . . . 189
17.2.2 La scrittura matriciale . . . 190
17.2.3 Le equazioni normali . . . 190
17.2.4 Che cosa cambia cambiando base . . . 191
17.3 Un esempio svolto a mano: cinque punti, una parabola . . . 191
17.3.1 Prima versione: la base monomiale . . . 191
17.3.2 Seconda versione: una base ortogonale sui nodi . . . 192
17.3.3 Terza versione: costruire la base dai dati . . . 193
17.3.4 Le due basi a confronto, in R . . . 196
17.4 Tre esempi con l'algoritmo . . . 197
17.4.1 Esempio A: la base che i dati si costruiscono addosso . . . 198
17.4.2 Esempio B: alzare il grado non ricalcola nulla . . . 200
17.4.3 Esempio C: dati di qualità diversa . . . 202
17.5 Come si risolve davvero: la fattorizzazione QR . . . 203
17.5.1 Perché le equazioni normali sono una cattiva idea . . . 203
17.5.2 Che cosa sono Q e R . . . 203
17.5.3 Perché la QR risolve il problema . . . 203
17.5.4 Il sistema triangolare e il suo costo . . . 204
17.5.5 La QR sull'esempio dei cinque punti . . . 204
17.6 Prima domanda: quanto si guadagna cambiando base . . . 205
17.6.1 Le tre basi a confronto . . . 205
17.6.2 Riscalare l'intervallo . . . 207
17.7 Seconda domanda: lo spettro dei coefficienti e il plateau . . . 208
17.7.1 I coefficienti come spettro . . . 208
17.7.2 Perché compare un plateau . . . 209
17.7.3 Il criterio di arresto . . . 209
17.7.4 L'esperimento . . . 210
17.8 Terza domanda: il fitting incrementale . . . 212
17.8.1 L'algoritmo . . . 212
17.8.2 L'implementazione in R . . . 214
17.8.3 L'algoritmo sui dati rumorosi . . . 214

17.9 Il problema continuo . . . 215
17.9.1 La proiezione in norma pesata . . . 215
17.9.2 Esempio: proiezione di e^x e dipendenza dal peso . . . 215
17.9.3 Esempio: cambiare il peso cambia la proiezione . . . 217
17.10 Esercizi . . . 219

18 Quadrature di Gauss e zeri dei polinomi . . . 221
18.1 Esattezza di grado $2n-1$ . . . 221
18.1.1 Dall'ortogonalità all'esattezza . . . 221
18.1.2 Positività dei pesi . . . 222
18.2 Gauss–Legendre . . . 223
18.2.1 Nodi e pesi . . . 223
18.2.2 I pesi sono la funzione di Christoffel . . . 223
18.2.3 Esempio: costruzione e verifica in R base . . . 225
18.3 Gauss–Jacobi e cambio di intervallo . . . 227
18.3.1 Peso di Jacobi e mappa affine . . . 227
18.3.2 Esempio: Gauss–Jacobi asimmetrica . . . 227
18.3.3 Esempio: i coefficienti di una serie ortogonale . . . 228
18.4 Errore e varianti con nodi vincolati . . . 229
18.5 Esercizi . . . 229

19 Approssimazione spettrale e altre applicazioni . . . 231
19.1 Approssimazione spettrale . . . 231
19.1.1 Quanto velocemente decadono i coefficienti . . . 231
19.1.2 Esempio: convergenza dello sviluppo di Legendre di e^x . . . 232
19.2 I limiti: discontinuità e fenomeno di Gibbs . . . 232
19.3 Metodi spettrali . . . 234
19.3.1 Collocazione per un problema ai limiti . . . 234
19.4 Altre applicazioni . . . 235
19.4.1 Probabilità, chaos polinomiale e armoniche sferiche . . . 235
19.4.2 Gli Hermite e la misura gaussiana . . . 235
19.5 Esercizi . . . 236

VI Zeri di funzione

20 Zeri di funzione . . . 239
20.1 Classificazione dei metodi . . . 239
20.2 Notazione . . . 240
20.3 Metodo di bisezione (o dicotomico) . . . 240
20.4 Criteri di arresto e accuratezza raggiungibile . . . 242
20.4.1 Intervallo di incertezza e condizionamento della radice . . . 242
20.4.2 Criteri pratici . . . 243

20.5 Dalla bisezione ai metodi aperti . . . 243
20.6 Metodo di Newton-Raphson . . . 244
20.6.1 Derivazione e interpretazione geometrica . . . 244
20.6.2 Convergenza locale . . . 244
20.6.3 Convergenza globale per funzioni convesse . . . 246
20.6.4 Quando Newton fallisce . . . 246
20.7 Metodo delle secanti . . . 247
20.7.1 Convergenza locale . . . 248
20.7.2 Variante con differenza finita di sicurezza . . . 250
20.8 Metodo delle false posizioni (*regula falsi*) . . . 251
20.8.1 Convergenza della *regula falsi* . . . 252
20.9 Metodo delle corde (derivata congelata) . . . 253
20.10 Metodo di Müller: interpolazione parabolica su tre punti . . . 254
20.10.1 Costruzione del metodo . . . 254
20.10.2 Esempio numerico . . . 254
20.11 Metodo della parabola tangente (Taylor quadratico) . . . 255
20.12 Metodo di Halley . . . 257
20.13 Confronto numerico dei metodi . . . 259

21 Ordine di convergenza e radici multiple . . . 265
21.1 Velocità di convergenza . . . 265
21.1.1 Q-convergenza . . . 265
21.1.2 R-convergenza . . . 266
21.1.3 Ordine ed efficienza dei metodi del capitolo . . . 267
21.2 Verifica numerica dell'ordine . . . 268
21.2.1 Stima dell'ordine senza conoscere la radice . . . 270
21.3 Radici multiple e rallentamento della convergenza . . . 274
21.3.1 Rallentamento della convergenza . . . 274
21.3.2 Esempio in R . . . 274
21.3.3 Una radice doppia perturbata . . . 275
21.3.4 Correzione del metodo di Newton per radici multiple . . . 276
21.4 Esercizi . . . 277

22 Metodi ibridi e algoritmi con salvaguardia . . . 279
22.1 Lo schema generale . . . 279
22.1.1 Il problema di riferimento . . . 280
22.2 Metodo di Dekker . . . 280
22.2.1 Come funziona . . . 280
22.2.2 Esempio numerico . . . 281
22.3 Metodo di Brent–Dekker . . . 282
22.3.1 Come funziona . . . 282
22.3.2 Esempio numerico . . . 284
22.4 Metodo di Ridders . . . 285
22.4.1 Come funziona . . . 285
22.4.2 Esempio numerico . . . 285

22.5 Metodo TOMS 748 (Alefeld–Potra–Shi) . . . 286
22.5.1 Come funziona . . . 286
22.5.2 Esempio numerico . . . 287
22.6 Metodo di Chandrupatla . . . 290
22.6.1 Come funziona . . . 290
22.6.2 Esempio numerico . . . 290
22.7 Metodo ITP (Interpolate–Truncate–Project) . . . 292
22.7.1 Come funziona . . . 292
22.7.2 Esempio numerico . . . 292
22.8 Gli ibridi con derivate: uno schema comune . . . 294
22.8.1 Il test di ammissibilità . . . 294
22.9 Metodo Newton–Bisection . . . 296
22.10 Metodo Halley–Bisection . . . 297
22.11 Metodo Chebyshev–Bisection . . . 297
22.12 Metodo Householder–Bisection . . . 298
22.13 Confronto sperimentale . . . 299
22.14 Esercizi . . . 300

23 Iterazioni di punto fisso . . . 303
23.1 Elenco metodi numerici per zeri di funzione . . . 307
23.1.1 Newton-Raphson . . . 307
23.1.2 Secanti . . . 307
23.1.3 Secanti (variante Brent) . . . 307
23.1.4 Parabole tangenti . . . 307
23.1.5 Metodo di Halley . . . 307
23.1.6 Metodo di Steffensen . . . 308
23.1.7 Metodo delle corde tangenti . . . 308
23.1.8 Metodo di Ostrowski . . . 308

24 Zeri di polinomi . . . 309
24.1 Il metodo di Sturm . . . 309
24.1.1 Eliminazione delle radici multiple . . . 309
24.1.2 Algoritmo di Euclide . . . 311
24.1.3 Separazione delle radici . . . 313
24.1.4 Costruzione della successione di Sturm per un polinomio . . . 314
24.2 Limitazione delle radici . . . 315

VII Sistemi lineari: metodi iterativi

25 Preliminari: raggio spettrale, serie di Neumann, M-matrici . . . 321
25.1 Raggio spettrale . . . 321
25.2 Serie di Neumann . . . 321
25.3 M-matrici . . . 321

26 Metodi Iterativi per Sistemi Lineari . . . 323
26.1 Costruzione degli Schemi di Iterazione mediante Splitting . . . 324
26.2 Metodo di Jacobi . . . 324
26.3 Metodo di Gauss-Seidel . . . 325
26.4 Metodo SOR . . . 325
26.5 Derivazione per componenti dei metodi di Jacobi, Gauss–Seidel e SOR 326
26.5.1 Metodo di Jacobi . . . 326
26.5.2 Metodo di Gauss–Seidel . . . 326
26.5.3 Metodo SOR (Successive Over-Relaxation) . . . 327
26.6 Generalizzazione . . . 328
26.7 Convergenza degli schemi iterativi . . . 329
26.8 Come si trasforma l'errore secondo lo schema iterativo generale? . . . 329
26.8.1 Trasformazione del residuo secondo lo schema iterativo generale . . . 330
26.9 Controllo della Convergenza . . . 333
26.9.1 Stima della velocità di convergenza . . . 333
26.9.2 Criteri di Arresto . . . 333
26.10 Quale metodo converge più in fretta . . . 334

VIII Equazioni differenziali ordinarie

27 Equazioni Differenziali Ordinarie . . . 339
27.1 Introduzione ai Problemi a Valore Iniziale . . . 339
27.1.1 Equazione Differenziale Ordinaria . . . 339
27.1.2 ODE esplicita . . . 340
27.1.3 ODE autonoma . . . 340
27.1.4 Riduzione a sistemi del primo ordine . . . 340
27.1.5 Problema a Valore Iniziale (IVP) . . . 341
27.2 Buona posizione dei problemi a valore iniziale . . . 341
27.2.1 Condizione di Lipschitz . . . 341
27.2.2 Funzione Localmente Lipschitziana . . . 341
27.3 Teoremi Fondamentali . . . 342
27.3.1 Teorema di Esistenza di Peano . . . 342
27.3.2 Teorema di Picard-Lindelöf . . . 343
27.3.3 Lemma di Grönwall . . . 344
27.3.4 Dimostrazione di Picard-Lindelöf . . . 345
27.3.5 Dipendenza continua dai dati iniziali . . . 346
27.3.6 Prolungamento delle soluzioni . . . 347

28 Soluzione numerica . . . 349
28.1 Passo Temporale . . . 349
28.2 Principali metodi numerici per IVP . . . 349
28.2.1 Metodi a Un Passo . . . 349
28.2.2 Metodi a Più Passi . . . 349
28.2.3 Metodi Espliciti . . . 349
28.2.4 Metodi Impliciti . . . 350

28.3 Il Metodo di Eulero . . . 350
28.3.1 Metodo di Eulero Esplicito . . . 350
28.3.2 Metodo di Eulero Implicito . . . 350
28.4 Convergenza Metodo di Eulero . . . 350
28.4.1 Convergenza Metodo di Eulero Esplicito . . . 350
28.4.2 Convergenza Metodo di Eulero Implicito . . . 352
28.4.3 Generalizzazione . . . 354
28.4.4 Funzione di avanzamento per altri schemi . . . 355
28.4.5 Propagazione dell'errore . . . 355
28.4.6 Convergenza dei metodi a un passo . . . 356
28.4.7 Generalizzazione (metodi impliciti) . . . 358
28.4.8 Dimostrazione della convergenza . . . 359
28.5 Verifica Numerica dell'Ordine . . . 360
28.5.1 Analisi Teorica . . . 360

29 Metodi di Taylor per ODE . . . 365
29.1 Metodo di Eulero e Serie di Taylor . . . 365
29.2 Metodo di Taylor al Secondo Ordine . . . 366
29.3 Metodo di Taylor al Terzo Ordine . . . 366
29.3.1 Notazione Compatta . . . 367
29.3.2 Metodo di Taylor al Terzo Ordine . . . 367
29.3.3 Considerazioni sul metodo . . . 368
29.4 Metodo basato su Taylor con $f(t, y)$ nota . . . 368
29.4.1 Definizione ricorsiva di $D_k(t, y)$ . . . 368
29.4.2 Espansione della soluzione con $D_k(t, y)$ . . . 369
29.4.3 Esempio: $f(t, y) = t + y + ty$ . . . 369
29.5 Metodo di Taylor migliorato . . . 370
29.5.1 Esempio: $f(t, y) = t + y + ty$ . . . 371
29.6 Sistemi di ODE . . . 372
29.6.1 Il caso vettoriale: la ricorrenza non cambia . . . 372
29.6.2 Dalle derivate ai coefficienti di Taylor . . . 374
29.6.3 La regola di Leibniz e i sistemi polinomiali . . . 374
29.6.4 Esempio: il modello preda–predatore . . . 374
29.6.5 Esempio: il modello SIR . . . 377
29.6.6 Ordine e costo . . . 380

30 Metodi di Runge-Kutta . . . 383
30.1 Formulazione Generale . . . 383
Richiamo: sviluppo di Taylor in due variabili . . . 383
Calcolo dell'Errore Locale di Troncamento . . . 384
Sviluppo dell'Errore Locale di Troncamento . . . 384
Famiglia RK del secondo ordine . . . 385
Estensione della Procedura . . . 385
30.2 Condizioni di Ordine . . . 386
Ordine 1 (Metodo di Eulero) . . . 386
Ordine 2 . . . 386
Ordine 3 . . . 386
Ordine 4 . . . 387

Ordini Superiori . . . 387
30.3 Metodi di Runge-Kutta Espliciti . . . 387
30.3.1 Tableau di Butcher . . . 388
30.4 Controllo adattivo del passo . . . 391
Coppie annidate . . . 391
Accettazione del passo e controllore . . . 392
Esempio: la coppia di Bogacki-Shampine 3(2) . . . 392
30.5 Metodi di Runge-Kutta Impliciti . . . 393
30.5.1 Soluzione degli stadi impliciti . . . 393
Metodo di Eulero Implicito . . . 394
Metodo del Punto Medio Implicito . . . 394
Metodo di Crank-Nicolson . . . 395
Metodi di collocazione . . . 395
Metodi di Gauss-Legendre . . . 396
Metodi di Lobatto . . . 397
Metodi di Radau . . . 398

31 Stabilità assoluta . . . 401
31.1 L'equazione test di Dahlquist . . . 401
31.2 Funzione e regione di stabilità . . . 401
31.3 I metodi fondamentali . . . 402
31.3.1 Eulero esplicito . . . 402
31.3.2 Runge-Kutta espliciti del secondo ordine . . . 403
31.3.3 Runge-Kutta classico del quarto ordine . . . 403
31.3.4 Eulero implicito . . . 403
31.3.5 Metodo dei trapezi (Crank-Nicolson) . . . 403
31.4 A-stabilità e L-stabilità . . . 404
31.5 Dal caso scalare ai sistemi . . . 404
31.6 Diagrammi di stabilità . . . 405
31.7 Stabilità e accuratezza . . . 406
31.7.1 Problemi oscillanti e asse immaginario . . . 406
31.7.2 Procedura pratica . . . 407
31.8 Quadro riassuntivo . . . 407

References . . . 409

Appendici

IX Complementi sui polinomi ortogonali

A Complementi sui polinomi ortogonali . . . 417
A.1 Simmetria: cosa succede se il peso è pari . . . 417
A.2 Derivate dei polinomi ortogonali . . . 418
A.3 La formula di Christoffel–Darboux . . . 419
A.4 Completezza e proiezione ortogonale . . . 420

X Perron-Frobenius, M-matrici e splitting

B Il teorema di Perron-Frobenius . . . 425

B.1 Introduzione . . . 425
B.1.1 Autovalori e raggio spettrale . . . 425
B.1.2 Autovalore matrice irriducibile . . . 425
B.1.3 Autovalori e raggio spettrale . . . 427
B.1.4 Limitazione autovalori . . . 427
B.1.5 Raggio spettrale e autovalore . . . 428

B.2 Teorema di Perron-Frobenius . . . 428
B.2.1 Monotonia raggio spettrale . . . 429

C Confronto fra splitting e velocità di convergenza . . . 431

C.1 Autovalori del prodotto . . . 431
C.2 Splitting convergente . . . 431
C.3 Splitting vari . . . 431
C.4 Confronto raggio spettrale . . . 432
C.5 Splitting e inversa . . . 433
C.6 Monotonia e splitting debole . . . 433
C.6.1 Proprietà splitting debole . . . 433
C.7 Splitting debolemente non negativi . . . 435
C.8 Raggio spettrale e splitting debolmente non negativi . . . 436
C.9 Splitting non negativi e convergenza . . . 436
C.10 Confronto della velocità di convergenza fra alcuni metodi iterativi . . . 436
C.10.1 Specializzazione nel caso a due colori . . . 438
C.10.2 M-matrice e elementi diagonali . . . 438
C.10.3 AGA . . . 440
C.10.4 Confronto raggi spettrali . . . 441
C.10.5 M-matrice caratterizzazione . . . 441
C.10.6 Mappa isotona . . . 442



Indice

- **Riassuntone per l'esame** — volume a parte
  - Calcolo Numerico
  - Elenco domande
- **Esercizi svolti** — volume a parte
  - Esempio di fattorizzazione LU
  - Scelta a priori del numero di sottointervalli
  - Zeri di funzione (esempi)
  - Criteri per controllo ordine
- **Sistemi lineari: metodi diretti**
  - Metodo di eliminazione di Gauss
  - Fattorizzazione di Cholesky
  - Esempio di fattorizzazione LU
- **Fattorizzazione QR**
  - Metodo QR per sistemi lineari
  - Costruzione della fattorizzazione QR
  - La pseudoinversa di Moore–Penrose
- **Interpolazione polinomiale**
  - Interpolazione polinomiale
  - Interpolazione di Lagrange
  - Interpolazione di Newton
  - Esempio di Runge
- **Integrazione numerica**
  - Integrazione numerica
  - Integrazione composita
  - Formule di quadratura
  - Stime a posteriori
  - Scelta a priori del numero di sottointervalli
- **Polinomi ortogonali**
  - Da Gram–Schmidt ai polinomi di Legendre
  - Peso di Jacobi e famiglie classiche
  - Ricorrenze, zeri e stabilità
  - Minimi quadrati con basi ortogonali
  - Quadratura di Gauss
  - Approssimazione spettrale e applicazioni

- **Zeri di funzione**
  - Zeri di funzione
  - Ordine di convergenza e radici multiple
  - Metodi ibridi e algoritmi con salvaguardia
  - Iterazioni di punto fisso
  - Zeri di funzione: esempi
  - Zeri di polinomi
- **Sistemi lineari: metodi iterativi**
  - Preliminari: raggio spettrale, serie di Neumann, M-matrici
  - Metodi iterativi per sistemi lineari
- **Equazioni differenziali ordinarie**
  - Equazioni differenziali ordinarie
  - Soluzioni numeriche per problemi a valore iniziale (IVP)
  - Metodi numerici per ODE basati sulla serie di Taylor
  - Metodi di Runge–Kutta
  - Stabilità assoluta dei metodi a un passo
  - Criteri per controllo ordine
- **Bibliografia**
- **Appendice: complementi sui polinomi ortogonali**
  - Complementi sui polinomi ortogonali
- **Appendice: Perron–Frobenius, M-matrici e splitting**
  - Il teorema di Perron–Frobenius
  - Confronto fra splitting e velocità di convergenza

Sistemi lineari: metodi diretti

1 **Metodo di eliminazione di Gauss** 19
1.1 Introduzione 19
1.2 Forma Matriciale del Metodo di Gauss 24
1.3 Proprietà delle matrici di Frobenius 25
1.4 Primo passo del metodo di Gauss 28
1.5 k–esimo passo del metodo di Gauss 29
1.6 Algoritmo di Gauss in presenza di pivoting 31

2 **Fattorizzazione di Cholesky** 37
2.1 Come si riconosce una matrice SPD: il criterio di Sylvester 40
2.2 Algoritmo di calcolo 42
2.3 Connessioni con la fattorizzazione di Gauss . . . 44



1. Metodo di eliminazione di Gauss

1.1 Introduzione

Il metodo di eliminazione di Gauss[1] è una procedura sistematica per risolvere sistemi di equazioni lineari. Consiste nel trasformare il sistema iniziale in un sistema equivalente di struttura più semplice, detto *in forma triangolare*, nel quale ogni equazione contiene un numero non crescente di incognite.

L'obiettivo del metodo è annullare, passo dopo passo, i coefficienti al di sotto della diagonale principale della matrice associata al sistema, mediante operazioni elementari sulle righe (scambi, moltiplicazioni per uno scalare e somme di righe). Una volta ottenuta la forma triangolare, il sistema si risolve mediante *sostituzioni all'indietro*: si parte dall'ultima equazione e si procede a ritroso fino alla prima.

Il metodo di Gauss è alla base di numerose tecniche numeriche per la risoluzione dei sistemi lineari.

Applicato a un sistema lineare della forma:

$$\begin{cases} A_{1,1}x_1 + A_{1,2}x_2 + \cdots + A_{1,n}x_n = b_1 \\ A_{2,1}x_1 + A_{2,2}x_2 + \cdots + A_{2,n}x_n = b_2 \\ \vdots \\ A_{n,1}x_1 + A_{n,2}x_2 + \cdots + A_{n,n}x_n = b_n \end{cases}$$

il metodo lo trasforma in un sistema equivalente, cioè con lo stesso insieme di soluzioni, della forma:

$$\begin{cases} C_{1,1}x_1 + C_{1,2}x_2 + \cdots + C_{1,n}x_n = d_1 \\ C_{2,2}x_2 + \cdots + C_{2,n}x_n = d_2 \\ \vdots \\ C_{n,n}x_n = d_n \end{cases} \tag{1.1}$$

detta *forma triangolare*.

Un sistema in forma triangolare si risolve mediante *sostituzioni all'indietro*: si parte dall'ultima incognita, il cui valore è determinato dall'ultima equazione, e si procede risalendo il sistema. In formula:

[1] Johann Carl Friedrich Gauss (1777-1855)

$$\begin{aligned} x_n &= \frac{d_n}{C_{n,n}} \\ x_{n-1} &= \frac{d_{n-1} - C_{n-1,n}x_n}{C_{n-1,n-1}} \\ x_{n-2} &= \frac{d_{n-2} - C_{n-2,n}x_n - C_{n-2,n-1}x_{n-1}}{C_{n-2,n-2}} \\ &\vdots \\ x_1 &= \frac{d_1 - C_{1,n}x_n - C_{1,n-1}x_{n-1} - \cdots - C_{1,2}x_2}{C_{1,1}} \end{aligned} \tag{1.2}$$

▪ **Example 1.1 — Esempio.** La soluzione del sistema in forma triangolare:

$$\begin{cases} x + y + z = 1 \\ 4y + 2z = 0 \\ z = 2 \end{cases}$$

(nel quale abbiamo posto $x = x_1$, $y = x_2$ e $z = x_3$), si ottiene immediatamente sostituendo all'indietro le variabili come mostrato in 1.2,

$$\begin{aligned} z &= \frac{2}{1} = 2, \\ y &= \frac{0 - 2z}{4} = \frac{0 - 2 \cdot 2}{4} = -1, \\ x &= \frac{1 - 1z - 1y}{1} = \frac{1 - 1(-1) - 1 \cdot 2}{1} = 0. \end{aligned}$$

▪

L'algoritmo di eliminazione di Gauss elimina, a ogni passo, una variabile esprimendola come combinazione lineare di quelle non ancora eliminate. A questo scopo si sceglie un'equazione, che viene poi usata per sostituire la variabile nelle altre equazioni. In termini matriciali, il procedimento consiste nel costruire opportune combinazioni lineari delle equazioni, e quindi delle righe della matrice dei coefficienti.

Per fissare le idee, illustriamo il procedimento con un esempio dettagliato.

▪ **Example 1.2 — Esempio.** Sia dato il seguente sistema lineare

$$\begin{cases} x + y + z + w = 1 & [1] \\ -x - y + 4z = 0 & [2] \\ x - z - w = 2 & [3] \\ x + w = 0 & [4] \end{cases}$$

Le equazioni sono numerate a destra tra parentesi quadre per facilitarne il riferimento. Trasformiamo il sistema in forma triangolare eliminando, un passo alla volta, le incognite x, y, z[2] mediante combinazioni lineari delle equazioni.

Per eliminare l'incognita x dalle equazioni [2], [3] e [4], eseguiremo le seguenti operazioni:

- **Eliminare x da** [2]: Sommiamo l'equazione [1] all'equazione [2] per ottenere la nuova equazione [2′]:

$$[2'] \leftarrow [2] + [1]$$

[2] Perché non si elimina l'incognita w?

- **Eliminare** x **da** $[3]$: Sottraiamo l'equazione $[1]$ dall'equazione $[3]$ per ottenere la nuova equazione $[3']$:
$$[3'] \leftarrow [3] - [1]$$
- **Eliminare** x **da** $[4]$: Sottraiamo l'equazione $[1]$ dall'equazione $[4]$ per ottenere la nuova equazione $[4']$:
$$[4'] \leftarrow [4] - [1]$$

Si ottiene:

$$\begin{cases} x + y + z + w = 1 & [1] \\ 5z + w = 1 & [2'] \\ -y - 2z - 2w = 1 & [3'] \\ -y - z = -1 & [4'] \end{cases}$$

Osserviamo che l'equazione $[2']$ non contiene la variabile y, mentre l'equazione $[3']$ la contiene. Per semplificare il sistema, scambiamo l'equazione $[2']$ con la $[3']$ prima di procedere all'eliminazione successiva:

$$\begin{cases} x + y + z + w = 1 & [1] \\ -y - 2z - 2w = 1 & [3'] \\ 5z + w = 1 & [2'] \\ -y - z = -1 & [4'] \end{cases}$$

Per eliminare l'incognita y dall'equazione $[4']$, eseguiremo la seguente operazione:

- **Eliminare** y **da** $[4']$: Sommiamo l'equazione $[3']$ all'equazione $[4']$ per ottenere la nuova equazione $[4'']$:
$$[4''] \leftarrow [4'] - [3']$$

Si ottiene:

$$\begin{cases} x + y + z + w = 1 & [1] \\ -y - 2z - 2w = 1 & [3'] \\ 5z + w = 1 & [2'] \\ z + 2w = -2 & [4''] \end{cases}$$

Per eliminare l'incognita z dall'equazione $[4'']$, eseguiremo la seguente operazione:

- **Eliminare** z **da** $[4'']$: Sottraiamo $\frac{1}{5}$ volte l'equazione $[2']$ dall'equazione $[4'']$ per ottenere la nuova equazione $[4''']$:
$$[4'''] \leftarrow [4''] - \frac{1}{5}[2']$$

Si ottiene:

$$\begin{cases} x + y + z + w = 1 & [1] \\ -y - 2z - 2w = 1 & [3'] \\ 5z + w = 1 & [2'] \\ \frac{9}{5}w = -\frac{11}{5} & [4'''] \end{cases}$$

Il sistema è ora in forma triangolare e, tramite il procedimento di sostituzioni all'indietro 1.2, possiamo calcolare la soluzione:

$$
\begin{aligned}
w &= \frac{-\frac{11}{5}}{\frac{9}{5}} = -\frac{11}{9} \\
z &= \frac{1-\left(-\frac{11}{9}\right)}{5} = \frac{4}{9} \\
y &= \frac{1+2\left(-\frac{11}{9}\right)+2\frac{4}{9}}{-1} = \frac{5}{9} \\
x &= \frac{1-\left(-\frac{11}{9}\right)-\frac{4}{9}-\frac{5}{9}}{1} = \frac{11}{9}
\end{aligned}
$$

▪

Osservazione

Per trasformare un sistema lineare da una forma generica a una forma triangolare, sono sufficienti solo due tipi di operazioni:

1. **Somma di un'equazione con un'altra moltiplicata per un opportuno scalare**: questa operazione consente di eliminare progressivamente le incognite da una o più equazioni e di portare il sistema in forma triangolare.
2. **Scambio di due equazioni**: Questa operazione è utile per posizionare le equazioni in modo che il sistema possa essere facilmente risolto tramite sostituzioni all'indietro.

Si noti che la soluzione di un sistema non cambia se:

- Si scambiano due equazioni tra loro.
- Si sostituisce un'equazione con la somma di se stessa e di un'altra equazione moltiplicata per uno scalare opportuno.

Queste operazioni mantengono l'equivalenza del sistema, preservando le soluzioni originali e facilitando il processo di riduzione a forma triangolare.

L'algoritmo di Gauss si può quindi descrivere come segue:

Metodo di Eliminazione di Gauss

Consideriamo un sistema lineare:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

dove

$$
\boldsymbol{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & A_{n-1,n} \\ A_{n,1} & \cdots & A_{n,n-1} & A_{n,n} \end{pmatrix}, \quad \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}.
$$

Metodo di eliminazione di Gauss

Dati: matrice $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ non singolare; termine noto $\boldsymbol{b} \in \mathbb{R}^n$
Restituisce: la soluzione $\boldsymbol{x}$ di $\boldsymbol{Ax} = \boldsymbol{b}$

Eliminazione progressiva
per $i = 1, \dots, n-1$ **fai**
  sia $k \geq i$ il primo indice con $A_{k,i} \neq 0$ ▷ *esiste: A non singolare*
  se $k \neq i$ **allora**
    scambia la riga i con la riga k (anche in $\boldsymbol{b}$)
  per $r = i+1, \dots, n$ **fai**
    $\beta \leftarrow A_{r,i} \big/ A_{i,i}$ ▷ *moltiplicatore*
    $b_r \leftarrow b_r - \beta\, b_i$
    per $j = i+1, \dots, n$ **fai**
      $A_{r,j} \leftarrow A_{r,j} - \beta\, A_{i,j}$ ▷ *la colonna i si annulla*

Sostituzione all'indietro
$x_n \leftarrow b_n \big/ A_{n,n}$
per $k = n-1, n-2, \dots, 1$ **fai**
  $\beta \leftarrow b_k$
  per $j = k+1, \dots, n$ **fai**
    $\beta \leftarrow \beta - A_{k,j}\, x_j$
  $x_k \leftarrow \beta \big/ A_{k,k}$

restituisci $\boldsymbol{x}$

Figure 1.1: Il metodo di eliminazione di Gauss: l'eliminazione progressiva porta il sistema in forma triangolare superiore operando contemporaneamente su $\boldsymbol{A}$ e su $\boldsymbol{b}$, la sostituzione all'indietro ricava poi le incognite a partire dall'ultima.

Alla fine dell'eliminazione progressiva la matrice $\boldsymbol{A}$ e il termine noto $\boldsymbol{b}$ hanno la forma

$$\boldsymbol{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ 0 & A_{2,2} & \cdots & A_{2,n} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & & A_{n-1,n} \\ 0 & 0 & \cdots & A_{n,n} \end{pmatrix}, \qquad \boldsymbol{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix},$$

e la sostituzione all'indietro può procedere dall'ultima equazione verso la prima.

Osservazione

Nel metodo di eliminazione di Gauss, l'operazione di scambio di equazioni è fondamentale per garantire che la matrice venga trasformata correttamente in una forma triangolare superiore. L'algoritmo standard prevede di scambiare le righe per trovare un elemento diverso da zero nella colonna corrente, ma esiste un criterio alternativo chiamato *pivoting* che può migliorare la stabilità numerica del metodo.

Pivoting

Il criterio di pivoting consiste nel selezionare, tra tutte le righe dalla riga corrente in poi, quella con il valore assoluto massimo nella colonna corrente. Precisamente, si procede come segue:

- Sia k l'indice tale che
$$\left|A^{(i)}_{k,i}\right| \geq \left|A^{(i)}_{j,i}\right|, \quad \forall j \geq i.$$
In altre parole, k è l'indice della riga con il valore assoluto più grande nella colonna i, a partire dalla riga i.

- **If** $k \neq i$ **then**
  - Scambia la i-esima riga con la k-esima riga nella matrice $\boldsymbol{A}^{(i)}$.
  - Scambia anche la componente i-esima con la componente k-esima del vettore $\boldsymbol{b}^{(i)}$.

Motivazione
Il pivoting ha lo scopo di migliorare l'accuratezza della soluzione calcolata. Questo è particolarmente importante quando si lavora con numeri molto piccoli. Infatti, se si divide per un numero molto vicino a zero, si rischia di ottenere risultati numerici imprecisi a causa dell'amplificazione degli errori di arrotondamento. Scegliendo l'elemento massimo nella colonna come pivot, si riduce il rischio di divisioni per numeri molto piccoli e si migliora la stabilità numerica dell'algoritmo.
In sintesi, il pivoting è una tecnica che aiuta a prevenire problemi numerici e a ottenere soluzioni più accurate per i sistemi di equazioni lineari.

1.2 Forma Matriciale del Metodo di Gauss

Il metodo di Gauss per la risoluzione di sistemi lineari può essere interpretato usando il formalismo delle matrici. Consideriamo un sistema lineare nella forma matriciale:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

dove $\boldsymbol{A}$ è una matrice quadrata $n \times n$, $\boldsymbol{x}$ è un vettore colonna di incognite, e $\boldsymbol{b}$ è un vettore colonna di termini noti.

Trasformazione in Forma Triangolare Superiore

L'obiettivo del metodo di Gauss è trasformare il sistema $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ in un sistema equivalente della forma:

$$\boldsymbol{U}\boldsymbol{x} = \boldsymbol{c}$$

dove $\boldsymbol{U}$ è una matrice triangolare superiore e $\boldsymbol{c}$ è un vettore modificato di termini noti. La matrice $\boldsymbol{U}$ è ottenuta applicando una serie di operazioni elementari al sistema originale.

Passaggi per la Trasformazione

La trasformazione dal sistema originale $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ al sistema triangolare superiore $\boldsymbol{U}\boldsymbol{x} = \boldsymbol{c}$ avviene attraverso i seguenti passaggi:

1. **Operazioni Elementari**: Utilizziamo matrici elementari, chiamate *matrici di Frobenius*, per applicare operazioni elementari alle righe della matrice $\boldsymbol{A}$. Queste operazioni nel caso del metodo di Gauss si riducono a:
   - **Addizione di un multiplo di una riga a un'altra riga**.
2. **Matrici di Scambio**: Se necessario, scambiamo due righe della matrice $\boldsymbol{A}$ utilizzando matrici di permutazione, che sono matrici identità con due righe scambiate. Queste matrici assicurano che l'algoritmo non fallisca a causa di zeri nei pivots (elementi diagonali).

Matrici di Frobenius

Le matrici di Frobenius[3] sono matrici elementari utilizzate per eseguire operazioni sui sistemi lineari. Ogni matrice di Frobenius rappresenta una trasformazione elementare specifica:

[3] Ferdinand Georg Frobenius (1849-1917)

1. **Addizione di Multipli di Righe ad Altre Righe**: Una matrice di Frobenius per aggiungere multipli $\beta_{k+1}, \beta_{k+2}, \dots, \beta_n$ della riga k-esima alle righe $k+1$, $k+2$ fino a n è una matrice identità con gli β_j posizionati nella k-esima colonna. Per scriverla esplicitamente consideriamo il vettore

$$\boldsymbol{v} = \left(0, \dots, 0, \beta_{k+1}, \dots, \beta_n\right)^T$$

cioè tale che

$$v_i = 0, \qquad i = 1, 2, \dots, k$$

in questo modo la matrice di Frobenius $\boldsymbol{F}$ diventa $\boldsymbol{F} = \boldsymbol{I} + \boldsymbol{v}\boldsymbol{e}_k^T$, o più esplicitamente:

$$\boldsymbol{F} = \left(\begin{array}{ccc|c|ccc} 1 & & & 0 & 0 & \cdots & 0 \\ & \ddots & & \vdots & \vdots & & \vdots \\ & & 1 & 0 & 0 & \cdots & 0 \\ \hline 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \hline 0 & \cdots & 0 & \beta_{k+1} & 1 & & \\ \vdots & & \vdots & \vdots & & \ddots & \\ 0 & \cdots & 0 & \beta_n & & & 1 \end{array}\right)$$

2. **Scambio di Due Righe**: Una matrice di permutazione per scambiare la i-esima e la j-esima riga (denotata con $\boldsymbol{S}_{i,j}$) è una matrice identità con le righe i e j scambiate. Esplicitamente prende la forma:

$$\boldsymbol{S}_{i,j} = \left(\begin{array}{ccc|ccc|cc} 1 & & & & & & & \\ & \ddots & & & & & & \\ & & 1 & & & & & \\ \hline & & & 0 & & & 1 & \\ \hline & & & & 1 & & & \\ & & & & & \ddots & & \\ & & & & & & 1 & \\ & & & 1 & & & 0 & \\ \hline & & & & & & & 1 \\ \hline & & & & & & & \ddots \\ & & & & & & & & 1 \end{array}\right) \tag{1.3}$$

La stessa matrice ammette la seguente rappresentazione più compatta:

$$\boldsymbol{S}_{i,j} = \boldsymbol{I} - (\boldsymbol{e}_i - \boldsymbol{e}_j)(\boldsymbol{e}_i - \boldsymbol{e}_j)^T$$

1.3 Proprietà delle matrici di Frobenius

Consideriamo la matrice di Frobenius $\boldsymbol{F}$ definita in 1.3 e la matrice $\boldsymbol{A}$,

$$\boldsymbol{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ \vdots & \vdots & & \vdots \\ A_{k,1} & A_{k,2} & \cdots & A_{k,n} \\ A_{k+1,1} & A_{k+1,2} & \cdots & A_{k+1,n} \\ \vdots & \vdots & & \vdots \\ A_{n,1} & A_{n,2} & \cdots & A_{n,n} \end{pmatrix},$$

il prodotto $\boldsymbol{F}\boldsymbol{A}$ è

$$\boldsymbol{F}\boldsymbol{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ \vdots & \vdots & & \vdots \\ A_{k,1} & A_{k,2} & \cdots & A_{k,n} \\ A_{k+1,1} + \beta_{k+1}A_{k,1} & A_{k+1,2} + \beta_{k+1}A_{k,2} & \cdots & A_{k+1,n} + \beta_{k+1}A_{k,n} \\ \vdots & \vdots & & \vdots \\ A_{n,1} + \beta_n A_{k,1} & A_{n,2} + \beta_n A_{k,2} & \cdots & A_{n,n} + \beta_n A_{k,n} \end{pmatrix}.$$

L'effetto della moltiplicazione di $\boldsymbol{F}$ sulla matrice $\boldsymbol{A}$ può essere descritto come segue:

- La riga i-esima della matrice $\boldsymbol{A}$ viene sostituita dalla somma della riga i-esima originale con un multiplo della riga k-esima, dove il multiplo è dato dallo scalare v_i.

In altre parole, stiamo effettuando le seguenti modifiche alle righe della matrice $\boldsymbol{A}$:

- Sostituiamo la riga $(k+1)$-esima con la riga

$$A_{k+1,\bullet} \leftarrow A_{k+1,\bullet} + \beta_{k+1} \cdot A_{k,\bullet}$$

- Sostituiamo la riga $(k+2)$-esima con la riga

$$A_{k+2,\bullet} \leftarrow A_{k+2,\bullet} + \beta_{k+2} \cdot A_{k,\bullet}$$

- E così via fino a:

$$A_{n,\bullet} \leftarrow A_{n,\bullet} + \beta_n \cdot A_{k,\bullet}$$

Lemma 1.1 — Prodotto di Matrici di Frobenius. Siano $\boldsymbol{F}_i = \boldsymbol{I} + \boldsymbol{v}_i \boldsymbol{e}_i^T$ le matrici di Frobenius, dove $\boldsymbol{v}_i$ è un vettore con zeri nelle prime i componenti, ossia della forma:

$$\boldsymbol{v}_i = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \beta_{i+1} \\ \beta_{i+2} \\ \vdots \\ \beta_n \end{pmatrix}$$

Allora, per ogni $2 \leq k \leq n-1$, si ha che:

$$\boldsymbol{F}_1 \boldsymbol{F}_2 \cdots \boldsymbol{F}_k = \boldsymbol{I} + \boldsymbol{v}_1 \boldsymbol{e}_1^T + \boldsymbol{v}_2 \boldsymbol{e}_2^T + \cdots + \boldsymbol{v}_k \boldsymbol{e}_k^T$$

Proof. La dimostrazione procede per induzione sull'indice k.

Base dell'Induzione ($k = 2$):

Calcoliamo il prodotto delle due matrici:

$$\begin{aligned} \boldsymbol{F}_1 \boldsymbol{F}_2 &= (\boldsymbol{I} + \boldsymbol{v}_1 \boldsymbol{e}_1^T)(\boldsymbol{I} + \boldsymbol{v}_2 \boldsymbol{e}_2^T) \\ &= \boldsymbol{I} + \boldsymbol{v}_1 \boldsymbol{e}_1^T + \boldsymbol{v}_2 \boldsymbol{e}_2^T + \boldsymbol{v}_1 (\boldsymbol{e}_1^T \boldsymbol{v}_2) \boldsymbol{e}_2^T \\ &= \boldsymbol{I} + \boldsymbol{v}_1 \boldsymbol{e}_1^T + \boldsymbol{v}_2 \boldsymbol{e}_2^T, \end{aligned}$$

poiché $\boldsymbol{e}_1^T \boldsymbol{v}_2 = (\boldsymbol{v}_2)_1 = 0$.

Passo Induttivo:

Supponiamo che il lemma sia vero per un indice generico $k-1$, cioè:

$$\boldsymbol{F}_1 \boldsymbol{F}_2 \cdots \boldsymbol{F}_{k-1} = \boldsymbol{I} + \boldsymbol{v}_1 \boldsymbol{e}_1^T + \boldsymbol{v}_2 \boldsymbol{e}_2^T + \cdots + \boldsymbol{v}_{k-1} \boldsymbol{e}_{k-1}^T.$$

Ora, calcoliamo:

$$\begin{aligned}(\boldsymbol{F}_1\boldsymbol{F}_2\cdots\boldsymbol{F}_{k-1})\,\boldsymbol{F}_k &= \left(\boldsymbol{I}+\boldsymbol{v}_1\boldsymbol{e}_1^T+\boldsymbol{v}_2\boldsymbol{e}_2^T+\cdots+\boldsymbol{v}_{k-1}\boldsymbol{e}_{k-1}^T\right)\left(\boldsymbol{I}+\boldsymbol{v}_k\boldsymbol{e}_k^T\right)\\ &= \boldsymbol{I}+\boldsymbol{v}_1\boldsymbol{e}_1^T+\boldsymbol{v}_2\boldsymbol{e}_2^T+\cdots+\boldsymbol{v}_{k-1}\boldsymbol{e}_{k-1}^T+\boldsymbol{v}_k\boldsymbol{e}_k^T\\ &\quad+\boldsymbol{v}_1(\boldsymbol{e}_1^T\boldsymbol{v}_k)\boldsymbol{e}_k^T+\cdots+\boldsymbol{v}_{k-1}(\boldsymbol{e}_{k-1}^T\boldsymbol{v}_k)\boldsymbol{e}_k^T\\ &= \boldsymbol{I}+\boldsymbol{v}_1\boldsymbol{e}_1^T+\boldsymbol{v}_2\boldsymbol{e}_2^T+\cdots+\boldsymbol{v}_{k-1}\boldsymbol{e}_{k-1}^T+\boldsymbol{v}_k\boldsymbol{e}_k^T,\end{aligned}$$

poiché tutti i termini misti sono nulli, essendo

$$\boldsymbol{e}_i^T\boldsymbol{v}_k=(\boldsymbol{v}_k)_i=0,\qquad i=1,2,\dots,k-1.$$

■

Lemma 1.2 — Inversa della Matrice di Frobenius. Consideriamo la matrice di Frobenius $\boldsymbol{F}$ definita come:

$$\boldsymbol{F}=\boldsymbol{I}+\boldsymbol{v}\boldsymbol{e}_k^T$$

dove $\boldsymbol{I}$ è la matrice identità, $\boldsymbol{v}$ è un vettore colonna e $\boldsymbol{e}_k^T$ è il vettore riga unitario con 1 nella k-esima posizione e zeri altrove.

Quando $v_k=0$, l'inversa di $\boldsymbol{F}$ è data dalla matrice:

$$\boldsymbol{F}^{-1}=\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T.$$

Proof. Per verificare che $\boldsymbol{F}^{-1}=\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T$ sia effettivamente l'inversa di $\boldsymbol{F}$, dobbiamo mostrare che:

$$\boldsymbol{F}\cdot\boldsymbol{F}^{-1}=\boldsymbol{I}$$

Calcoliamo il prodotto:

$$\begin{aligned}\left(\boldsymbol{I}+\boldsymbol{v}\boldsymbol{e}_k^T\right)\left(\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T\right) &= \boldsymbol{I}\cdot\boldsymbol{I}-\boldsymbol{I}\cdot\boldsymbol{v}\boldsymbol{e}_k^T+\boldsymbol{v}\boldsymbol{e}_k^T\cdot\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T\cdot\boldsymbol{v}\boldsymbol{e}_k^T\\ &= \boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T+\boldsymbol{v}\boldsymbol{e}_k^T-\boldsymbol{v}(\boldsymbol{e}_k^T\boldsymbol{v})\boldsymbol{e}_k^T\\ &= \boldsymbol{I}-\boldsymbol{v}(\boldsymbol{e}_k^T\boldsymbol{v})\boldsymbol{e}_k^T\end{aligned}$$

Notiamo che i termini $\boldsymbol{v}\boldsymbol{e}_k^T$ si cancellano:

$$-\boldsymbol{v}\boldsymbol{e}_k^T+\boldsymbol{v}\boldsymbol{e}_k^T=0$$

Pertanto, rimane:

$$\left(\boldsymbol{I}+\boldsymbol{v}\boldsymbol{e}_k^T\right)\left(\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T\right)=\boldsymbol{I}-\boldsymbol{v}(\boldsymbol{e}_k^T\boldsymbol{v})\boldsymbol{e}_k^T$$

Per dimostrare che $\boldsymbol{F}^{-1}=\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T$ è effettivamente l'inversa di $\boldsymbol{F}$, dobbiamo verificare che il termine $\boldsymbol{v}(\boldsymbol{e}_k^T\boldsymbol{v})\boldsymbol{e}_k^T$ sia nullo. Poiché $\boldsymbol{e}_k^T\boldsymbol{v}$ è il k-esimo elemento del vettore $\boldsymbol{v}$, denotato come v_k, abbiamo:

$$\boldsymbol{e}_k^T\boldsymbol{v}=v_k$$

Se $v_k=0$ (cioè l'k-esimo elemento di $\boldsymbol{v}$ è zero), allora:

$$\left(\boldsymbol{I}+\boldsymbol{v}\boldsymbol{e}_k^T\right)\left(\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T\right)=\boldsymbol{I}$$

Poiché $v_k=0$ allora $\boldsymbol{F}^{-1}=\boldsymbol{I}-\boldsymbol{v}\boldsymbol{e}_k^T$ è l'inversa della matrice di Frobenius. ■

Osservazione

- Data la matrice $\boldsymbol{A}$ dove $\boldsymbol{A}_{i,\bullet}$ è la riga $i-$esima della matrice $\boldsymbol{A}$, l'effetto di $\boldsymbol{S}_{i,j}\boldsymbol{A}$ è il seguente:

$$\boldsymbol{A} = \begin{pmatrix} \vdots \\ \boldsymbol{A}_{i,\bullet} \\ \vdots \\ \boldsymbol{A}_{j,\bullet} \\ \vdots \end{pmatrix}, \qquad \boldsymbol{SA} = \begin{pmatrix} \vdots \\ \boldsymbol{A}_{j,\bullet} \\ \vdots \\ \boldsymbol{A}_{i,\bullet} \\ \vdots \end{pmatrix},$$

  cioè scambia la riga $i-$esima con la riga $j-$esima.
- Scambiando due volte le stesse righe di una matrice riotteniamo la matrice iniziale. Quindi, se $\boldsymbol{S}_{i,j}$ è una matrice di scambio allora $\boldsymbol{S}_{i,j}\boldsymbol{S}_{i,j} = \boldsymbol{I}$ cioè $\boldsymbol{S}_{i,j}$ coincide con la sua inversa, cioè è una matrice ortogonale.
- Il prodotto di due matrici di scambio è una matrice di permutazione, che modifica l'ordinamento delle righe di una matrice.
- Le matrici di permutazione si possono scrivere come il prodotto di matrici di scambio. In generale il prodotto di matrici ortogonali è una matrice ortogonale[4], e quindi l'inversa della matrice di permutazione coincide con la sua trasposta.

1.4 Primo passo del metodo di Gauss

Dato il sistema $\boldsymbol{Ax} = \boldsymbol{b}$ poniamo $\boldsymbol{A}^{(1)} = \boldsymbol{A}$ e $\boldsymbol{b}^{(1)} = \boldsymbol{b}$,

$$\boldsymbol{A}^{(1)} = \begin{pmatrix} A^{(1)}_{1,1} & A^{(1)}_{1,2} & \cdots & A^{(1)}_{1,n} \\ A^{(1)}_{2,1} & A^{(1)}_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & A^{(1)}_{n-1,n} \\ A^{(1)}_{n,1} & \cdots & A^{(1)}_{n,n-1} & A^{(1)}_{n,n} \end{pmatrix}.$$

Supponendo $A^{(1)}_{1,1} \neq 0$ definiamo

$$v^{(1)}_i = \begin{cases} 0 & i = 1 \\ \dfrac{A^{(1)}_{i,1}}{A^{(1)}_{1,1}} & i = 2, 3, \dots, n \end{cases},$$

$$\boldsymbol{L}_1 = \boldsymbol{I} - \boldsymbol{v}^{(1)}\boldsymbol{e}_1^T = \begin{pmatrix} 1 & 0 & \cdots & & 0 \\ -v^{(1)}_2 & 1 & & & \vdots \\ \vdots & & \ddots & & 0 \\ -v^{(1)}_{n-1} & & & 1 & 0 \\ -v^{(1)}_n & 0 & \cdots & 0 & 1 \end{pmatrix}.$$

Moltiplicando il sistema $\boldsymbol{A}^{(1)}\boldsymbol{x} = \boldsymbol{b}^{(1)}$ a sinistra per la matrice di Frobenius $\boldsymbol{L}_1$ otteniamo il sistema trasformato $\boldsymbol{A}^{(2)}\boldsymbol{x} = \boldsymbol{b}^{(2)}$ dove,

[4] Se $\boldsymbol{A}$ e $\boldsymbol{B}$ sono matrici ortogonali, allora $(\boldsymbol{AB})^T(\boldsymbol{AB}) = \boldsymbol{B}^T(\boldsymbol{A}^T\boldsymbol{A})\boldsymbol{B} = \boldsymbol{B}^T\boldsymbol{IB} = \boldsymbol{B}^T\boldsymbol{B} = \boldsymbol{I}$ e quindi il prodotto è una matrice ortogonale.

$$\boldsymbol{A}^{(2)} = \boldsymbol{L}_1\boldsymbol{A}^{(1)} = \left(\begin{array}{c|ccc} A^{(1)}_{1,1} & A^{(1)}_{1,2} & A^{(1)}_{1,3} & \cdots & A^{(1)}_{1,n} \\ \hline 0 & A^{(2)}_{2,2} & A^{(2)}_{2,3} & \cdots & A^{(2)}_{2,n} \\ 0 & A^{(2)}_{3,2} & A^{(2)}_{3,3} & \cdots & A^{(2)}_{3,n} \\ \vdots & \vdots & & & \vdots \\ 0 & A^{(2)}_{n,2} & A^{(2)}_{n,3} & \cdots & A^{(2)}_{n,n} \end{array}\right), \quad \boldsymbol{b}^{(2)} = \boldsymbol{L}_1\boldsymbol{b}^{(1)} = \begin{pmatrix} b^{(1)}_1 \\ b^{(2)}_2 \\ b^{(2)}_3 \\ \vdots \\ b^{(2)}_n \end{pmatrix}.$$

ed inoltre si ha

$$\begin{aligned} A^{(2)}_{i,j} &= A^{(1)}_{i,j} - v^{(1)}_i A^{(1)}_{1,j}, && i,j = 2,3,\dots,n \\ b^{(2)}_i &= b^{(1)}_i - v^{(1)}_i b^{(1)}_1 && i = 2,3,\dots,n. \end{aligned}$$

1.5 $k-$esimo passo del metodo di Gauss

Supponiamo di aver effettuato $k-1$ passi dell'algoritmo di Gauss (senza permutazione di righe). Siamo quindi nella situazione

$$\boldsymbol{A}^{(k)} = \boldsymbol{L}_{k-1}\cdots\boldsymbol{L}_2\boldsymbol{L}_1\boldsymbol{A}^{(1)}$$

e

$$\boldsymbol{b}^{(k)} = \boldsymbol{L}_{k-1}\cdots\boldsymbol{L}_2\boldsymbol{L}_1\boldsymbol{b}^{(1)}$$

dove

$$\boldsymbol{A}^{(k)} = \left(\begin{array}{ccc|ccc} A^{(1)}_{1,1} & \cdots & A^{(1)}_{1,k-1} & {}^{(1)}_{1,k} & \cdots & A^{(1)}_{1,n} \\ & \ddots & \vdots & \vdots & & \vdots \\ 0 & & A^{(k-1)}_{k-1,k-1} & A^{(k-1)}_{k-1,k} & \cdots & A^{(k-1)}_{k-1,n} \\ \hline 0 & \cdots & 0 & A^{(k)}_{k,k} & \cdots & A^{(k)}_{k,n} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & A^{(k)}_{n,k} & \cdots & A^{(k)}_{n,n} \end{array}\right), \quad \boldsymbol{b}^{(k)} = \begin{pmatrix} b^{(1)}_1 \\ \vdots \\ b^{(k-1)}_{k-1} \\ b^{(k)}_k \\ \vdots \\ b^{(k)}_n \end{pmatrix}.$$

Supponendo $A^{(k)}_{k,k} \neq 0$ e ponendo

$$v^{(k)}_i = \begin{cases} 0 & i = 1,2,\dots,k \\ \dfrac{A^{(k)}_{i,k}}{A^{(k)}_{k,k}} & i = k+1, k+2, \dots, n \end{cases},$$

$$\boldsymbol{L}_k = \boldsymbol{I} - \boldsymbol{v}_k\boldsymbol{e}_k^T = \left(\begin{array}{ccc|c|ccc} 1 & & & 0 & 0 & \cdots & 0 \\ & \ddots & & \vdots & \vdots & & \vdots \\ & & 1 & 0 & 0 & \cdots & 0 \\ \hline 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ \hline 0 & \cdots & 0 & -v^{(k)}_{k+1} & 1 & & \\ \vdots & & \vdots & \vdots & & \ddots & \\ 0 & \cdots & 0 & -v^{(k)}_n & & & 1 \end{array}\right),$$

otteniamo

$$\boldsymbol{L}_k\boldsymbol{A}^{(k)} = \left(\begin{array}{ccc|ccc} A_{1,1}^{(1)} & \cdots & A_{1,k-1}^{(1)} & A_{1,k}^{(1)} & A_{1,k+1}^{(1)} & \cdots & A_{1,n}^{(1)} \\ & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & & A_{k-1,k-1}^{(k-1)} & A_{k-1,k}^{(k-1)} & A_{k-1,k+1}^{(k-1)} & \cdots & A_{k-1,n}^{(k-1)} \\ \hline 0 & \cdots & 0 & A_{k,k}^{(k)} & A_{k,k+1}^{(k)} & \cdots & A_{k,n}^{(k)} \\ 0 & \cdots & 0 & 0 & A_{k+1,k+1}^{(k+1)} & \cdots & A_{k+1,n}^{(k+1)} \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & 0 & A_{n,k+1}^{(k+1)} & \cdots & A_{n,n}^{(k+1)} \end{array}\right), \quad \boldsymbol{L}_k\boldsymbol{b}^{(k)} = \begin{pmatrix} b_1^{(1)} \\ \vdots \\ b_{k-1}^{(k-1)} \\ b_k^{(k)} \\ b_{k+1}^{(k+1)} \\ \vdots \\ b_n^{(k+1)} \end{pmatrix},$$

dove

$$\begin{aligned} A_{i,j}^{(k+1)} &= A_{i,j}^{(k)} - v_i^{(k)} A_{k,j}^{(k)}, & i,j &= k+1, k+2, \dots, n \\ b_i^{(k+1)} &= b_i^{(k)} - v_i^{(k)} b_i^{(k)} & i &= k+1, k+2, \dots, n. \end{aligned}$$

L'algoritmo di Gauss applica lo stesso processo iterativo su matrici sempre più piccole fino a ottenere la triangolarizzazione della matrice originale. Ad ogni passo, le operazioni elementari di riga riducono progressivamente la dimensione della matrice fino a raggiungere una forma triangolare superiore.

Osservazione

Consideriamo una matrice quadrata $\boldsymbol{A}$ di dimensione $n \times n$ e supponiamo che ogni passo dell'algoritmo di Gauss sia applicabile, cioè $A_{k,k}^{(k)} \neq 0$ per ogni k. Dopo $n-1$ passi dell'algoritmo, otteniamo:

$$\boldsymbol{L}_{n-1}\boldsymbol{L}_{n-2}\cdots\boldsymbol{L}_1\boldsymbol{A} = \boldsymbol{U},$$

ponendo

$$\boldsymbol{L} = \boldsymbol{L}_1^{-1}\boldsymbol{L}_2^{-1}\cdots\boldsymbol{L}_{n-1}^{-1}.$$

possiamo scrivere

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}.$$

Inoltre:

1. Le matrici $\boldsymbol{L}_k = \boldsymbol{I} - \boldsymbol{v}_k\boldsymbol{e}_k^T$ sono matrici di Frobenius.
2. Le matrici $\boldsymbol{L}_k^{-1} = \boldsymbol{I} + \boldsymbol{v}_k\boldsymbol{e}_k^T$ sono ancora matrici di Frobenius.
3. Applicando il lemma Lemma 1.1, abbiamo:

$$\begin{aligned} \boldsymbol{L} &= \boldsymbol{L}_1^{-1}\boldsymbol{L}_2^{-1}\cdots\boldsymbol{L}_{n-1}^{-1}, \\ &= (\boldsymbol{I} + \boldsymbol{v}_1\boldsymbol{e}_1^T)(\boldsymbol{I} + \boldsymbol{v}_2\boldsymbol{e}_2^T)\cdots(\boldsymbol{I} + \boldsymbol{v}_{n-1}\boldsymbol{e}_{n-1}^T), \\ &= \boldsymbol{I} + \boldsymbol{v}_1\boldsymbol{e}_1^T + \boldsymbol{v}_2\boldsymbol{e}_2^T + \cdots + \boldsymbol{v}_{n-1}\boldsymbol{e}_{n-1}^T. \end{aligned}$$

Quindi, $\boldsymbol{L}$ è una matrice triangolare inferiore.
La matrice $\boldsymbol{A}$ è stata quindi decomposta come il prodotto di una matrice triangolare inferiore $\boldsymbol{L}$ e una matrice triangolare superiore $\boldsymbol{U}$. Questa decomposizione è nota come *decomposizione LU* o *fattorizzazione LU*.

Osservazione

Dalla decomposizione LU della matrice $\boldsymbol{A}$, abbiamo:

$$\boldsymbol{L} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ v_2^{(1)} & 1 & 0 & \cdots & 0 \\ v_3^{(1)} & v_3^{(2)} & \ddots & \ddots & \vdots \\ \vdots & \vdots & & 1 & 0 \\ v_n^{(1)} & v_n^{(2)} & \cdots & v_n^{(n-1)} & 1 \end{pmatrix}, \quad \text{dove} \quad v_i^{(j)} = \frac{A_{i,j}^{(j)}}{A_{j,j}^{(j)}}$$

$$\boldsymbol{U} = \begin{pmatrix} A_{1,1}^{(1)} & A_{1,2}^{(1)} & A_{1,3}^{(1)} & \cdots & A_{1,n}^{(1)} \\ 0 & A_{2,2}^{(2)} & A_{2,3}^{(1)} & \cdots & A_{2,n}^{(2)} \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \vdots & & A_{n-1,n-1}^{(n-1)} & A_{n-1,n}^{(n-1)} \\ 0 & 0 & \cdots & 0 & A_{n,n}^{(n)} \end{pmatrix}$$

Da quanto sopra, è evidente che le matrici $\boldsymbol{L}$ e $\boldsymbol{U}$ possono essere memorizzate all'interno della matrice originale $\boldsymbol{A}$, senza memorizzare esplicitamente la diagonale di $\boldsymbol{L}$. Questo approccio consente di implementare un algoritmo che calcola la decomposizione LU e salva direttamente il risultato nella stessa matrice $\boldsymbol{A}$.

L'algoritmo modifica la matrice $\boldsymbol{A}$ *in-place*: i coefficienti di eliminazione prendono il posto degli zeri sotto la diagonale, formando $\boldsymbol{L}$ (con 1 sottintesi sulla diagonale), mentre sopra la diagonale resta $\boldsymbol{U}$.

Decomposizione LU (senza pivoting)

Dati: matrice $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ con minori principali non nulli
Restituisce: $\boldsymbol{A}$ sovrascritta: $\boldsymbol{L}$ sotto la diagonale (con 1 implicito), $\boldsymbol{U}$ sulla diagonale e sopra

```
per i = 1, ..., n − 1 fai
    per r = i + 1, ..., n fai
        β ← A_{r,i} / A_{i,i}                ▷ moltiplicatore
        per j = i + 1, ..., n fai
            A_{r,j} ← A_{r,j} − β A_{i,j}    ▷ complemento di Schur
        A_{r,i} ← β                          ▷ al posto dello zero: L in-place
```

Figure 1.2: La decomposizione LU senza pivoting: è l'eliminazione di Gauss in cui il moltiplicatore, invece di essere buttato via, viene memorizzato al posto dell'elemento che ha appena annullato.

1.6 Algoritmo di Gauss in presenza di pivoting

Nel processo di eliminazione di Gauss di base, l'algoritmo esegue operazioni sulle righe per convertire una matrice in una forma triangolare superiore. Tuttavia, questo metodo può incontrare instabilità numerica, specialmente quando la matrice presenta elementi di pivot molto piccoli o è mal condizionata. Per migliorare la stabilità numerica e l'accuratezza, si utilizza il **pivoting parziale**.

Il **pivoting parziale** consiste nel selezionare l'elemento di valore assoluto massimo nella colonna corrente come pivot e scambiare le righe per portare questo elemento massimo nella posizione del pivot. Questa strategia aiuta a minimizzare gli errori di arrotondamento e migliora la robustezza dell'algoritmo.

Per analizzare l'algoritmo di eliminazione di Gauss con pivoting parziale, è necessario gestire

anche le matrici di scambio che vengono introdotte. Queste matrici sono utilizzate per scambiare righe e devono essere propagate attraverso le operazioni di fattorizzazione. Introduciamo quindi il seguente lemma per capire questo processo:

Lemma 1.3 — Frobenius e Scambi. Sia $\boldsymbol{F} = \boldsymbol{I} - \boldsymbol{v}\boldsymbol{e}_k^T$ una matrice di Frobenius e $\boldsymbol{S}_{i,j}$ una matrice di scambio che scambia le righe i e j. Se i e j sono entrambi maggiori di k, allora la matrice risultante $\boldsymbol{F}\boldsymbol{S}_{i,j}$ può essere scritta come $\boldsymbol{S}_{i,j}\boldsymbol{F}'$, dove $\boldsymbol{F}'$ è anch'essa una matrice di Frobenius. In particolare:

$$\boldsymbol{F}' = \boldsymbol{I} - \boldsymbol{w}\boldsymbol{e}_k^T, \quad \text{dove} \quad \boldsymbol{w} = \boldsymbol{S}_{i,j}\boldsymbol{v}.$$

Proof. Calcoliamo direttamente il prodotto $\boldsymbol{F}\boldsymbol{S}_{i,j}$ e confrontiamolo con la forma $\boldsymbol{S}_{i,j}\boldsymbol{F}'$.

1. **Osservazione sui vettori unitari**: Nota che $\boldsymbol{e}_k^T\boldsymbol{S}_{i,j}\boldsymbol{x} = x_k = \boldsymbol{e}_k^T\boldsymbol{x}$ se $i \neq k$ e $j \neq k$. Questo perché, quando i e j sono diversi da k, la matrice di scambio $\boldsymbol{S}_{i,j}$ non sposta la k-esima componente. Quindi possiamo scrivere $\boldsymbol{e}_k^T\boldsymbol{S}_{i,j} = \boldsymbol{e}_k^T$.
2. **Calcolo di $\boldsymbol{F}\boldsymbol{S}_{i,j}$**: Consideriamo il prodotto:

$$\begin{aligned}\boldsymbol{F}\boldsymbol{S}_{i,j} &= (\boldsymbol{I} - \boldsymbol{v}\boldsymbol{e}_k^T)\boldsymbol{S}_{i,j} \\ &= \boldsymbol{S}_{i,j} - \boldsymbol{v}\boldsymbol{e}_k^T\boldsymbol{S}_{i,j} \\ &= \boldsymbol{S}_{i,j} - \boldsymbol{v}\boldsymbol{e}_k^T \qquad \text{[per il punto 1]}\end{aligned}$$

3. **Calcolo di $\boldsymbol{S}_{i,j}\boldsymbol{F}'$**: Sostituendo $\boldsymbol{F}' = \boldsymbol{I} - \boldsymbol{w}\boldsymbol{e}_k^T$, otteniamo:

$$\begin{aligned}\boldsymbol{S}_{i,j}\boldsymbol{F}' &= \boldsymbol{S}_{i,j}(\boldsymbol{I} - \boldsymbol{w}\boldsymbol{e}_k^T) \\ &= \boldsymbol{S}_{i,j} - \boldsymbol{S}_{i,j}\boldsymbol{w}\boldsymbol{e}_k^T, \\ &= \boldsymbol{S}_{i,j} - \boldsymbol{S}_{i,j}\boldsymbol{S}_{i,j}\boldsymbol{v}\boldsymbol{e}_k^T. \\ &= \boldsymbol{S}_{i,j} - \boldsymbol{v}\boldsymbol{e}_k^T. \qquad \text{[per la proprietà } \boldsymbol{S}_{i,j}\boldsymbol{S}_{i,j} = \boldsymbol{I}\text{]}\end{aligned}$$

e quindi $\boldsymbol{F}\boldsymbol{S}_{i,j} = \boldsymbol{S}_{i,j}\boldsymbol{F}'$. ∎

L'algoritmo di Gauss con pivoting può essere formulato come segue:

$$\boldsymbol{L}_{n-1}\boldsymbol{P}_{n-1}\boldsymbol{L}_{n-2}\boldsymbol{P}_{n-2}\cdots\boldsymbol{L}_2\boldsymbol{P}_2\boldsymbol{L}_1\boldsymbol{P}_1\boldsymbol{A} = \boldsymbol{U},$$

dove $\boldsymbol{P}_k$ è una matrice di scambio, utilizzata se è necessario scambiare righe al k-esimo passo dell'algoritmo di Gauss. Quando non è necessario scambiare righe, $\boldsymbol{P}_k$ è la matrice identità. Si osserva che $\boldsymbol{P}_k = \boldsymbol{S}_{i,j}$ per $i, j \geq k$, e quindi, applicando il lemma Lemma 1.3, otteniamo che:

$$\boldsymbol{P}_k\boldsymbol{L}_{k-1} = \boldsymbol{L}'_{k-1}\boldsymbol{P}_k,$$

dove $\boldsymbol{L}'_{k-1} = \boldsymbol{I} - \boldsymbol{w}_k\boldsymbol{e}_k^T$ e $\boldsymbol{w}_k = \boldsymbol{P}_k\cdots\boldsymbol{P}_2\boldsymbol{P}_1\boldsymbol{v}_k$. Spostando le matrici di scambio a destra nell'equazione precedente, otteniamo:

$$\left(\boldsymbol{L}'_{n-1}\boldsymbol{L}'_{n-2}\cdots\boldsymbol{L}'_2\boldsymbol{L}'_1\right)\left(\boldsymbol{P}_{n-1}\boldsymbol{P}_{n-2}\cdots\boldsymbol{P}_2\boldsymbol{P}_1\right)\boldsymbol{A} = \boldsymbol{U},$$

dove la matrice $\boldsymbol{U}$ è triangolare superiore e le matrici $\boldsymbol{L}'_k$ sono matrici di Frobenius date da:

$$\boldsymbol{L}'_k = \boldsymbol{I} - \boldsymbol{w}_k\boldsymbol{e}_k^T, \qquad \boldsymbol{w}_k = \boldsymbol{P}_k\cdots\boldsymbol{P}_2\boldsymbol{P}_1\boldsymbol{v}_k.$$

Definiamo ora:

$$\boldsymbol{L} = (\boldsymbol{L}'_1)^{-1}(\boldsymbol{L}'_2)^{-1}\cdots(\boldsymbol{L}'_{n-1})^{-1}, \qquad \boldsymbol{P} = \boldsymbol{P}_{n-1}\boldsymbol{P}_{n-2}\cdots\boldsymbol{P}_2\boldsymbol{P}_1,$$

e otteniamo:

$$\boldsymbol{PA} = \boldsymbol{LU}.$$

Quindi, l'algoritmo di eliminazione consente di decomporre la matrice $\boldsymbol{PA}$ in un prodotto $\boldsymbol{LU}$, dove $\boldsymbol{P}$ rappresenta la sequenza di scambi di righe applicati alla matrice $\boldsymbol{A}$, e $\boldsymbol{L}$ e $\boldsymbol{U}$ sono rispettivamente la matrice triangolare inferiore e superiore ottenute tramite la decomposizione LU.

Osservazione

Si è illustrato il funzionamento dell'algoritmo di fattorizzazione con pivoting, noto come *pivot parziale*, che prevede solo lo scambio delle righe. Questo tipo di pivoting equivale a riordinare le equazioni del sistema lineare.
Esiste però un pivoting più generale, chiamato *pivot totale*, che consente sia scambi di righe che di colonne. In questo caso, lo scambio di colonne corrisponde a una diversa numerazione delle variabili nel sistema lineare.
Il metodo di fattorizzazione **LU** con pivot parziale si estende facilmente al pivot totale, e si ottiene la seguente decomposizione:

$$\boldsymbol{PAQ} = \boldsymbol{LU},$$

dove $\boldsymbol{P}$ è la matrice di permutazione delle righe, ottenuta come prodotto delle matrici elementari di scambio delle righe, e $\boldsymbol{Q}$ è la matrice di permutazione delle colonne, ottenuta come prodotto delle matrici elementari di scambio delle colonne.

Decomposizione LU con pivoting parziale

Dati: matrice $\boldsymbol{A} \in \mathbb{R}^{n\times n}$
Restituisce: $\boldsymbol{A}$ sovrascritta con i fattori $\boldsymbol{L}$ e $\boldsymbol{U}$;
vettore delle permutazioni $\boldsymbol{p}$

per $i = 1, \dots, n$ **fai**
  $p_i \leftarrow i$ ▷ *permutazione identica*
per $i = 1, \dots, n-1$ **fai**
  $i_p \leftarrow p_i$
  $k \leftarrow i, \quad k_p \leftarrow i_p$ ▷ *candidato pivot*
  per $j = i+1, \dots, n$ **fai**
    $j_p \leftarrow p_j$
    se $|A_{j_p,i}| > |A_{k_p,i}|$ **allora**
      $k \leftarrow j, \quad k_p \leftarrow j_p$ ▷ *pivot parziale: massimo in colonna*
  scambia p_i con p_k, $\quad i_p \leftarrow p_i$ ▷ *lo scambio è solo sugli indici*
  per $r = i+1, \dots, n$ **fai**
    $r_p \leftarrow p_r$
    $\beta \leftarrow A_{r_p,i} / A_{i_p,i}$ ▷ *moltiplicatore*
    per $j = i+1, \dots, n$ **fai**
      $A_{r_p,j} \leftarrow A_{r_p,j} - \beta\, A_{i_p,j}$ ▷ *complemento di Schur*
    $A_{r_p,i} \leftarrow \beta$ ▷ *L sotto la diagonale*
restituisci $\boldsymbol{A}$, $\boldsymbol{p}$

Osservazione: Pivot Numerico

Idealmente, se l'elemento pivotale è non nullo, non sarebbe necessario eseguire uno scambio di righe. Tuttavia, se l'elemento pivotale è molto piccolo in valore assoluto, l'algoritmo di fattorizzazione può produrre complementi di Schur[5] molto grandi, il che potrebbe compro-

mettere la stabilità numerica dell'algoritmo.
Per affrontare questo problema, nella pratica si adotta una strategia di controllo: si sceglie l'elemento pivotale con il valore assoluto maggiore nella colonna — noto come *pivot parziale* — o nella sottomatrice — noto come *pivot totale* — composta dagli elementi che devono ancora essere fattorizzati, quando l'elemento pivotale corrente è al di sotto di una soglia prestabilita.
Questa tecnica, chiamata *pivot numerico*, è progettata per garantire la stabilità numerica dell'algoritmo di fattorizzazione, prevenendo problemi che potrebbero derivare da piccoli valori degli elementi pivotali.

Osservazione: Pivot Simbolico

Oltre alla stabilità numerica, c'è un'altra considerazione importante nella scelta del pivot: la preservazione della sparsità della matrice.
Nella pratica, spesso si lavora con matrici *sparse*, cioè matrici in cui solo un numero limitato di elementi è diverso da zero, tipicamente dell'ordine della dimensione della matrice piuttosto che del suo quadrato. È cruciale mantenere questa sparsità durante il processo di fattorizzazione.
Il meccanismo del complemento di Schur tende infatti a generare molti elementi non nulli, il che può portare a una crescita esponenziale nel numero di elementi non nulli e, conseguentemente, a un eccessivo consumo di memoria per memorizzare le matrici $\boldsymbol{L}$ e $\boldsymbol{U}$ risultanti.
Per controllare la crescita di elementi non nulli e preservare la sparsità, è utile eseguire scambi di righe, che corrispondono a scambi delle equazioni nel sistema. Questo approccio può aiutare a mantenere la matrice più sparsa possibile.
Mostriamo questo effetto con il seguente esempio:

1.6.1 Esempio

Consideriamo la fattorizzazione LU della matrice $\boldsymbol{A} \in \mathbb{R}^{10\times 10}$

$$\boldsymbol{A} = \begin{pmatrix} \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \end{pmatrix}$$

e della matrice che si ottiene prendendo righe e colonne nell'ordine inverso, cioè

[5] Issai Schur (1875-1941)

$$\boldsymbol{PAP} = \begin{pmatrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & \mathbf{1} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \end{pmatrix}$$

Abbiamo considerato gli elementi non nulli di $\boldsymbol{A}$ come aventi valore 1, ma l'argomento che desideriamo illustrare riguarda esclusivamente il processo di formazione di nuovi elementi non nulli durante la fattorizzazione, noto con il termine inglese *fill-in*. Questo è distinto dal valore che tali elementi assumono nei fattori di Gauss $\boldsymbol{L}$ e $\boldsymbol{U}$.

In seguito, analizzeremo la struttura degli elementi non nulli nella matrice $\boldsymbol{A}$, nella matrice permutata simmetricamente $\boldsymbol{PAP}$, e nei rispettivi fattori triangolari superiori e inferiori ottenuti dall'algoritmo di Gauss, assumendo che non sia necessario eseguire pivoting numerico.

È importante notare che nella forma originale della matrice $\boldsymbol{A}$, i fattori risultanti dall'algoritmo di Gauss sono *pieni*, il che implica un riempimento totale.

Invece, nella forma permutata, non si generano nuovi elementi non nulli. Inoltre, poiché la diagonale di $\boldsymbol{L}$ è composta esclusivamente da 1, non è necessario memorizzarla. Di conseguenza, non viene richiesta ulteriore memoria rispetto a quella necessaria per memorizzare gli elementi non nulli della matrice.

In sintesi

- L'eliminazione di Gauss trasforma $\boldsymbol{Ax} = \boldsymbol{b}$ in un sistema triangolare con $\sim 2n^3/3$ flop; la risoluzione all'indietro costa poi solo $\mathcal{O}(n^2)$.
- In forma matriciale ogni passo è la moltiplicazione per una **matrice di Frobenius** $\boldsymbol{F}_k$, la cui inversa si ottiene cambiando di segno i moltiplicatori: da qui $\boldsymbol{A} = \boldsymbol{LU}$ a costo nullo.
- La fattorizzazione LU va calcolata **una volta sola**: sistemi con la stessa matrice e termini noti diversi costano $\mathcal{O}(n^2)$ ciascuno.
- Il **pivoting parziale** serve alla stabilità, non all'esistenza: scambia le righe per massimizzare il pivot e dà $\boldsymbol{PA} = \boldsymbol{LU}$.
- Pivot *simbolicamente* nullo e pivot *numericamente* piccolo sono problemi diversi: il primo riguarda la singolarità, il secondo l'amplificazione degli errori di arrotondamento.

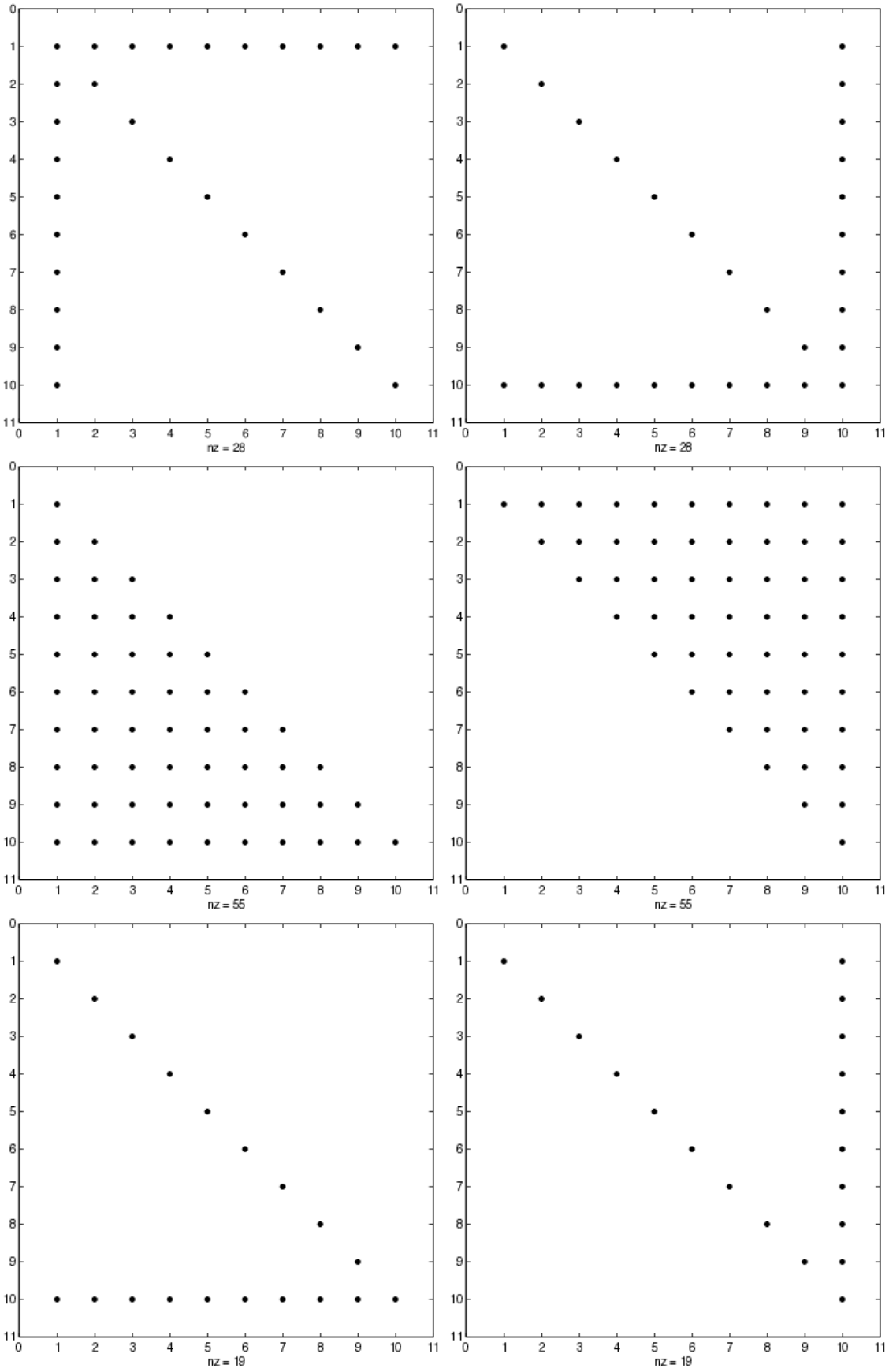


Figure 1.3: In alto mostriamo la struttura dei non zeri di $\boldsymbol{A}$ e di $\boldsymbol{PAP}$, in mezzo quella dei fattori di $\boldsymbol{A}$ ed in basso quella dei fattori di $\boldsymbol{PAP}$.



2. Fattorizzazione di Cholesky

Sia $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ una matrice non singolare e $\boldsymbol{b} \in \mathbb{R}^n$. Vogliamo determinare $\boldsymbol{x} \in \mathbb{R}^n$ tale che

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}.$$

Sappiamo che è possibile risolvere il problema tramite la fattorizzazione di Gauss[1] $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$, dove, per semplicità di esposizione, assumiamo che non sia necessario il pivoting. Questo approccio richiede di risolvere due sistemi lineari con matrici triangolari, uno inferiore ($\boldsymbol{L}$) e uno superiore ($\boldsymbol{U}$):

$$\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}, \qquad \boldsymbol{U}\boldsymbol{x} = \boldsymbol{y}.$$

Il costo computazionale di questa procedura non risiede soltanto nella soluzione dei due sistemi, ma anche nel processo di calcolo dei fattori $\boldsymbol{L}$ e $\boldsymbol{U}$. Inoltre, il calcolatore deve disporre di sufficiente memoria per memorizzare entrambi i fattori. Questo non rappresenta un problema per matrici *piene*, poiché la memoria già occupata dalla matrice originale può essere riutilizzata. Tuttavia, per matrici *sparse*, il processo di *fill-in* (creazione di nuovi elementi non nulli durante la fattorizzazione) può incrementare notevolmente i requisiti di memoria, rendendo difficile la memorizzazione dei due fattori.

Per queste ragioni, è utile considerare situazioni in cui è possibile una fattorizzazione alternativa del tipo:

$$\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R},$$

dove $\boldsymbol{R}$ è una matrice triangolare superiore, detta *fattorizzazione di Cholesky*[2].

Poiché questa fattorizzazione richiede il calcolo di un solo fattore anziché due, possiamo aspettarci una riduzione sia del costo computazionale sia dell'occupazione di memoria. Sebbene il guadagno non sia necessariamente dimezzato, si ottiene comunque un notevole risparmio[3].

Il seguente teorema descrive le matrici per le quali la fattorizzazione di Cholesky è possibile.

Theorem 2.1 — Fattorizzazione di Cholesky e matrici SPD. Le matrici quadrate non-singolari $\boldsymbol{A}$ per cui è possibile una fattorizzazione della forma $\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}$ con $\boldsymbol{R}$ matrice triangolare superiore sono *tutte e sole* le matrici *simmetriche e definite positive*, (*SPD*).

[1] Carl Friedrich Gauss (1777-1855)
[2] André-Louis Cholesky (1875-1918)
[3] In sintesi, il risparmio è considerevole!

Proof. Mostriamo innanzitutto la *necessità*, cioè che la fattorizzazione di Cholesky è possibile solo per matrici simmetriche e definite positive. Mostreremo in seguito la *sufficienza*, cioè che data una matrice che ha tali proprietà è sempre possibile calcolare il suo *fattore di Cholesky* $\boldsymbol{R}$.

- **Necessità**
  Sia $\boldsymbol{A}$, $\boldsymbol{R} \in \mathbb{R}^{n\times n}$ con $\boldsymbol{A}$ non singolare ed $\boldsymbol{R}$ triangolare superiore tale che valga la decomposizione $\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}$.
  - **Simmetria**
    $\boldsymbol{A}^T = (\boldsymbol{R}^T\boldsymbol{R})^T = \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{A}$.
  - **Positività**
    Se $\boldsymbol{A}$ è non-singolare, allora anche $\boldsymbol{R}$ è non-singolare perché dal teorema di Binet segue
    $\left|\boldsymbol{A}\right| = \left|\boldsymbol{R}^T\boldsymbol{R}\right| = \left|\boldsymbol{R}^T\right|\left|\boldsymbol{R}\right| = \left|\boldsymbol{R}\right|^2$.
    Inoltre, se $\boldsymbol{R}$ è non-singolare, si ha $\boldsymbol{R}\boldsymbol{x} \neq \boldsymbol{0}$ per ogni vettore $\boldsymbol{x} \neq \boldsymbol{0}$. Quindi,
    $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{x} = \|\boldsymbol{R}\boldsymbol{x}\|_2^2 > 0$
- **Sufficienza**
  La dimostrazione procede per induzione sull'ordine della matrice, ma è anche in parte costruttiva. Illustreremo un algoritmo di riduzione, che permette di ricondurre il problema della fattorizzazione di una matrice *SPD* di ordine n a quello di una matrice *SPD* di ordine $n-1$. Per induzione, si troverà che tutte le matrici *SPD* sono fattorizzabili.
  Per facilitare la costruzione dell'algoritmo di riduzione, riscriviamo la matrice simmetrica $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ in una forma equivalente a blocchi

$$\boldsymbol{A} = \left(\begin{array}{c|ccc} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ \hline A_{2,1} & A_{2,2} & \cdots & A_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n,1} & A_{n,2} & \cdots & A_{n,n} \end{array}\right) \begin{pmatrix} \alpha & \boldsymbol{a}^T \\ \boldsymbol{a} & \tilde{\boldsymbol{A}} \end{pmatrix}$$

  introducendo la sottomatrice
  $\tilde{\boldsymbol{A}} \in \mathbb{R}^{(n-1)\times(n-1)}$, il vettore $\boldsymbol{a} \in \mathbb{R}^{n-1}$ e lo scalare $\alpha \in \mathbb{R}^n$

$$\tilde{\boldsymbol{A}} = \begin{pmatrix} A_{2,2} & \cdots & A_{2,n} \\ \vdots & \ddots & \vdots \\ A_{n,2} & \cdots & A_{n,n} \end{pmatrix}, \qquad \boldsymbol{a} = \begin{pmatrix} A_{2,1} \\ \vdots \\ A_{n,1} \end{pmatrix}, \qquad \alpha = A_{1,1}.$$

  Scriviamo la matrice $\boldsymbol{R}$ a blocchi in modo simile ad $\boldsymbol{A}$

$$\boldsymbol{R} = \left(\begin{array}{c|c} \rho & \boldsymbol{r}^T \\ \hline \boldsymbol{0} & \boldsymbol{R}' \end{array}\right)$$

  e sviluppiamo il prodotto $\boldsymbol{R}^T\boldsymbol{R}$

$$\boldsymbol{R}^T\boldsymbol{R} = \begin{pmatrix} \rho & \boldsymbol{0}^T \\ \boldsymbol{r} & (\boldsymbol{R}')^T \end{pmatrix} \begin{pmatrix} \rho & \boldsymbol{r}^T \\ \boldsymbol{0} & \boldsymbol{R}' \end{pmatrix} = \begin{pmatrix} \rho^2 & \rho\boldsymbol{r}^T \\ \rho\boldsymbol{r} & \boldsymbol{r}\boldsymbol{r}^T + (\boldsymbol{R}')^T\boldsymbol{R}' \end{pmatrix}.$$

  La fattorizzazione $\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}$ si può esprimere uguagliando i blocchi nelle posizioni corrispondenti di $\boldsymbol{R}^T\boldsymbol{R}$ ed $\boldsymbol{A}$

$$\begin{pmatrix} \alpha & \boldsymbol{a}^T \\ \boldsymbol{a} & \tilde{\boldsymbol{A}} \end{pmatrix} = \begin{pmatrix} \rho^2 & \rho\boldsymbol{r}^T \\ \rho\boldsymbol{r} & \boldsymbol{r}\boldsymbol{r}^T + (\boldsymbol{R}')^T\boldsymbol{R}' \end{pmatrix},$$

  per cui si ottengono immediatamente le relazioni seguenti

$$\alpha = \rho^2, \qquad \boldsymbol{a} = \rho\boldsymbol{r},$$

o equivalentemente

$$\boldsymbol{a}^T = \rho \boldsymbol{r}^T, \qquad \tilde{\boldsymbol{A}} = (\boldsymbol{R}')^T \boldsymbol{R}' + \boldsymbol{r}\boldsymbol{r}^T. \tag{2.1}$$

Le prime due relazioni forniscono una espressione diretta per determinare ρ ed $\boldsymbol{r}$, cioè

$$\rho = \sqrt{\alpha}, \qquad \text{e} \qquad \boldsymbol{r} = \frac{1}{\sqrt{\alpha}}\boldsymbol{a},$$

dove supponiamo per il momento[4] che α sia un numero reale strettamente positivo. L'ultima relazione in 2.1, riscritta come

$$(\boldsymbol{R}')^T \boldsymbol{R}' = \tilde{\boldsymbol{A}} - \frac{1}{\alpha}\boldsymbol{a}\boldsymbol{a}^T,$$

suggerisce che la matrice $\tilde{\boldsymbol{R}}$ sia a sua volta il fattore triangolare superiore – ammesso ovviamente che esista – della fattorizzazione di Cholesky della matrice *ridotta* $\boldsymbol{A}' \in \mathbb{R}^{(n-1)\times(n-1)}$

$$\boldsymbol{A}' = \tilde{\boldsymbol{A}} - \frac{1}{\alpha}\boldsymbol{a}\boldsymbol{a}^T.$$

Procedendo in questo modo potremmo scrivere il fattore triangolare superiore $\boldsymbol{R}$ di Cholesky come

$$\boldsymbol{R} = \begin{pmatrix} \sqrt{\alpha} & \frac{1}{\sqrt{\alpha}}\boldsymbol{a}^T \\ \boldsymbol{0} & \boldsymbol{R}' \end{pmatrix},$$

in cui la prima riga sarebbe completamente determinata e resterebbe da calcolare il fattore triangolare superiore $\boldsymbol{R}'$ della matrice $\boldsymbol{A}'$ di ordine $n-1$. Di fatto, avremmo ridotto di uno la dimensione del problema ed è evidente che se si potesse ripetere questo procedimento n volte, arriveremmo alla fine a una matrice di ordine 1 la cui fattorizzazione sarebbe banalmente

$$\left(A_{1,1}\right) = \left(\sqrt{A_{1,1}}\right)\left(\sqrt{A_{1,1}}\right).$$

Mostreremo che l'ipotesi che $\boldsymbol{A}$ sia *SPD* garantisce che tutto il procedimento ipotizzato è in realtà ben definito. Innanzitutto, osserviamo che valgono le seguenti proprietà.

- (a) **Se $\boldsymbol{A}$ è SPD, allora** $\alpha > 0$
  Il risultato si ottiene applicando la definizione di matrice definita positiva ad $\boldsymbol{e}_1$, primo vettore della base canonica:
  $\alpha = A_{1,1} = \boldsymbol{e}_1^T \boldsymbol{A}\boldsymbol{e}_1 > 0$.
- (b) **Se $\boldsymbol{A}$ è SPD, allora $\boldsymbol{A}'$ è SPD**
  La simmetria di $\boldsymbol{A}'$ è evidente dalla sua definizione. Vogliamo mostrare inoltre che per ogni vettore $\boldsymbol{y} \in \mathbb{R}^{n-1}$ non nullo si ha

  $$\boldsymbol{y}^T \boldsymbol{A}' \boldsymbol{y} > 0 \qquad \text{cioè} \qquad \boldsymbol{y}^T \left(\tilde{\boldsymbol{A}} - \alpha^{-1}\boldsymbol{a}^T\boldsymbol{a}\right)\boldsymbol{y} > 0,$$

  nell'ipotesi che $\boldsymbol{A}$ sia definita positiva. Dato un vettore generico $\boldsymbol{y} \in \mathbb{R}^{n-1}$, introduciamo un vettore ausiliario $\boldsymbol{z} = \begin{pmatrix} \eta & \boldsymbol{y}^T \end{pmatrix}^T \in \mathbb{R}^n$, dove η è a sua volta un qualsiasi numero reale non nullo. La positività di $\boldsymbol{A}$ implica che $\boldsymbol{z}^T\boldsymbol{A}\boldsymbol{z} > 0$, cioè

  $$\begin{pmatrix} \eta & \boldsymbol{y}^T \end{pmatrix} \begin{pmatrix} \alpha & \boldsymbol{a}^T \\ \boldsymbol{a} & \boldsymbol{A}' \end{pmatrix} \begin{pmatrix} \eta \\ \boldsymbol{y} \end{pmatrix} = \alpha\,\eta^2 + 2\boldsymbol{a}^T\boldsymbol{y}\,\eta + \boldsymbol{y}^T\tilde{\boldsymbol{A}}\boldsymbol{y} > 0.$$

[4] Se α fosse strettamente negativo, il ragionamento che segue avrebbe ancora senso ammettendo di calcolare la radice quadrata in campo complesso. Questo tuttavia non produrrebbe una algoritmo numerico efficiente. Se α fosse nullo, il ragionamento che segue non sarebbe possibile. Il problema, in realtà, non si pone, perché dimostreremo tra un attimo che l'ipotesi *SPD* assicura la positività stretta di questo scalare.

La positività di questo polinomio di secondo grado in η implica che il discriminante sia negativo (ovvero non ci sono radici reali), per cui sicuramente vale

$$\Delta = \left(\boldsymbol{a}^T \boldsymbol{y}\right)^2 - \alpha \boldsymbol{y}^T \tilde{\boldsymbol{A}} \boldsymbol{y} < 0,$$

e questa è proprio la condizione che dobbiamo verificare!

Sfruttando le due proprietà (a) e (b) possiamo concludere la dimostrazione procedendo per induzione sull'ordine della matrice. Assumiamo che $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ sia *SPD*, e che si abbia

- per $n = 1$ una matrice *SPD* si fattorizza come già riportato prima, e cioè

$$\left(A_{1,1}\right) = \left(\sqrt{A_{1,1}}\right)\left(\sqrt{A_{1,1}}\right).$$

- per ogni matrice $\boldsymbol{A}'$ *SPD* di ordine $n-1$ esiste una matrice triangolare superiore $\boldsymbol{R}$ che la fattorizza nella forma di Cholesky $\boldsymbol{A}' = \boldsymbol{R}'(\boldsymbol{R}')^T$.

Allora, le proprietà (a) e (b) implicano che se $\boldsymbol{A}$ è *SPD* è possibile operare la riduzione, e dalle ipotesi induttive segue subito che la matrice è fattorizzabile in forma di Cholesky. ∎

2.1 Come si riconosce una matrice SPD: il criterio di Sylvester

Il Teorema 2.1 caratterizza le matrici fattorizzabili come le matrici *SPD*, ma la definizione di matrice definita positiva — $\boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} > 0$ per ogni $\boldsymbol{x} \neq \boldsymbol{0}$ — non è verificabile in pratica: richiederebbe di controllare infiniti vettori. Serve un criterio **finito**.

Theorem 2.2 — Criterio di Sylvester. Sia $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ simmetrica e siano

$$\boldsymbol{A}_k := \boldsymbol{A}_{1:k,1:k}, \qquad k = 1, \dots, n$$

le sue *sottomatrici principali di testa*. Allora

$$\boxed{\boldsymbol{A} \succ 0 \quad \Longleftrightarrow \quad \det(\boldsymbol{A}_k) > 0, \qquad k = 1, \dots, n}$$

cioè una matrice reale simmetrica è definita positiva se e solo se tutti i suoi **minori principali di testa** sono strettamente positivi.

Proof. **Necessità.** Sia $\boldsymbol{A}$ *SPD*. Fissato k, si prenda $\boldsymbol{w} \in \mathbb{R}^k$, $\boldsymbol{w} \neq \boldsymbol{0}$, e lo si immerga in $\mathbb{R}^n$ come

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{w} \\ \boldsymbol{0} \end{bmatrix}$$

Allora $\boldsymbol{w}^T \boldsymbol{A}_k \boldsymbol{w} = \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} > 0$, quindi ogni $\boldsymbol{A}_k$ è a sua volta *SPD*. I suoi autovalori sono dunque positivi e

$$\det(\boldsymbol{A}_k) = \prod_{i=1}^{k} \lambda_i(\boldsymbol{A}_k) > 0$$

Sufficienza. Si procede per induzione su k, mostrando che ogni $\boldsymbol{A}_k$ è definita positiva. Per $k = 1$ si ha $\boldsymbol{A}_1 = [a_{11}]$ e $\det(\boldsymbol{A}_1) = a_{11} > 0$.

Sia $\boldsymbol{A}_{k-1} \succ 0$ e si scriva a blocchi

$$\boldsymbol{A}_k = \begin{bmatrix} \boldsymbol{A}_{k-1} & \boldsymbol{b} \\ \boldsymbol{b}^T & c \end{bmatrix}$$

Poiché $\boldsymbol{A}_{k-1} \succ 0$ essa è invertibile, e si può introdurre il suo *complemento di Schur*

$$d := c - \boldsymbol{b}^T \boldsymbol{A}_{k-1}^{-1} \boldsymbol{b}$$

esattamente come nel passo generico dell'algoritmo di Cholesky. Vale la fattorizzazione

$$\boldsymbol{A}_k = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{b}^T \boldsymbol{A}_{k-1}^{-1} & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{k-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & d \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{A}_{k-1}^{-1}\boldsymbol{b} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \tag{2.2}$$

Il primo e il terzo fattore sono triangolari con diagonale unitaria, quindi hanno determinante 1 e

$$\det(\boldsymbol{A}_k) = \det(\boldsymbol{A}_{k-1})\, d \qquad \Longrightarrow \qquad d = \frac{\det(\boldsymbol{A}_k)}{\det(\boldsymbol{A}_{k-1})} > 0$$

Dalla 2.2, posto $\boldsymbol{x} = \begin{bmatrix} \boldsymbol{u} \\ \mu \end{bmatrix} \neq \boldsymbol{0}$ e $\boldsymbol{v} = \boldsymbol{u} + \mu \boldsymbol{A}_{k-1}^{-1}\boldsymbol{b}$, si ottiene

$$\boldsymbol{x}^T \boldsymbol{A}_k \boldsymbol{x} = \boldsymbol{v}^T \boldsymbol{A}_{k-1} \boldsymbol{v} + d\,\mu^2$$

somma di due termini non negativi, nulla solo se $\mu = 0$ e $\boldsymbol{v} = \boldsymbol{0}$, cioè solo se $\boldsymbol{x} = \boldsymbol{0}$. Dunque $\boldsymbol{A}_k \succ 0$, e per $k = n$ si conclude. □ ■

Due osservazioni utili

- La quantità d della dimostrazione è **lo stesso** complemento di Schur che compare al passo generico dell'algoritmo di Cholesky, e il rapporto $d = \det(\boldsymbol{A}_k)/\det(\boldsymbol{A}_{k-1})$ dice che i pivot di Cholesky sono rapporti di minori consecutivi. In particolare

$$\det(\boldsymbol{A}) = \prod_{k=1}^{n} r_{kk}^2$$

- $\boldsymbol{A}$ è **definita negativa** se e solo se $-\boldsymbol{A}$ è definita positiva, cioè se e solo se i minori di testa si alternano di segno, $(-1)^k \det(\boldsymbol{A}_k) > 0$.

Attenzione: non vale per il semidefinito

Il criterio richiede disuguaglianze **strette** e minori **di testa**. Sostituire > 0 con ≥ 0 è falso: la matrice

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}$$

ha $\det(\boldsymbol{A}_1) = 0$ e $\det(\boldsymbol{A}_2) = 0$, entrambi ≥ 0, ma non è semidefinita positiva, perché $\boldsymbol{x} = (0,1)^T$ dà $\boldsymbol{x}^T \boldsymbol{A} \boldsymbol{x} = -1$.
Per il semidefinito occorre richiedere la non negatività di **tutti** i minori principali, non solo di quelli di testa: in questo esempio il minore principale di indice $\{2\}$ vale -1 ed è quello che smaschera la matrice.

In pratica

Per n grande il criterio di Sylvester non è il modo con cui si verifica la positività: calcolare n determinanti costa più della fattorizzazione stessa. Il test operativo è **tentare la fattorizzazione di Cholesky**: se tutti i pivot r_{kk}^2 risultano positivi la matrice è *SPD*, altrimenti l'algoritmo si arresta e la matrice non lo è. Costa $\sim n^3/3$ flop e dà in omaggio la fattorizzazione. Il criterio di Sylvester resta lo strumento **teorico** e il metodo pratico per matrici piccole o con parametri simbolici.

2.2 Algoritmo di calcolo

In questo paragrafo, descriviamo il processo logico della fattorizzazione mediante il metodo di Cholesky. Analizziamo in dettaglio il primo e il secondo step, lo step generico k e

Step 1

Il teorema appena dimostrato ci permette di scrivere la matrice $\boldsymbol{A}$ con una decomposizione del tipo:

$$\begin{aligned}\boldsymbol{A} &= \begin{pmatrix} \alpha & \boldsymbol{a}^T \\ \boldsymbol{a} & \tilde{\boldsymbol{A}} \end{pmatrix} \\ &= \begin{pmatrix} \sqrt{\alpha} & 0 \\ \dfrac{1}{\sqrt{\alpha}}\boldsymbol{a} & \boldsymbol{I}_{n-1} \end{pmatrix} \begin{pmatrix} 1 & 0^T \\ 0 & \boldsymbol{A}' \end{pmatrix} \begin{pmatrix} \sqrt{\alpha} & \dfrac{1}{\sqrt{\alpha}}\boldsymbol{a}^T \\ 0 & \boldsymbol{I}_{n-1} \end{pmatrix} \\ &= \boldsymbol{R}_1^T \boldsymbol{A}_1 \boldsymbol{R}_1\end{aligned}$$

dove le matrici $\boldsymbol{A}_1$ e $\boldsymbol{R}_1$ sono definite in modo naturale. Inoltre, abbiamo introdotto la matrice *ridotta* di ordine $(n-1)$:

$$\boldsymbol{A}' = \tilde{\boldsymbol{A}} - \frac{1}{\alpha}\boldsymbol{a}\boldsymbol{a}^T,$$

che risulta essere *SPD* (definita positiva), poiché la matrice originale $\boldsymbol{A}$ è anch'essa *SPD*. La matrice $\boldsymbol{A}'$ è chiamata *complemento di Schur*[5] della matrice $\tilde{\boldsymbol{A}}$ rispetto all'elemento pivotale α.

Step 2

Poiché la matrice ridotta $\boldsymbol{A}'$ è ancora *SPD* (definita positiva), possiamo ripetere su di essa lo stesso processo del primo step. Iniziamo riscrivendo $\boldsymbol{A}'$ in forma a blocchi:

$$\boldsymbol{A}' = \begin{pmatrix} \alpha' & (\boldsymbol{a}')^T \\ \boldsymbol{a}' & \tilde{\boldsymbol{A}}' \end{pmatrix}$$

Applichiamo quindi la riduzione, decomponendo $\boldsymbol{A}'$ come segue:

$$\begin{pmatrix} \alpha' & (\boldsymbol{a}')^T \\ \boldsymbol{a}' & \tilde{\boldsymbol{A}}' \end{pmatrix} = \begin{pmatrix} \sqrt{\alpha'} & 0 \\ \dfrac{1}{\sqrt{\alpha'}}\boldsymbol{a}' & \boldsymbol{I}_{n-2} \end{pmatrix} \begin{pmatrix} 1 & 0^T \\ 0 & \boldsymbol{A}'' \end{pmatrix} \begin{pmatrix} \sqrt{\alpha'} & \dfrac{1}{\sqrt{\alpha'}}(\boldsymbol{a}')^T \\ 0 & \boldsymbol{I}_{n-2} \end{pmatrix}.$$

La matrice ridotta $\boldsymbol{A}'' \in \mathbb{R}^{(n-2)\times(n-2)}$ è data da:

$$\boldsymbol{A}'' = \tilde{\boldsymbol{A}}' - \frac{1}{\alpha'}\boldsymbol{a}'(\boldsymbol{a}')^T,$$

che rappresenta il *complemento di Schur* di $\tilde{\boldsymbol{A}}'$ rispetto al pivot α'. Dato che $\boldsymbol{A}'$ è *SPD*, anche $\boldsymbol{A}''$ sarà *SPD*.

A questo punto abbiamo decomposto la matrice $\boldsymbol{A}_1$, ottenuta nel primo step, come segue:

$$\begin{aligned}\boldsymbol{A}_1 &= \begin{pmatrix} 1 & 0^T \\ 0 & \begin{pmatrix} \alpha' & (\boldsymbol{a}')^T \\ \boldsymbol{a}' & \tilde{\boldsymbol{A}}' \end{pmatrix} \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0^T \\ 0 & \begin{pmatrix} \sqrt{\alpha'} & 0 \\ \dfrac{1}{\sqrt{\alpha'}}\boldsymbol{a}' & \boldsymbol{I}_{n-2} \end{pmatrix} \end{pmatrix} \begin{pmatrix} 1 & 0^T \\ 0 & \begin{pmatrix} 1 & 0^T \\ 0 & \boldsymbol{A}'' \end{pmatrix} \end{pmatrix} \begin{pmatrix} 1 & 0^T \\ 0 & \begin{pmatrix} \sqrt{\alpha'} & \dfrac{1}{\sqrt{\alpha'}}(\boldsymbol{a}')^T \\ 0 & \boldsymbol{I}_{n-2} \end{pmatrix} \end{pmatrix} \\ &= \boldsymbol{R}_2^T \boldsymbol{A}_2 \boldsymbol{R}_2,\end{aligned}$$

[5] Issai Schur (1875-1941)

dove $\boldsymbol{R}_2$ e $\boldsymbol{A}_2$ sono definite in modo naturale. Dopo questi due step, abbiamo ottenuto la seguente decomposizione per la matrice $\boldsymbol{A}$:

$$\boldsymbol{A} = \boldsymbol{R}_1^T \boldsymbol{A}_1 \boldsymbol{R}_1 = \boldsymbol{R}_1^T \boldsymbol{R}_2^T \boldsymbol{A}_2 \boldsymbol{R}_2 \boldsymbol{R}_1.$$

Step k generico

Dopo $k-1$ step, abbiamo decomposto la matrice $\boldsymbol{A}$ nel prodotto dei fattori:

$$\boldsymbol{A} = \boldsymbol{R}_1^T \boldsymbol{R}_2^T \dots \boldsymbol{R}_{k-1}^T \boldsymbol{A}_{k-1} \boldsymbol{R}_{k-1} \dots \boldsymbol{R}_2 \boldsymbol{R}_1,$$

dove

$$\boldsymbol{A}_{k-1} = \begin{pmatrix} \boldsymbol{I}_{k-1} & \boldsymbol{0}^T \\ \boldsymbol{0} & \boldsymbol{A}^{(k-1)} \end{pmatrix}.$$

Dal teorema generale, si deduce che il blocco di matrice $\boldsymbol{A}^{(k-1)} \in \mathbb{R}^{(n-k+1)\times(n-k+1)}$ è dato da:

$$\boldsymbol{A}^{(k-1)} = \begin{pmatrix} \alpha^{(k-1)} & (\boldsymbol{a}^{(k-1)})^T \\ \boldsymbol{a}^{(k-1)} & \tilde{\boldsymbol{A}}^{(k-1)} \end{pmatrix},$$

ed è una matrice *SPD*, il che consente di continuare con la decomposizione.

Procediamo ora a decomporre a blocchi la matrice $\boldsymbol{A}_{k-1}$:

$$\begin{aligned}
\boldsymbol{A}_{k-1} &= \begin{pmatrix} \boldsymbol{I}_{k-1} & \boldsymbol{0}^T \\ \boldsymbol{0} & \begin{pmatrix} \alpha^{(k-1)} & (\boldsymbol{a}^{(k-1)})^T \\ \boldsymbol{a}^{(k-1)} & \tilde{\boldsymbol{A}}^{(k-1)} \end{pmatrix} \end{pmatrix} \\
&= \begin{pmatrix} \boldsymbol{I}_{k-1} & \boldsymbol{0}^T \\ \boldsymbol{0} & \begin{pmatrix} \sqrt{\alpha^{(k-1)}} & \boldsymbol{0} \\ \dfrac{1}{\sqrt{\alpha^{(k-1)}}} \boldsymbol{a}^{(k-1)} & \boldsymbol{I}_{n-k} \end{pmatrix} \end{pmatrix} \\
&\quad \begin{pmatrix} \boldsymbol{I}_{k-1} & \boldsymbol{0}^T \\ \boldsymbol{0} & \begin{pmatrix} 1 & \boldsymbol{0}^T \\ \boldsymbol{0} & \boldsymbol{A}^{(k)} \end{pmatrix} \end{pmatrix} \\
&\quad \begin{pmatrix} \boldsymbol{I}_{k-1} & \boldsymbol{0}^T \\ \boldsymbol{0} & \begin{pmatrix} \sqrt{\alpha^{(k-1)}} & \dfrac{1}{\sqrt{\alpha^{(k-1)}}} (\boldsymbol{a}^{(k-1)})^T \\ \boldsymbol{0} & \boldsymbol{I}_{n-k} \end{pmatrix} \end{pmatrix}
\end{aligned}$$

Dopo k step, abbiamo quindi decomposto la matrice $\boldsymbol{A}$ nella forma seguente:

$$\boldsymbol{A} = \boldsymbol{R}_1^T \boldsymbol{R}_2^T \dots \boldsymbol{R}_k^T \boldsymbol{A}_k \boldsymbol{R}_k \dots \boldsymbol{R}_2 \boldsymbol{R}_1.$$

Step n

Procedendo in questo modo, è chiaro che dopo n passi otterremo la seguente decomposizione della matrice $\boldsymbol{A}$ originale:

$$\boldsymbol{A} = \boldsymbol{R}_1^T \boldsymbol{R}_2^T \dots \boldsymbol{R}_n^T \boldsymbol{I}_n \boldsymbol{R}_n \boldsymbol{R}_{n-1} \dots \boldsymbol{R}_1.$$

Le matrici $\boldsymbol{R}_i^T$ per $i = 1, 2, \dots, n$ non sono solo triangolari inferiori, ma anche matrici di Frobenius[6]. In particolare, ogni matrice $\boldsymbol{R}_i^T$ è una matrice di Frobenius di indice i. Poiché il prodotto di queste matrici (e delle loro trasposte) è disposto nell'ordine corretto, esso risulta in una matrice triangolare inferiore che, nella colonna i-esima, contiene l'i-esima colonna di $\boldsymbol{R}_i^T$.

Possiamo definire la matrice $\boldsymbol{R}^T$ come:

[6] Ferdinand Georg Frobenius (1849-1917)

$$\boldsymbol{R}^T = \boldsymbol{R}_1^T \boldsymbol{R}_2^T \dots \boldsymbol{R}_n^T,$$

e concludere che, ad ogni passo dell'algoritmo, stiamo effettivamente calcolando una colonna (o una riga) del fattore di Cholesky (o del suo trasposto).

> Osservazione
>
> Si osservi che la simmetria della matrice $\boldsymbol{A}$ consente di *ottimizzare* il calcolo degli elementi dei vari complementi di Schur, poiché anch'essi risulteranno simmetrici in base al teorema generale. Nel metodo di fattorizzazione di Cholesky, quindi, si lavora esclusivamente sulla diagonale e sulla metà superiore (o inferiore) della matrice, mentre gli altri elementi sono noti per trasposizione.

2.3 Connessioni con la fattorizzazione di Gauss

> Osservazione
>
> La fattorizzazione di Cholesky rappresenta una variante *intelligente* della fattorizzazione di Gauss per matrici *SPD*. Procediamo decomponendo la matrice $\boldsymbol{A}$ a blocchi nel seguente modo:
>
> $$\begin{pmatrix} \alpha\beta & \boldsymbol{r}^T \\ \boldsymbol{c} & \boldsymbol{A}' \end{pmatrix} = \begin{pmatrix} \alpha & \boldsymbol{0} \\ \frac{1}{\beta}\boldsymbol{c} & \boldsymbol{L}' \end{pmatrix} \begin{pmatrix} \beta & \frac{1}{\alpha}\boldsymbol{r}^T \\ \boldsymbol{c} & \boldsymbol{U}' \end{pmatrix},$$
>
> dove $\alpha\beta$ rappresenta l'elemento pivotale, mentre $\boldsymbol{r}$ e $\boldsymbol{c}$ indicano la prima riga e colonna di $\boldsymbol{A}$, escluse le prime voci. Supponiamo inoltre che esistano due matrici triangolari, $\boldsymbol{L}'$ e $\boldsymbol{U}'$, tali che:
>
> $$\boldsymbol{A}' = \boldsymbol{L}'\boldsymbol{U}' + \frac{1}{\alpha\beta}\boldsymbol{c}\boldsymbol{r}^T.$$
>
> - Se $\alpha = 1$ e $\mathrm{diag}(\boldsymbol{L}') = 1$, otteniamo la fattorizzazione di Gauss (senza pivot) $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$.
> - Se $\alpha = \beta$, $\boldsymbol{L}' = \boldsymbol{U}'$, e $\boldsymbol{c} = \boldsymbol{r}$, allora si ha la fattorizzazione di Cholesky $\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}$, con $\boldsymbol{R} = \boldsymbol{U}'$.
>
> In quest'ultimo caso, la matrice deve necessariamente essere simmetrica.

In entrambi i casi, la matrice $\boldsymbol{A}''$ ottenuta durante il processo è *il complemento di Schur* del blocco di elementi corrispondenti alle righe e colonne $i, j = 2, \dots, n$ di $\boldsymbol{A}$, calcolato rispetto all'elemento pivotale.

> Osservazione
>
> Dato che il teorema generale garantisce che l'elemento pivotale è sempre strettamente positivo, si potrebbe pensare che non è necessario operare alcun pivoting. Valgono tuttavia le stesse considerazioni fatte a proposito del pivot numerico e del pivot simbolico per l'algoritmo di Gauss. Nel caso di matrici sparse è sempre consigliabile operare un pivot simbolico per controllare la formazione dei non-zeri.

In sintesi

- Le matrici per cui esiste $\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}$ sono tutte e sole le **simmetriche definite positive** (Teorema 2.1).

- Il criterio di Sylvester (Teorema 2.2) riduce la verifica di positività a n determinanti; i pivot di Cholesky sono rapporti di minori consecutivi.
- L'algoritmo è l'eliminazione di Gauss che sfrutta la simmetria: costa $\sim n^3/3$ flop, **metà** di Gauss, e non richiede pivoting.
- Il passo generico è un **complemento di Schur**, lo stesso oggetto che compare nella dimostrazione di Sylvester.
- In pratica la fattorizzazione è anche il test di positività: se si arresta, la matrice non è *SPD*.

II Fattorizzazione QR

3 Metodo QR per sistemi lineari 49
3.1 Problema dei minimi quadrati 49

4 Costruzione della fattorizzazione QR . . 57
4.1 Matrici Ortogonali 57
4.2 Rotazioni di Givens 58
4.3 Matrici di Householder 59
4.4 Fattorizzazione QR 61
4.5 Algoritmo di Householder 65

5 La pseudo inversa di Moore-Penrose . 69
5.1 Pseudo-inversa definizione 69
5.2 Proprietà della pseudo-inversa 69
5.3 Connessioni con la fattorizzazione QR 71
5.4 Pseudo inversa e fattorizzazione LQ 72



3. Metodo QR per sistemi lineari

3.1 Problema dei minimi quadrati

Consideriamo il seguente sistema lineare

$$\begin{cases} x = 0 \\ x = 1 \end{cases}$$

È evidente che questo sistema non ha soluzione. Supponiamo ora che x rappresenti una quantità ottenuta tramite misurazioni soggette a errore, e che $x = 0$ e $x = 1$ siano i valori di due misure. Possiamo quindi riformulare il problema come

$$\begin{cases} x = 0 + \epsilon_1 \\ x = 1 + \epsilon_2 \end{cases}$$

dove ϵ_1 ed ϵ_2 rappresentano gli errori di misura. Poiché tali errori non sono noti, possiamo distribuire l'errore tra le due misure, ottenendo

$$x = \frac{1}{2} \quad \Longrightarrow \quad \epsilon_1 = \frac{1}{2}, \quad \epsilon_1 = -\frac{1}{2}$$

se però abbiamo tre misure

$$\left.\begin{cases} x = 0 + \epsilon_1 \\ x = 1 + \epsilon_2 \\ x = 1 + \epsilon_3 \end{cases}\right\} \text{ come distribuisco gli errori?}$$

alcune possibilità

- minimizzo $|\epsilon_1| + |\epsilon_2| + |\epsilon_3|$, ma la funzione valore assoluto non è differenziabile sullo 0.
- minimizzo $\epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2$, questa funzione è differenziabile.
- posso migliorare se minimizzo $w_1\epsilon_1^2 + w_2\epsilon_2^2 + w_3\epsilon_3^2$, dove w_k sono dei pesi positivi che stanno a indicare l'affidabilità della misura. Se w_k è grande la misura è affidabile se w_k è piccolo lo è meno.

Consideriamo quindi la funzione

$$f(\epsilon_1, \epsilon_2, \epsilon_3) = \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2$$

dove

$$\begin{cases} \epsilon_1 = x \\ \epsilon_2 = x - 1 \\ \epsilon_3 = x - 1 \end{cases}$$

e quindi

$$f(\epsilon_1, \epsilon_2, \epsilon_3) = \epsilon_1^2 + \epsilon_2^2 + \epsilon_3^2 = x^2 + (x-1)^2 + (x-1)^2 = g(x)$$

cercando il minimo

$$g'(x) = 2x + 4(x-1) = 6x - 4 = 0, \quad \Longrightarrow \quad x = \frac{2}{3}$$

Generalizzando a n misure:

$$\begin{cases} x = y_1 + \epsilon_1 \\ x = y_2 + \epsilon_2 \\ x = y_3 + \epsilon_3 \\ \quad \vdots \\ x = y_n + \epsilon_n \end{cases}$$

per cui

$$f(\epsilon) = \sum_{k=1}^{n} \epsilon_k^2 = \sum_{k=1}^{n} (x - y_k)^2 = g(x)$$

e cercando il minimo

$$g'(x) = 2 \sum_{k=1}^{n} (x - y_k) = 0, \quad \Longrightarrow \quad x = \frac{1}{n} \sum_{k=1}^{n} y_k$$

che è la media delle misure. Soluzione molto intuitiva.
Questo approccio si può generalizzare e prende il nome di metodo dei **minimi quadrati**.

3.1.1 Regressione lineare

Consideriamo una serie di punti (x_k, y_k) e supponiamo di sapere che questi punti si distribuiscono su una retta

$$R(x) = A + Bx$$

ovvero $y_k = R(x_k)$ con A e B da determinare. Questa si chiama **retta di regressione lineare**. Nel caso di dati perfetti basterebbero due punti ma i dati sono affetti da errore e quindi

$$\begin{cases} A + Bx_0 = y_0 + \epsilon_0 \\ A + Bx_1 = y_1 + \epsilon_1 \\ A + Bx_2 = y_2 + \epsilon_2 \\ \qquad \vdots \\ A + Bx_n = y_n + \epsilon_n \end{cases}$$

per cui la somma dei quadrati degli errori diventa

$$\sum_{k=0}^{n} \epsilon_k^2 = \sum_{k=0}^{n} (A + Bx_k - y_k)^2 = f(A, B)$$

i punti di minimo soddisfano $\nabla f(A, B) = 0$ ovvero

$$\begin{aligned}\frac{\partial f(A,B)}{\partial A} &= 2\sum_{k=0}^{n}(A + Bx_k - y_k)\\ &= 2(n+1)A + 2\left(\sum_{k=0}^{n} x_k\right)B - 2\sum_{k=0}^{n} y_k = 0\\ \frac{\partial f(A,B)}{\partial B} &= 2\sum_{k=0}^{n}(A + Bx_k - y_k)x_k\\ &= 2\left(\sum_{k=0}^{n} x_k\right)A + 2\left(\sum_{k=0}^{n} x_k^2\right)B - 2\sum_{k=0}^{n} x_k y_k = 0\end{aligned}$$

e quindi otteniamo il sistema lineare

$$\begin{pmatrix} n+1 & \sum_{k=0}^{n} x_k \\ \sum_{k=0}^{n} x_k & \sum_{k=0}^{n} x_k^2 \end{pmatrix}\begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} \sum_{k=0}^{n} y_k \\ \sum_{k=0}^{n} x_k y_k \end{pmatrix}$$

la cui soluzione produce la retta di regressione lineare cercata.
Usando la libreria di R per fare il fitting otteniamo

```r
# Generazione di dati sintetici
set.seed(123)  # Per riproducibilità
x <- seq(1, 10, by = 0.5)  # Variabile indipendente
y <- 2.5 * x + rnorm(length(x), mean = 0, sd = 3)  # Variabile dipendente con rumore

# Fitting lineare
fit <- lm(y ~ x)

# Sommario del modello
summary(fit)
```

```
Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-6.2312 -1.9646 -0.0946  1.4675  4.9818

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.7839     1.5685   0.500    0.624
x             2.4524     0.2553   9.606 2.78e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

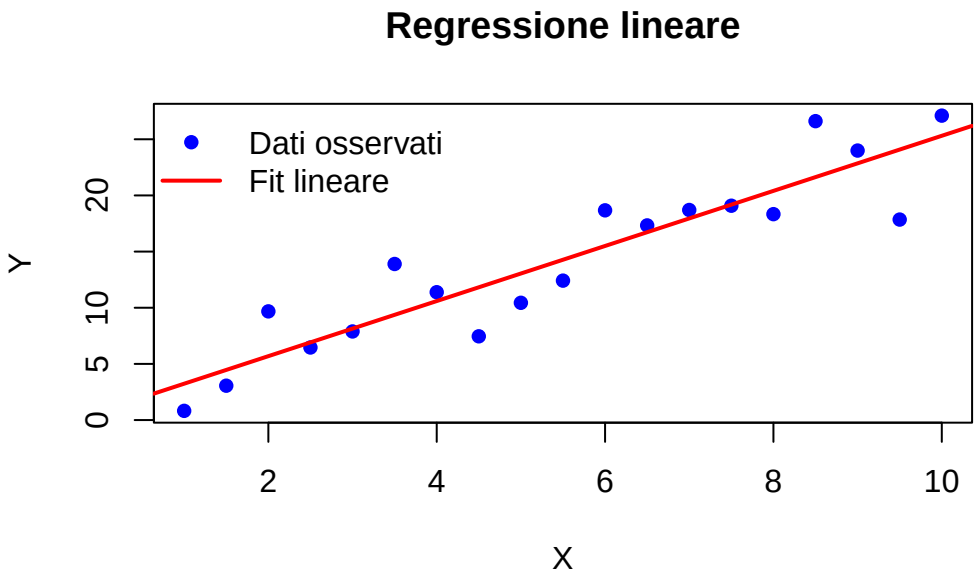
Residual standard error: 3.047 on 17 degrees of freedom
Multiple R-squared:  0.8444,    Adjusted R-squared:  0.8353
F-statistic: 92.28 on 1 and 17 DF,  p-value: 2.781e-08
```

e poi plottiamo la soluzione

```r
# Plot dei dati e della retta di regressione
plot(x, y, main = "Regressione lineare",
     xlab = "X", ylab = "Y", pch = 16, col = "blue")
```

```
abline(fit, col = "red", lwd = 2)
# Aggiunge la retta di regressione

# Aggiunta della legenda
legend("topleft", legend = c("Dati osservati", "Fit lineare"),
       col = c("blue", "red"), pch = c(16, NA), lwd = c(NA, 2), bty = "n")
```




```
cat("Coefficiente angolare (B):", coef(fit)[2], "\n")
```

```
Coefficiente angolare (B): 2.452363
```

```
cat("Intercetta (A):", coef(fit)[1], "\n")
```

```
Intercetta (A): 0.7838889
```

3.1.2 Fitting polinomiale

Consideriamo una serie di punti (x_k, y_k) e supponiamo di sapere che questi punti si distribuiscono su una parabola

$$R(x) = A + Bx + Cx^2$$

ovvero $y_k = R(x_k)$ con A, B e C da determinare. Nel caso di dati perfetti basterebbero tre punti ma i dati sono affetti da errore e quindi

$$\begin{cases} A + Bx_0 + Cx_0^2 = y_0 + \epsilon_0 \\ A + Bx_1 + Cx_1^2 = y_1 + \epsilon_1 \\ A + Bx_2 + Cx_2^2 = y_2 + \epsilon_2 \\ \vdots \\ A + Bx_n + Cx_n^2 = y_n + \epsilon_n \end{cases}$$

per cui la somma dei quadrati degli errori diventa

$$\frac{1}{2}\sum_{k=0}^{n} \epsilon_k^2 = \frac{1}{2}\sum_{k=0}^{n} (A + Bx_k + Cx_k^2 - y_k)^2 = f(A, B, C)$$

i punti di minimo soddisfano $\nabla f(A, B) = 0$ ovvero

$$\begin{aligned}
\frac{\partial f(A,B,C)}{\partial A} &= \sum (A + Bx_k + Cx_k^2 - y_k) \\
&= (n+1)A + \left(\sum x_k\right) B + \left(\sum x_k^2\right) C - \sum y_k = 0 \\
\frac{\partial f(A,B,C)}{\partial B} &= \sum (A + Bx_k + Cx_k^2 - y_k)x_k \\
&= \left(\sum x_k\right) A + \left(\sum x_k^2\right) B + \left(\sum x_k^3\right) C - \sum x_k y_k = 0 \\
\frac{\partial f(A,B,C)}{\partial C} &= \sum (A + Bx_k + Cx_k^2 - y_k)x_k \\
&= \left(\sum x_k^2\right) A + \left(\sum x_k^3\right) B + \left(\sum x_k^4\right) C - \sum x_k^2 y_k = 0
\end{aligned}$$

e quindi otteniamo il sistema lineare

$$\begin{pmatrix} n+1 & \sum x_k & \sum x_k^2 \\ \sum x_k & \sum x_k^2 & \sum x_k^3 \\ \sum x_k^2 & \sum x_k^3 & \sum x_k^4 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \end{pmatrix} = \begin{pmatrix} \sum y_k \\ \sum x_k y_k \\ \sum x_k^2 y_k \end{pmatrix}$$

la cui soluzione produce la parabola di regressione cercata.

3.1.3 Forma matriciale del problema

In forma matricale il problema dei minimi quadrati diventa

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} + \boldsymbol{\epsilon}$$

per cui il problema diventa minimizzare la norma euclidea di $\boldsymbol{\epsilon}$ ovvero

$$\text{minimize:} \qquad \sqrt{\boldsymbol{\epsilon} \cdot \boldsymbol{\epsilon}} = \|\boldsymbol{\epsilon}\|_2 = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2$$

dove

$$\boldsymbol{A} = \begin{pmatrix} A_{00} & A_{01} & \cdots & A_{0m} \\ A_{10} & A_{11} & \cdots & A_{1m} \\ A_{20} & A_{21} & \cdots & A_{2m} \\ A_{30} & A_{31} & \cdots & A_{3m} \\ \vdots & \vdots & & \vdots \\ A_{n0} & A_{11} & \cdots & A_{nm} \end{pmatrix}$$

e

$$\boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_0 \\ \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \vdots \\ \epsilon_n \end{pmatrix} \quad \boldsymbol{x} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_m \end{pmatrix} \quad \boldsymbol{b} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}$$

quindi minimizzare $\|\boldsymbol{\epsilon}\|_2$ è equivalente a minimizzare $\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2$.

Anticipiamo alcuni risultati che verranno discussi negli appunti sulla costruzione della fattorizzazione QR:

- Una matrice $\boldsymbol{Q}$ è ortogonale se la sua inversa coincide con la trasposta, ovvero $\boldsymbol{Q}^{-1} = \boldsymbol{A}^\top$
- Le matrici ortogonali applicate a un vettore ne conservano la lunghezza:

$$\|\boldsymbol{Q}\boldsymbol{v}\|_2 = \|\boldsymbol{v}\|_2$$

$$\mathbf{A} \times \mathbf{x} = \mathbf{b} + \mathbf{e}$$

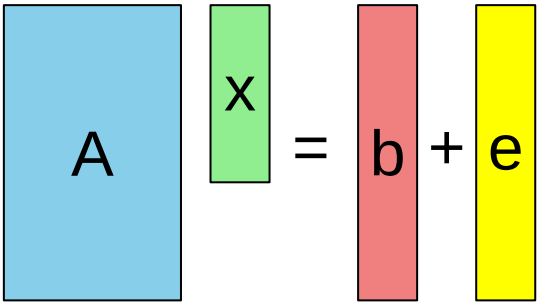


Figure 3.1: Schema del problema ai minimi quadrati

- Una matrice $\boldsymbol{A}$ quadrata, o rettangolare con più righe che colonne e di rango massimo, si può decomporre come

$$\boldsymbol{A} = \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}$$

dove $\boldsymbol{Q}$ è una matrice ortogonale e $\boldsymbol{R}$ è una matrice (quadrata) triangolare superiore.

Torniamo ora al problema di minimizzare $\|\boldsymbol{Ax} - \boldsymbol{b}\|$. Usando la decomposizione QR possiamo scrivere

$$\begin{aligned} \|\boldsymbol{Ax} - \boldsymbol{b}\|_2 &= \left\| \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{x} - \boldsymbol{b} \right\|_2 \\ &= \left\| \boldsymbol{Q} \left[\begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{x} - \boldsymbol{Q}^\top \boldsymbol{b} \right] \right\|_2 \\ &= \left\| \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{x} - \boldsymbol{Q}^\top \boldsymbol{b} \right\|_2 \end{aligned}$$

partizionando il vettore $\boldsymbol{Q}^\top \boldsymbol{b}$

$$\begin{aligned} \|\boldsymbol{Ax} - \boldsymbol{b}\|_2 &= \left\| \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix} \boldsymbol{x} - \begin{pmatrix} \boldsymbol{Q}_1^\top \\ \boldsymbol{Q}_2^\top \end{pmatrix} \boldsymbol{b} \right\|_2 \\ &= \left\| \begin{pmatrix} \boldsymbol{Rx} - \boldsymbol{Q}_1^\top \boldsymbol{b} \\ \boldsymbol{Q}_2^\top \boldsymbol{b} \end{pmatrix} \right\|_2 \end{aligned}$$

usando la definizione di $\|\cdot\|_2$ è facile verificare che per vettori partizionati vale

$$\left\| \begin{matrix} \boldsymbol{x} \\ \boldsymbol{y} \end{matrix} \right\|_2^2 = \|\boldsymbol{x}\|_2^2 + \|\boldsymbol{y}\|_2^2$$

per cui possiamo scrivere

$$\|\boldsymbol{Ax} - \boldsymbol{b}\|_2^2 = \left\| \begin{pmatrix} \boldsymbol{Rx} - \boldsymbol{Q}_1^\top \boldsymbol{b} \\ \boldsymbol{Q}_2^\top \boldsymbol{b} \end{pmatrix} \right\|_2^2 = \underbrace{\|\boldsymbol{Rx} - \boldsymbol{Q}_1^\top \boldsymbol{b}\|_2^2}_{\text{può essere messo a 0}} + \underbrace{\|\boldsymbol{Q}_2^\top \boldsymbol{b}\|_2^2}_{\text{non si pùo modificare}}$$

è chiaro quindi che il minimo si ottiene quando $\boldsymbol{Rx} - \boldsymbol{Q}_1^\top \boldsymbol{b} = \boldsymbol{0}$

Con la decomposizione QR quindi il problema dei minimi quadrati si risolve facilmente:

- Dato $\boldsymbol{Ax} = \boldsymbol{b} + \boldsymbol{\epsilon}$ il minimo dell'errore (in norma euclidea) si ottiene con i seguenti passaggi:

- Calcolo decomposizione QR di $\boldsymbol{A}$ ovvero

$$\boldsymbol{A} = \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}$$

- Moltiplico $\boldsymbol{Q}^\top$ per $\boldsymbol{b}$ e partiziono il risultato in due parti, la prima di lunghezza il numero delle colonne di $\boldsymbol{A}$:

$$\boldsymbol{Q}^\top \boldsymbol{b} = \begin{pmatrix} \boldsymbol{Q}_1^\top \boldsymbol{b} \\ \boldsymbol{Q}_2^\top \boldsymbol{b} \end{pmatrix}$$

- La norma $\|\boldsymbol{Q}_2^\top \boldsymbol{b}\|_2^2$ è il valore minimo della minimizzazione
- La soluzione del sistema lineare ridotto

$$\boldsymbol{R}\boldsymbol{x} = \boldsymbol{Q}_1^\top \boldsymbol{b}$$

è la soluzione cercata.

In sintesi

- Un sistema sovradeterminato $\boldsymbol{A}\boldsymbol{x} \simeq \boldsymbol{b}$ con $m > n$ non ha in generale soluzione: si cerca il **minimo dei quadrati** dei residui.
- Regressione lineare e fitting polinomiale sono lo stesso problema con matrici di disegno diverse: cambia $\boldsymbol{A}$, non il metodo.
- Le equazioni normali $\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}^T\boldsymbol{b}$ risolvono il problema ma **elevano al quadrato il condizionamento**: è il motivo per cui si preferisce la fattorizzazione QR.



4. Costruzione della fattorizzazione QR

4.1 Matrici Ortogonali

Per risolvere il problema dei minimi quadrati abbiamo usato la fattorizzazione QR, che si basa su matrici ortogonali. Richiamiamo qui alcune definizioni e proprietà.

Definition 4.1 Una matrice quadrata $\boldsymbol{Q}$ di ordine n è ortogonale se:

$$\boldsymbol{Q}^\top \boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{Q}^\top = \boldsymbol{I}$$

dove $\boldsymbol{I}$ è la matrice identità.

Questa definizione implica che la matrice ortogonale è invertibile e la sua inversa coincide con la sua trasposta:

$$\boldsymbol{Q}^{-1} = \boldsymbol{Q}^\top$$

4.1.1 Proprietà

Le matrici ortogonali hanno le seguenti proprietà:

1. **Preservazione delle lunghezze**: La moltiplicazione di un vettore per una matrice ortogonale non cambia la sua lunghezza:

$$\|\boldsymbol{Q}\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$$

2. **Preservazione degli angoli**: La matrice ortogonale preserva l'angolo tra due vettori $\boldsymbol{u}$ e $\boldsymbol{v}$:

$$(\boldsymbol{Q}\boldsymbol{u}) \cdot (\boldsymbol{Q}\boldsymbol{v}) = \boldsymbol{u}^\top \boldsymbol{Q}^\top \boldsymbol{Q}\boldsymbol{v} = \boldsymbol{u}^\top \boldsymbol{v} = \boldsymbol{u} \cdot \boldsymbol{v}$$

3. **Determinante**: Il determinante di una matrice ortogonale è sempre ± 1.

$$1 = |\boldsymbol{I}| = |\boldsymbol{Q}^{-1}\boldsymbol{Q}| = |\boldsymbol{Q}^\top \boldsymbol{Q}| = |\boldsymbol{Q}^\top|\,|\boldsymbol{Q}| = |\boldsymbol{Q}|^2$$

4. **Colonne e righe ortogonali**: Le colonne e le righe di una matrice ortogonale sono ortogonali tra loro e hanno norma unitaria:

$$\boldsymbol{c}_i \cdot \boldsymbol{c}_j = \delta_{ij}$$

dove δ_{ij} è il **delta di Kronecker**[1], che vale 1 quando $i = j$ e 0 quando $i \neq j$. Questa proprietà è immediata conseguenza di $\boldsymbol{Q}^\top \boldsymbol{Q} = \boldsymbol{I}$.

5. **Prodotto:** Il prodotto di matrici ortogonali è una matrice ortogonale. Infatti vale

$$\boldsymbol{M} = \boldsymbol{Q}_1 \boldsymbol{Q}_2 \cdots \boldsymbol{Q}_m$$

inoltre

$$\boldsymbol{M}\boldsymbol{M}^\top = \boldsymbol{Q}_1 \boldsymbol{Q}_2 \cdots \underbrace{\boldsymbol{Q}_m \boldsymbol{Q}_m^\top}_{=\boldsymbol{I}} \boldsymbol{Q}_{m-1}^\top \cdots \boldsymbol{Q}_1^\top = \boldsymbol{I}$$

4.2 Rotazioni di Givens

Le rotazioni di Givens[2] sono trasformazioni che agiscono su coppie di righe di una matrice per eliminare elementi specifici. Una **matrice di rotazione di Givens** $\boldsymbol{G}(i, j, \theta)$ è una matrice ortogonale di dimensione $n \times n$ che opera sulla riga i e j per annullare un elemento specifico di $\boldsymbol{A}$. In altre parole, moltiplicando $\boldsymbol{A}$ per una rotazione di Givens, possiamo rendere zero un elemento di $\boldsymbol{A}$ senza modificare altri elementi in colonna.

Prima di considerare il caso generale consideriamo la matrice di rotazione nel piano $\mathbb{R}^2$:

$$\boldsymbol{G} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

Questa matrice rappresenta una rotazione di un angolo θ nel piano. Se moltiplichiamo un vettore $\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$ per $\boldsymbol{G}$, otteniamo il vettore ruotato, ma la lunghezza del vettore rimane invariata:

$$\|\boldsymbol{G}\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$$

La matrice $\boldsymbol{G}$ è ortogonale, poiché:

$$\begin{aligned} \boldsymbol{G}^\top \boldsymbol{G} &= \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \\ &= \begin{pmatrix} (\sin\theta)^2 + (\cos\theta)^2 & 0 \\ 0 & (\sin\theta)^2 + (\cos\theta)^2 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \boldsymbol{I} \end{aligned}$$

4.2.1 Forma della Matrice di Rotazione di Givens

La matrice di rotazione di Givens $\boldsymbol{G}(i, j, \theta)$ è una matrice identità con le seguenti modifiche:

1. Gli elementi in posizione (i, i) e (j, j) sono sostituiti da $\cos\theta$.
2. Gli elementi in posizione (i, j) e (j, i) sono sostituiti rispettivamente da $\sin\theta$ e $-\sin\theta$.

Quindi $\boldsymbol{G}(i, j, \theta)$ assume la forma:

[1] Leopold Kronecker (1823-1891)
[2] James Wallace Givens Jr. (1910-1993)

$$G(i,j,\theta) = \begin{pmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & \cos\theta & \cdots & \sin\theta & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -\sin\theta & \cdots & \cos\theta & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{pmatrix}$$

In questo caso, l'elemento della riga i e della colonna j viene annullato se scegliamo il seno e il coseno dell'angolo θ in modo appropriato.

4.2.2 Calcolo di $\cos\theta$ e $\sin\theta$

Dato che vogliamo trasformare un vettore $\begin{pmatrix} a \\ b \end{pmatrix}$ in $\begin{pmatrix} r \\ 0 \end{pmatrix}$, dove $r = \sqrt{a^2 + b^2}$, definiamo $\cos\theta$ e $\sin\theta$ come segue:

$$\cos\theta = \frac{a}{\sqrt{a^2 + b^2}}, \quad \sin\theta = \frac{b}{\sqrt{a^2 + b^2}}$$

Con questa scelta, applicando la rotazione di Givens, b sarà annullato e a verrà trasformato in r.

4.3 Matrici di Householder

Le **matrici di Householder**[3] sono una classe speciale di matrici ortogonali che vengono utilizzate principalmente nella decomposizione QR di una matrice e in altre applicazioni di analisi numerica. Queste matrici sono particolarmente utili per ridurre una matrice a una forma più semplice, come una matrice triangolare superiore, senza alterarne il rango.

Definition 4.2 Una matrice di Householder è una matrice ortogonale $\boldsymbol{H}$ definita come:

$$\boldsymbol{H} = \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v}\cdot\boldsymbol{v}}$$

dove $\boldsymbol{v}$ è un vettore (detto **vettore di Householder**) e $\boldsymbol{I}$ è la matrice identità.

Questa matrice riflette i vettori rispetto al piano ortogonale con normale $\boldsymbol{v}$, cioè applicando $\boldsymbol{H}$ a un vettore, si ottiene un riflesso rispetto a un iperpiano che passa per l'origine e ortogonale a $\boldsymbol{v}$.

4.3.1 Proprietà

- La matrice $\boldsymbol{H}$ è simmetrica e coincide con la sua inversa:

$$\boldsymbol{H}^\top = \boldsymbol{H} \qquad \boldsymbol{H}\boldsymbol{H} = \boldsymbol{I}$$

- La matrice riflette il vettore $\boldsymbol{v}$ infatti

$$\boldsymbol{H}\boldsymbol{v} = \left(\boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v}\cdot\boldsymbol{v}}\right)\boldsymbol{v} = \boldsymbol{v} - 2\frac{\boldsymbol{v}\cancel{\boldsymbol{v}^\top\boldsymbol{v}}}{\cancel{\boldsymbol{v}\cdot\boldsymbol{v}}} = \boldsymbol{v} - 2\boldsymbol{v} = -\boldsymbol{v}$$

Possiamo costruire matrici di Householder particolari che sono utili nella costruzione della fattorizzazione QR.

[3] Alston Scott Householder (1904-1993)

Lemma 4.1 Dato un vettore $\boldsymbol{x}$ possiamo costruire una matrice di Householder che lo riflette nel vettore $\pm\|x\|_2 \boldsymbol{e}_k$ dove

$$\boldsymbol{e}_k = \big(0, \cdots, 0, \underbrace{1}_{k-\text{position}}, 0, \cdots, 0\big)^\top$$

con l'elemento 1 alla posizione k. La matrice è ottenuta usando il vettore $\boldsymbol{v} = \boldsymbol{x} + \text{sign}(x_k)\|\boldsymbol{x}\|_2 \boldsymbol{e}_k$.

Proof. Basta considerare il vettore

$$\boldsymbol{v} = \boldsymbol{x} + s\|x\|_2 \boldsymbol{e}_k, \qquad s = \pm 1$$

infatti

$$\begin{aligned}
(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k)\cdot(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k) &= \overbrace{\boldsymbol{x}\cdot\boldsymbol{x}}^{=\|\boldsymbol{x}\|_2^2} + 2s\|\boldsymbol{x}\|_2 x_k + \overbrace{s^2}^{=1}\|\boldsymbol{x}\|_2^2 \\
&= 2\|\boldsymbol{x}\|_2^2 + 2s\|\boldsymbol{x}\|_2 x_k \\
&= 2\|\boldsymbol{x}\|_2(\|\boldsymbol{x}\|_2 + sx_k) \\
(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k)\cdot\boldsymbol{x} &= \|\boldsymbol{x}\|_2^2 + sx_k\|\boldsymbol{x}\|_2 \\
&= \|\boldsymbol{x}\|_2(\|\boldsymbol{x}\|_2 + sx_k)
\end{aligned}$$

e quindi

$$\begin{aligned}
\boldsymbol{H}\boldsymbol{v} &= \left(\boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v}\cdot\boldsymbol{v}}\right)\boldsymbol{x} \\
&= \left(\boldsymbol{I} - 2\frac{(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k)(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k)^\top}{(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k)\cdot(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k)}\right)\boldsymbol{x} \\
&= \boldsymbol{x} - 2\frac{(\boldsymbol{x} + s\|x\|_2\boldsymbol{e}_k)\cancel{\|\boldsymbol{x}\|_2(\|\boldsymbol{x}\|_2 + sx_k)}}{\cancel{2\|\boldsymbol{x}\|_2(\|\boldsymbol{x}\|_2 + sx_k)}} \\
&= \boldsymbol{x} - \boldsymbol{x} - s\|x\|_2\boldsymbol{e}_k = -s\|x\|_2\boldsymbol{e}_k
\end{aligned}$$

il segno $s = \text{sign}(x_k)$ in modo che $sx_k = |x_k|$ e si evitano cancellazioni numeriche. ∎

4.3.2 Esempio di Matrice di Householder

Consideriamo il vettore

$$\boldsymbol{v} = \begin{pmatrix} 9 \\ 3 \end{pmatrix}, \qquad \boldsymbol{v}\cdot\boldsymbol{v} = 9^2 + 3^2 = 90$$

La matrice di Householder è:

$$\begin{aligned}
\boldsymbol{H} &= \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v}\cdot\boldsymbol{v}} \\
&= \boldsymbol{I} - 2\frac{\begin{pmatrix} 9 \\ 3 \end{pmatrix}\begin{pmatrix} 9 & 3 \end{pmatrix}}{90} \\
&= \boldsymbol{I} - \frac{1}{45}\begin{pmatrix} 81 & 27 \\ 27 & 9 \end{pmatrix} \\
&= \frac{1}{45}\begin{pmatrix} -34 & -27 \\ -27 & 36 \end{pmatrix}
\end{aligned}$$

Moltiplicando $\boldsymbol{H}$ per $\boldsymbol{v}$

$$\boldsymbol{H}\boldsymbol{v} = \frac{1}{45}\begin{pmatrix} -34 & -27 \\ -27 & 36 \end{pmatrix}\begin{pmatrix} 9 \\ 3 \end{pmatrix}$$
$$= \frac{1}{15}\begin{pmatrix} -129 \\ \\ -45 \end{pmatrix} = -\begin{pmatrix} 9 \\ 3 \end{pmatrix} = -\boldsymbol{v}$$

4.4 Fattorizzazione QR

La fattorizzazione QR rappresenta una matrice $\boldsymbol{A}$ come prodotto di due matrici: una matrice ortogonale $\boldsymbol{Q}$ e una matrice triangolare superiore $\boldsymbol{R}$. È utile in diverse applicazioni dell'algebra lineare, tra cui la risoluzione di sistemi lineari e il calcolo di autovalori e autovettori.

4.4.1 Algoritmo di Givens

Il **metodo di Givens** è ampiamente usato per calcolare la fattorizzazione QR. Esso applica una successione di **rotazioni di Givens** per annullare gli elementi sotto la diagonale principale di $\boldsymbol{A}$ e trasformarla gradualmente nella matrice triangolare superiore $\boldsymbol{R}$.

L'algoritmo per ottenere la fattorizzazione QR di una matrice $\boldsymbol{A}$ con il metodo di Givens è il seguente:

1. Si parte da $\boldsymbol{A}$ e si applica una serie di rotazioni di Givens $\boldsymbol{G}_k$, ognuna delle quali annulla un elemento sotto la diagonale.
2. La matrice risultante è triangolare superiore e coincide con $\boldsymbol{R}$.
3. La matrice ortogonale $\boldsymbol{Q}$ sarà data dal prodotto delle trasposte delle matrici di Givens, cioè $\boldsymbol{Q} = \boldsymbol{G}_1^\top \boldsymbol{G}_2^\top \cdots \boldsymbol{G}_m^\top$.

▪ **Example 4.1 — Fattorizzazione QR con il Metodo di Givens.** Consideriamo la matrice $\boldsymbol{A}$ di dimensione 3×2:

$$\boldsymbol{A} = \begin{pmatrix} 4 & -2 & 2 \\ 3 & 6 & 2 \\ 2 & 2 & 3 \end{pmatrix}$$

L'obiettivo è decomporre questa matrice in una matrice ortogonale $\boldsymbol{Q}$ e una matrice triangolare superiore $\boldsymbol{R}$ tali che:

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$$

- **Passo 1:** Applicazione della prima rotazione di Givens per annullare l'elemento A_{21}
  La rotazione di Givens è una matrice $\boldsymbol{G}_1$ che agisce sulla prima e seconda riga di $\boldsymbol{A}$. La rotazione annullerà l'elemento A_{21}.
  1. Calcoliamo l'angolo θ utilizzando la formula per ottenere i coseni e seni per la rotazione. Vogliamo trasformare il vettore $(4, 3)^\top$ in un vettore con il secondo elemento uguale a zero. Pertanto, calcoliamo il raggio r come:

$$r = \sqrt{A_{11}^2 + A_{21}^2} = \sqrt{4^2 + 3^2} = \sqrt{16 + 9} = 5$$

  2. Ora calcoliamo il coseno e il seno dell'angolo θ:

$$\cos\theta = \frac{A_{11}}{r} = \frac{4}{5}, \quad \sin\theta = \frac{A_{21}}{r} = \frac{3}{5}$$

  La matrice di rotazione di Givens è quindi:

$$\boldsymbol{G}_1 = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} = \frac{1}{5}\begin{pmatrix} 4 & 3 & 0 \\ -3 & 4 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

3. Applichiamo la rotazione $\boldsymbol{G}_1$ alla matrice $\boldsymbol{A}$:

$$\begin{aligned} \boldsymbol{A}^{(1)} = \boldsymbol{G}_1\boldsymbol{A} &= \frac{1}{5}\begin{pmatrix} 4 & 3 & 0 \\ -3 & 4 & 0 \\ 0 & 0 & 5 \end{pmatrix}\begin{pmatrix} 4 & -2 & 2 \\ 3 & 6 & 2 \\ 2 & 2 & 3 \end{pmatrix} \\ &= \frac{1}{5}\begin{pmatrix} 25 & 10 & 14 \\ 0 & 30 & 2 \\ 10 & 10 & 15 \end{pmatrix} \end{aligned}$$

- **Passo 2:** Applicazione della seconda rotazione di Givens per annullare l'elemento $A_{31}^{(1)}$
  La rotazione di Givens è una matrice $\boldsymbol{G}_2$ che agisce sulla prima e terza riga di $\boldsymbol{A}^{(1)}$. La rotazione annullerà l'elemento $A_{31}^{(1)}$.
  1. Calcoliamo l'angolo θ utilizzando la formula per ottenere i coseni e seni per la rotazione. Vogliamo trasformare il vettore $(5, 2)^\top$ in un vettore con il secondo elemento uguale a zero. Pertanto, calcoliamo il raggio r come:

$$r = \sqrt{A_{11}^2 + A_{31}^2} = \sqrt{5^2 + 2^2} = \sqrt{25 + 4} = \sqrt{29}$$

  2. Ora calcoliamo il coseno e il seno dell'angolo θ:

$$\cos\theta = \frac{A_{11}^{(1)}}{r} = \frac{5}{\sqrt{29}}, \quad \sin\theta = \frac{A_{31}^{(1)}}{r} = \frac{2}{\sqrt{29}}$$

     La matrice di rotazione di Givens $\boldsymbol{G}_2$ per annullare l'elemento $A_{31}^{(1)}$ è quindi:

$$\boldsymbol{G}_2 = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} = \frac{1}{\sqrt{29}}\begin{pmatrix} 5 & 0 & 2 \\ 0 & \sqrt{29} & 0 \\ -2 & 0 & 5 \end{pmatrix}$$

  3. Applichiamo la rotazione $\boldsymbol{G}_2$ alla matrice $\boldsymbol{A}^{(1)}$.

$$\begin{aligned} \boldsymbol{A}^{(2)} = \boldsymbol{G}_2\boldsymbol{A}^{(1)} &= \frac{1}{\sqrt{29}}\begin{pmatrix} 5 & 0 & 2 \\ 0 & \sqrt{29} & 0 \\ -2 & 0 & 5 \end{pmatrix}\begin{pmatrix} 5 & 2 & 14/5 \\ 0 & 6 & 2/5 \\ 2 & 2 & 3 \end{pmatrix} \\ &= \frac{1}{\sqrt{29}}\begin{pmatrix} 29 & 14 & 20 \\ 0 & 6\sqrt{29} & 2\sqrt{29}/5 \\ 0 & 6 & 47/5 \end{pmatrix} \end{aligned}$$

- **Passo 3:** Applicazione della terza rotazione di Givens per annullare l'elemento $A_{32}^{(2)}$
  La rotazione di Givens è una matrice $\boldsymbol{G}_3$ che agisce sulla prima e terza riga di $\boldsymbol{A}^{(3)}$. La rotazione annullerà l'elemento $A_{32}^{(3)}$.
  1. Calcoliamo l'angolo θ utilizzando la formula per ottenere i coseni e seni per la rotazione. Vogliamo trasformare il vettore $(6, 6/\sqrt{29})^\top$ in un vettore con il secondo elemento uguale a zero. Pertanto, calcoliamo il raggio r come:

$$r = \sqrt{A_{22}^2 + A_{32}^2} = \sqrt{6^2 + 6^2/29} = 6\sqrt{30/29}$$

  2. Ora calcoliamo il coseno e il seno dell'angolo θ:

$$\cos\theta = \frac{A_{22}^{(2)}}{r} = \frac{\sqrt{29}}{\sqrt{30}}, \quad \sin\theta = \frac{A_{32}^{(2)}}{r} = \frac{1}{\sqrt{30}}$$

La matrice di rotazione di Givens $\boldsymbol{G}_3$ per annullare l'elemento $A_{32}^{(2)}$ è quindi:

$$\boldsymbol{G}_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix} = \frac{1}{\sqrt{30}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{29} & 1 \\ 0 & -1 & \sqrt{29} \end{pmatrix}$$

3. Applichiamo la rotazione $\boldsymbol{G}_3$ alla matrice $\boldsymbol{A}^{(2)}$:

$$\begin{aligned} \boldsymbol{A}^{(3)} = \boldsymbol{G}_3 \boldsymbol{A}^{(2)} &= \frac{1}{\sqrt{30}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{29} & 1 \\ 0 & -1 & \sqrt{29} \end{pmatrix} \times \frac{1}{\sqrt{29}} \begin{pmatrix} 29 & 14 & 20 \\ 0 & 6\sqrt{29} & 2\sqrt{29}/5 \\ 0 & 6 & 47/5 \end{pmatrix} \\ &= \frac{1}{\sqrt{30 \cdot 29}} \begin{pmatrix} 29 & 14 & 20 \\ 0 & 174 & 58/5 \\ 0 & 0 & 9\sqrt{29} \end{pmatrix} \end{aligned}$$

▪

Le rotazioni di Given non sono limitate a matrici quadrate ma si possono applicare a matrici rettangolari.

▪ **Example 4.2 — Fattorizzazione QR con Rotazioni di Givens per una Matrice** 3×2**.** Consideriamo la matrice $\boldsymbol{A}$ di dimensione 3×3:

$$\boldsymbol{A} = \begin{pmatrix} 4 & 1 \\ 3 & 2 \\ 1 & 3 \end{pmatrix}$$

L'obiettivo è decomporre questa matrice in una matrice ortogonale $\boldsymbol{Q}$ e una matrice triangolare superiore $\boldsymbol{R}$ tali che:

$$\boldsymbol{A} = \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}$$

- **Passo 1:** Applicazione della prima rotazione di Givens per annullare l'elemento A_{21}
  1. Calcoliamo l'angolo θ utilizzando la formula per ottenere i coseni e seni per la rotazione.

$$r = \sqrt{A_{11}^2 + A_{21}^2} = \sqrt{4^2 + 3^2} = \sqrt{16 + 9} = 5$$

  2. Ora calcoliamo il coseno e il seno dell'angolo θ:

$$\cos\theta = \frac{A_{11}}{r} = \frac{4}{5}, \quad \sin\theta = \frac{A_{21}}{r} = \frac{3}{5}$$

  La matrice di rotazione di Givens è quindi:

$$\boldsymbol{G}_1 = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} = \frac{1}{5} \begin{pmatrix} 4 & 3 & 0 \\ -3 & 4 & 0 \\ 0 & 0 & 5 \end{pmatrix}$$

  3. Applichiamo la rotazione $\boldsymbol{G}_1$ alla matrice $\boldsymbol{A}$:

$$\boldsymbol{A}^{(1)} = \boldsymbol{G}_1\boldsymbol{A} = \frac{1}{5}\begin{pmatrix} 4 & 3 & 0 \\ -3 & 4 & 0 \\ 0 & 0 & 5 \end{pmatrix}\begin{pmatrix} 4 & 1 \\ 3 & 2 \\ 1 & 3 \end{pmatrix}$$
$$= \frac{1}{5}\begin{pmatrix} 25 & 10 \\ 0 & 5 \\ 5 & 15 \end{pmatrix} = \begin{pmatrix} 5 & 2 \\ 0 & 1 \\ 1 & 3 \end{pmatrix}$$

- **Passo 2:** Applicazione della seconda rotazione di Givens per annullare l'elemento $A_{31}^{(1)}$
  1. Calcoliamo il coseno e il seno della rotazione

$$r = \sqrt{A_{11}^2 + A_{31}^2} = \sqrt{5^2 + 1^2} = \sqrt{25 + 1} = \sqrt{26}$$

$$\cos\theta = \frac{A_{11}^{(1)}}{r} = \frac{5}{\sqrt{26}}, \quad \sin\theta = \frac{A_{31}^{(1)}}{r} = \frac{1}{\sqrt{26}}$$

  La matrice di rotazione di Givens $\boldsymbol{G}_2$ per annullare l'elemento $A_{31}^{(1)}$ è quindi:

$$\boldsymbol{G}_2 = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix} = \frac{1}{\sqrt{26}}\begin{pmatrix} 5 & 0 & 1 \\ 0 & \sqrt{26} & 0 \\ -1 & 0 & 5 \end{pmatrix}$$

  2. Applichiamo la rotazione $\boldsymbol{G}_2$ alla matrice $\boldsymbol{A}^{(1)}$.

$$\boldsymbol{A}^{(2)} = \boldsymbol{G}_2\boldsymbol{A}^{(1)} = \frac{1}{\sqrt{26}}\begin{pmatrix} 5 & 0 & 1 \\ 0 & \sqrt{26} & 0 \\ -1 & 0 & 5 \end{pmatrix}\begin{pmatrix} 5 & 2 \\ 0 & 1 \\ 1 & 3 \end{pmatrix}$$
$$= \frac{1}{\sqrt{26}}\begin{pmatrix} 26 & 13 \\ 0 & \sqrt{26} \\ 0 & 13 \end{pmatrix}$$

- **Passo 3:** Applicazione della terza rotazione di Givens per annullare l'elemento $A_{32}^{(2)}$
  1. Calcoliamo il coseno e il seno dell'angolo θ:

$$r = \sqrt{A_{22}^2 + A_{32}^2} = \sqrt{1^2 + 13^2/26} = \sqrt{15/2}$$

$$\cos\theta = \frac{A_{22}^{(2)}}{r} = \frac{\sqrt{2}}{\sqrt{15}}, \quad \sin\theta = \frac{A_{32}^{(2)}}{r} = \frac{\sqrt{13}}{\sqrt{15}}$$

  La matrice di rotazione di Givens $\boldsymbol{G}_3$ per annullare l'elemento $A_{32}^{(2)}$ è quindi:

$$\boldsymbol{G}_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{pmatrix} = \frac{1}{\sqrt{15}}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{2} & \sqrt{13} \\ 0 & -\sqrt{13} & \sqrt{2} \end{pmatrix}$$

  2. Applichiamo la rotazione $\boldsymbol{G}_3$ alla matrice $\boldsymbol{A}^{(2)}$:

$$\boldsymbol{A}^{(3)} = \boldsymbol{G}_3\boldsymbol{A}^{(2)} = \frac{1}{\sqrt{15}}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \sqrt{2} & \sqrt{13} \\ 0 & -\sqrt{13} & \sqrt{2} \end{pmatrix} \times \frac{1}{\sqrt{26}}\begin{pmatrix} 26 & 13 \\ 0 & \sqrt{26} \\ 0 & 13 \end{pmatrix}$$
$$= \frac{1}{\sqrt{390}}\begin{pmatrix} 26 & 13 \\ 0 & 15\sqrt{13} \\ 0 & 0 \end{pmatrix}$$

∎

4.5 Algoritmo di Householder

Data una matrice $\boldsymbol{A}$ di dimensioni $m \times n$, vogliamo applicare una trasformazione di Householder che agisca su una colonna di $\boldsymbol{A}$, trasformando i valori sotto la diagonale in zeri. La trasformazione di Householder è definita come:

$$\boldsymbol{H} = \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v} \cdot \boldsymbol{v}}$$

dove: - $\boldsymbol{I}$ è la matrice identità di dimensione appropriata. - v è un **vettore di Householder**, scelto in modo tale da creare una riflessione che annulli i valori desiderati.

4.5.1 Costruzione del Vettore di Householder

Dato un vettore colonna $\boldsymbol{x}$ di dimensione m, il vettore $\boldsymbol{v}$ di Householder è costruito come:

$$\boldsymbol{v} = \boldsymbol{x} + \text{sign}(x_1)\|\boldsymbol{x}\|_2 \boldsymbol{e}_1$$

dove: - $\text{sign}(x_1)$ è il segno del primo elemento di $\boldsymbol{x}$ (serve per evitare problemi di cancellazione numerica). - $\|\boldsymbol{x}\|_2$ è la norma euclidea di $\boldsymbol{x}$. - $\boldsymbol{e}_1$ è il vettore unitario che ha 1 come primo elemento e 0 altrove.

In questo modo, il vettore $\boldsymbol{v}$ è parallelo alla direzione di $\boldsymbol{x}$ ma modificato per creare una riflessione che annulli tutti gli elementi di $\boldsymbol{x}$ tranne il primo.

La fattorizzazione QR può essere ottenuta applicando una serie di trasformazioni di Householder a una matrice $\boldsymbol{A}$, così da ridurla alla forma triangolare superiore $\boldsymbol{R}$. Ogni trasformazione annulla gli elementi sotto la diagonale di una colonna specifica.

4.5.2 Passaggi per la Fattorizzazione QR

Supponiamo di voler fattorizzare una matrice $\boldsymbol{A}$ di dimensioni $m \times n$:

1. **Inizializzazione**:
   - Poniamo $\boldsymbol{Q} = \boldsymbol{I}$ (matrice identità) e $\boldsymbol{R} = \boldsymbol{A}$.
2. **Applica le Trasformazioni di Householder**:
   - Per ogni colonna j della matrice $\boldsymbol{R}$, dal primo all'ultimo elemento diagonale:
     1. Estrarre il vettore colonna $\boldsymbol{x}$ che parte dall'elemento (j, j) fino all'ultimo elemento della colonna.
     2. Costruire il vettore di Householder $\boldsymbol{v}$ in base a $\boldsymbol{x}$.
     3. Creare la matrice di Householder $\boldsymbol{H}_j = \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v} \cdot \boldsymbol{v}}$.
     4. Applicare $\boldsymbol{H}_j$ alla matrice $\boldsymbol{R}$: $\boldsymbol{R} \leftarrow \boldsymbol{H}_j \boldsymbol{R}$.
     5. Aggiornare $\boldsymbol{Q}$ moltiplicandolo a sinistra per $\boldsymbol{H}_j$: $\boldsymbol{Q} \leftarrow \boldsymbol{Q}\boldsymbol{H}_j$.
3. **Itera fino a Ottenere $\boldsymbol{R}$ in Forma Triangolare**:
   - Continua applicando le trasformazioni di Householder a tutte le colonne successive, fino a che $\boldsymbol{R}$ non è in forma triangolare superiore.

Alla fine della procedura, avremo:

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$$

dove $\boldsymbol{Q}$ è il prodotto delle matrici di Householder e risulta ortogonale ($\boldsymbol{Q}^\top\boldsymbol{Q} = \boldsymbol{I}$), e $\boldsymbol{R}$ è triangolare superiore.

▪ **Example 4.3 — Fattorizzazione QR con Matrici di Householder.** Consideriamo una matrice $\boldsymbol{A}$ di dimensione 3×2:

$$\boldsymbol{A} = \begin{bmatrix} 4 & 1 \\ 3 & 2 \\ 0 & -1 \end{bmatrix}$$

- **Passo 1:** Trasformazione della Prima Colonna
  1. Prendiamo la prima colonna di $\boldsymbol{A}$, ovvero il vettore $\boldsymbol{x}^\top = (4, 3, 12)$.
  2. Calcoliamo la norma di $\boldsymbol{x}$: $\|\boldsymbol{x}\|_2 = \sqrt{4^2 + 3^2 + 0^2} = 5$.
  3. Costruiamo il vettore di Householder $\boldsymbol{v}$:

$$\boldsymbol{v} = \boldsymbol{x} + \text{sign}(4)\|\boldsymbol{x}\|_2 \boldsymbol{e}_1 = \begin{bmatrix} 4 \\ 3 \\ 0 \end{bmatrix} + 5 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 9 \\ 3 \\ 0 \end{bmatrix}$$

  4. Costruiamo

$$\begin{aligned} \boldsymbol{H}_1 &= \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v} \cdot \boldsymbol{v}} \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \frac{1}{90} \begin{bmatrix} 9 \\ 3 \\ 0 \end{bmatrix} \begin{bmatrix} 9 \\ 3 \\ 0 \end{bmatrix}^\top \\ &= \begin{bmatrix} -4/5 & -3/5 & 0 \\ -3/5 & 4/4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

  5. Moltiplichiamo $\boldsymbol{H}_1$ per $\boldsymbol{A}$ per ottenere una nuova matrice $\boldsymbol{R}$ con zero sotto l'elemento $(1, 1)$.

$$\boldsymbol{H}_1 \boldsymbol{A} = \begin{bmatrix} -4/5 & -3/5 & 0 \\ -3/5 & 4/4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 4 & 1 \\ 3 & 2 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} -5 & -2 \\ 0 & 1 \\ 0 & -1 \end{bmatrix}$$

- **Passo 2:** Trasformazione della Seconda Colonna
  1. Prendiamo la seconda colonna di $\boldsymbol{A}$, ovvero il vettore $\boldsymbol{x}^\top = (0, 1, -1)$.
  2. Calcoliamo la norma di $\boldsymbol{x}$: $\|\boldsymbol{x}\|_2 = \sqrt{0^2 + 1^2 + (-1)^2} = \sqrt{2}$.
  3. Costruiamo il vettore di Householder $\boldsymbol{v}$:

$$\boldsymbol{v} = \boldsymbol{x} + \text{sign}(1)\|\boldsymbol{x}\|_2 \boldsymbol{e}_2 = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} + \sqrt{2} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 + \sqrt{2} \\ -1 \end{bmatrix}$$

  4. Costruiamo

$$\begin{aligned} \boldsymbol{H}_2 &= \boldsymbol{I} - 2\frac{\boldsymbol{v}\boldsymbol{v}^\top}{\boldsymbol{v} \cdot \boldsymbol{v}} \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \frac{1}{2 + \sqrt{2}} \begin{bmatrix} 0 \\ 1 + \sqrt{2} \\ -1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 + \sqrt{2} \\ -1 \end{bmatrix}^\top \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & -\sqrt{2}/2 & (1 + \sqrt{2})/(2 + \sqrt{2}) \\ 0 & (1 + \sqrt{2})/(2 + \sqrt{2}) & \sqrt{2}/2 \end{bmatrix} \end{aligned}$$

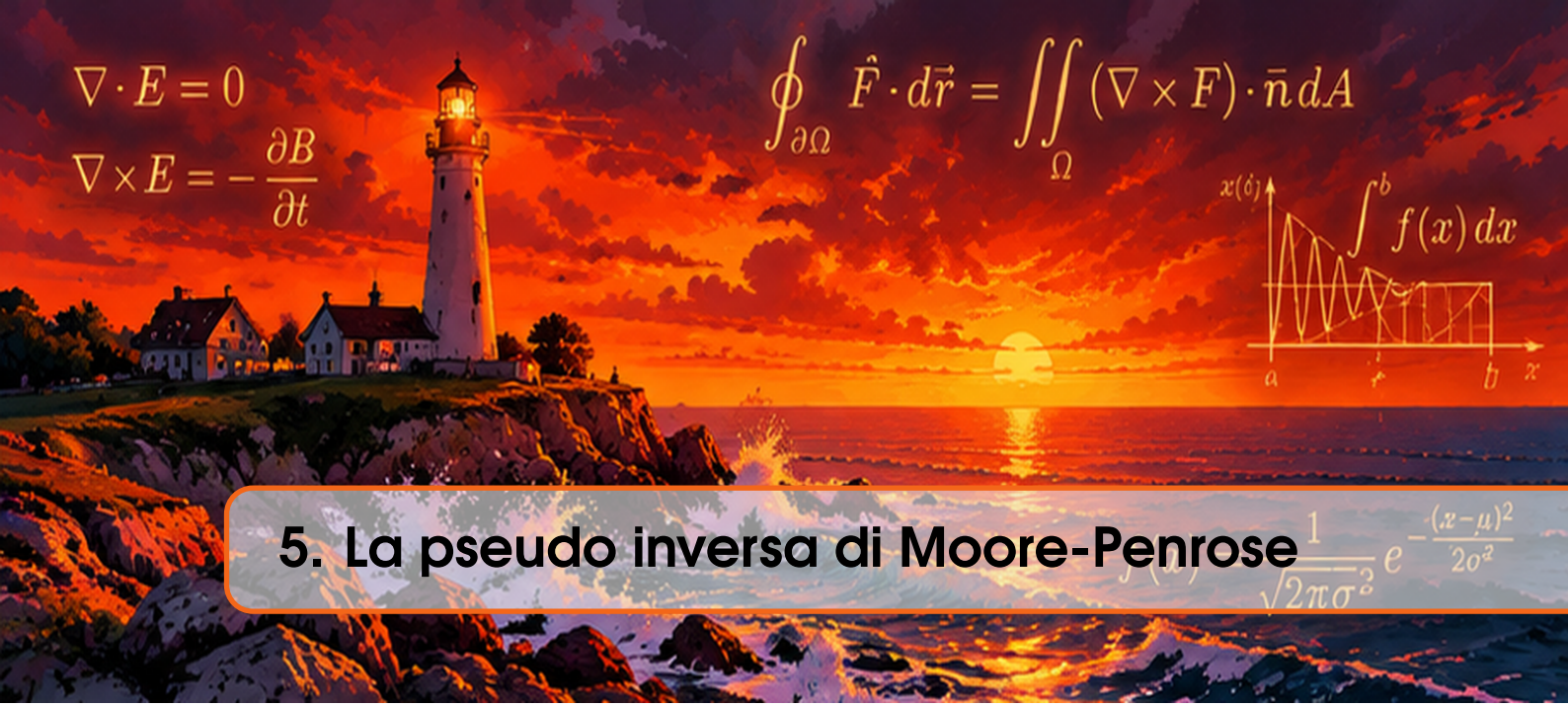
  5. Moltiplichiamo $\boldsymbol{H}_2$ per $\boldsymbol{H}_1\boldsymbol{A}$ per ottenere una nuova matrice $\boldsymbol{R}$ con zero sotto l'elemento $(2, 2)$.

$$\boldsymbol{H}_2 \boldsymbol{H}_1 \boldsymbol{A} = \begin{bmatrix} -5 & -2 \\ 0 & -\sqrt{2} \\ 0 & 0 \end{bmatrix}$$

∎

In sintesi

- Le matrici ortogonali conservano la norma euclidea: applicate ai minimi quadrati non alterano il residuo, e per questo sono lo strumento giusto.
- Le **rotazioni di Givens** annullano un elemento alla volta: convenienti per matrici sparse o quasi triangolari.
- Le **riflessioni di Householder** annullano un'intera colonna sotto la diagonale in un colpo solo: è la scelta standard per matrici piene.
- In entrambi i casi si ottiene $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$ e il problema dei minimi quadrati diventa triangolare, senza mai formare $\boldsymbol{A}^T\boldsymbol{A}$.



5. La pseudo inversa di Moore-Penrose

5.1 Pseudo-inversa definizione

Data una matrice $\boldsymbol{A} \in \mathbb{R}^{n \times m}$, definiamo pseudo-inversa di Moore-Penrose[1] la matrice $\boldsymbol{A}^+ \in \mathbb{R}^{m \times n}$ che soddisfa le seguenti quattro proprietà:

1. $\boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{A}$
2. $\boldsymbol{A}^+\boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}^+$
3. $\boldsymbol{A}^+\boldsymbol{A}$ è simmetrica (o hermitiana nel caso complesso)
4. $\boldsymbol{A}\boldsymbol{A}^+$ è simmetrica (o hermitiana nel caso complesso)

Ovviamente, per una matrice quadrata non singolare vale $\boldsymbol{A}^+ = \boldsymbol{A}^{-1}$, infatti:

1. $\boldsymbol{A}\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{A}$
2. $\boldsymbol{A}^{-1}\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}$
3. $\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{I}$ è simmetrica!
4. $\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I}$ è simmetrica!

5.2 Proprietà della pseudo-inversa

5.2.1 La pseudo-inversa di una matrice è unica

Supponiamo che esistano due pseudo-inverse della matrice $\boldsymbol{A}$, ovvero $\boldsymbol{A}^+$ e $\boldsymbol{B}$. Definiamo la matrice

$$\boldsymbol{M} = \boldsymbol{A}\boldsymbol{B} - \boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}(\boldsymbol{B} - \boldsymbol{A}^+)$$

e quindi

$$\begin{aligned}\boldsymbol{M}^2 &= \boldsymbol{A}(\boldsymbol{B} - \boldsymbol{A}^+)\boldsymbol{A}(\boldsymbol{B} - \boldsymbol{A}^+) \\ &= (\boldsymbol{A}\boldsymbol{B}\boldsymbol{A} - \boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A})(\boldsymbol{B} - \boldsymbol{A}^+) \\ &= (\boldsymbol{A} - \boldsymbol{A})(\boldsymbol{B} - \boldsymbol{A}^+) \\ &= \boldsymbol{0}(\boldsymbol{B} - \boldsymbol{A}^+) = \boldsymbol{0}\end{aligned}$$

Per la proprietà 4, la matrice $\boldsymbol{M}$ è simmetrica, perché è la differenza delle matrici simmetriche $\boldsymbol{A}\boldsymbol{B}$ e $\boldsymbol{A}\boldsymbol{A}^+$. Pertanto:

$$\boldsymbol{0} = \boldsymbol{M}^2 = \boldsymbol{M}\boldsymbol{M} = \boldsymbol{M}\boldsymbol{M}^\top$$

[1] Eliakim Hastings Moore (1862-1932) e Roger Penrose (1931)

Ciò implica che $\boldsymbol{M} = \boldsymbol{0}$: infatti, la k-esima colonna di $\boldsymbol{M}$ è $\boldsymbol{M}\boldsymbol{e}_k$ e vale

$$\|\boldsymbol{M}\boldsymbol{e}_k\|^2 = (\boldsymbol{M}\boldsymbol{e}_k) \cdot (\boldsymbol{M}\boldsymbol{e}_k) = \boldsymbol{e}_k^\top \boldsymbol{M}^\top \boldsymbol{M}\boldsymbol{e}_k = \boldsymbol{e}_k^\top \boldsymbol{0}\boldsymbol{e}_k = 0$$

e quindi ogni colonna di $\boldsymbol{M}$ è nulla. Usando lo stesso argomento definendo la matrice

$$\boldsymbol{N} = \boldsymbol{B}\boldsymbol{A} - \boldsymbol{A}^+\boldsymbol{A} = (\boldsymbol{B} - \boldsymbol{A}^+)\boldsymbol{A}$$

otteniamo che $\boldsymbol{N} = \boldsymbol{0}$. Quindi abbiamo

$$\begin{aligned} \boldsymbol{0} = \boldsymbol{M} = \boldsymbol{A}\boldsymbol{B} - \boldsymbol{A}\boldsymbol{A}^+ \quad &\implies \quad \boldsymbol{A}\boldsymbol{B} = \boldsymbol{A}\boldsymbol{A}^+ \\ \boldsymbol{0} = \boldsymbol{N} = \boldsymbol{B}\boldsymbol{A} - \boldsymbol{A}^+\boldsymbol{A} \quad &\implies \quad \boldsymbol{B}\boldsymbol{A} = \boldsymbol{A}^+\boldsymbol{A} \end{aligned}$$

e usando queste ultime proprietà

$$\boldsymbol{A}^+ = \boldsymbol{A}^+\boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}^+\boldsymbol{A}\boldsymbol{B} = \boldsymbol{B}\boldsymbol{A}\boldsymbol{B} = \boldsymbol{B}$$

5.2.2 La pseudo-inversa di una matrice non singolare è l'inversa "classica"

Quando esiste, l'inversa è anche una pseudo-inversa di Moore-Penrose. Per l'unicità della pseudo-inversa, essa coincide quindi con l'inversa.

5.2.3 La pseudo-inversa della pseudo-inversa è la matrice originale

ovvero $(\boldsymbol{A}^+)^+ = \boldsymbol{A}$. Prendiamo $\boldsymbol{B} = \boldsymbol{A}^+$ e usiamo le proprietà della pseudo-inversa

1. $\boldsymbol{B}\boldsymbol{B}^+\boldsymbol{B} = \boldsymbol{B}$ ovvero $\boldsymbol{A}^+\boldsymbol{B}^+\boldsymbol{A}^+ = \boldsymbol{A}^+$
2. $\boldsymbol{B}^+\boldsymbol{B}\boldsymbol{B}^+ = \boldsymbol{B}^+$ ovvero $\boldsymbol{B}^+\boldsymbol{A}^+\boldsymbol{B}^+ = \boldsymbol{B}^+$
3. $\boldsymbol{B}^+\boldsymbol{B}$ ovvero $\boldsymbol{B}^+\boldsymbol{A}^+$
4. $\boldsymbol{B}\boldsymbol{B}^+$ ovvero $\boldsymbol{A}^+\boldsymbol{B}^+$

La matrice $\boldsymbol{B}^+$ soddisfa le proprietà 1–4 della pseudo-inversa, scambiando tra loro le proprietà 1 e 2 e le proprietà 3 e 4. Quindi $\boldsymbol{A}^+$ è la pseudo-inversa di $\boldsymbol{B}^+$; per l'unicità della pseudo-inversa, ne segue che $\boldsymbol{B}^+ = \boldsymbol{A}$.

5.2.4 La pseudo-inverse di un vettore $\boldsymbol{a}$ è $\boldsymbol{a}^+ = (\boldsymbol{a} \cdot \boldsymbol{a})^{-1}\boldsymbol{a}^\top$

Basta verificare le quattro condizioni (poniamo $c = 1/(\boldsymbol{a} \cdot \boldsymbol{a})$).

1. $\boldsymbol{a}\boldsymbol{a}^+\boldsymbol{a} = \boldsymbol{a}c\boldsymbol{a}^\top\boldsymbol{a} = \boldsymbol{a}\cancel{c(\boldsymbol{a} \cdot \boldsymbol{a})} = \boldsymbol{a}$
2. $\boldsymbol{a}^+\boldsymbol{a}\boldsymbol{a}^+ = c\boldsymbol{a}^\top\boldsymbol{a}c\boldsymbol{a}^\top = (\boldsymbol{a} \cdot \boldsymbol{a})^{-1}\boldsymbol{a}^\top = \boldsymbol{a}^+$
3. $\boldsymbol{a}^+\boldsymbol{a} = c\boldsymbol{a}^\top\boldsymbol{a} = 1$ è uno scalare chiaramente simmetrico
4. $\boldsymbol{a}\boldsymbol{a}^+ = \boldsymbol{a}c\boldsymbol{a}^\top$ e anche questa è una matrice simmetrica

5.2.5 Pseudo-inversa di una matrice diagonale

Sia $\boldsymbol{D}$ una matrice diagonale. Allora $\boldsymbol{D}^+$ è una matrice diagonale con

$$D_{ii}^+ = \begin{cases} \dfrac{1}{D_{ii}} & D_{ii} \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

5.2.6 Pseudo-inversa ed SVD

Consideriamo una matrice rettangolare $\boldsymbol{A}$. La sua fattorizzazione SVD, che esiste sempre, è

$$\boldsymbol{A} = \boldsymbol{U}^\top \boldsymbol{\Sigma} \boldsymbol{V}$$

dove

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 & & & \\ 0 & \sigma_2 & & \vdots & & & \\ \vdots & & \ddots & 0 & & & \\ 0 & \cdots & 0 & \sigma_k & & & \\ & & & & 0 & \cdots & 0 \\ & & & & \vdots & \ddots & \vdots \\ & & & & 0 & \cdots & 0 \end{pmatrix}$$

e $\boldsymbol{U}$ e $\boldsymbol{V}$ sono matrici ortogonali. Allora

$$\boldsymbol{A}^+ = \boldsymbol{V}^\top \boldsymbol{\Sigma}^+ \boldsymbol{U}$$

dove

$$\boldsymbol{\Sigma}^+ = \begin{pmatrix} \sigma_1^{-1} & 0 & \cdots & 0 & & & \\ 0 & \sigma_2^{-1} & & \vdots & & & \\ \vdots & & \ddots & 0 & & & \\ 0 & \cdots & 0 & \sigma_k^{-1} & & & \\ & & & & 0 & \cdots & 0 \\ & & & & \vdots & \ddots & \vdots \\ & & & & 0 & \cdots & 0 \end{pmatrix}^\top$$

5.3 Connessioni con la fattorizzazione QR

Nel caso $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ con $n \geq m$ e $\boldsymbol{A}$ di rango massimo possiamo calcolare la fattorizzazione QR

$$\boldsymbol{A} = \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}$$

e possiamo definire la pseudo-inversa come

$$\boldsymbol{A}^+ = \begin{pmatrix} \boldsymbol{R}^{-1} & \boldsymbol{0} \end{pmatrix} \boldsymbol{Q}^T$$

infatti vale

5.3.1 Proprietà 1: $\boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{A}$

$$\begin{aligned} \boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A} &= \underbrace{\boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}}_{\boldsymbol{A}} \underbrace{\begin{pmatrix} \boldsymbol{R}^{-1} & \boldsymbol{0} \end{pmatrix} \boldsymbol{Q}^T}_{\boldsymbol{A}^+} \underbrace{\boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}}_{\boldsymbol{A}} \\ &= \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{R}^{-1} & \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix} \\ &= \boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{R}^{-1}\boldsymbol{R} \end{pmatrix} \\ &= \underbrace{\boldsymbol{Q} \begin{pmatrix} \boldsymbol{R} \\ \boldsymbol{0} \end{pmatrix}}_{\boldsymbol{A}} \begin{pmatrix} \boldsymbol{I} \end{pmatrix} = \boldsymbol{A} \end{aligned}$$

5.3.2 Proprietà 2: $A^+AA^+ = A^+$

$$\begin{aligned} A^+AA^+ &= \underbrace{\begin{pmatrix} R^{-1} & 0 \end{pmatrix} Q^T}_{A^+} \underbrace{Q \begin{pmatrix} R \\ 0 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} R^{-1} & 0 \end{pmatrix} Q^T}_{A^+} \\ &= \begin{pmatrix} R^{-1} & 0 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} \begin{pmatrix} R^{-1} & 0 \end{pmatrix} Q^T \\ &= I \underbrace{\begin{pmatrix} R^{-1} & 0 \end{pmatrix} Q^T}_{A^+} = A^+ \end{aligned}$$

5.3.3 Proprietà 3: A^+A simmetrica

$$\begin{aligned} A^+A &= \underbrace{\begin{pmatrix} R^{-1} & 0 \end{pmatrix} Q^T}_{A^+} \underbrace{Q \begin{pmatrix} R \\ 0 \end{pmatrix}}_{A} \\ &= \begin{pmatrix} R^{-1} & 0 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} \\ &= I, \qquad \text{[ovviamente simmetrica]} \end{aligned}$$

5.3.4 Proprietà 4: AA^+ simmetrica

$$\begin{aligned} AA^+ &= \underbrace{Q \begin{pmatrix} R \\ 0 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} R^{-1} & 0 \end{pmatrix} Q^T}_{A^+} \\ &= Q \begin{pmatrix} R \\ 0 \end{pmatrix} \begin{pmatrix} R^{-1} & 0 \end{pmatrix} Q^T \\ &= Q \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} Q^T, \qquad \text{[ovviamente simmetrica]} \end{aligned}$$

5.4 Pseudo inversa e fattorizzazione LQ

Data una matrice $A \in \mathbb{R}^{n \times m}$ con $m > n$, non possiamo calcolarne la fattorizzazione QR; possiamo invece fattorizzarne la trasposta.

$$A^T = Q \begin{pmatrix} R \\ 0 \end{pmatrix}$$

e quindi

$$A = \begin{pmatrix} R^T & 0 \end{pmatrix} Q^T$$

La matrice R^T è triangolare inferiore e possiamo indicarla con L. In generale, una matrice $A \in \mathbb{R}^{n \times m}$ di rango massimo si può decomporre come

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \qquad \text{se } n \geq m$$

$$A = \begin{pmatrix} L & 0 \end{pmatrix} Q \qquad \text{se } n \leq m$$

Il caso $n \geq m$ lo abbiamo appena visto. Consideriamo ora il caso $n \leq m$. In questo caso

$$A^+ = Q^T \begin{pmatrix} L^{-1} & 0 \end{pmatrix}$$

infatti vale

5.4.1 Proprietà 1: $AA^+A = A$

$$\begin{aligned} AA^+A &= \underbrace{\begin{pmatrix} L & 0 \end{pmatrix} Q}_{A} \underbrace{Q^T \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix}}_{A^+} \underbrace{\begin{pmatrix} L & 0 \end{pmatrix} Q}_{A} \\ &= \begin{pmatrix} L & 0 \end{pmatrix} \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix} \underbrace{\begin{pmatrix} L & 0 \end{pmatrix} Q}_{A} \\ &= I \underbrace{\begin{pmatrix} L & 0 \end{pmatrix} Q}_{A} \\ &= A \end{aligned}$$

5.4.2 Proprietà 2: $A^+AA^+ = A^+$

$$\begin{aligned} A^+AA^+ &= \underbrace{Q^T \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix}}_{A^+} \underbrace{\begin{pmatrix} L & 0 \end{pmatrix} Q}_{A} \underbrace{Q^T \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix}}_{A^+} \\ &= \underbrace{Q \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix}}_{A^+} \begin{pmatrix} L & 0 \end{pmatrix} \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix} \\ &= \underbrace{Q \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix}}_{A^+} I = A^+ \end{aligned}$$

5.4.3 Proprietà 3: A^+A simmetrica

$$\begin{aligned} A^+A &= \underbrace{Q \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix}}_{A^+} \underbrace{\begin{pmatrix} L & 0 \end{pmatrix} Q}_{A} \\ &= Q \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} Q, \qquad \text{[ovviamente simmetrica]} \end{aligned}$$

5.4.4 Proprietà 4: AA^+ simmetrica

$$\begin{aligned} AA^+ &= \underbrace{\begin{pmatrix} L & 0 \end{pmatrix} Q}_{A} \underbrace{Q^T \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix}}_{A^+} \\ &= \begin{pmatrix} L & 0 \end{pmatrix} \begin{pmatrix} L^{-1} \\ 0 \end{pmatrix} = I, \qquad \text{[ovviamente simmetrica]} \end{aligned}$$

In sintesi

- La pseudo-inversa A^+ è l'unica matrice che soddisfa le quattro condizioni di Moore-Penrose; per A non singolare coincide con A^{-1}.
- $x = A^+b$ è la soluzione dei minimi quadrati di **norma minima**: risolve insieme il caso sovradeterminato e quello sottodeterminato.
- Si calcola dalla fattorizzazione QR (caso $m > n$) o LQ (caso $m < n$), e dalla SVD nel caso generale, senza mai invertire esplicitamente nulla.

III Interpolazione polinomiale

6 Interpolazione Polinomiale 77
6.1 Unicità del polinomio interpolante 81
6.2 Generalizzazione 84
6.3 Condizione di Haar (o unisolvenza) 84
6.4 Errore di interpolazione 84

7 Interpolazione di Lagrange 89
7.1 Derivazione pratica della interpolazione di Lagrange 91
7.2 Polinomio di miglior approssimazione 93

8 Interpolazione di Newton 95
8.1 Proprietà delle differenze divise 98
8.2 Invarianza per permutazioni dei nodi 100
8.3 Formula ricorrente 100
8.4 L'algoritmo di Aitken-Neville 104

9 Esempio di Runge 107
9.1 Approssimazione di funzioni continue 107
9.2 Controesempio di Runge 109
9.3 Conclusioni 114



6. Interpolazione Polinomiale

Qui ci occuperemo di approssimare una funzione f assegnata con un polinomio p di grado assegnato e come descrivere con un polinomio un insieme di misurazioni. I due problemi sono spesso equivalenti, ad esempio se abbiamo un certo insieme di misurazioni $\{(x_i, y_i)\}_{i=0}^m$, le possiamo interpretare come i valori di una funzione f incognita che vogliamo approssimare, ad esempio con un polinomio:

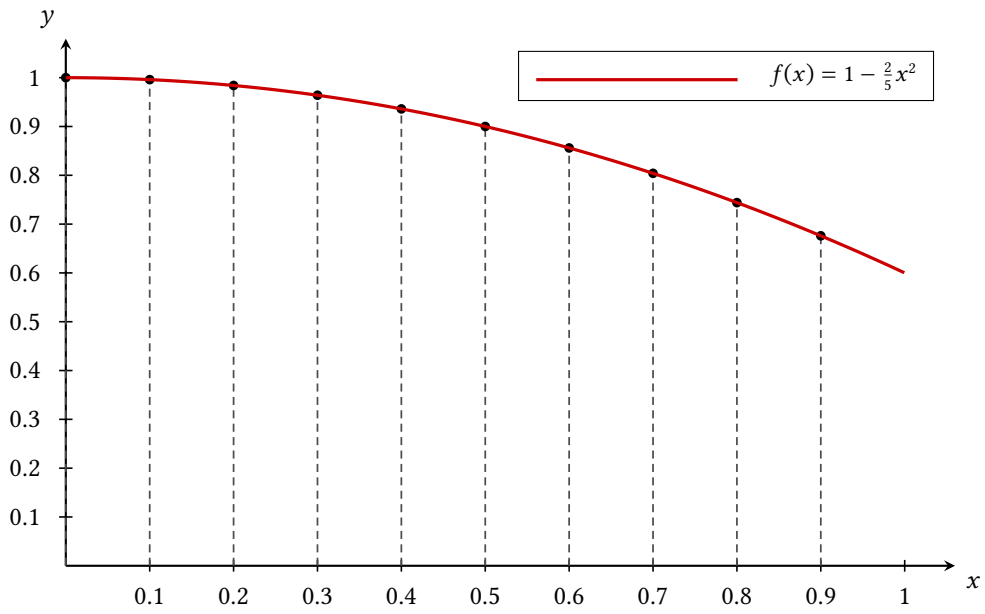


Figure 6.1: Campionamento della funzione $f(x) = 1 - \frac{2}{5}x^2$ in nodi equispaziati. Le coppie $(x_i, f(x_i))$, indicate dai punti neri, costituiscono i dati da interpolare con un polinomio.

Indichiamo con $\mathbb{R}_n[x]$ lo spazio vettoriale di polinomi di grado al più n a coefficienti reali. Vogliamo trovare un polinomio $p(x) \in \mathbb{R}_n[x]$ tale che $p(x_i) = y_i$ per ogni $i = 1, \dots, m$. Ciò corrisponde a risolvere un sistema lineare del tipo:

$$
\begin{aligned}
f(x) \approx p(x) &= a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n \\
p(x_0) &= a_0 + a_1 x_0 + a_2 x_0^2 + \ldots + a_n x_0^n \\
p(x_1) &= a_0 + a_1 x_1 + a_2 x_1^2 + \ldots + a_n x_1^n \\
&\vdots \\
p(x_m) &= a_0 + a_1 x_m + a_2 x_m^2 + \ldots + a_n x_m^n
\end{aligned}
$$

▪ **Example 6.1** Se per esempio $m = 2$ e $n = 3$, con i dati $\{(0,1),(1,2),(-1,3)\}$, cerchiamo di determinare i coefficienti di $p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$ tramite il sistema:

$$
\begin{aligned}
p(x_0) &= a_0 + a_1 0 + a_2 0^2 + a_3 0^3 = 1 = y_0 \\
p(x_1) &= a_0 + a_1 1 + a_2 1^2 + a_3 1^3 = 2 = y_1 \\
p(x_2) &= a_0 + a_1(-1) + a_2(-1)^2 + a_3(-1)^3 = 3 = y_2.
\end{aligned}
$$

▪

In generale, un sistema rettangolare (cioè con $n \neq m$) può non avere soluzioni oppure averne infinite. Per garantire esistenza e unicità è necessario che $n = m$ e che i nodi siano distinti, cioè $x_i \neq x_j$ per $i \neq j$. D'ora in poi assumeremo che queste condizioni siano soddisfatte.

▪ **Example 6.2** Nel caso di $n = 0$ il polinomio si riduce a $p(x) = a_0 = y_0$ ovvero al polinomio costante; se $n = 1$ avremo $p(x) = a_0 + a_1 x$ cioè una retta; se $n = 2$ avremo una parabola.

▪

▪ **Example 6.3** Supponiamo di voler interpolare i dati $\{(-1,1),(2,7),(0,1)\}$, avremo bisogno di un polinomio di secondo grado $p(x) = a_0 + a_1 x + a_2 x^2$. Il sistema da risolvere è quindi:

$$
\begin{aligned}
a_0 + a_1(-1) + a_2(-1)^2 &= 1 \\
a_0 + a_1 2 + a_2 2^2 &= 7 \\
a_0 + a_1 0 + a_2 0^2 &= 1,
\end{aligned}
$$

da cui

$$
\begin{aligned}
a_0 - a_1 + a_2 &= 1 \\
a_0 + 2a_1 + 4a_2 &= 7 \\
a_0 &= 1,
\end{aligned}
$$

e quindi semplificando, $-a_1 + a_2 = 0$ e $2a_1 + 4a_2 = 6$ da cui $a_2 = a_1$ e $6a_2 = 6$. La soluzione è quindi $a_0 = a_1 = a_2 = 1$, ovvero $p(x) = 1 + x + x^2$.

▪

Questo metodo mostra che approssimare una funzione a partire da dati campionati equivale a risolvere un sistema lineare. Il procedimento è però macchinoso; in questa sezione ne vedremo di più efficienti. Iniziamo con un teorema che garantisce l'esistenza di un polinomio di questo tipo.

Theorem 6.1 — Stone-Weierstrass. Sia f una funzione continua definita sull'intervallo $[a,b]$, allora per ogni $\varepsilon > 0$ esiste un numero naturale n (dipendente da ε) e un polinomio $p \in \mathbb{R}_n[x]$ tali che $\|f - p\|_\infty < \varepsilon$.

Ricordiamo che la norma infinito di un vettore v significa $\|v\|_\infty := \max |v|$, e nel caso di funzioni corrisponde a $\|f\|_\infty := \max_{a \leq x \leq b} |f(x)|$. Il teorema non ci dice quale sia questo polinomio approssimante, né che grado abbia. Tuttavia è possibile costruire una successione di polinomi b_n, detti polinomi di Bernstein[1], per i quali vale il seguente limite:

[1] Sergej Natanovič Bernštejn (1880-1968)

$$\lim_{n\to\infty} \|f - b_n\|_\infty = 0.$$

Inoltre se $f \in C^k[a,b]$ allora

$$\lim_{n\to\infty} \|f^{(k)} - b_n^{(k)}\|_\infty = 0,$$

ovvero il polinomio approssima bene anche le derivate di f.

I polinomi di Bernstein sono molto utili in quanto sono la base delle Curve di Bèzier, molto usate in computer graphics e in altre applicazioni. Senza perdita di generalità si possono scrivere sull'intervallo $[0,1]$ come:

$$b_n(x) = \sum_{i=0}^{n} c_k B_i^n(x) = \sum_{i=0}^{n} c_i \binom{n}{i} x^i (1-x)^{n-i}.$$

I coefficienti c_i sono dati dai valori della funzione da approssimare su punti equispaziati di $[0,1]$, dunque

$$f : [0,1] \mapsto \mathbb{R}, \qquad f(x) \approx b_n(x) = \sum_{i=0}^{n} f(i/n) \binom{n}{i} x^i (1-x)^{n-i}$$

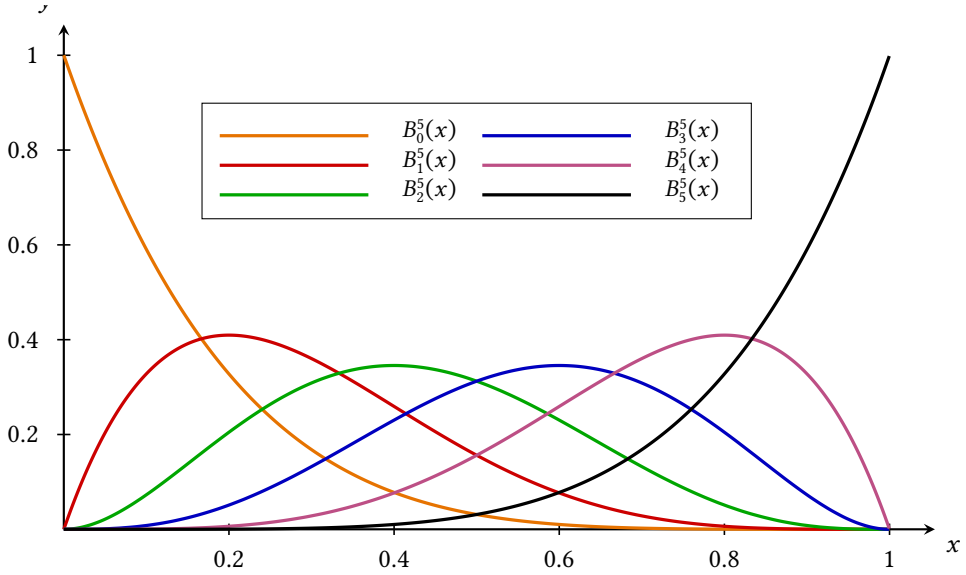


Figure 6.2: La base di Bernstein dei polinomi di quinto grado $B_i^5(x)$ per $i = 0, 1, \ldots, 5$.

Il polinomio di Bernstein è una media pesata dei valori di f nei punti di suddivisione. I pesi sono i polinomi $B_i^n(x) = \binom{n}{i} x^i (1-x)^{n-i}$, detti polinomi base di Bernstein (si veda Figura 6.2). Essi attribuiscono pesi maggiori ai valori di f associati ai punti di suddivisione più vicini e pesi trascurabili a quelli più lontani. Questa proprietà si può interpretare in termini probabilistici mediante la legge dei grandi numeri. La media dei valori di f è dunque effettivamente una media pesata, poiché i coefficienti hanno somma pari a 1:

$$\sum_{i=0}^{n} \binom{n}{i} x^i (1-x)^{n-i} = (x + (1-x))^n = 1.$$

L'area sottesa a ogni peso è pari a $1/(n+1)$, dunque la loro somma è 1 e l'area è equidivisa. Supponiamo ora che x rappresenti la probabilità di un certo evento, ad esempio $x = 1/6$ la probabilità che esca 6 lanciando un dado. Se facciamo n lanci indipendenti, allora $B_i^n(x)$ è la

probabilità che l'evento si verifichi esattamente i volte. Allora, a patto di prendere n abbastanza grande, possiamo rendere arbitrariamente grande la probabilità che la frequenza i/n del verificarsi dell'evento sia vicina (più di ε) alla sua probabilità x, ovvero la legge dei grandi numeri.

▪ **Example 6.4** Approssimiamo la funzione $f(x) = \cos(x)$ su $[a, b] = [0, 1]$ con alcuni polinomi di Bernstein. Come si vede dalla Figura 6.3, via via che il grado del polinomio si eleva, la qualità dell'approssimazione migliora.

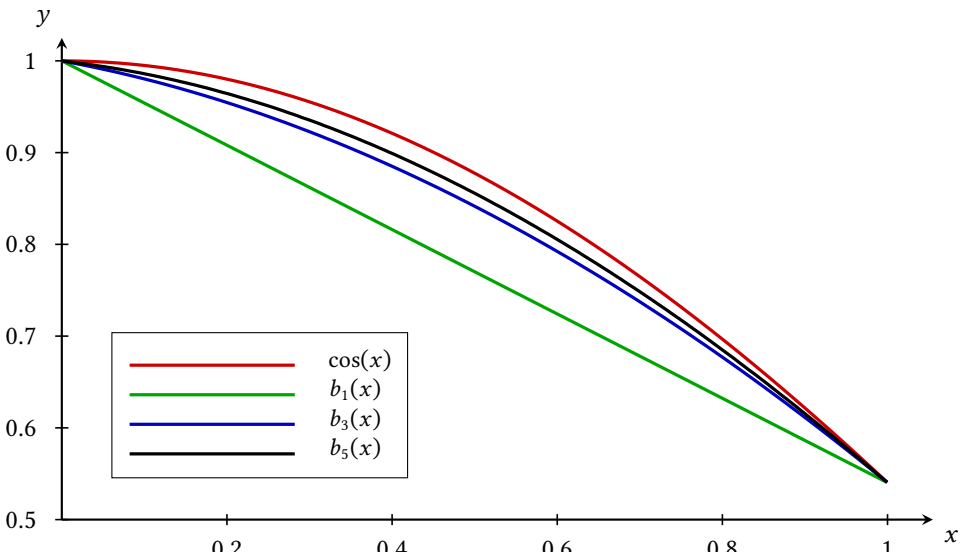


Figure 6.3: La funzione $\cos(x)$ approssimata su $[0, 1]$ con b_1, b_3 e b_5. All'aumentare del grado l'approssimazione migliora, ma la convergenza è lenta.

▪

La convergenza è estremamente lenta, anche nel caso in cui f stessa sia un polinomio. Dal Corso di Analisi Matematica possiamo usare il Teorema di Taylor[2], nel caso in cui f sia differenziabile.

Theorem 6.2 — Taylor. Sia f una funzione continua su $[a, b]$ differenziabile di classe almeno $C^{n+1}[a, b]$, allora per ogni punto $x_0 \in (a, b)$ fissato, esiste un polinomio $p_n(x) \in \mathbb{R}_n[x]$ (il polinomio di Taylor) tale che

$$f(x) = p_n(x) + R_{n+1}(x),$$
$$p_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k.$$

Il resto $R_{n+1}(x)$ si può esprimere in diversi modi; tra i più noti ricordiamo le forme di Peano e di Lagrange[a]. Quest'ultima è utile ai nostri fini perché consente di stimare l'errore di approssimazione. Il resto di Lagrange è

$$R_{n+1}(x) = \frac{(x - x_0)^{n+1}}{(n + 1)!} f^{(n+1)}(\xi_x),$$

con ξ_x (incognito) compreso tra x e x_0, ovvero $\xi_x \in (\min(x, x_0), \max(x_0, x))$.

[a] Joseph-Louis Lagrange (1736-1813)

Questo approccio sembra più promettente, ma vale solo localmente, in un intorno di x_0.

[2] Brook Taylor (1685-1731)

▪ **Example 6.5** Riprendiamo l'esempio precedente, usando questa volta alcuni polinomi di Taylor centrati in $x_0 = \frac{1}{2}$. Il risultato per il polinomio p_0 è la costante, quello al primo ordine è la retta tangente. Le approssimazioni sono via via migliori ma solamente intorno a x_0, Figura 6.4.

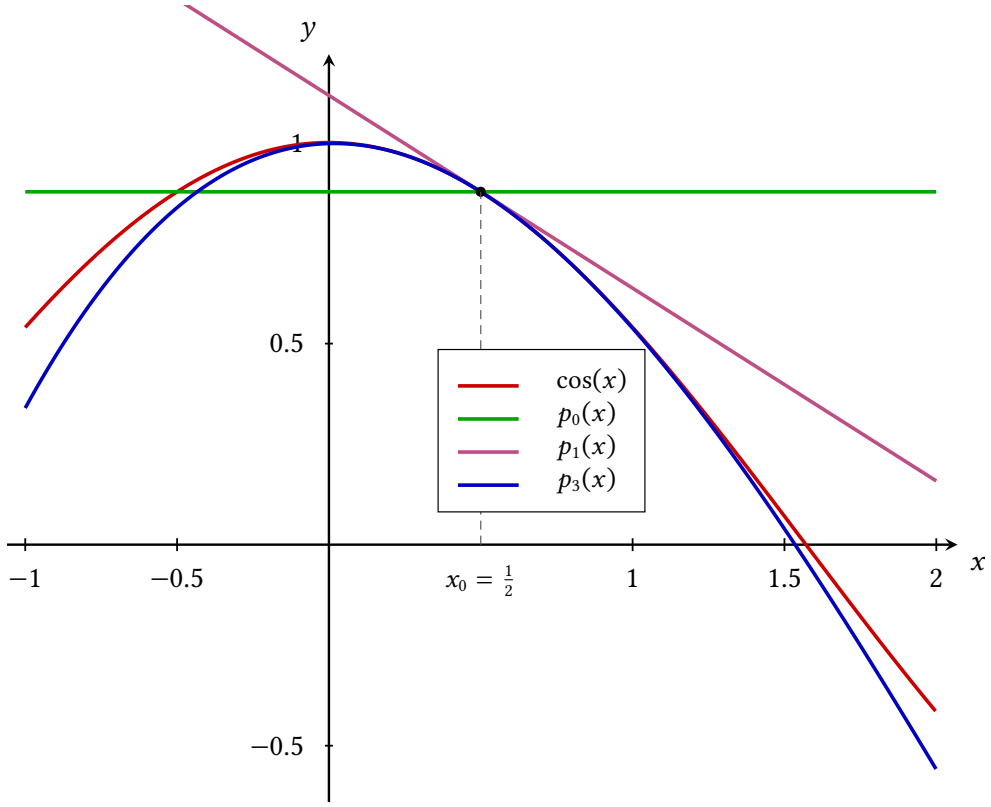


Figure 6.4: La funzione $\cos(x)$ approssimata in un intorno di $x_0 = 1/2$ con i polinomi di Taylor p_0, p_1 e p_3.

▪

Una terza via è data dalla costruzione dei polinomi interpolatori. Essi saranno utili anche nei prossimi capitoli quando affronteremo l'integrazione numerica e le equazioni differenziali.

6.1 Unicità del polinomio interpolante

Theorem 6.3 — interpolazione polinomiale. Siano $x_0, x_1, \dots, x_n \in \mathbb{R}$, $n+1$ punti distinti ed $f : \mathbb{R} \mapsto \mathbb{R}$ una funzione, allora *esiste un unico* polinomio di grado al più n, $p(x)$ per cui vale

$$p(x_i) = f(x_i), \qquad i = 0, 1, \dots, n$$

Proof. Consideriamo un generico polinomio di grado al più n,

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n,$$

ed imponiamo che soddisfi le $n+1$ condizioni della forma $(x_i, f(x_i))$ per $i = 0, 1, \dots, n$. Si ottiene il sistema lineare

$$\begin{cases} a_0 + a_1 x_0 + a_2 x^2 + \cdots + a_n x^n = f(x_0) \\ a_0 + a_1 x_1 + a_2 x_1^2 + \cdots + a_n x_1^n = f(x_1) \\ \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ a_0 + a_1 x_n + a_2 x_n^2 + \cdots + a_n x_n^n = f(x_n) \end{cases}$$

Il sistema si scrive in forma compatta matriciale $\boldsymbol{M}\boldsymbol{a} = \boldsymbol{b}$ con

$$\boldsymbol{M} = \begin{pmatrix} 1 & x_0 & x^2 & \cdots & x^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & & & \vdots & \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix}, \quad \text{ed} \quad \boldsymbol{a} = \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix}, \quad \boldsymbol{b} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix},$$

quindi il problema ha una unica soluzione se e solo se il determinante della matrice $\boldsymbol{M}$ è non nullo.

La matrice $\boldsymbol{M}$ è nota col nome di matrice di Vandermonde [3] ed il suo determinante ha la forma seguente:

$$\boxed{\prod_{i>j}^{n} (\boldsymbol{x}_i - x_j)} \tag{6.1}$$

Dall'ipotesi che $x_i \neq x_j$ per $i \neq j$ (punti distinti) segue che il determinante di $\boldsymbol{M}$ è non nullo ed il problema ha una unica soluzione.

Per completare la dimostrazione calcoliamo il determinante 6.1. Sia

$$V(x_0, x_1, \ldots, x_{n-1}, x) = \begin{vmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & & & \vdots & \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^n \\ 1 & x & x^2 & \cdots & x^n \end{vmatrix}. \tag{6.2}$$

Sviluppando il determinante in 6.2 per cofattori sull'ultima riga si ottiene un polinomio nella variabile x di grado al più n. Osserviamo che sostituendo al posto di x successivamente i valori x_0, x_1,... fino a $ x_{n-1}$ il determinante si annulla, cioè

$$V(x_0, x_1, \ldots, x_{n-1}, x_k]) = 0, \qquad k = 0, 1, \ldots, n-1.$$

Ciò accade perché si sta calcolando il determinante di una matrice con due righe uguali, l'ultima e di volta in volta la $k - esima$. Le ascisse x_0, x_1, ..., x_{n-1} sono radici del polinomio $V(x_0, x_1, \ldots, x_{n-1}, x)$, che può essere quindi fattorizzato nella forma

$$K(x - x_0)(x - x_1) \cdots (x - x_{n-1}), \tag{6.3}$$

dove lo scalare K è il coefficiente del termine di grado massimo.

Infatti, moltiplicando i binomi in 6.3

$$V(x_0, x_1, \ldots, x_{n-1}, x) = Kx^n + Lx^{n-1} \cdots$$

si vede subito che K è il coefficiente che moltiplica x^n.

Il determinante della matrice di Vandermonde coincide con il valore del polinomio in 6.3 calcolato nell'ascissa x_n, per cui si rende necessario determinare esplicitamente il coefficiente K. Sviluppando $V(x_0, x_1, \ldots, x_{n-1}, x)$ per minori rispetto alla ultima riga otteniamo

[3] Alexandre Théophile Vandermonde (1735-1796), l'attribuzione è dovuta a Henri Lèon Lebesgue (1875-1941), tuttavia questa matrice non compare nei lavori di Vandermonde.

$$V(x_0, x_1, \dots, x_{n-1}, x) = \begin{vmatrix} 1 & x_0 & x^2 & \cdots & x^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & & & & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^n \\ 1 & x & x^2 & \cdots & x^n \end{vmatrix}$$

$$= x^n \begin{vmatrix} 1 & x_0 & x^2 & \cdots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ \vdots & & & & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \end{vmatrix} + x^{n-1} \cdots$$

da cui segue immediatamente che

$$K = \begin{vmatrix} 1 & x_0 & x^2 & \cdots & x_0^{n-1} \\ 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ \vdots & & & & \vdots \\ 1 & x_{n-1} & x_{n-1}^2 & \cdots & x_{n-1}^{n-1} \end{vmatrix} = V(x_0, x_1, \dots, x_{n-1}).$$

Sostituendo questa espressione per K in 6.3 abbiamo

$$V(x_0, x_1, \dots, x_{n-1}, x_n) = V(x_0, x_1, \dots, x_{n-1})(x_n - x_0)(x_n - x_1) \cdots (x_n - x_{n-1}), \tag{6.4}$$

in cui è facile riconoscere una relazione di ricorrenza

$$\begin{aligned} V(x_0, x_1, \dots, x_{n-1}, x_n) &= V(x_0, x_1, \dots, x_{n-1}) \prod_{i=0}^{n-1} (x_n - x_i), \\ &= V(x_0, x_1, \dots, x_{n-2}) \left(\prod_{i=0}^{n-2} (x_{n-1} - x_i) \right) \left(\prod_{i=0}^{n-1} (x_n - x_i) \right), \\ &\vdots \\ &= V(x_0, x_1, \dots, x_k) \prod_{j=k}^{n} \prod_{i=0}^{j-1} (x_j - x_i), \qquad k \geq 1. \end{aligned} \tag{6.5}$$

La ricorsione all'indietro si arresta sull'indice $k = 1$

$$V(x_0, x_1) = \begin{vmatrix} 1 & x_0 \\ 1 & x_1 \end{vmatrix} = x_1 - x_0.$$

che in 6.5 produce proprio la 6.1. ∎

Definition 6.1 — Nodi di interpolazione. Le coppie ordinate della forma (x_i, y_i) con gli x_i assunti distinti, cioè $x_i \neq x_j$ per $i \neq j$, si indicano normalmente col termine di *nodi di interpolazione*.

Se necessario, si può anche supporre che $y_i = f(x_i)$, cioè che i valori assunti nei nodi di interpolazione siano tabulati a partire da una funzione nota $y = f(x)$.

Definition 6.2 — Gradi di libertà. Gli $n + 1$ coefficienti a_i per $i = 0, 1, \dots n$ che determinano il polinomio di grado n si indicano normalmente col termine di *gradi di libertà*. Il numero a_n si chiama *coefficiente del termine di grado massimo* o *leading coefficient*, ed è ovviamente assunto non nullo quando il polinomio è di grado n.

Osservazione

Il teorema Teorema 6.3 afferma che dati $n+1$ nodi di interpolazione (distinti) il polinomio interpolante di grado al più n esiste ed è unico. Nel teorema è stato scritto come combinazione lineare dei polinomi $1, x, x^2, \dots, x^n$.
Tuttavia può essere scritto anche come combinazione lineare di funzioni polinomiali generali, che indicheremo con i simboli $\mathcal{L}_0(x), \mathcal{L}_1(x), \dots, \mathcal{L}_n(x)$.

6.2 Generalizzazione

Esaminiamo cosa succede se consideriamo una funzione della forma:

$$\mathcal{L}(x) = a_0\mathcal{L}_0(x) + a_1\mathcal{L}_1(x) + \cdots + a_n\mathcal{L}_n(x),$$

per cui valga

$$\mathcal{L}(x_k) = f(x_k), \qquad k = 0, 1, \dots, n$$

Imponendo esplicitamente il passaggio per i nodi di interpolazione si ha un sistema lineare

$$\begin{cases} a_0\mathcal{L}_0(x_0) + a_1\mathcal{L}_1(x_0) + \cdots + a_n\mathcal{L}_n(x_0) = f(x_0) \\ a_0\mathcal{L}_0(x_1) + a_1\mathcal{L}_1(x_1) + \cdots + a_n\mathcal{L}_n(x_1) = f(x_1) \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\vdots \\ a_0\mathcal{L}_0(x_n) + a_1\mathcal{L}_1(x_n) + \cdots + a_n\mathcal{L}_n(x_n) = f(x_n) \end{cases}$$

che scriviamo in forma matriciale compatta $\boldsymbol{M}\boldsymbol{a} = \boldsymbol{b}$ con

$$\boldsymbol{M} = \begin{pmatrix} \mathcal{L}_0(x_0) & \mathcal{L}_1(x_0) & \mathcal{L}_2(x_0) & \cdots & \mathcal{L}_n(x_0) \\ \mathcal{L}_0(x_1) & \mathcal{L}_1(x_1) & \mathcal{L}_2(x_1) & \cdots & \mathcal{L}_n x_1) \\ \vdots & & & & \vdots \\ \mathcal{L}_0(x_n) & \mathcal{L}_1(x_n) & \mathcal{L}_2(x_n) & \cdots & \mathcal{L}_n(x_n) \end{pmatrix}, \quad \text{ed} \quad \boldsymbol{b} = \begin{pmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix},$$

nelle incognite $\boldsymbol{a} = (a_0,\ a_1,\ \dots,\ a_n)^T$.

Quindi la soluzione esiste ed è unica se e solo se la matrice $\boldsymbol{M}$ ha determinante non nullo.

6.3 Condizione di Haar (o unisolvenza)

Definition 6.3 — Condizione di Haar (o unisolvenza). Date $n+1$ funzioni polinomiali generali $\mathcal{L}_0(x)$, $\mathcal{L}_1(x)$,...,$\mathcal{L}_n(x)$ ed $n+1$ nodi di interpolazione con ascisse (x_i) supposte distinte, sia $\boldsymbol{M}$ la matrice di componenti $M_{i,j} = \mathcal{L}_j(x_i)$. La condizione $|\boldsymbol{M}| \neq 0$ si chiama *condizione di Haar*[a] o più semplicemente si dice che la base è unisolvente.

[a] Alfréd Haar (1885-1933)

Osservazione

La *condizione di Haar* esprime l'indipendenza lineare delle funzioni polinomiali $\mathcal{L}_i(x)$ per $i = 0, 1, \dots n$ sui nodi di interpolazione ed è *necessaria e sufficiente* per la risolubilità del problema di interpolazione.

6.4 Errore di interpolazione

Definition 6.4 — errore di interpolazione. Data la funzione $f(x)$ ed il polinomio $p(x)$ interpolante gli $n+1$ nodi (x_i, f_i) per $i = 0, 1, \dots n$, definiamo come *errore di interpolazione* nel punto x la differenza (in valore assoluto) tra il valore calcolato con la funzione ed il valore calcolato con il polinomio

$$E(x) = |\, f(x) - p(x)\,|.$$

È possibile calcolare l'errore di approssimazione che si commette a sostituire $p_n(x)$ alla funzione di partenza $f(x)$.

Theorem 6.4 — Errore di interpolazione. Sia

- $f(x)$ differenziabile di classe C^{n+1} sull'intervallo $[a, b]$
- $p_n(x)$ il suo polinomio interpolatore sui nodi distinti $\{x_i\}_{i=0}^n$
- $E_n(x) = f(x) - p_n(x)$ l'errore di interpolazione

allora per ogni x vale

$$E_n(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!}\omega_{n+1}(x), \qquad \xi_x \in I(x, x_0, \dots, x_n)$$

dove

$$I(x, x_0, \dots, x_n) = (\min\{x, x_0, \dots, x_n\}, \max\{x, x_0, \dots, x_n\}),$$

ovvero il più piccolo intervallo contenente i punti $x_0, x_1, \dots, x_n$ ed x, ed

$$\omega_{n+1}(x) = (x - x_0)(x - x_1)\cdots(x - x_n)$$

ovvero un polinomio monico le cui radici sono le ascisse di interpolazione.

Proof. L'errore di interpolazione è ovviamente zero se x coincide con uno dei nodi $\{x_i\}_{i=0}^n$. Negli altri casi fissiamo $x = \bar{x}$ e definiamo una funzione ausiliaria $G(z)$ come

$$G(z) := f(z) - p_n(z) - \frac{\omega_{n+1}(z)E_n(\bar{x})}{\omega_{n+1}(\bar{x})},$$

con la proprietà che G è di classe C^{n+1} su I in quanto lo è f, mentre le restanti funzioni sono polinomi, inoltre G ha $n+2$ zeri distinti, dati dai nodi e da $\bar{x}$. Infatti per $i = 0, 1, \dots, n$:

$$\begin{aligned} G(x_i) &= f(x_i) - p_n(x_i) - \frac{\omega_{n+1}(x_i)E_n(\bar{x})}{\omega_{n+1}(\bar{x})} \\ &= f(x_i) - f(x_i) - 0 = 0, \end{aligned}$$

ed

$$\begin{aligned} G(\bar{x}) &= f(\bar{x}) - p_n(\bar{x}) - \frac{\omega_{n+1}(\bar{x})E_n(\bar{x})}{\omega_{n+1}(\bar{x})} \\ &= E_n(\bar{x}) - E_n(\bar{x}) = 0. \end{aligned}$$

Per il teorema di Rolle[4] (o del valor medio), $G'(z)$ ha $n+1$ zeri distinti; analogamente, $G^{(k)}(z)$ ha $n+2-k$ zeri distinti. Pertanto la derivata $(n+1)$-esima di $G(z)$ si annulla in un unico punto ξ_x. Inoltre, la derivata $(n+1)$-esima di E_n coincide con la derivata $(n+1)$-esima di f, poiché p_n ha grado n, mentre la derivata $(n+1)$-esima di ω_{n+1} è $(n+1)!$. Si ottiene quindi

$$G^{(n+1)}(z) = f^{(n+1)}(z) - \frac{(n+1)!E_n(\bar{x})}{\omega_{n+1}(\bar{x})}$$

[4] Michel Rolle (1652-1719)

L'espressione precedente si annulla per $z = \xi_x$, quindi si ricava la relazione cercata per $E_n(x)$. ∎

Corollario

Come corollario, se si conosce una stima della derivata $(n+1)$-esima di $f(x)$, si ottiene immediatamente la seguente stima dell'errore:

$$|f(x) - p_n(x)| \leq \sup_{\xi \in I} \left| \frac{f^{(n+1)}(\xi)}{(n+1)!} \right| |\omega_{n+1}(x)|.$$

Un'ulteriore caratterizzazione si ottiene nel caso di nodi equispaziati, cioè quando $x_{i+1} - x_i = h$. In tal caso si può esplicitare la seguente stima, non ottimale, del polinomio $\omega_{n+1}(x)$ per $x \in (a, b)$:

$$|\omega_{n+1}(x)| = \left| \prod_{i=0}^{n} (x - x_i) \right| \leq (b-a)^{n+1}.$$

ma si può migliorare facilmente con la seguente osservazione. Se il punto x cade nell'intervallo $[x_{i-1}, x_i]$ allora

$$\begin{aligned} 0 &\leq |x - x_i| \leq h, \\ h &\leq |x - x_{i\pm1}| \leq 2h, \\ 2h &\leq |x - x_{i\pm2}| \leq 3h, \\ &\dots \end{aligned}$$

così possiamo scrivere

$$|\omega_{n+1}(x)| \leq h(2h)(3h)\cdots(nh)((n+1)h) = (n+1)!h^{n+1}. \tag{6.6}$$

In casi particolari si può fare di meglio. Ad esempio per $n = 1$ abbiamo

$$|\omega_2(x)| = |(x - x_0)(x - x_1)| = |x^2 - (a+b)x + ab|,$$

essendo una parabola è facile trovare il massimo in $[a, b]$ ottenendo con alcuni passaggi facili $|\omega_2(x)| \leq h^2/4$ che è migliore di $2h^2$ che si ottiene con la 6.6.

Con questa osservazione si può migliorare un poco la stima 6.6 osservando che se il punto x cade nell'intervallo $[x_{i-1}, x_i]$ allora

$$|x - x_{i-1}| \cdot |x - x_i| \leq \frac{h^2}{4}, \quad \Longrightarrow \quad |\omega_{n+1}(x)| \leq \frac{(n+1)!}{4} h^{n+1}. \tag{6.7}$$

Usando la stima 6.7 si ottiene per i nodi equispaziati

$$|E_n(x)| \leq \sup_{\xi \in I} \left| \frac{f^{(n+1)}(\xi)}{(n+1)!} \right| |\omega_{n+1}(x)| \leq \sup_{\xi \in I} |f^{(n+1)}(\xi)| \frac{h^{n+1}}{4}$$

Bisogna stare attenti a non concludere che il limite per $n \to \infty$ dell'espressione dell'errore sia zero. Infatti se derivata $n+1$-esima di $f(x)$ cresce più velocemente di h^{n+1}, allora la convergenza non è garantita. Un famoso controesempio (di Runge[5]) mostra che tale limite può addirittura essere infinito (vedi note su esempio di Runge)

[5] Carl David Tolmé Runge (1856-1927)

In sintesi

- Weierstrass garantisce che ogni funzione continua su un compatto sia approssimabile uniformemente da polinomi; Bernstein lo rende costruttivo.
- Dati $n+1$ nodi **distinti**, il polinomio interpolante di grado $\leq n$ esiste ed è **unico**: è la condizione di Haar (unisolvenza).
- L'errore di interpolazione è $\dfrac{f^{(n+1)}(\zeta)}{(n+1)!}\prod_i (t-t_i)$: dipende dalla regolarità di f **e** dalla posizione dei nodi.
- Il teorema di Faber avverte che non esiste una successione di nodi che faccia convergere l'interpolazione per **ogni** funzione continua: aumentare il grado non è di per sé una strategia.



7. Interpolazione di Lagrange

Il metodo più noto per costruire un polinomio che interpola un insieme di punti è quello dei **polinomi interpolatori di Lagrange**.

La costruzione di questo polinomio può essere ottenuta attraverso una dimostrazione alternativa rispetto a quella che si basa sul determinante di Vandermonde[1], comunemente utilizzata per dimostrare l'esistenza e l'unicità del polinomio interpolatore.

Questa dimostrazione, inoltre, fornisce l'algoritmo stesso per l'interpolazione secondo il metodo di Lagrange.

Theorem 7.1 — Lagrange. Sia $f(x)$ una funzione sull'intervallo $[a, b]$ e siano $\{x_i\}_{i=0}^n$ punti distinti di $[a, b]$, allora esiste un unico polinomio $p_n(x) \in \mathbb{R}_n[x]$ tale che

$$p_n(x_i) = f(x_i), \qquad \forall\, i = 1, 2, \dots, n.$$

I punti $\{x_i\}_i$ sono detti nodi di interpolazione e $p_n(x)$ si dice polinomio interpolatore di Lagrange nei nodi $\{x_i\}_i$.

Proof. Come prima cosa si dimostra l'unicità di questo polinomio, l'esistenza si dimostra esibendo il polinomio cercato. Siano dunque $p(x)$ e $q(x)$ due polinomi di grado n che soddisfano

$$p(x_i) = f(x_i) \qquad e \qquad q(x_i) = f(x_i), \qquad i = 0, 1, 2, \dots, n.$$

Allora il polinomio $p(x) - q(x)$ ha anche grado n ed ha come gli $n + 1$ punti x_i, infatti $p(x_i) - q(x_i) = f(x_i) - f(x_i) = 0$ e quindi deve essere il polinomio nullo, pertanto $p(x) \equiv q(x)$.

Costruiamo ora il polinomio interpolatore, sia $\mathcal{L}_i \in \mathbb{R}[x]$ definito come $\mathcal{L}_i(x_j) = \delta_{ij}$, dove δ_{ij} è il delta di Kronecker[2], ovvero una funzione che vale 1 se e solo se $i = j$ e zero in tutti gli altri casi. Senza perdita di generalità si può supporre $n \geq 1$. Tale polinomio $\mathcal{L}_i$ prende il nome di polinomio caratteristico di Lagrange, ed è facilmente scrivibile come

$$\mathcal{L}_i(x) = \prod_{\substack{0 \leq k \leq n \\ k \neq i}} \frac{x - x_k}{x_i - x_k}.$$

Ad esempio se ci sono solo quattro nodi $\{x_0, x_1, x_2, x_3\}$, il polinomio $\mathcal{L}_2(x)$ sarà definito come

[1] Alexandre-Théophile Vandermonde (1735-1796)
[2] Leopold Kronecker (1823-1891)

$$\mathcal{L}_2(x) = \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1} \cdot \frac{x - x_3}{x_2 - x_3},$$
$$\mathcal{L}_2(x_0) = \frac{x_0 - x_0}{x_2 - x_0} \cdot \dots = 0,$$
$$\mathcal{L}_2(x_2) = \frac{x_2 - x_0}{x_2 - x_0} \cdot \frac{x_2 - x_1}{x_2 - x_1} \cdot \frac{x_2 - x_3}{x_2 - x_3} = 1.$$

Il grafico di $\mathcal{L}_2(x)$ per $x_i = \{1, 1.5, 3, 4\}$ è disegnato in Figura 7.1.

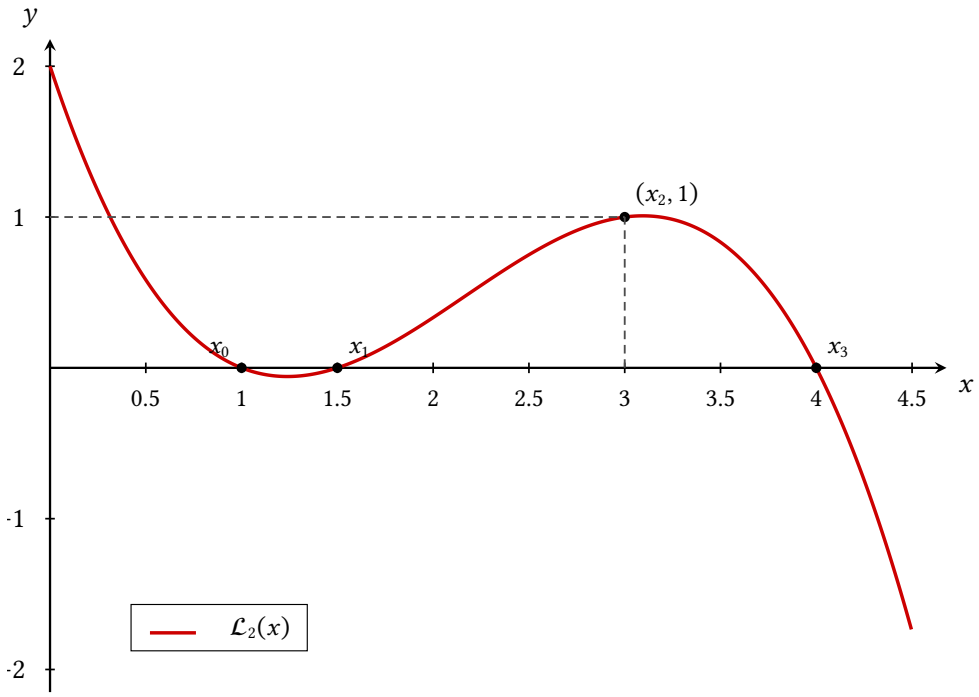


Figure 7.1: Il polinomio caratteristico di Lagrange $\mathcal{L}_2(x)$ per i quattro nodi $x_0 = 1$, $x_1 = 1.5$, $x_2 = 3$ e $x_3 = 4$.

Si sfrutta questa costruzione per scrivere il polinomio interpolatore di Lagrange nel modo seguente:

$$p_n(x) = \sum_{i=0}^{n} f(x_i)\mathcal{L}_i(x) \tag{7.1}$$

che con l'ausilio dell'esempio precedente è un polinomio che vale $f(x_i)$ nel nodo x_i, questo polinomio ha il giusto grado e quindi è il polinomio cercato. ■

Un altro modo per scrivere un generico polinomio caratteristico di Lagrange è usare il polinomio

$$\omega_{n+1}(x) = \prod_{i=0}^{n}(x - x_i) \in \mathbb{R}_{n+1}[x]$$
$$\omega'_{n+1}(x) = \sum_{i=0}^{n}\prod_{\substack{j=0 \\ j\neq i}}^{n}(x - x_j)$$
$$= (x - x_1)(x - x_2)\cdots(x - x_n)$$
$$+ (x - x_0)(x - x_2)\cdots(x - x_n)$$
$$+ \dots (x - x_0)(x - x_1)\cdots(x - x_{n-1}).$$

Il polinomio caratteristico $\mathcal{L}_i(x)$ si riscrive in modo più compatto come

$$\mathcal{L}_i(x) = \frac{\omega_{n+1}(x)}{(x - x_i)\omega'_{n+1}(x_i)}.$$

In pratica, la derivata di $\omega_{n+1}(x)$ valutata su un nodo x_i non è altro che il polinomio

$$\prod_{j \neq i}^{n} (x_i - x_j)$$

Infatti di tutti gli addendi di $\omega'_{n+1}(x)$ sopravvive solo quello che non contiene $(x - x_i)$ (che azzera tutto il termine).

▪ **Example 7.1** Supponiamo di voler interpolare i dati $\{(-1, 1), (2, 7), (0, 1)\}$, e svolgiamolo con il metodo di Lagrange. Abbiamo bisogno di calcolare i polinomi caratteristici $\mathcal{L}_0$, $\mathcal{L}_1$ ed $\mathcal{L}_2$:

$$\begin{aligned}
\mathcal{L}_0(x) &= \frac{x - x_1}{x_0 - x_1}\frac{x - x_2}{x_0 - x_2} &&= \frac{x}{-1}\frac{x-2}{-1-2} &&= \frac{x(x-2)}{3} \\
\mathcal{L}_1(x) &= \frac{x - x_0}{x_1 - x_0}\frac{x - x_2}{x_1 - x_2} &&= \frac{x+1}{1}\frac{x-2}{0-2} &&= -\frac{(x+1)(x-2)}{2} \\
\mathcal{L}_2(x) &= \frac{x - x_0}{x_2 - x_0}\frac{x - x_1}{x_2 - x_1} &&= \frac{x+1}{2+1}\frac{x}{2} &&= \frac{x(x+1)}{6}
\end{aligned}$$

Ora basta calcolare il polinomio interpolatore usando la 7.1, ovvero

$$\begin{aligned}
p_n(x) = p_2(x) &= \sum_{i=0}^{n} f(x_i) L_i(x) \\
&= 1 \cdot \frac{x(x-2)}{3} + 1 \cdot \frac{(x+1)(x-2)}{-2} + 7 \cdot \frac{x(x+1)}{6} \\
&= \left(\frac{1}{3} - \frac{1}{2} + \frac{7}{6}\right) x^2 + \left(-\frac{2}{3} + 1 - \frac{1}{2} + \frac{7}{6}\right) x + 1 \\
&= x^2 + x + 1,
\end{aligned}$$

che è proprio il polinomio calcolato con il metodo del sistema lineare.

▪

7.1 Derivazione pratica della interpolazione di Lagrange

L'interpolazione di Lagrange[3] permette di costruire un polinomio di grado n che passa esattamente per un insieme di punti dati. I polinomi di Lagrange sono definiti come:

$$\ell_k(x) = \prod_{\substack{i=0 \\ i \neq k}}^{n} (x - x_i),$$

Questi polinomi hanno una proprietà fondamentale:

$$\ell_k(x_i) = \begin{cases} 0 & i \neq k \\ \text{non nullo} & i = k \end{cases}$$

Da questa proprietà possiamo costruire i polinomi di Lagrange normalizzati, detti $\mathcal{L}_k(x)$, come:

$$\mathcal{L}_k(x) = \frac{\ell_k(x)}{\ell_k(x_k)}$$

[3] Joseph-Louis Lagrange (1736-1813)

Tali polinomi soddisfano:

$$\mathcal{L}_k(x_i) = \begin{cases} 0 & i \neq k \\ 1 & i = k \end{cases}$$

Osservazione

I polinomi di Lagrange possono essere espressi in forma compatta utilizzando il polinomio ausiliario $\omega_{n+1}(x)$ di grado $n+1$:

$$\mathcal{L}_k(x) = \frac{\omega_{n+1}(x)}{(x - x_k)\omega'_{n+1}(x_k)}$$

dove $\omega_{n+1}(x)$ è definito come:

$$\omega_{n+1}(x) = (x - x_0)(x - x_1)\cdots(x - x_n),$$
$$\omega'_{n+1}(x) = \sum_{k=0}^{n} \prod_{\substack{i=0 \\ i\neq k}}^{n} (x - x_i).$$

e la derivata $\omega'_{n+1}(x_k)$, calcolata nel punto x_k, assume una forma particolarmente semplice:

$$\omega'_{n+1}(x_k) = \prod_{\substack{i=0 \\ i\neq k}}^{n} (x_k - x_i).$$

Definition 7.1 — Polinomi elementari di Lagrange. I polinomi di Lagrange $\mathcal{L}_k(x)$, detti anche polinomi elementari di Lagrange, hanno la forma esplicita:

$$\mathcal{L}_k(x) = \frac{\omega_{n+1}(x)}{(x - x_k)\omega'_{n+1}(x_k)} = \prod_{\substack{i=0 \\ i\neq k}}^{n} \left(\frac{x - x_i}{x_k - x_i} \right).$$

Questi polinomi sono fondamentali nell'interpolazione poiché soddisfano la proprietà δ_{ik}, ossia $L_k(x_i) = \delta_{ik}$, con δ_{ik} simbolo di Kronecker.

Si ha quindi che la matrice dei coefficienti del problema di interpolazione soddisfa $\boldsymbol{M} = \boldsymbol{I}$ ed il problema della interpolazione si riduce a $a_k = f(x_k), \quad k = 0, 1 \ldots n$.

Polinomio interpolante

Utilizzando i polinomi $\mathcal{L}_k(x)$, possiamo scrivere il polinomio interpolante di grado n che passa per i punti $(x_i, f(x_i))$ come:

$$\boxed{p(x) = \sum_{k=0}^{n} f(x_k)\mathcal{L}_k(x) = \sum_{k=0}^{n} f(x_k) \frac{\omega_{n+1}(x)}{(x - x_k)\omega'_{n+1}(x_k)}}$$

7.1.1 Stabilità

In questa sezione vedremo che il processo di interpolazione con nodi equispaziati non è stabile rispetto alla variazione dei dati. Consideriamo un tipo di errore dato dall'inesattezza della misurazione del valore della funzione, ad esempio indichiamo con $\hat{f}(x_i)$ il valore misurato effettivamente (quindi con errori di arrotondamento, misurazione, ecc.) di $f(x_i)$. Se interpoliamo questi dati otterremo un polinomio $\hat{p}_n$, ci chiediamo quanto sia grande la differenza tra i due polinomi. Si ha, usando il metodo di interpolazione di Lagrange, che

$$\begin{aligned}\max_{[a,b]} |\hat{p}_n(x) - p_n(x)| &= \max_{[a,b]} \left| \sum_{i=0}^{n} \left(\hat{f}(x_i) - f(x_i) \right) \mathcal{L}_i(x) \right| \\ &\leq \max \sum_{i=0}^{n} |\mathcal{L}_i(x)| \cdot \max_{0 \leq i \leq n} |\hat{f}(x_i) - f(x_i)|, \\ &\leq \Lambda_n \max_{0 \leq i \leq n} |\hat{f}(x_i) - f(x_i)|.\end{aligned}$$

La costante $\Lambda_n = \max \sum_{i=0}^{n} |\mathcal{L}_i(x)|$ è detta *costante di Lebesgue*[4] che dipende dai nodi di interpolazione. Quindi a piccole variazioni sui dati corrispondono piccole variazioni sul polinomio interpolatore fatta salva la *costante di Lebesgue*. Se essa è piccola, anche la variazione è piccola, ma se è grande può causare grandi differenze. Questa costante ha il ruolo di numero di condizionamento del problema. Nel caso dell'interpolazione di Lagrange su nodi equispaziati si può approssimare bene con

$$\Lambda_n \approx \frac{2^{n+1}}{ne(\ln(n) + \gamma)},$$

dove e è il numero di Nepero e $\gamma \approx 0.5477$ è la costante di Eulero-Mascheroni.

7.2 Polinomio di miglior approssimazione

Fissati i nodi e il grado n, accanto al polinomio interpolante ha senso considerare il *polinomio di miglior approssimazione* $p_n^\star(x)$, cioè quello che rende minimo l'errore in norma infinito fra tutti i polinomi di grado n:

$$E_n^\star := \|f - p_n^\star\|_\infty \leq \|f - p_n\|_\infty.$$

Il polinomio di miglior approssimazione in generale **non** interpola f nei nodi, quindi non è calcolabile con le formule viste finora. La costante di Lebesgue permette però di confrontare i due errori: se $E_n := \|f - p_n\|_\infty$ è l'errore del polinomio interpolante, vale

$$E_n \leq (1 + \Lambda_n) E_n^\star. \tag{7.2}$$

L'interpolazione sui nodi dati è quindi lontana dalla miglior approssimazione al più di un fattore $1 + \Lambda_n$: minimizzare Λ_n significa scegliere i nodi in modo che il problema sia ben condizionato.

Non è possibile spingere Λ_n sotto una soglia che cresce con n. Il teorema di Natanson[5] afferma infatti che, per *ogni* scelta dei nodi, esiste una costante $C > 0$ tale che

$$\Lambda_n > \frac{2}{\pi} \ln(n + 1) - C, \tag{7.3}$$

e quindi $\Lambda_n \to \infty$ per $n \to \infty$. Nessuna successione di nodi rende l'interpolazione polinomiale convergente per tutte le funzioni continue: è la sostanza del teorema di Faber[6], discusso nel capitolo sul fenomeno di Runge[7]. Il meglio che si può fare è avvicinarsi alla crescita logaritmica di 7.3, ed è quello che fanno i nodi di Chebyshev[8].

[4] Costante di Lebesgue da Henri Léon Lebesgue
[5] Isidor Pavlovič Natanson (1906-1964)
[6] Georg Faber (1877-1966)
[7] Carl David Tolmé Runge (1856-1927)
[8] Pafnutij L'vovič Čebyšëv (1821-1894)

In sintesi

- La base di Lagrange ℓ_i rende il polinomio interpolante immediato da scrivere, ma costosa da valutare e da aggiornare.
- La **costante di Lebesgue** Λ_n è il numero di condizionamento dell'interpolazione: lega l'errore sui dati all'errore sul polinomio.
- Su nodi equispaziati Λ_n cresce esponenzialmente (fenomeno di Runge), sui nodi di Chebyshev solo come $\log n$.
- Il polinomio interpolante non è quello di miglior approssimazione, ma su buoni nodi gli è vicino a meno del fattore $1 + \Lambda_n$.



8. Interpolazione di Newton

Un limite dell'approccio di Lagrange[1] è che, una volta calcolato il polinomio $p_n(x)$ che interpola i dati,

$$\{(x_i, y_i)\}_{i=0}^{n}$$

per aggiungere un ulteriore punto e calcolare $p_{n+1}(x)$ occorre ripetere l'intero procedimento: in generale, non è possibile ottenere direttamente p_{n+1} a partire da p_n. L'idea di Newton[2] consente invece di costruire il polinomio in modo incrementale, scrivendo

$$p_{n+1}(x) = p_n(x) + g(x)$$

per un polinomio $g(x)$ da determinare. Se valutiamo p_{n+1} sui nodi $x_0, x_1, \dots, x_{n+1}$ avremo che:

$$\begin{aligned}
y_0 &= p_{n+1}(x_0) = p_n(x_0) + g(x_0) && \implies g(x_0) = 0 \\
y_1 &= p_{n+1}(x_1) = p_n(x_1) + g(x_1) && \implies g(x_1) = 0 \\
&\vdots && \quad\vdots \\
y_n &= p_{n+1}(x_n) = p_n(x_n) + g(x_n) && \implies g(x_n) = 0 \\
y_{n+1} &= p_{n+1}(x_{n+1}) = p_n(x_{n+1}) + g(x_{n+1}) && \implies g(x_{n+1}) = y_{n+1} - p_n(x_{n+1})
\end{aligned}$$

Dunque $g(x)$ è un polinomio che deve valere zero sui nodi $x_0, \dots, x_n$ e quindi si scriverà come

$$g(x) = c\omega_{n+1}(x) = c \cdot (x - x_0)(x - x_1) \cdot \dots \cdot (x - x_n)$$

Allora

$$p_{n+1}(x) = p_n(x) + c\omega_{n+1}(x)$$

con $\omega_{n+1}(x_i) = 0$ per $i = 0, \dots, n$. Il grado complessivo di $p_{n+1}(x)$ è proprio $n + 1$ per il contributo di $\omega_{n+1}(x)$. Valutando $p_{n+1}(x)$ sul nodo x_{n+1} si determina facilmente la costante c:

$$\begin{aligned}
p_{n+1}(x_{n+1}) &= p_n(x_{n+1}) + c\omega_{n+1}(x_{n+1}) = y_{n+1} \\
a_{n+1} = c &= \frac{y_{n+1} - p_n(x_{n+1})}{\omega_{n+1}(x_{n+1})}.
\end{aligned} \tag{8.1}$$

[1] Joseph-Louis Lagrange (1736-1813)
[2] Isaac Newton (1642-1727)

▪ **Example 8.1** Vediamo un esempio di costruzione incrementale col metodo di Newton. Consideriamo i dati $\{(0,1),(2,3),(1,0)\}$. I passi da fare sono:

- $n = 0$. Il primo polinomio interpolante è $p_0(x) = y_0 = 1$.
- $n = 1$. Il polinomio $p_1(x) = p_0(x) + c\omega_1(x)$. Si ha immediatamente che $\omega_1(x) = x - x_0 = x$ e quindi

$$\begin{aligned} p_1(x) &= p_0(x) + c\omega_1(x) = 1 + cx = 3 \quad \Longrightarrow \quad c = 1 \\ &= 1 + x \end{aligned}$$

- $n = 2$. Il polinomio $p_2(x) = p_1(x) + c\omega_2(x)$ con

$$\omega_2(x) = (x - x_0)(x - x_1) = x(x-2) = x^2 - 2x.$$

Allora si calcola c a partire da p_2 mediante

$$\begin{aligned} p_2(x) &= p_1(x) + c\omega_2(x) = (1 + x) + c(x^2 - 2x) \\ p_2(x_2) &= p_1(1) + c\omega_2(1) = (1 + 1) + c(1^2 - 2 \cdot 1) \quad \Longrightarrow \; c = 2 \\ p_2(x) &= 1 + x + 2(x^2 - 2x) = 1 - 3x + 2x^2 \end{aligned}$$

Il polinomio interpolante è quindi $1 - 3x + 2x^2$.

▪

Riprendiamo in modo più formale e schematico quanto appena visto. Per prima cosa definiamo il polinomio $\omega_k(x)$ di grado k come

$$\omega_k(x) := \prod_{j=0}^{k-1}(x - x_j).$$

Si nota immediatamente che $\omega_{k+1}(x) = (x - x_k)\omega_k(x)$ e la costruzione incrementale si può scrivere come

$$\begin{aligned} p_0(x) &= a_0 = f(x_0) = y_0 \\ p_1(x) &= p_0(x) + a_1\omega_1(x) = a_0 + a_1\omega_1(x) \\ p_2(x) &= p_1(x) + a_2\omega_2(x) = a_0 + a_1\omega_1(x) + a_2\omega_2(x) \\ &\vdots \\ p_n(x) &= p_{n-1}(x) + a_n\omega_n(x) = a_0 + a_1\omega_1(x) + \ldots + a_n\omega_n(x) \end{aligned}$$

La costruzione di Newton può essere formulata nuovamente come un sistema lineare, questa volta triangolare inferiore anziché rettangolare, come si vede dalla seguente espressione:

$$\begin{aligned} f(x_0) &= a_0 \\ f(x_1) &= a_0 + a_1\omega_1(x_1) \\ f(x_2) &= a_0 + a_1\omega_1(x_2) + a_2\omega_2(x_2) \\ f(x_3) &= a_0 + a_1\omega_1(x_3) + a_2\omega_2(x_3) + a_3\omega_3(x_3) \\ &\vdots \\ f(x_n) &= a_0 + a_1\omega_1(x_n) + a_2\omega_2(x_n) + \ldots + a_n\omega_n(x_n). \end{aligned}$$

Equivalentemente, in forma matriciale si ha $\boldsymbol{M}\boldsymbol{a} = \boldsymbol{b}$:

$$\begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ 1 & \omega_1(x_1) & 0 & \ldots & 0 \\ \vdots & & & \vdots & \\ 1 & \omega_1(x_n) & \omega_2(x_n) & \ldots & \omega_n(x_n) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_n) \end{bmatrix}. \tag{8.2}$$

Il determinante di $\boldsymbol{M}$ è dato dal prodotto dei polinomi ω_i e vale

$$\det \boldsymbol{M} = \omega_0(x_0)\omega_1(x_1)\cdots\omega_n(x_n) = \prod_{k=0}^{n}\omega_k(x_k) = \prod_{k=0}^{n}\prod_{j=0}^{k-1}(x_k - x_j).$$

Di conseguenza, nell'ipotesi fatta di avere nodi distinti, cioè $x_i \neq x_j$ se $i \neq j$, il determinante è non nullo ed il problema ammette un'unica soluzione.

Relazioni di ricorrenza

La ricorsione vista all'8.1 permette la costruzione incrementale del polinomio interpolante. Sia la $n+1$-upla di coefficienti $(a_0, a_1, \dots, a_n)$ una soluzione del problema di interpolazione. Allora gli $n+1$ polinomi

$$p_k(x) = a_0\omega_0(x) + a_1\omega_1(x) + \cdots + a_k\omega_k(x),$$

per $k = 0, 1, 2, \dots, n$ interpolano i nodi $(x_0, f(x_0)), (x_1, f(x_1)), \dots, (x_k, f(x_k))$.
Ovviamente, per $k = n$ ritroviamo il polinomio di interpolazione, cioè $p_n(x) = p(x)$.
Per $i \leq k$ si ha

$$\omega_{k+1}(x_i) = \omega_{k+2}(x_i) = \dots = \omega_n(x_i) = 0.$$

Infatti, se per esempio esaminiamo il primo di questi polinomi, quello con indice $k+1$, notiamo che

$$\omega_{k+1}(x_i) = (x_i - x_0)(x_i - x_1)\dots(x_i - x_k),$$

per cui, essendo $i \leq k$, uno dei binomi è sicuramente nullo. Lo stesso ragionamento si può ripetere anche per gli altri polinomi con indici $k+2$, $k+3$, $\dots$, n.
Il polinomio interpolatore calcolato nel nodo $ x_i$ si scrive formalmente come

$$\begin{aligned} p_k(x_i) &= \underbrace{a_0\omega_0(x_i) + a_1\omega_1(x_i) + \cdots + a_k\omega_k(x_i)}_{=p(x_i)} + \\ &+ \underbrace{a_{k+1}\omega_{k+1}(x_i) + \cdots + a_n\omega_n(x_i)}_{=0}, \\ &= p(x_i), \qquad i = 1, 2, \dots, k. \end{aligned}$$

Possiamo esprimere questa proprietà per mezzo delle seguenti *relazioni di ricorrenza* per $k \geq 0$

$$\begin{aligned} \omega_{k+1}(x) &= (x - x_k)\omega_k(x), \\ p_{k+1}(x) &= p_k(x) + a_{k+1}\omega_{k+1}(x). \end{aligned}$$

La condizione di interpolazione $p_{k+1}(x_{k+1}) = f(x_{k+1})$ permette di ricavare immediatamente un'espressione per a_{k+1}, cioè per il *coefficiente del termine di grado massimo* del polinomio $p_{k+1}(x)$

$$a_{k+1} = \frac{f(x_{k+1}) - p_k(x_{k+1})}{\omega_{k+1}(x_{k+1})}, \tag{8.3}$$

che sostituita nelle relazioni di ricorrenza permette di scrivere un algoritmo di calcolo per il polinomio di interpolazione.

Seguendo la tradizione, si usa indicare il termine a_k come la *differenza divisa* di ordine k e si scrive con un simbolo speciale

$$a_k = f[x_0, x_1, \dots, x_k] \tag{8.4}$$

e come conseguenza, il polinomio interpolatore di Newton si scrive come

$$p_n(x) = \sum_{k=0}^{n} f[x_0, x_1, \dots, x_k]\omega_k(x).$$

La differenza divisa $f[x_0, x_1, \dots, x_k]$ è il coefficiente del monomio di grado massimo (k) del polinomio interpolatore dei primi $k+1$ punti, e non dipende dall'ordine con cui sono considerati i nodi. Come conseguenza, il polinomio di interpolazione non cambia se si cambia la sequenza dei nodi. Il modo efficiente di usare le differenze divise è quello di costruirne la tavola in modo ricorsivo. Siano x_i i nodi, la prima differenza divisa è $f[x_i] = f(x_i)$, e i passi successivi dipendono solo dai precedenti.

8.1 Proprietà delle differenze divise

Dalla relazione 8.3 e dalla definizione 8.4 si intuisce che c'è una ricorsione che permette di calcolare un coefficiente a partire dai precedenti:

$$\begin{aligned}
a_0 &= f[x_0] := f(x_0) \\
a_1 &= f[x_0, x_1] := \frac{f[x_1] - f[x_0]}{x_1 - x_0} \\
a_2 &= f[x_0, x_1, x_2] := \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_1} \\
&\vdots \\
a_k &= f[x_0, x_1, \dots, x_k] := \frac{f[x_1, \dots, x_k] - f[x_0, x_1, \dots, x_{k-1}]}{x_k - x_0}.
\end{aligned}$$

Il coefficiente di grado massimo del polinomio interpolatore $p_n(x)$ è proprio l'ultima differenza divisa $f[x_0, \dots, x_n]$. Infatti il polinomio ω_n è monico in quanto prodotto di fattori della forma $(x - x_i)$. In generale si possono definire le differenze divise per una sequenza di punti qualsiasi:

$$f[x_i, x_{i+1}, \dots, x_{i+k}] := \frac{f[x_{i+1}, \dots, x_{i+k}] - f[x_i, \dots, x_{i+k-1}]}{x_{i+k} - x_i} \tag{8.5}$$

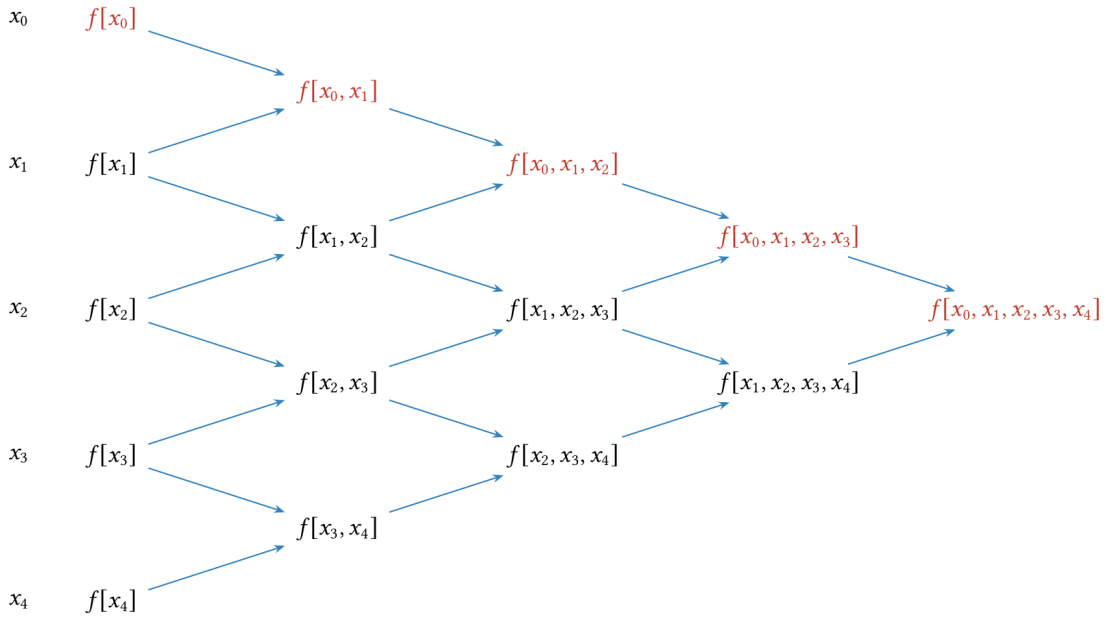
e da questa definizione si vede che per calcolare una differenza di ordine k servono le differenze divise di ordine $j = 0, \dots, k-1$. Questa dipendenza si esprime bene con la seguente tavola delle differenze divise:

La tavola si calcola una colonna alla volta da sinistra verso destra, utilizzando i valori precedenti per calcolare quelli nuovi. Osservando con attenzione la tavola, si nota che i coefficienti del polinomio interpolatore si trovano sul lato superiore del triangolo delle differenze divise. Notiamo anche che è necessario calcolare tutta la tavola, perché, ad esempio, per calcolare $f[x_0, x_1, x_2]$ servono prima $f[x_0, x_1]$, $f[x_1, x_2]$ ed in seconda istanza $f[x_0]$, $f[x_1]$ ed $f[x_2]$.

▪ **Example 8.2** Interpoliamo i punti $\{(-1, 1), (2, 7), (0, 1)\}$. La costruzione della tavola delle differenze divise è la seguente:

$$\begin{array}{lll}
x_0 & f[x_0] & \\
 & & f[x_0, x_1] \\
x_1 & f[x_1] & & f[x_0, x_1, x_2] \\
 & & f[x_1, x_2] \\
x_2 & f[x_2] &
\end{array}$$

che calcolata sui nodi specifici dell'esempio dà



in rosso: il lato superiore del triangolo, cioè i coefficienti del polinomio di Newton

Figure 8.1: La tavola delle differenze divise. Ogni freccia va da una differenza divisa a quella di ordine successivo che la usa: la colonna di ordine $k+1$ si ottiene tutta dalla colonna di ordine k, quindi la tavola si costruisce una colonna alla volta da sinistra verso destra.

$$
\begin{array}{cc}
-1 & \mathbf{1} \\
 & f[x_0, x_1] = \dfrac{1-7}{-1-2} = \mathbf{2} \\
2 & 7 \qquad\qquad f[x_0, x_1, x_2] = \dfrac{2-3}{-1-0} = \mathbf{1} \\
 & f[x_1, x_2] = \dfrac{7-1}{2-0} = 3 \\
0 & 1
\end{array}
$$

I coefficienti in rosso sul triangolo della tavola sono i coefficienti del polinomio interpolatore cercato, che si scriverà come combinazione lineare dei polinomi della base ω_k:

$$
\begin{aligned}
&\omega_0(x) = 1 \\
&\omega_1(x) = (x+1) \\
&\omega_2(x) = \omega_1(x)(x-2) = (x+1)(x-2) = x^2 - x - 2.
\end{aligned}
$$

Il polinomio interpolatore è quindi

$$
\begin{aligned}
p_2(x) &= \mathbf{1}\omega_0(x) + \mathbf{2}\omega_1(x) + \mathbf{1}\omega_2(x) \\
&= 1 + 2(x+1) + 1(x^2 - x - 2) = 1 + x + x^2.
\end{aligned}
$$

▪

▪ **Example 8.3** Vediamo un altro esempio con più punti da interpolare:

$$\{(0, 13), (-1, 25), (-2, 51), (1, 3), (3, 61)\}.$$

La tavola delle differenze divise è:

$$
\begin{array}{llllll}
0 & f[x_0] = \mathbf{13} & & & & \\
 & & \frac{13-25}{0-(-1)} = \mathbf{*12} & & & \\
-1 & f[x_1] = 25 & & \frac{-12-26}{0-(-2)} = \mathbf{7} & & \\
 & & \frac{25-51}{-1-(-2)} = -26 & & \frac{7-5}{0-1} = \mathbf{*2} & \\
-2 & f[x_2] = 51 & & \frac{-26-(-16)}{-1-1} = 5 & & \frac{-2-1}{0-3} = \mathbf{1} \\
 & & \frac{51-3}{-2-1} = -16 & & \frac{5-9}{-1-3} = 1 & \\
1 & f[x_3] = 3 & & \frac{-16-29}{-2-3} = 9 & & \\
 & & \frac{3-61}{1-3} = 29 & & & \\
3 & f[x_4] = 61 & & & &
\end{array}
$$

Il polinomio di interpolazione è allora

$$p_4(x) = 13 - 12\omega_1(x) + 7\omega_2(x) - 2\omega_3(x) + 1\omega_4(x).$$

▪

8.2 Invarianza per permutazioni dei nodi

La differenza divisa $f[x_0, \dots, x_k]$ non dipende dall'ordine in cui si considerano i nodi di interpolazione. Sia dunque $\sigma \in S_{k+1}$ una permutazione degli interi $0, 1, 2, \dots, k$, dimostriamo che

$$f[x_0, \dots, x_k] = f[x_{\sigma(0)}, \dots, x_{\sigma(k)}]$$

Si procede per induzione: la base dell'induzione è per $k = 1$ e dalla definizione si ha che

$$\begin{aligned} f[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} &= \frac{-(f(x_0) - f(x_1))}{-(x_0 - x_1)} \\ &= \frac{f(x_0) - f(x_1)}{x_0 - x_1} = f[x_1, x_0]. \end{aligned}$$

Per l'ipotesi induttiva, si può assumere che l'argomento è valido per $k > 1$ e resta da dimostrare il passo induttivo che porta da $k - 1$ a k. Possiamo scrivere il polinomio interpolatore $p_k(x)$ che deve coincidere sulla sequenza di punti $x_0, x_1, \dots, x_k$ sia sulla sequenza permutata $x_{\sigma(0)}, \dots, x_{\sigma(k)}$:

$$\begin{aligned} p_k(x) &= f[x_0]\omega_0(x) + \dots + f[x_0, \dots, x_k]\omega_k(x) \\ &= f[x_{\sigma(0)}]\hat{\omega}_0(x) + \dots + f[x_{\sigma(0)}, \dots, x_{\sigma(k)}]\hat{\omega}_k(x). \end{aligned}$$

Il termine di grado massimo (ovvero k) di $p_k(x)$ è dato dal termine di grado massimo di $\omega_k(x)$ e $\hat{\omega}_k(x)$ ed è esattamente x^k per entrambi, in altre parole i polinomi ω_j (e $\hat{\omega}_j$) sono polinomi monici. Allora per il principio di sovrapposizione dei polinomi, il coefficiente che moltiplica x^k di $p_k(x)$ deve essere lo stesso, e quindi $f[x_0, \dots, x_k] = f[x_{\sigma(0)}, \dots, x_{\sigma(k)}]$.

8.3 Formula ricorrente

In questo paragrafo dimostriamo la formula ricorrente 8.5 per le differenze divise che permette di costruire la tavola delle differenze divise. Questa dimostrazione si basa sull'invarianza per permutazioni mostrata in precedenza. Scriviamo il polinomio $p(x)$ che interpola i nodi

$$x_0, \dots, x_{k-2}, x_{k-1}, x_k$$

e il polinomio $\hat{p}(x)$ che interpola gli stessi nodi ma con gli ultimi due scambiati, ovvero

$$x_0, \dots, x_{k-2}, x_k, x_{k-1}$$

In questa situazione abbiamo

$$\begin{aligned}
\omega_0(x) &= 1 \\
\omega_1(x) &= \omega_0(x)(x - x_0) \\
&\vdots \\
\omega_{k-2}(x) &= \omega_{k-3}(x)(x - x_{k-3}) \\
\omega_{k-1}(x) &= \omega_{k-2}(x)(x - x_{k-2}) \\
\omega_k(x) &= \omega_{k-1}(x)(x - x_{k-1})
\end{aligned}$$

$$\begin{aligned}
\hat{\omega}_0(x) &= 1 \\
\hat{\omega}_1(x) &= \omega_0(x)(x - x_0) \\
&\vdots \\
\hat{\omega}_{k-2}(x) &= \hat{\omega}_{k-3}(x)(x - x_{n-3}) \\
\hat{\omega}_{k-1}(x) &= \hat{\omega}_{k-2}(x)(x - x_{k-2}) \\
\hat{\omega}_k(x) &= \hat{\omega}_{k-1}(x)(x - x_{k-2})
\end{aligned}$$

e quindi

$$\omega_j(x) = \hat{\omega}_j(x), \qquad j = 0, 1, \dots, k-1$$

Ovviamente abbiamo già visto che il polinomio interpolante è lo stesso, cioè $p(x) = \hat{p}(x)$ scriviamolo per esteso:

$$\begin{aligned}
\hat{p}(x) = {} & f[x_0]\omega_0(x) \\
& + f[x_0, x_1]\omega_1(x) \\
& \vdots \\
& + f[x_0, \dots, x_{k-2}]\omega_{k-2}(x) \\
& + f[x_0, \dots, x_{k-2}, x_{k-1}]\omega_{k-1}(x) \\
& + f[x_0, \dots, x_{k-1}, x_k]\omega_k(x)
\end{aligned}$$

$$\begin{aligned}
p_k(x) = {} & f[x_0]\omega_0(x) \\
& + f[x_0, x_1]\omega_1(x) \\
& \vdots \\
& + f[x_0, \dots, x_{k-2}]\hat{\omega}_{k-2}(x) \\
& + f[x_0, \dots, x_{k-2}, x_k]\hat{\omega}_{k-1}(x) \\
& + f[x_0, \dots, x_k, x_{k-1}]\hat{\omega}_k(x).
\end{aligned}$$

Sottraendo membro a membro, i primi $k-2$ addendi sono uguali in entrambi i casi, rimane solamente

$$\begin{aligned}
0 = {} & f[x_0, \dots, x_{k-2}, x_{k-1}]\omega_{k-1}(x) + f[x_0, \dots, x_{k-1}, x_k]\omega_k(x) \\
& - f[x_0, \dots, x_{k-2}, x_k]\hat{\omega}_{k-1}(x) - f[x_0, \dots, x_k, x_{k-1}]\hat{\omega}_k(x).
\end{aligned} \tag{8.6}$$

L'espressione precedente si può semplificare osservando che

$$\begin{aligned}\omega_{k-1}(x) &= \hat{\omega}_{k-1}(x),\\ \omega_k(x) &= \omega_{k-1}(x)(x - x_{k-1}),\\ \hat{\omega}_k(x) &= \omega_{k-1}(x)(x - x_{k-2}),\end{aligned}$$

e la 8.6 diventa, dopo aver diviso per il fattore comune ω_{k-1},

$$\begin{aligned}0 = f[x_0, \dots, x_{k-2}, x_{k-1}] + f[x_0, \dots, x_{k-1}, x_k](x - x_{k-1})\\ - f[x_0, \dots, x_{k-2},\ x_k] - f[x_0, \dots, x_k, x_{k-1}](x - x_k).\end{aligned}$$

Per la proprietà di invarianza per permutazione, l'ultimo addendo dell'espressione precedente diventa $f[x_0, \dots, x_{k-1}, x_k](x - x_k)$ e si può mettere a fattore comune con il secondo addendo, così da ottenere:

$$\begin{aligned}f[x_0, \dots, x_{k-2}, x_{k-1}] - f[x_0, \dots, x_{k-2},\ x_k] &= f[x_0, \dots, x_{k-1}, x_k](x - x_k - (x - x_{k-1}))\\ &= f[x_0, \dots, x_k](x_{k-1} - x_k).\end{aligned}$$

Ora possiamo raccogliere il termine $f[x_0, \dots, x_k]$ e scrivere

$$f[x_0, \dots, x_k] = \frac{f[x_0, \dots, x_{k-2}, x_{k-1}] - f[x_0, \dots, x_{k-2},\ x_k]}{x_{k-1} - x_k}. \tag{8.7}$$

Questa è già una formula ricorrente valida, tuttavia non è scritta nel modo standard, per ricondurla alla 8.5 basta operare una nuova permutazione dei nodi. Chiamiamo per comodità y la sequenza di nodi $x_0, \dots, x_{k-2}$, $a = x_{k-1}$ e $b = x_k$, allora l'espressione 8.7 si riscrive come

$$f[y, a, b] = \frac{f[y, a] - f[y, b]}{a - b},$$

e permutando l'ordine possiamo scrivere

$$f[a, y, b] = \frac{f[a, y] - f[y, b]}{a - b}$$

che è proprio la ricorrenza 8.5.

8.3.1 Riassunto sulle differenze divise

Il sistema lineare che definisce il problema di interpolazione secondo il metodo di Newton

$$\begin{cases} f(x_0) = a_0\omega_{k+1} \\ f(x_1) = a_0 + a_1\omega_1(x_1)\omega k + 1 \\ f(x_2) = a_0 + a_1\omega_1(x_2) + a_2\omega_2(x_2)\omega_{k+1} \\ \quad\vdots \\ f(x_n) = a_0 + a_1\omega_1(x_n) + \cdots + a_n\omega_n(x_n) \end{cases}$$

è triangolare inferiore e può essere risolto esplicitamente con la tecnica delle *sostituzioni in avanti*.

La soluzione del sistema lineare può essere scritta come

$$\begin{aligned}a_0 &= f(x_0),\\ a_1 &= \frac{f(x_1) - f(x_0)}{\omega_1(x_1)},\\ a_2 &= \frac{f(x_2) - f(x_0) - \omega_1(x_1)\dfrac{f(x_1) - f(x_0)}{\omega_1(x_1)}}{\omega_2(x_2)},\\ &\vdots\end{aligned}$$

Si noti che ogni a_i dipende dai nodi di interpolazione $(x_j, f(x_j))$ per $j = 0, 1, \dots i$ e non dai successivi.

> **Osservazione**
>
> Questo tipo di dipendenza non deve stupire, perché è ancora un modo per descrivere le relazioni di ricorrenza di cui si è già parlato.

Indichiamo questa dipendenza con il simbolo speciale (notare le parentesi quadre)

$$a_j = f[x_0, x_1, \dots, x_j],$$

che si chiama *differenza divisa* di ordine j.
Tramite le differenze divise possiamo scrivere il polinomio interpolante come segue

$$\boxed{p(x) = \sum_{k=0}^{n} f[x_0, x_1, \dots, x_k]\omega_k(x)}$$

Le differenze divise godono di una serie notevole di proprietà interessanti.

- *Proprietà 1*: La differenza divisa $f[x_0, x_1, \dots, x_k]$ è il coefficiente del termine di grado massimo k del polinomio interpolatore che interpola i $k + 1$ nodi di ascisse x_0, x_1, ..., x_k.
- *Proprietà 2:* La differenza divisa $f[x_0, x_1, \dots, x_k]$ non dipende dall'ordine con cui si prendono i nodi di interpolazione, cioè

  $$f[x_0, x_1, \dots, x_k] = f[x_{i_0}, x_{i_1}, \dots, x_{i_k}],$$

  dove $i_0, i_1, \dots, i_k$ è una qualunque permutazione dei numeri $0, 1, \dots, k$.
- *Proprietà 3* Definizione *ricorsiva* delle differenze divise di ogni ordine

  $$f[x_i] = f(x_i), \qquad f[\dots, x_i, x_j] = \frac{f[\dots, x_i] - f[\dots, x_j]}{x_i - x_j}.$$

  Il procedimento ricorsivo è quello illustrato in Figura 8.1.

8.3.2 Costi computazionali

In conclusione, abbiamo visto tre modi di calcolare il polinomio interpolatore dei dati, quello naïve, quello di Lagrange e quello di Newton, rispettivamente:

$$\begin{aligned}
p_n(x) &= a_0 + a_1 x + \dots + a_n x^n && \text{[sistema lineare]}\\
p_n(x) &= b_0 \mathcal{L}_0(x) + b_1 \mathcal{L}_1(x) + \dots + b_n \mathcal{L}_n(x) && \text{[Lagrange]}\\
p_n(x) &= c_0 + c_1 \omega_1(x) + \dots + c_n \omega_n(x) && \text{[Newton]}
\end{aligned}$$

In tutti e tre i casi, il polinomio è espresso come combinazione lineare di una base. Il primo approccio porta a un polinomio semplice da scrivere, ma richiede la soluzione di un sistema lineare e non consente di aggiornare efficientemente il risultato quando si aggiunge un nodo. Il costo computazionale è quindi quello della soluzione del sistema, pari a circa $\mathcal{O}(n^3)$.

Il metodo di Lagrange impiega polinomi base $\mathcal{L}_i(x)$ la cui espressione è articolata; consente tuttavia di scrivere direttamente la soluzione, pur non offrendo una costruzione incrementale. Per calcolare il polinomio interpolatore occorre formare $n + 1$ polinomi $\mathcal{L}_i(x)$; ciascuno richiede $n + 1$ somme e n prodotti sia al numeratore sia al denominatore. Il costo complessivo è dell'ordine di $\mathcal{O}(4n^2)$.

Infine, il metodo di Newton consente una costruzione incrementale con un costo computazionale contenuto. L'uso delle differenze divise rende il procedimento ancora più efficiente.

Può però verificarsi una perdita di cifre significative per cancellazione, quando, con l'infittirsi dei nodi di interpolazione, si sottraggono valori molto vicini.

Il calcolo con il metodo delle differenze divise ha una complessità di $\mathcal{O}(\frac{3}{2}n^2)$, ma ha complessità lineare se da p_n si vuol passare a p_{n+1} e non serve ripercorrere tutto il calcolo precedentemente fatto per p_n.

In generale una funzione f si può approssimare come una combinazione lineare del tipo

$$f(x) = a_0 b_0(x) + a_1 b_1(x) + \ldots + a_n b_n(x) + E_n(x),$$

che è una combinazione lineare degli elementi della base $\{b_i\}_{i=0}^n$ più un errore E_n. Abbiamo visto il caso in cui la base è una base polinomiale canonica $1, x, x^2, \ldots$, il caso della base di Lagrange $\mathcal{L}_i(x)$ e la base di Newton $\omega_i(x)$. Ci sono anche altre basi polinomiali, ma si possono scegliere anche altre forme funzionali, molto usate sono basi di funzioni esponenziali, trigonometriche o funzioni razionali.

8.4 L'algoritmo di Aitken-Neville

Per presentare l'algoritmo di Aitken-Neville [3]

introduciamo l'insieme formato da $n+1$ punti di $\mathbb{R} \times \mathbb{R}$

$$\boldsymbol{I} = \{(x_i, f(x_i)) \mid i = 0, 1, 2, \ldots, n,\ x_i \neq x_j\},$$

e tutti i suoi sottoinsiemi

$$I_{i,i+1,i+2,\ldots,j} = \{(x_k, f(x_k)) \mid k = i, i+1, i+2, \ldots, j\},$$

Ad ogni sottoinsieme $I_{0,1,2,\ldots,k}$ associamo il polinomio $p_{0,1,2,\ldots,k}$ di grado k che interpola i punti dell'insieme

$$p_{0,1,2,\ldots,k}(x_j) = f(x_j) \qquad j = 0, 1, \ldots, k.$$

Remark 8.1 Conoscendo i due polinomi interpolatori

- $p_{0,1,2,\ldots,k-1}(x)$ e
- $p_{1,2,\ldots,k-1,k}(x)$

possiamo facilmente costruire il polinomio interpolatore

- $p_{0,1,2,\ldots,k-1,k}(x)$,

che è dato dalla formula

$$p_{0,1,\ldots,k}(x) = \frac{(x_k - x)p_{0,1,\ldots,k-1}(x) + (x - x_0)p_{1,2,\ldots,k}(x)}{x_k - x_0}$$

Per $s = 1, 2, \ldots, k-1$ vale ovviamente

$$\begin{aligned} p_{0,1,\ldots,k}(x_s) &= \frac{(x_k - x_s)p_{0,1,\ldots,k-1}(x_s) + (x_s - x_0)p_{1,2,\ldots,k}(x_s)}{x_k - x_0}, \\ &= \frac{(x_k - x_s)f(x_s) + (x_s - x_0)f(x_s)}{x_k - x_0}, \\ &= f(x_s), \end{aligned}$$

Nel caso $s = 0$ abbiamo

[3] Alexander Craig Aitken 1895–1967

$$p_{0,1,\dots,k}(x_0) = \frac{(x_k - x_0)p_{0,1,\dots,k-1}(x_0) + (x_0 - x_0)p_{1,2,\dots,k}(x_0)}{x_k - x_0},$$
$$= \frac{(x_k - x_0)f(x_0)}{x_k - x_0} = f(x_0).$$

Nel caso $s = k$ abbiamo

$$p_{0,1,\dots,k}(x_k) = \frac{(x_k - x_k)p_{0,1,\dots,k-1}(x_k) + (x_k - x_0)p_{1,2,\dots,k}(x_k)}{x_k - x_0},$$
$$= \frac{(x_k - x_0)f(x_k)}{x_k - x_0} = f(x_k).$$

8.4.1 Osservazioni finali sull'algoritmo di Aitken-Neville

I polinomi $p_i(x)$ con $i = 0, 1, \dots, n$ sono semplicemente le costanti $f(x_0), f(x_1), \dots f(x_n)$; cioè:

$$p_i(x) = f(x_i), \qquad i = 0, 1, \dots, n$$

Analogamente

$$p_{i,i+1}(x) = \frac{(x_{i+1} - x)p_i(x) + (x - x_i)p_{i+1}(x)}{x_{i+1} - x_i},$$

per $i = 0, 1, \dots, n-1$.

La formula ricorrente si usa in modo efficiente per calcolare il valore del polinomio interpolatore in un punto generico $\bar{x}$ senza mai costruirne i coefficienti: si riempie una tavola triangolare colonna per colonna, partendo dai valori nei nodi.

Algoritmo di Aitken–Neville

Dati: nodi $x_0, \dots, x_n$ distinti; valori $f(x_0), \dots, f(x_n)$; punto $\bar{x}$
Restituisce: $p_{0,1,\dots,n}(\bar{x})$, il valore in $\bar{x}$ del polinomio interpolatore

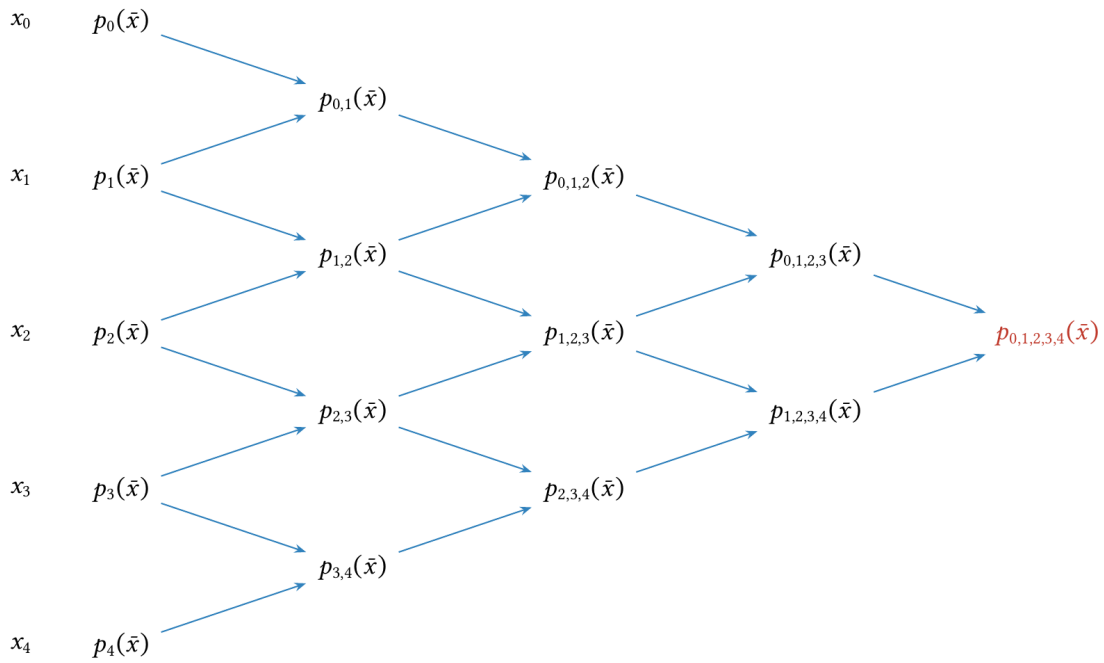
1 **per** $i = 0, 1, \dots, n$ **fai**
2 $p_i(\bar{x}) \leftarrow f(x_i)$ ▷ *colonna di ordine 0: le costanti*
3 **per** $k = 1, 2, \dots, n$ **fai**
  ▷ *la colonna di ordine k dalle due di ordine $k-1$*
4 **per** $i = 0, 1, \dots, n-k$ **fai**
5 $p_{i,\dots,i+k}(\bar{x}) \leftarrow \dfrac{(x_{i+k} - \bar{x})\, p_{i,\dots,i+k-1}(\bar{x}) + (\bar{x} - x_i)\, p_{i+1,\dots,i+k}(\bar{x})}{x_{i+k} - x_i}$
6 **restituisci** $p_{0,1,\dots,n}(\bar{x})$

Figure 8.2: L'algoritmo di Aitken–Neville. Ogni colonna contiene i polinomi di un grado, valutati in $\bar{x}$, e si ottiene come media pesata della colonna precedente: il grado cresce di uno a ogni colonna, fino all'unico polinomio che interpola tutti i nodi.

Lo schema di calcolo è lo stesso triangolo delle differenze divise (Figura 8.1), con i polinomi al posto delle differenze:

In sintesi

- La forma di Newton costruisce il polinomio **incrementalmente**: aggiungere un nodo costa $\mathcal{O}(n)$ e non richiede di rifare i conti.
- Le differenze divise sono invarianti per permutazione dei nodi e si calcolano con una ricorrenza a triangolo, in $\mathcal{O}(n^2)$.



in rosso: il valore cercato, cioè il polinomio interpolatore su tutti i nodi valutato in $\bar{x}$

Figure 8.3: Lo schema di Aitken–Neville. Ogni freccia va da un polinomio a quello di grado successivo che lo usa; a differenza della tavola di Newton, qui il risultato è il solo elemento in fondo a destra, e tutta la tavola serve per arrivarci.

- Vale $f[t_0, \dots, t_n] = \dfrac{f^{(n)}(\zeta)}{n!}$: le differenze divise sono derivate discrete, e questo spiega la forma dell'errore.
- L'algoritmo di **Aitken-Neville** valuta il polinomio in un punto senza costruirne i coefficienti: preferibile quando serve un solo valore.



9. Esempio di Runge

9.1 Approssimazione di funzioni continue

Presentiamo di seguito alcuni teoremi classici sull'approssimazione delle funzioni continue mediante polinomi. Il primo afferma che i polinomi sono *densi* nello spazio delle funzioni continue: ogni funzione continua su un intervallo $[a, b]$ può essere approssimata con precisione arbitraria da un polinomio.

Teorema di approssimazione di Weierstrass (Weierstraß, 1885)

Sia $f : [a, b] \to \mathbb{C}$ una funzione continua.
Allora per ogni $\varepsilon > 0$ esiste un polinomio $p(x)$ tale che

$$|f(x) - p(x)| \leq \varepsilon, \qquad \forall x \in [a, b]$$

Nel caso di funzioni a valori reali Bernstein da una dimostrazione costruttiva, ovvero da un algoritmo per la costruzione del polinomio approssimante.

Teorema di approssimazione di Bernstein (Sergej Natanovič Bernštejn, 1911)

Sia $f : [0, 1] \to \mathbb{R}$ una funzione continua e per $n \in \mathbb{N}$ l'n-esimo *polinomio di Bernstein* sia definito come

$$B_n(x, f) = \sum_{k=0}^{n} \binom{n}{k} x^k (1 - x)^{n-k} f\left(\frac{k}{n}\right)$$

allora la successione di polinomi $B_n(x, f)$ converge uniformemente a $f(x)$ ovvero per ogni $\varepsilon > 0$ esiste un $m \in \mathbb{N}$ tale che per ogni $n \geq m$ vale

$$|f(x) - B_n(x, f)| \leq \varepsilon, \qquad \forall x \in [0, 1]$$

Nel teorema di Bernstein il polinomio ha una costruzione ad hoc; nel caso dei polinomi *interpolanti*, le cose non vanno così bene, come mostra il teorema seguente:

Teorema di interpolazione di Faber (Georg Faber, 1914)

Data una qualunque successione di nodi in un intervallo chiuso $[a, b]$ esiste sempre una funzione continua $f : [a, b] \to \mathbb{R}$ che genera una sequenza di polinomi $\{p_n(x)\}_{n\in\mathbb{N}}$ di interpolazione che non convergono uniformemente a $f(x)$ in $[a, b]$. Ovvero

$$\lim_{n\to\infty} \left(\max_{x\in[a,b]} |f(x) - p_n(x)| \right) \neq 0$$

G.Faber, *Über die interpolatorische Darstellung stetiger Funktionen*, Jahresber. Deut. Math. Verein. 23 (1914), 192-210.

Per ottenere il fenomeno descritto dal teorema di Faber occorre costruire funzioni particolari. Se invece la funzione è sufficientemente regolare, per esempio se appartiene a $C^1[a, b]$ (cioè è derivabile), il comportamento è migliore. Per funzioni molto regolari si può ottenere una stima precisa dell'errore di interpolazione:

Errore di interpolazione (Augustin-Louis Cauchy, 1840)

Sia $f \in C^{n+1}[a, b]$ e siano $\{x_0, x_1, \dots, x_n\} \subset [a, b]$ punti distinti. Allora il polinomio interpolante $p_n(x)$ (ovvero $p_n(x_k) = f(x_k)$) soddisfa

$$f(x) - p_n(x) = f^{(n+1)}(\zeta)\frac{(x - x_0)(x - x_1)\cdots(x - x_n)}{(n + 1)!},$$

dove $\zeta \in I(x, x_0, x_1, \dots, x_n)$ ed $I(x, x_0, x_1, \dots, x_n)$ è il più piccolo intervallo aperto contenente i punti $x, x_0, x_1, \dots, x_n$.

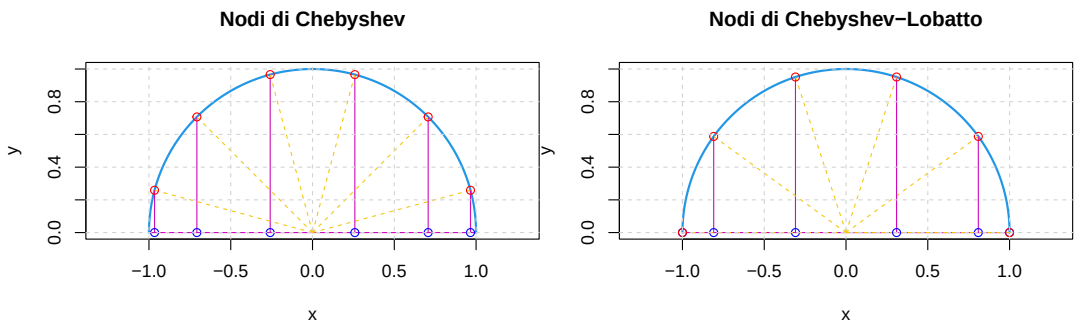
Per migliorare l'interpolazione non abbiamo il controllo di $f^{(n+1)}(\zeta)$ ma possiamo cercare di controllare il termine $(x-x_0)(x-x_1)\cdots(x-x_n)$ scegliendo opportunamente i nodi di interpolazione. Ci sono distribuzioni di nodi che minimizzano $\max_{x\in I(x_0,x_1,\dots,x_n)} |(x - x_0)(x - x_1)\cdots(x - x_n)|$ e sono i nodi di Chebyshev.

Successioni di nodi

Dato l'intervallo $[a, b]$ è possibile definire le sequenti successioni di nodi

- *nodi equispaziati*
$$x_k = a + \frac{k}{n}(b - a), \qquad k = 0, 1, \dots, n$$
- *nodi di Chebyshev*[1]
$$x_k = \frac{a + b}{2} + \frac{b - a}{2}\cos\left(\pi\frac{2k + 1}{2n + 2}\right) \qquad k = 0, 1, \dots, n$$
- *nodi di Chebyshev-Lobatto*
$$x_k = \frac{a + b}{2} + \frac{b - a}{2}\cos\left(\frac{k\,\pi}{n}\right) \qquad k = 0, 1, \dots, n$$

i nodi di Chebyshev (e Chebyshev-Lobatto) si possono interpretare come la proiezioni di nodi equispaziati sulla semi-circonferenza sull'asse delle x.



Per entrambe le distribuzioni si sa stimare la costante di Lebesgue[2], che cresce solo logaritmicamente invece che esponenzialmente come nel caso equispaziato:

$$\begin{aligned} \Lambda_n &< \frac{2}{\pi}\left(\ln n + \gamma + \ln\frac{8}{\pi}\right) + \frac{\pi}{72n^2} && \text{[Chebyshev-Lobatto]} \\ \Lambda_n &< \frac{2}{\pi}\left(\ln(n+1) + \gamma + \ln\frac{8}{\pi}\right) + \frac{\pi}{72(n+1)^2} && \text{[Chebyshev]} \end{aligned} \tag{9.1}$$

dove γ è la costante di Eulero-Mascheroni. Confrontando con il limite inferiore di Natanson[3] si vede che questi nodi sono ottimali a meno di una costante additiva, e per 7.2 l'errore di interpolazione resta entro un fattore $O(\ln n)$ dalla miglior approssimazione.

Scegliendo i nodi di Chebyshev per una funzione $C^1[a,b]$ abbiamo il seguente risultato

Secondo teorema di Bernstein

Per ogni funzione $f \in C^1[a,b]$ allora se $p_n(x)$ è il polinomio interpolante per gli $n+1$ nodi di Chebyshev allora l'errore

$$E_n = \max_{x\in[a,b]} |f(x) - p_n(x)|$$

soddisfa

$$\lim_{n\to\infty} E_n = 0$$

ovvero la successione di polinomi converge uniformemente a $f(x)$.

9.2 Controesempio di Runge

Anche se la funzione è molto regolare (ad esempio $C^\infty[a,b]$) sono necessari i nodi di Chebyshev per avere la convergenza. Il seguente controesempio lo trovate in

- **Carl Runge**[4], *Über empirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten*, Zeitschrift für Mathematik und Physik, 1901, vol. 46, pages 224-243

Innanzitutto costruiamo la funzione da interpolare $f(x) = 1/(1+x^2)$:

```
rfun = function ( x ) { 1 / ( 1 + x ^ 2 ) }
```

e salviamo un certo numero di punti per plottare la funzione

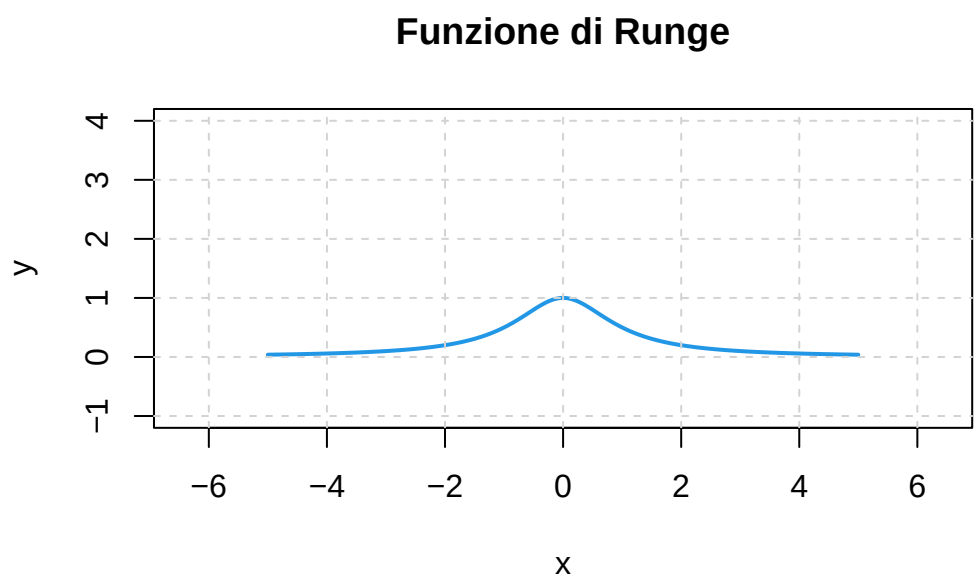
[1] Pafnutij L'vovič Čebyšëv (1821-1894)
[2] Henri Léon Lebesgue (1875-1941)
[3] Isidor Pavlovič Natanson (1906-1964)
[4] Carl David Tolmé Runge (1856-1927)

```
x = seq(from = -5, to = 5, by = 0.01)
```

plotta la funzione da interpolare

```
y = rfun( x );

# salva il plot in una funzione per riusarla piu avanti
plot_runge = function() {
  plot(
    x,y,type="l",lwd=2,col=4,
    main="Funzione di Runge",
    xlim=c(-5,5), ylim=c(-1,4),
    xlab="x", ylab="y", asp=1
  )
  # Turn on grid
  grid(
    lty = 2,           # Grid line type
    col = "lightgray", # Grid line color
    lwd = 1
  )
}
plot_runge()
```



costruiamo un primo polinomio interpolante

```
library(polynom)

# punti di interpolazione
x1 = seq(from = -5, to = 5, by = 2)
y1 = rfun( x1 )

# costruzione del polinomio interopolante
poly1 = poly.calc(x1, y1)

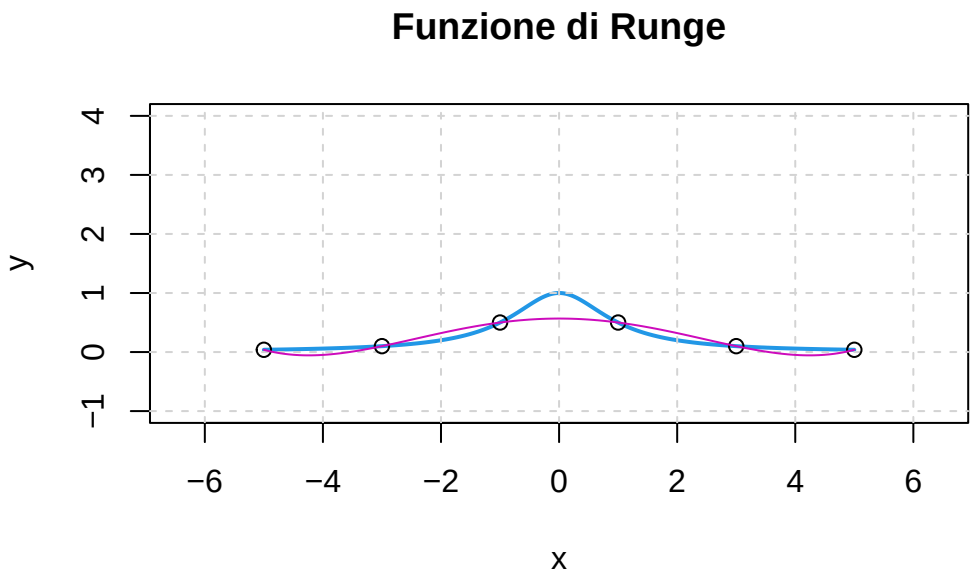
# polinomio interpolante come funzione
fun1 = as.function(poly1)

# valutazione del polinomio
yy1 = fun1(x)

# plotta funzione di Runge
```

```
plot_runge()

# punti di interpolazione
points(x1, y1)
lines(x, yy1, type = "l", lty = 1, col=6)
```



ne aggiungiamo un secondo

```
library(polynom)

# punti di interpolazione
x2 = seq(from = -5, to = 5, by = 1)
y2 = rfun( x2 )

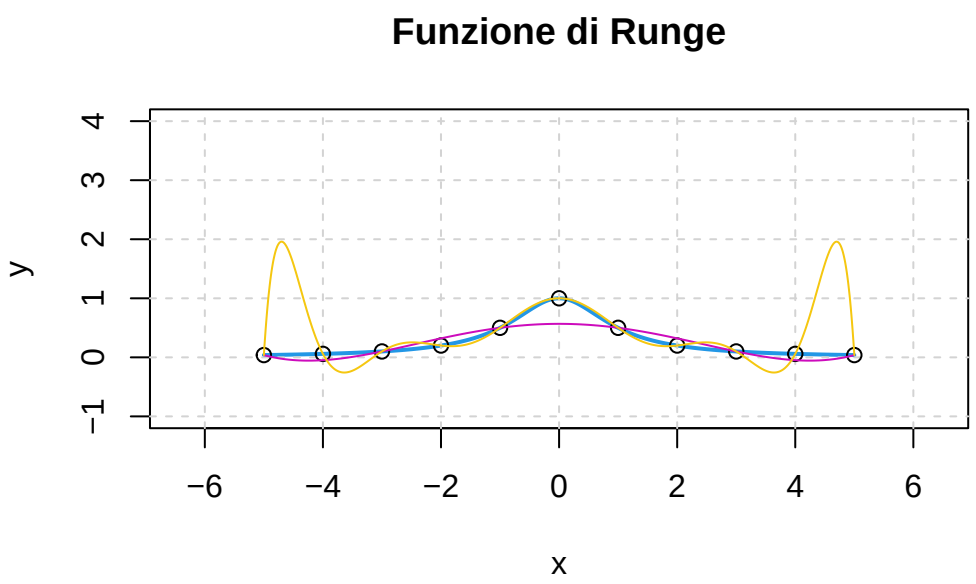
# costruzione del polinomio interopolante
poly2 = poly.calc(x2, y2)

# polinomio interpolante come funzione
fun2 = as.function(poly2)

# valutazione del polinomio
yy2 = fun2(x)

# plotta funzione di Runge
plot_runge()

# punti di interpolazione
points(x2, y2)
lines(x, yy1, type = "l", lty = 1, col=6)
lines(x, yy2, type = "l", lty = 1, col=7)
```

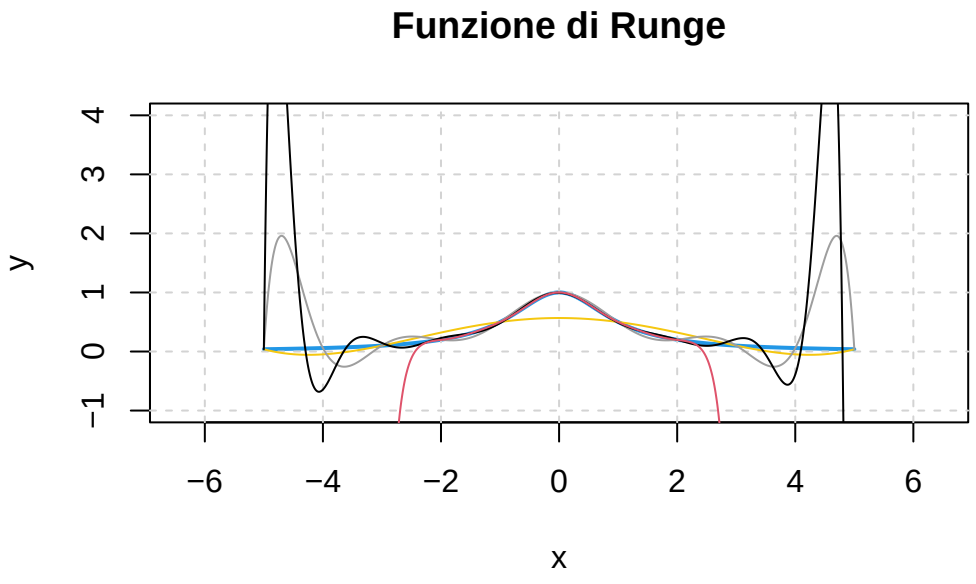


aggiunge una serie di polinomi di grado crescente

```
library(polynom)

# plotta funzione di Runge
plot_runge()

kk = 1;
for ( delta in c(2,1,0.7,0.5) ) {
  xk    = seq(from = -5, to = 5, by = delta)
  yk    = rfun( xk )
  polyk = poly.calc(xk, yk)
  funk  = as.function(polyk)
  yyk   = funk(x)
  lines(x, yyk, type = "l", lty = 1, col=6+kk)
  kk = kk+1
}
```

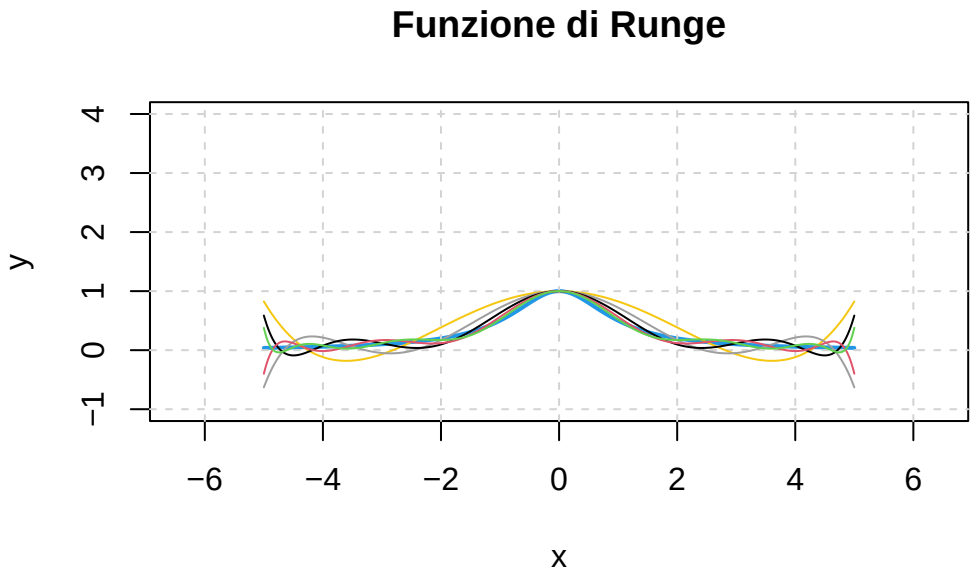


si vede chiaramente che man mano che aumentano i punti di interpolazione i polinomi approssimano sempre peggio la funzione $f(x) = 1/(1 + x^2)$. Ma usando i nodi di Chebyshev si vede che le cose vanno meglio:

```
library(polynom)

# plotta funzione di Runge
plot_runge()

kk = 1;
for ( n in c(4,6,8,10,12) ) {
  tk    = seq(from = 1, to = n+1, by = 1)/(n+2)
  xk    = 5*cos(pi*tk)
  yk    = rfun( xk )
  polyk = poly.calc(xk, yk)
  funk  = as.function(polyk)
  yyk   = funk(x)
  lines(x, yyk, type = "l", lty = 1, col=6+kk)
  kk = kk+1
}
```

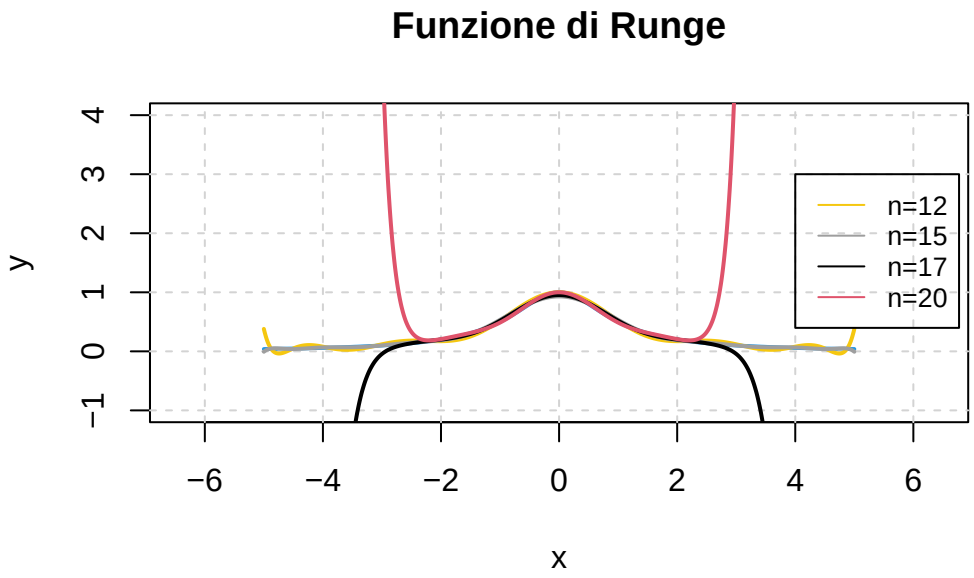


ma solo fino a un certo punto. Se il grado del polinomio diventa troppo elevato, gli errori in virgola mobile si accumulano e possono rendere del tutto inattendibile la valutazione:

```
library(polynom)

# plotta funzione di Runge
plot_runge()

kk = 1;
for ( n in c(12,15,17,20) ) {
  tk    = seq(from = 1, to = n+1, by = 1)/(n+2)
  xk    = 5*cos(pi*tk)
  yk    = rfun( xk )
  polyk = poly.calc(xk, yk)
  funk  = as.function(polyk)
  yyk   = funk(x)
  lines(x, yyk, type = "l", lwd=2, lty = 1, col=6+kk)
  kk = kk+1
}
# Add a legend
legend(4, 3, legend=c("n=12", "n=15", "n=17", "n=20"),
       col=c(7,8,9,10), lty=1, cex=0.8)
```



9.3 Conclusioni

- I polinomi sono densi nelle funzioni continue ma approssimare una funzione continua con polinomi interpolanti non è facile.
- Anche se la funzione è regolare i nodi di interpolazione vanno scelti con cura per ridurre l'errore (vedi teorema di Cauchy)
- Usando i nodi di Chebyshev, per le funzioni regolari si ottiene teoricamente la convergenza uniforme del polinomio interpolante alla funzione da approssimare.
- In ogni caso la valutazione di un polinomio è instabile e sopra gradi moderatamente alti (grado 10) non si dovrebbe usare un polinomio per approssimare una funzione.
  - Si possono usare polinomi a tratti (spline) di grado basso per approssiamare funzioni continue
  - Per migliorare la valutazione di un polinomio di grado si possono usare polinomi speciali (polinomi ortogonali con le loro ricorrenze)

IV Integrazione numerica

10 Integrazione numerica 117
10.1 Problema dell'integrazione numerica 117
10.2 Strategia *interpolatoria* 118
10.3 Classificazione (largamente incompleta) 118
10.4 Formule di Newton-Cotes 119
10.5 Accuratezza 120
10.6 Metodo dei Coefficienti Indeterminati 121
10.7 Stima dell'errore di integrazione 122

11 Integrazione composita 129
11.1 Stima a priori degli intervalli 130
11.2 Trapezi e Simpson composite 131
11.3 Scelta a priori del numero di sottointervalli . . . 133

12 Formule di quadratura 137
12.1 Principali formule di quadratura 137

13 Stime a posteriori 145
13.1 Dalla stima a posteriori a un algoritmo adattativo 146
13.2 Perché cercare di equidistribuire l'errore 147
13.3 Esperimenti numerici: stima ed errore effettivo 148



10. Integrazione numerica

10.1 Problema dell'integrazione numerica

Data una funzione $f : \mathbb{R} \mapsto \mathbb{R}$ e l'intervallo chiuso e limitato $[a,\ b]$ si vuole stimare *numericamente* l'integrale

$$\int_a^b f(x)\,\mathrm{d}x$$

Si parla di *stima numerica* perché non si vuole calcolare il valore esatto dell'integrale, definito come differenza dei valori assunti dalla primitiva della funzione integranda f agli estremi di integrazione. Si vuole invece determinare un valore approssimato dell'integrale che dipenda soltanto dalla conoscenza della funzione integranda f, e non della sua primitiva.

Il problema è particolarmente interessante perché, anche quando l'integrale esiste ed è finito, la funzione integranda potrebbe non ammettere una primitiva esprimibile analiticamente, oppure la primitiva potrebbe essere molto complessa da determinare e costosa da calcolare. La funzione integranda f potrebbe persino non essere nota in forma analitica, ma soltanto tabulata in alcuni punti dell'intervallo $[a, b]$; in tal caso, cercarne una primitiva sarebbe di fatto impossibile.

Riprendendo la *vecchia idea* di interpretare l'integrale come una sorta di *somma generalizzata*, stimeremo il valore dell'integrale di f per mezzo di espressioni del tipo:

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{k=0}^{n} w_k\, f(x_k)$$

in cui la funzione da integrare è valutata in un insieme di punti $x_i \in [a,\ b]$ e l'integrale è approssimato mediante una somma *pesata*, con opportuni coefficienti w_k.

Definition 10.1 — Formula di quadratura. L'espressione della forma

$$\mathcal{I}(f; a, b) = \sum_{k=0}^{n} w_k\, f(x_k)$$

definita da $\{x_i,\, w_i\}_{i=0,1,\ldots,n}$ si chiama *formula di quadratura*. I punti $\{x_i\}_{i=0,1,\ldots,n}$ sono i *nodi* della formula di quadratura ed i coefficienti $\{w_i\}_{i=0,1,\ldots,n}$ sono i *pesi*.

Ricordiamo che la formula di quadratura fornisce una stima, non il valore esatto dell'integrale; comporta quindi un errore di approssimazione, che in alcune situazioni può essere significativo. Ci chiederemo quanto sia precisa una data formula di quadratura nella stima dell'integrale di una funzione generica e da quali fattori dipenda l'accuratezza del risultato numerico.

Per definire una formula di quadratura è necessario

- scegliere i nodi $\{x_i\}_{i=0,1,\dots,n}$ della formula di quadratura;
- scegliere i pesi $\{w_i\}_{i=0,1,\dots,n}$ della formula di quadratura.

Il criterio con cui si scelgono nodi e pesi è dunque fondamentale per determinare l'accuratezza di una formula di quadratura.

È inoltre ragionevole aspettarsi che la precisione del valore stimato dipenda anche dalla funzione da integrare. In altre parole, la stessa formula di quadratura può fornire un risultato molto accurato — o persino esatto — per alcune famiglie di funzioni e un risultato poco accurato in altri casi.

Questi argomenti saranno oggetto di discussione nel presente capitolo.

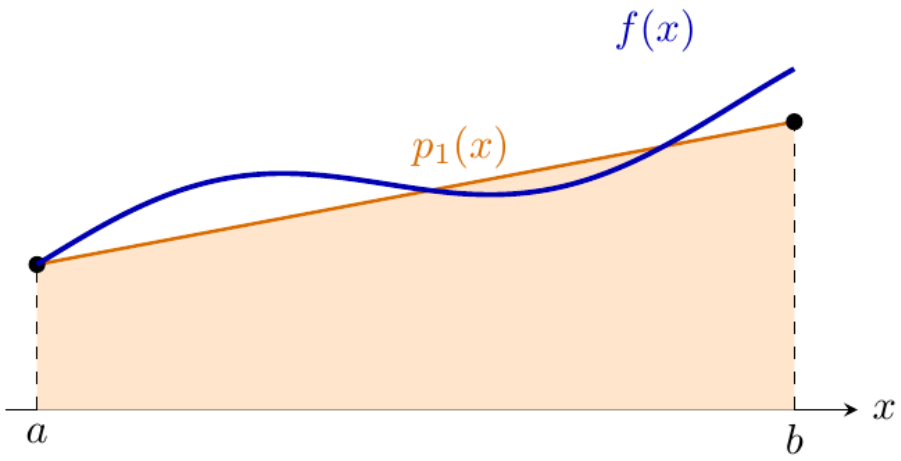


Figure 10.1: Schema geometrico di quadratura: quadratura_01

10.2 Strategia *interpolatoria*

(i) approssimiamo la funzione f con un polinomio interpolatore di grado n scegliendo con un *criterio da stabilire* $n+1$ nodi di interpolazione con ascisse distinte $x_i \in [a, b]$ per $i = 0, 1, \dots, n$; \

(ii) stimiamo l'integrale di f calcolando l'integrale del polinomio interpolante:

$$\int_a^b f(x)\,\mathrm{d}x \approx \int_a^b p_n(x)\,\mathrm{d}x.$$

Definition 10.2 — Formule di quadratura. Le formule di quadratura costruite con la strategia interpolatoria prendono il nome di *formule di quadratura interpolatorie*

10.3 Classificazione (largamente incompleta)

- **Formule di Newton-Cotes**
  i nodi di integrazione sono scelti *equidistanti* nell'intervallo $[a, b]$; distinguiamo tra
  - *formule aperte*: gli estremi a e b non sono compresi;
  - *formule chiuse*: $x_0 = a$ e $x_n = b$;
- **Formule Gaussiane**
  i nodi di integrazione sono eventualmente gli estremi dell'intervallo di integrazione e gli *zeri* di polinomi appartenenti a famiglie speciali. Citiamo, per esempio, i *polinomi ortogonali*

che producono le formule di *Gauss-Legendre*[1], di *Gauss-Lobatto*, di *Gauss-Radau*, etc. Nella nostra esposizione tratteremo solo il caso delle formule di Newton-Cotes.

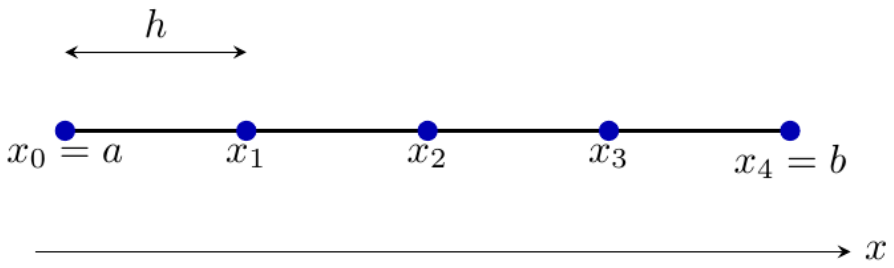


Figure 10.2: Schema geometrico di quadratura: quadratura_02

10.4 Formule di Newton-Cotes

I pesi si determinano immediatamente calcolando l'integrale del polinomio interpolatore. Esaminiamo il caso delle formule *chiuse*:

$$\begin{aligned} h &= \frac{b-a}{n}, && && \text{passo di integrazione,} \\ x_i &= a + i\,h, && i = 0, 1, \ldots, n, && \text{nodo } i, \\ x &= a + t\,h, && t \in [0,\ n], && \text{generico punto } x \in [a,\ b]. \end{aligned}$$

Sia $p_n(x)$ il polinomio interpolatore di Lagrange[2]

$$p_n(x) = f(x_0)\boldsymbol{L}_0(x) + f(x_1)\boldsymbol{L}_1(x) + \ldots + f(x_n)\boldsymbol{L}_n(x),$$

dove gli $\boldsymbol{L}_k(x)$ sono i polinomi elementari di Lagrange, definiti dalla relazione $\boldsymbol{L}_i(x_j) = \delta_{ij}$. Ricordiamo la definizione data nel capitolo sull'interpolazione,

$$\begin{aligned} \boldsymbol{L}_i(x) &= \frac{(x-x_0)\ldots(x-x_{i-1}(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)\ldots(x_i-x_{i-1})(x-x_{i+1})\ldots(x_i-x_n)} \\ &= \prod_{\substack{j=0 \\ j\neq i}}^{n} \left(\frac{x-x_j}{x_i-x_j}\right). \end{aligned}$$

Dato che possiamo scrivere per ogni indice i e j

$$\begin{aligned} x - x_j &= (a + t\,h) - (a + j\,h) = (t-j)\,h \\ x_i - x_j &= (a + i\,h) - (a + j\,h) = (i-j)\,h, \end{aligned}$$

si ha dalla definizione dei polinomi di Lagrange un'espressione particolare per nodi di interpolazione equidistanti

[1] Carl Friedrich Gauss (1777-1855) e Adrien-Marie Legendre (1752-1833)
[2] Joseph-Louis Lagrange (1736-1813)

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x &\approx \int_a^b p_n(x)\,\mathrm{d}x,\\ &= \int_a^b \sum_{j=0}^{n} f(x_j)\mathcal{L}_j(x)\,\mathrm{d}x,\\ &= \sum_{j=0}^{n} f(x_j)\int_a^b \mathcal{L}_j(x)\,\mathrm{d}x,\\ &= \sum_{j=0}^{n} f(x_j)\int_a^b \prod_{\substack{j=0\\ j\neq i}}^{n}\left(\frac{x-x_j}{x_i-x_j}\right)\mathrm{d}x,\\ &= \sum_{j=0}^{n} f(x_j)\,h\int_0^n \prod_{\substack{j=0\\ j\neq i}}^{n}\left(\frac{t-j}{i-j}\right)\mathrm{d}t.\end{aligned}$$

Si ottiene quindi la formula di quadratura

$$\mathcal{I}(f;a,b)=\sum_{k=0}^{n} w_k\,f(x_k)=\sum_{j=0}^{n} f(x_j)\,h\int_0^n \prod_{\substack{j=0\\ j\neq i}}^{n}\left(\frac{t-j}{i-j}\right)\mathrm{d}t.$$

definita dai nodi e dai pesi

$$x_i=a+i\,h,\quad i=0,1,\dots n,\qquad w_i=h\int_0^n \prod_{\substack{j=0\\ j\neq i}}^{n}\left(\frac{t-j}{i-j}\right)\mathrm{d}t.$$

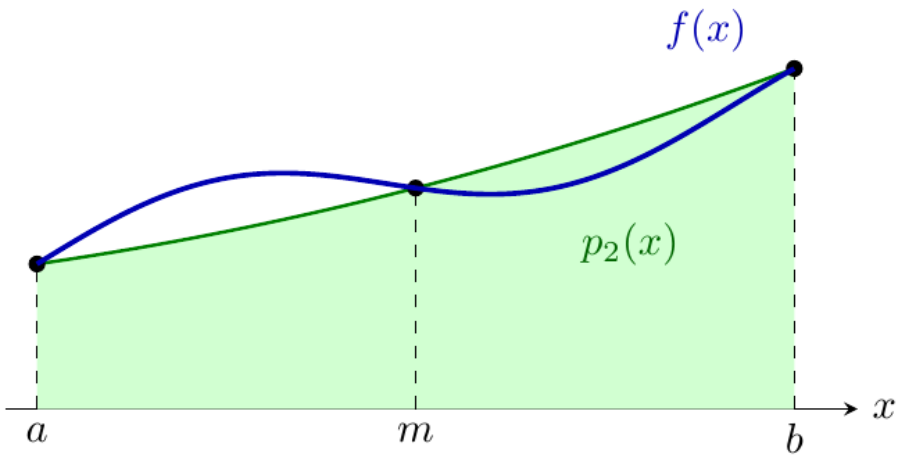


Figure 10.3: Schema geometrico di quadratura: quadratura_03

10.5 Accuratezza

Definition 10.3 — Errore di integrazione. Si chiama *errore di integrazione* la differenza tra il valore esatto dell'integrale ed il suo valore stimato con la formula di quadratura

$$\mathcal{E}(f;a,b)=\int_a^b f(x)\,\mathrm{d}x-\sum_{k=0}^{n} w_k\,f(x_k)$$

Ovviamente $\mathcal{E}(f; a, b)$ deve dipendere dalla funzione $f(x)$ e dall'intervallo di integrazione $[a,\ b]$.

Definition 10.4 — Ordine di accuratezza. L'*ordine di accuratezza* o di *precisione* della formula di quadratura è dato dal massimo grado dei polinomi che sono integrati esattamente dal metodo. Più correttamente diremo che una formula di quadratura

- ha ordine **almeno** k se integra **esattamente tutti** i polinomi di grado fino a k;
- ha ordine k se ha ordine almeno k ed **esiste almeno un** polinomio di grado $k+1$ che non è integrato *esattamente*.

10.6 Metodo dei Coefficienti Indeterminati

Il metodo dei Coefficienti Indeterminati permette di ricondurre il calcolo dei pesi delle formule di quadratura, una volta stabiliti i nodi, alla risoluzione di un problema algebrico.

Osserviamo che l'integrale di un polinomio $p_n(x)$ della forma

$$p_n(x) = a_n x^n + a_{n-1}x^{n-1} + \dots a_1 x + a_0$$

si scrive come

$$\begin{aligned}\int_a^b p_n(x)\,\mathrm{d}x &= \int_a^b \left(a_n x^n + a_{n-1}x^{n-1} + \dots + a_1 x + a_0\right)\,\mathrm{d}x \\ &= a_n \int_a^b x^n\,\mathrm{d}x + a_{n-1}\int_a^b x^{n-1}\,\mathrm{d}x + \cdots + a_1\int_a^b x\,\mathrm{d}x + a_0\int_a^b 1\,\mathrm{d}x\end{aligned}$$

Quindi l'integrale è esatto *indipendentemente* dai coefficienti $a_0, a_1, \dots, a_n$ se sono esatti *tutti* gli integrali della forma

$$\int_a^b x^i\,\mathrm{d}x, \qquad \text{per } i = 0, 1, \dots n.$$

L'osservazione precedente suggerisce il seguente procedimento per la determinazione dei *pesi* di una formula di quadratura, una volta scelti i nodi (non necessariamente equidistanti).

- Si applica la formula di quadratura a tutti i monomi x^i, per $i = 0, 1, \dots, n$ e si impone che il risultato sia uguale al risultato dell'integrazione esatta.

In questo modo si ricava un sistema nelle $n+1$ equazioni per $i = 0, 1, \dots, n$ nelle $n+1$ incognite $w_0, w_1, \dots w_n$.

Praticamente, si deve imporre la condizione

$$\sum_{j=0}^{n} w_j x_j^i = \int_a^b x^i\,\mathrm{d}x = \frac{b^{i+1} - a^{i+1}}{i+1}, \qquad \text{per } i = 0, 1, \dots, n,$$

per cui si ottiene un sistema lineare nelle w_i

$$\begin{aligned}
i = 0 \quad & w_0 + w_1 + \dots + w_n && = b - a \\
i = 1 \quad & w_0 x_0 + w_1 x_1 + \dots + w_n x_n && = (b^2 - a^2)/2 \\
i = 2 \quad & w_0 x_0^2 + w_1 x_1^2 + \dots + w_n x_n^2 && = (b^3 - a^3)/3 \\
\dots & && \\
i = n \quad & w_0 x_0^n + w_1 x_1^n + \dots + w_n x_n^n && = (b^{n+1} - a^{n+1})/(n+1).
\end{aligned}$$

Il sistema lineare si può riscrivere in forma matriciale compatta $\boldsymbol{V}\boldsymbol{w} = \boldsymbol{q}$ dove

$$\boldsymbol{V} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ x_0 & x_1 & \cdots & x_n \\ x_0^2 & x_1^2 & \cdots & x_n^2 \\ \vdots & & & \vdots \\ x_0^n & x_1^n & \cdots & x_n^n \end{pmatrix}, \qquad \boldsymbol{q} = \begin{pmatrix} b-a \\ (b^2-a^2)/2 \\ (b^3-a^3)/3 \\ \vdots \\ (b^{n+1}-a^{n+1})/(n+1) \end{pmatrix},$$

nelle incognite $w_0, w_1, \dots, w_n$.
La matrice $\boldsymbol{V}$ è una matrice di *Vandermonde*[3], il cui determinante

$$V(x_0, x_1, \dots, x_n) = \prod_{i>j}^{n} (x_i - x_j)$$

è sicuramente diverso da zero se i nodi sono scelti a due a due distinti, cioè vale $x_i \neq x_j$ per $i \neq j$. La non singolarità della matrice $\boldsymbol{V}$ garantisce l'esistenza di una ed una sola soluzione formata da una $n+1$-upla di pesi $w_0, w_1, \dots, w_n$.

Osservazione

La formula di quadratura così costruita ha ordine *almeno n*. Qual è il massimo ordine che può avere?
Se lasciamo libertà di scegliere sia gli $n+1$ nodi (purché distinti) che gli $n+1$ pesi, abbiamo in totale $2(n+1)$ *gradi di libertà*, per cui possiamo chiedere che il metodo dei coefficienti indeterminati soddisfi un ugual numero di condizioni, il che ci da la possibilità di integrare esattamente tutti i monomi fino all'ordine $2n+1$ [4]

10.7 Stima dell'errore di integrazione

10.7.1 Richiamo: errore della regola dei rettangoli

Per una singola formula di quadratura si possono ottenere stime dirette mediante una funzione ausiliaria. Consideriamo come esempio la regola di quadratura che approssima l'integrale in un intervallo usando il rettangolo con altezza il valore della funzione all'estremo a sinistra dell'intervallo:

$$\int_a^b f(x)\mathrm{d}x \approx f(a)(b-a)$$

Lemma 10.1 — Errore regola dei rettangoli. L'errore di integrazione con la regola dei rettangoli a sinistra è il seguente

$$\mathcal{E}(f, a, b) = \frac{(b-a)^2}{2} f'(\omega),$$

Proof. Se introduciamo la primitiva di $f(x)$ indicandola con $F(x)$, possiamo definire la funzione ausiliaria

$$G(t) := F(a+t) - F(a) - tf(a) - \frac{t^2}{h^2}E, \qquad h = b-a$$

La funzione G si annulla per $t = 0$ e per $t = h$, infatti:

[3] Alexandre-Théophile Vandermonde (1735-1796)

[4] la numerazione degli esponenti parte da zero, essendo $x^0 = 1$ il primo monomio di ogni polinomio di qualsiasi ordine.

$$G(0) = F(a) - F(a) - 0 - 0 = 0$$
$$G(h) = F(a+h) - F(a) - hf(a) - \frac{h^2}{h^2}\mathcal{E}(f,a,b)$$
$$= \underbrace{\int_a^b f(x)\mathrm{d}x - hf(a)}_{\text{per definizione}=\mathcal{E}(f,a,b)} - \mathcal{E}(f,a,b) = 0.$$

Allora per il teorema di Rolle[5] (o del valor medio) esiste un punto α compreso in $0 < \alpha < h$ tale che $G'(\alpha) = 0$. Possiamo ripetere il procedimento precedente applicandolo a G':

$$G'(t) = f(a+t) - f(a) - \frac{2t}{h^2}\mathcal{E}(f,a,b).$$

Anche questa volta $G'(0) = 0$, infatti $G'(0) = f(a) - f(a) = 0$, e per il teorema di Rolle esiste β tale che $0 < \beta < \alpha < h$ per cui $G''(\beta) = 0$. La derivata seconda di G è

$$G''(t) = f'(a+t) - \frac{2}{h^2}\mathcal{E}(f,a,b),$$

dunque

$$G''(\beta) = f'(a+\beta) - \frac{2}{h^2}\mathcal{E}(f,a,b) = 0 \Rightarrow \mathcal{E}(f,a,b) = \frac{h^2 f'(a+\beta)}{2}.$$

e ponendo $\omega = a + \beta$ e ricordando che $h = b - a$ si conclude la dimostrazione. ∎

In maniera molto simile si possono calcolare le stime dell'errore per la regola dei trapezi (interpolazione lineare)

$$\int_a^b f(x)\mathrm{d}x \approx \frac{f(a)+f(b)}{2}(b-a)$$

e la regola di Simpson (interpolazione quadratica)

$$\int_a^b f(x)\mathrm{d}x \approx \frac{f(a)+4f((a+b)/2)+f(b)}{6}(b-a)$$

10.7.2 Stime dirette per trapezi e Simpson mediante Rolle

Come nella dimostrazione per i rettangoli a sinistra, poniamo $h = b - a > 0$, introduciamo una primitiva F di f e costruiamo una **funzione ausiliaria** $G(t)$, dove $t \in [0, h]$ rappresenta la lunghezza dell'intervallo $[a, a+t]$. Indichiamo con $\mathcal{E}$ la differenza fra integrale esatto e formula di quadratura.

10.7.2.1 Errore della regola dei trapezi

Supponiamo $f \in C^2([a,b])$ e poniamo

$$\mathcal{E}_T = F(b) - F(a) - \frac{h}{2}\big(f(a)+f(b)\big).$$

Sia p_1 l'interpolante lineare di f agli estremi. Allora $\int_a^b p_1(x)\,\mathrm{d}x = \frac{h}{2}(f(a)+f(b))$. Definiamo

$$\boxed{G(t) = F(a+t) - F(a) - \int_0^t p_1(a+s)\,\mathrm{d}s - \frac{\mathcal{E}_T}{h^3}\,t^2(3h-2t).}$$

[5] Michel Rolle (1652-1719)

Il polinomio correttivo è stato scelto in modo che valga 0 per $t = 0$, valga h^3 per $t = h$ e abbia derivata nulla in $t = 0, h$. Pertanto

$$G(0) = G(h) = 0, \qquad G'(0) = G'(h) = 0.$$

Per Rolle esiste $\alpha \in (0, h)$ con $G'(\alpha) = 0$. Applicando Rolle a G' negli intervalli $(0, \alpha)$ e (α, h), troviamo due punti $\beta_1 < \beta_2$ con $G''(\beta_1) = G''(\beta_2) = 0$. Un'ulteriore applicazione di Rolle produce $\gamma \in (\beta_1, \beta_2)$ tale che $G'''(\gamma) = 0$. Ma

$$G'''(t) = f''(a + t) + \frac{12\mathcal{E}_T}{h^3}.$$

Ne segue, posto $\xi = a + \gamma$,

$$\boxed{\mathcal{E}_T = -\frac{h^3}{12} f''(\xi), \qquad \xi \in (a, b).}$$

10.7.2.2 Errore della regola di Simpson

Supponiamo $f \in C^4([a, b])$. Per semplificare le formule trasliamo l'origine nel punto medio $m = (a + b)/2$ e poniamo $r = (b - a)/2$. Scriviamo ancora $f(x)$ per $f(m + x)$, con $x \in [-r, r]$. L'errore di Simpson è

$$\mathcal{E}_S = \int_{-r}^{r} f(x)\,\mathrm{d}x - \frac{r}{3}\big(f(-r) + 4f(0) + f(r)\big).$$

La funzione trucco dipende dall'ampiezza t dell'intervallo simmetrico. Definiamo

$$\boxed{G(t) = \int_{-t}^{t} f(x)\,\mathrm{d}x - \frac{t}{3}\big(f(-t) + 4f(0) + f(t)\big) - \frac{\mathcal{E}_S}{r^5} t^5, \qquad 0 \le t \le r.}$$

Per costruzione $G(0) = G(r) = 0$. Si noti che il fattore t **davanti alla formula di Simpson è indispensabile** per le dimensioni e per ottenere la corretta formula di quadratura.

La funzione G è di classe C^4 e, derivando mediante la regola di Leibniz, si ottiene

$$\begin{aligned}
G'(t) &= \frac{2}{3}\big(f(t) + f(-t) - 2f(0)\big) - \frac{t}{3}\big(f'(t) - f'(-t)\big) - \frac{5\mathcal{E}_S}{r^5} t^4,\\
G''(t) &= \frac{1}{3}\big(f'(t) - f'(-t)\big) - \frac{t}{3}\big(f''(t) + f''(-t)\big) - \frac{20\mathcal{E}_S}{r^5} t^3,\\
G'''(t) &= -\frac{t}{3}\big(f'''(t) - f'''(-t)\big) - \frac{60\mathcal{E}_S}{r^5} t^2,\\
G^{(4)}(t) &= -\frac{1}{3}\big(f'''(t) - f'''(-t)\big) - \frac{t}{3}\big(f^{(4)}(t) + f^{(4)}(-t)\big) - \frac{120\mathcal{E}_S}{r^5} t.
\end{aligned}$$

In particolare,

$$G(0) = G'(0) = G''(0) = G'''(0) = 0, \qquad G(r) = 0.$$

Applicando **quattro volte il teorema di Rolle** (partendo dallo zero di molteplicità almeno quattro in $t = 0$ e dallo zero in $t = r$) esiste $\beta \in (0, r)$ tale che $G^{(4)}(\beta) = 0$. Pertanto

$$\frac{120\mathcal{E}_S}{r^5}\,\beta = -\frac{1}{3}\big(f'''(\beta) - f'''(-\beta)\big) - \frac{\beta}{3}\big(f^{(4)}(\beta) + f^{(4)}(-\beta)\big).$$

Qui non occorre derivare una quinta volta. Per il **teorema di Lagrange** applicato a f''' sull'intervallo $[-\beta, \beta]$ esiste $\eta \in (-\beta, \beta)$ tale che

$$f'''(\beta) - f'''(-\beta) = 2\beta\, f^{(4)}(\eta).$$

Sostituendo e dividendo per $\beta > 0$ si ricava

$$\mathcal{E}_S = -\frac{r^5}{360}\big(2f^{(4)}(\eta) + f^{(4)}(\beta) + f^{(4)}(-\beta)\big).$$

Il valore tra parentesi è quattro volte una media pesata positiva di valori della funzione continua $f^{(4)}$. Per il teorema dei valori intermedi esiste $\xi \in [-\beta, \beta] \subset (-r, r)$ tale che

$$\frac{2f^{(4)}(\eta) + f^{(4)}(\beta) + f^{(4)}(-\beta)}{4} = f^{(4)}(\xi).$$

Si conclude quindi che

$$\boxed{\mathcal{E}_S = -\frac{r^5}{90} f^{(4)}(\xi) = -\frac{(b-a)^5}{2880} f^{(4)}(\xi), \qquad \xi \in (a, b).}$$

La dimostrazione utilizza soltanto $f \in C^4([a, b])$: quattro applicazioni di Rolle alla funzione trucco $G(t)$, seguite dal teorema di Lagrange per la differenza $f'''(\beta) - f'''(-\beta)$.

10.7.3 Teorema generale per le formule di Newton-Cotes

Dopo le dimostrazioni dirette, riportiamo l'enunciato di un teorema assai importante, che permette di stimare l'ordine di accuratezza per le formule di Newton-Cotes [6][7].

Il teorema si riassume usualmente con l'affermazione che *le formule di Newton-Cotes di grado pari guadagnano un ordine di accuratezza.* Per esempio, la formula di Simpson[8], pur essendo costruibile come un formula interpolatoria che integra esattamente tutti i polinomi di grado *fino a due*, ha in realtà ordine *quattro*, e non *tre* come si potrebbe erroneamente pensare.

Questa proprietà dipende essenzialmente dal fatto che l'integrale di $\omega_{n+1}(x)$, che compare nell'espressione dell'errore di fronte al termine $n+1$ dello sviluppo di Taylor[9], si annulla sull'intervallo $[a, b]$ per ragioni di simmetria. L'errore è dominato dal termine successivo dello sviluppo di Taylor che coinvolge la derivata $n+2$-esima.

L'enunciato del teorema mostra la forma generale dell'errore di integrazione per formule interpolatorie alla Newton-Cotes che interpolano esattamente polinomi di ordine fino ad n sia pari che dispari. Ricordiamo tuttavia che nella pratica non si usano formule di Newton-Cotes con n grandi, maggiori di 7-8, per ragioni di stabilità numerica.

La dimostrazione si può trovare per esempio in [6].

Theorem 10.1 — Stima errore quadratura. Sia

$$\mathcal{E}(f; a, b) = \int_a^b f(x)\,\mathrm{d}x - \mathcal{I}(f; a, b)$$

l'errore di integrazione che si ha applicando una formula di quadratura di Newton-Cotes con $n+1$ nodi

$$\mathcal{I}(f; a, b) = \sum_{k=0}^{n} w_k\, f(x_k)$$

per il calcolo dell'integrale della funzione $f(x)$ su $[a\,b]$:

[6] Sir Isaac Newton (1643-1727)
[7] Roger Cotes (1682-1716)
[8] Thomas Simpson (1710-1761)
[9] Brook Taylor (1685-1731)

$$\int_a^b f(x)\,\mathrm{d}x.$$

Allora valgono i seguenti risultati:

(i) sia $f \in C^{n+2}(a,b)$, con n **pari**; allora esiste un $\bar{x} \in (a,b)$ tale che

$$\mathcal{E}(f;a,b) = \frac{1}{(n+2)!} f^{n+2}(\bar{x}) \int_a^b x\,\omega_{n+1}(x)\,\mathrm{d}x$$

(ii) sia $f \in C^{n+1}(a,b)$, con n **dispari**; allora esiste un $\bar{x} \in (a,b)$ tale che

$$\mathcal{E}(f;a,b) = \frac{1}{(n+1)!} f^{n+1}(\bar{x}) \int_a^b \omega_{n+1}(x)\,\mathrm{d}x$$

dove $\omega_{n+1}(x) = (x-x_0)(x-x_1)\cdots(x-x_n)$.

10.7.4 Verifica esplicita del teorema per trapezi e Simpson

Verifichiamo che il teorema Teorema 10.1 restituisca esattamente i coefficienti ottenuti con le funzioni ausiliarie. In entrambi i casi poniamo $h = b-a$ e indichiamo con $\mathcal{E}$ l'integrale esatto meno la formula di quadratura.

10.7.4.1 Regola dei trapezi: $n = 1$ (dispari)

I nodi sono $x_0 = a$ e $x_1 = b$, quindi

$$\omega_2(x) = (x-a)(x-b).$$

Con il cambio di variabile $s = x-a$, per cui $0 \le s \le h$, calcoliamo **esplicitamente**

$$\begin{aligned}\int_a^b \omega_2(x)\,\mathrm{d}x &= \int_0^h s(s-h)\,\mathrm{d}s \\ &= \left[\frac{s^3}{3} - \frac{hs^2}{2}\right]_0^h \\ &= \frac{h^3}{3} - \frac{h^3}{2} = -\frac{h^3}{6}.\end{aligned}$$

Il caso (ii) del teorema, con $(n+1)! = 2$, fornisce

$$\boxed{\mathcal{E}_T = \frac{f''(\xi)}{2!}\left(-\frac{h^3}{6}\right) = -\frac{(b-a)^3}{12} f''(\xi).}$$

10.7.4.2 Regola di Simpson: $n = 2$ (pari)

Poniamo $m = (a+b)/2$ e $r = (b-a)/2$. I tre nodi sono $m-r, m, m+r$, quindi

$$\omega_3(x) = (x-m+r)(x-m)(x-m-r).$$

Il caso (i) del teorema richiede l'integrale $\int_a^b x\omega_3(x)\,\mathrm{d}x$. Con $y = x-m$ otteniamo

$$\begin{aligned}\int_a^b x\omega_3(x)\,\mathrm{d}x &= \int_{-r}^{r}(m+y)y(y^2-r^2)\,\mathrm{d}y \\ &= m\underbrace{\int_{-r}^{r} y(y^2-r^2)\,\mathrm{d}y}_{=0\text{ (funzione dispari)}} + \int_{-r}^{r}(y^4-r^2y^2)\,\mathrm{d}y \\ &= 2\left[\frac{y^5}{5}-\frac{r^2y^3}{3}\right]_0^r \\ &= 2r^5\left(\frac{1}{5}-\frac{1}{3}\right) = -\frac{4r^5}{15}.\end{aligned}$$

Poiché $(n+2)! = 4! = 24$, il caso (i) dà

$$\begin{aligned}\mathcal{E}_S &= \frac{f^{(4)}(\xi)}{4!}\left(-\frac{4r^5}{15}\right) = -\frac{r^5}{90}f^{(4)}(\xi) \\ &= \boxed{-\frac{(b-a)^5}{2880}f^{(4)}(\xi)}.\end{aligned}$$

Le due espressioni coincidono con quelle dimostrate direttamente mediante la funzione $G(t)$ e il teorema di Rolle.

In sintesi

- Una formula di quadratura $\sum_i w_i f(t_i)$ nasce dall'integrazione del polinomio interpolante: nodi e pesi derivano dalla strategia **interpolatoria**.
- Le formule di **Newton-Cotes** usano nodi equispaziati; il grado di precisione è n se n è dispari e $n+1$ se n è pari, per simmetria.
- Il **metodo dei coefficienti indeterminati** ricava i pesi imponendo l'esattezza sui monomi: è la via operativa per costruire o verificare una formula.
- L'errore di quadratura si stima dal nucleo di Peano o dallo sviluppo di Taylor e dipende da una derivata di f di ordine legato al grado di precisione.
- Newton-Cotes di grado alto è instabile (pesi di segno alterno): si preferisce il composito o la quadratura gaussiana.



11. Integrazione composita

Nella applicazione delle formule di quadratura è conveniente calcolare l'integrale approssimato usando la formula di quadratura su dei sottointervalli. Definendo

$$H = \frac{b-a}{N}, \qquad x_0 = a, \quad x_1 = a + H, \quad \dots \quad x_N = a + NH = b$$

possiamo spezzare l'integrale sui sottointervalli $[x_{k-1}, x_k]$ come segue:

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x &= \sum_{k=1}^{N} \int_{x_{k-1}}^{x_k} f(x)\,\mathrm{d}x \\ &= \sum_{k=1}^{N} \Big(\mathcal{I}(f; x_{k-1}, x_k) + \mathcal{E}(f; x_{k-1}, x_k) \Big) \\ &= \underbrace{\sum_{k=1}^{N} \mathcal{I}(f; x_{k-1}, x_k)}_{\text{formula composita}} + \underbrace{\sum_{k=1}^{N} \mathcal{E}(f; x_{k-1}, x_k)}_{\text{errore globale}}\end{aligned}$$

ancora meglio è definire nodi e pesi con stima dell'errore in un intervallo di riferimento standard facendo un cambio di variabile. L'intervallo di riferimento è l'intervallo $[-1, 1]$. Facendo il cambio di variabile:

$$\begin{aligned} x &= \frac{x_{k-1} + x_k}{2} + z\frac{H}{2}, \\ \mathrm{d}x &= \frac{H}{2}\mathrm{d}z, \\ g_k(z) &= f\left(\frac{x_{k-1} + x_k}{2} + z\frac{H}{2}\right)\end{aligned}$$

e quindi

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{k=1}^{N} \int_{x_{k-1}}^{x_k} f(x)\,\mathrm{d}x = \frac{H}{2} \sum_{k=1}^{N} \int_{-1}^{1} g_k(z)\,\mathrm{d}z$$

La formula di quadratura per l'intervallo $[-1, 1]$ (grazie al teorema Teorema 10.1 avrà la forma

$$\int_{-1}^{1} g(z)\mathrm{d}z = \sum_{j=1}^{m} w_j g(z_j) + \underbrace{C_q\, g^{(q)}(\omega)}_{\text{errore}}, \qquad \omega \in (-1, 1)$$

dove C_q è una costante che dipende dalla formula di quadratura, e quindi

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x &= \frac{H}{2}\sum_{k=1}^{N}\int_{-1}^{1} g_k(z)\,\mathrm{d}z \\ &= \underbrace{\frac{H}{2}\sum_{k=1}^{N}\left(\sum_{j=1}^{m} w_j g_k(z_j)\right)}_{\text{formula composita}} + \underbrace{\frac{H}{2}\sum_{k=1}^{N} C_q\, g_k^{(q)}(\omega_k)}_{\text{errore globale}} \\ &= \sum_{k=1}^{N}\left(\sum_{j=1}^{m}\left(\frac{H}{2}w_j\right) f\left(\frac{x_{k-1}+x_k}{2}+z_j\frac{H}{2}\right)\right) + \frac{NH}{2}\left(\frac{1}{N}\sum_{k=1}^{N} C_q\, g_k^{(q)}(\omega_k)\right)\end{aligned}$$

osservando

$$\begin{aligned}g_k^{(q)}(\omega_k) &= \left(\frac{\mathrm{d}}{\mathrm{d}x}\right)^q f\left(\frac{x_{k-1}+x_k}{2}+z\frac{H}{2}\right) \\ &= \frac{H^q}{2^q} f^{(q)}\left(\frac{x_{k-1}+x_k}{2}+z\frac{H}{2}\right)\end{aligned}$$

e definendo i nodi

$$z_{k,j} = \frac{x_{k-1}+x_k}{2} + z_j\frac{H}{2}$$

e i pesi

$$W_j = \frac{H}{2}w_j$$

otteniamo la formula di quadratura composita

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x &= \sum_{k=1}^{N}\sum_{j=1}^{m} W_j f(z_{k,j}) + C_q\frac{b-a}{2}\frac{H^q}{2^q}\left(\frac{1}{N}\sum_{k=1}^{N} f^{(q)}\left(\frac{x_{k-1}+x_k}{2}+\omega_j\frac{H}{2}\right)\right) \\ &= \underbrace{\sum_{k=1}^{N}\sum_{j=1}^{m} W_j f(z_{k,j})}_{\text{formula composita}} + \underbrace{C_q(b-a)\frac{H^q}{2^{q+1}} f^{(q)}(\omega),}_{\text{errore globale}} \qquad \omega\in(a,b)\end{aligned}$$

11.1 Stima a priori degli intervalli

Quando si usa una formula di quadratura (composita) è importante dare una stima a priori del numero di intervalli in cui suddividere l'intervallo $[a,b]$ di integrazione in modo da mantenere l'errore sotto una tolleranza assegnata. Cioè dato ϵ tolleranza assegnata vogliamo calcolare N in modo che

$$\left|\int_a^b f(x)\,\mathrm{d}x - \underbrace{\sum_{k=1}^{N}\sum_{j=1}^{m} W_j f(z_{k,j})}_{\text{formula composita}}\right| = \left|\underbrace{C_q(b-a)\frac{H^q}{2^{q+1}} f^{(q)}(\omega)}_{\text{errore globale}}\right| \le \epsilon$$

data la costante

$$M_q = \max_{x\in[a,b]}\left|f^{(q)}(x)\right|$$

otteniamo

$$\left|C_q(b-a)\frac{H^q}{2^{q+1}}f^{(q)}(\omega)\right| \leq |C_q|(b-a)\frac{H^q}{2^{q+1}}\left|f^{(q)}(\omega)\right| \leq |C_q|(b-a)\frac{H^q}{2^{q+1}}M_q$$

scegliendo H in modo che soddisfi

$$|C_q|(b-a)\frac{H^q}{2^{q+1}}M_q \leq \epsilon$$

allora possiamo garantire che l'errore globale è sotto la tolleranza ϵ. Dalla identità $NH = (b-a)$ possiamo ricavare una diseguaglianza per N:

$$|C_q|(b-a)\frac{\left(\frac{b-a}{N}\right)^q}{2^{q+1}}M_q \leq \epsilon, \qquad \Rightarrow \qquad |C_q|\frac{(b-a)^{q+1}}{\epsilon\, 2^{q+1}}M_q \leq N^q$$

da cui

Stima intervalli dato errore quadratura

$$N \geq \frac{b-a}{2}\left(\frac{(b-a)}{2\epsilon}M_q|C_q|\right)^{1/q} \tag{11.1}$$

per cui il più piccolo intero N che soddisfa 11.1 può essere usato nella formula di quadratura composita per calcolare l'integrale con la tolleranza assegnata.

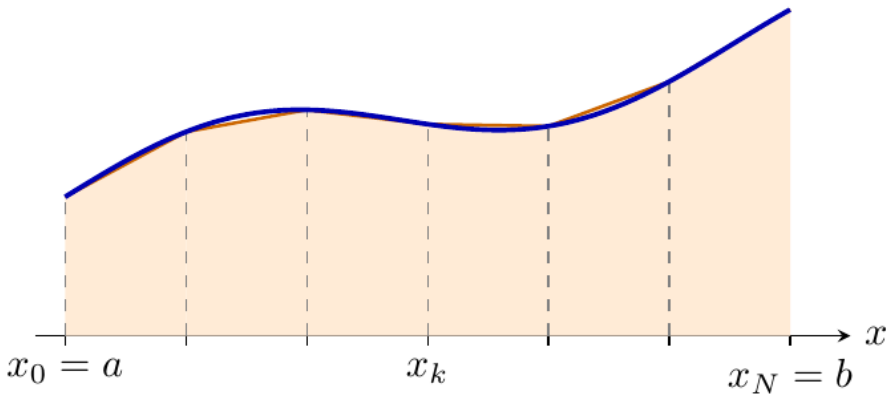


Figure 11.1: Schema geometrico di quadratura: quadratura_04

11.2 Trapezi e Simpson composite

La formula dei trapezi composita si ottiene approssimando l'integrale con polinomi lineari a tratti e prende la forma

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x &= \sum_{k=1}^{N}\frac{H}{2}(f(x_{k-1})+f(x_k)) - \frac{b-a}{12}H^2 f''(\omega)\\ &= \underbrace{\frac{H}{2}(f(a)+f(b)) + H\sum_{k=1}^{N-1}f(x_k)}_{\text{formula composita}} - \underbrace{\frac{b-a}{12}H^2 f''(\omega)}_{\text{errore}}\end{aligned}$$

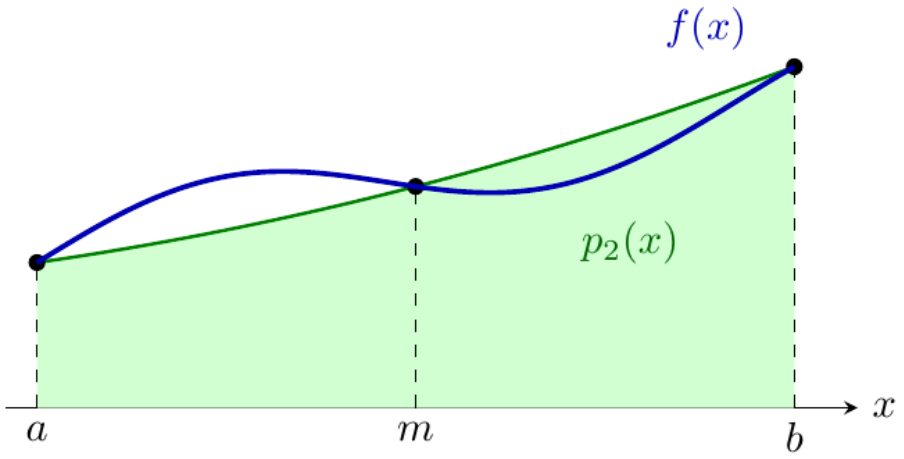


Figure 11.2: Schema geometrico di quadratura: quadratura_05

Mentre la formula composita di Simpson[1] ottenuta approssimando l'integrale con polinomi quadratici a tratti (parabole) prende la forma

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{k=1}^{N} \frac{H}{6}(f(x_{k-1}) + 4f(x_{k-1/2}) + f(x_k)) - \frac{b-a}{2880}H^4 f''''(\omega)$$

dove $x_{k-1/2} = (x_{k-1} + x_k)/2$. Nel caso di Simpson si possono definire i nodi della formula composita come segue:

$$n = 2N, \quad h = \frac{b-a}{n} = \frac{b-a}{2N} = \frac{H}{2}, \qquad \tilde{x}_k = a + kh, \quad k = 0, 1, \ldots, n$$

Con queste definizioni, la formula di Simpson diventa

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{k=1,3,5,\ldots}^{n-1} \frac{h}{3}(f(\tilde{x}_{k-1}) + 4f(\tilde{x}_k) + f(\tilde{x}_{k+1})) - \frac{b-a}{180}h^4 f''''(\omega)$$

che di solito si scrive nella forma seguente

$$\int_a^b f(x)\,\mathrm{d}x = \frac{h}{3}\Big(f(a) + 4f(\tilde{x}_1) + 2f(\tilde{x}_2) + 4f(\tilde{x}_3) + 2f(\tilde{x}_4) + \cdots + \\ + 2f(\tilde{x}_{n-2}) + 4f(\tilde{x}_{n-1}) + f(b)\Big) - \frac{b-a}{180}h^4 f''''(\omega)$$

dove si riconosce il pattern dei pesi $[1, 4, 2, 4, 2, \ldots, 2, 4, 1]$ della formula di quadratura.

> Attenzione
>
> Se si usa h per determinare gli intervalli di integrazione della formula di Simpson, si ricordi che n deve essere un numero *pari*. Infatti $n = 2N$.

In sintesi

- Invece di alzare il grado si suddivide l'intervallo e si applica una formula di basso grado su ogni sottointervallo: è la strategia **composita**.

[1] Thomas Simpson (1710-1761)

- L'errore passa da $\mathcal{O}(h^p)$ locale a $\mathcal{O}(h^{p-1})$ globale sull'intervallo, perché i contributi si sommano su $\sim (b-a)/h$ pezzi.
- Trapezi composita è $\mathcal{O}(h^2)$, Simpson composita $\mathcal{O}(h^4)$: da qui la stima a priori del numero di sottointervalli necessari a una tolleranza data.
- La stima a priori richiede una limitazione delle derivate di f; quando non è disponibile si passa alle stime **a posteriori** e all'adattività.

11.3 Scelta a priori del numero di sottointervalli

Le formule di errore delle regole composite permettono di **dimensionare la discretizzazione prima di calcolare l'integrale**. Fissata una tolleranza assoluta $\mathrm{tol} > 0$, cerchiamo il più piccolo intero $N \geq 1$ per cui la *maggiorazione teorica* dell'errore non supera tol. Questo è il minimo numero di sottointervalli **certificato dalla stima**, non necessariamente il minimo che produce effettivamente un errore inferiore alla tolleranza.

Ponendo $L = b - a$, $H = L/N$ e

$$M_2 \geq \max_{[a,b]} |f''(x)|, \qquad M_4 \geq \max_{[a,b]} |f^{(4)}(x)|,$$

le formule composite danno

$$|E_T| \leq \frac{L^3 M_2}{12 N^2}, \qquad |E_S| \leq \frac{L^5 M_4}{2880 N^4}.$$

Pertanto, per $M_2 > 0$ e $M_4 > 0$, arrotondando **sempre per eccesso**,

$$\boxed{N_T = \left\lceil \sqrt{\frac{L^3 M_2}{12\,\mathrm{tol}}} \right\rceil}, \qquad \boxed{N_S = \left\lceil \left(\frac{L^5 M_4}{2880\,\mathrm{tol}} \right)^{1/4} \right\rceil}.$$

Il simbolo $\lceil x \rceil$ (parentesi quadre con il tratto inferiore omesso) indica l'**arrotondamento all'intero superiore**, detto *ceiling* o *ceil*: è il più piccolo intero maggiore o uguale a x. Ad esempio, $\lceil 15.051 \rceil = 16$, mentre $\lceil 16 \rceil = 16$.

Se la maggiorazione della derivata è nulla, la rispettiva regola è esatta e basta un solo intervallo.

Qui N_S conta gli **intervalli di Simpson** di ampiezza H: ciascun intervallo è diviso dal punto medio in due sottointervalli di ampiezza $h = H/2$. Se si conta invece il numero n di sottointervalli, allora $n = 2N_S$ deve essere pari. Le maggiorazioni richiedono rispettivamente $f \in C^2([a,b])$ e $f \in C^4([a,b])$.

Quattro esercizi su questo dimensionamento — trapezi ed esponenziale, Simpson e seno, trapezi e funzione razionale, Simpson e polinomio — sono svolti nel volume degli esercizi. Qui ne resta uno solo, che confronta fra loro tutte le regole viste.

11.3.1 Esercizio — Confronto tra trapezi, Simpson e Gauss-Legendre

Per approssimare

$$I = \int_0^1 e^x \, \mathrm{d}x = e - 1, \qquad \mathrm{tol} = 10^{-8},$$

determinare il minimo numero di intervalli uniformi **certificato dalle stime a priori** usando trapezi, Simpson e Gauss-Legendre con $m = 2, 3, 4, 5$ nodi per intervallo. Confrontare anche il numero di valutazioni di f.

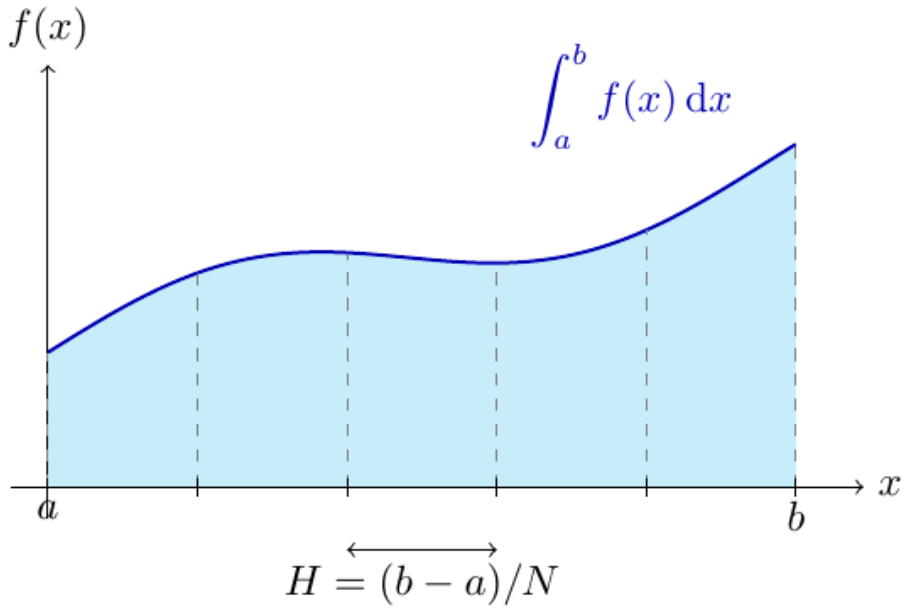


Figure 11.3: Interpretazione geometrica: un integrale è suddiviso in sottointervalli uguali; la regione colorata rappresenta l'area da approssimare.

11.3.1.1 Svolgimento

Dividiamo $[0,1]$ in N intervalli di ampiezza $H = 1/N$. Per Simpson ciascun intervallo contiene due sottointervalli di ampiezza $h = H/2$. Tutte le derivate di $f(x) = e^x$ coincidono con e^x, quindi possiamo usare la maggiorazione $M_k = e$ per ogni ordine k.

Le stime per trapezi e Simpson sono

$$|E_T| \le \frac{e}{12N^2}, \qquad |E_S| \le \frac{e}{2880N^4}.$$

Per Gauss-Legendre con m nodi, il cambio di variabile da $[-1,1]$ a un intervallo di ampiezza H dà la stima locale

$$|E_{G_m,\mathrm{loc}}| \le C_m H^{2m+1} M_{2m}, \qquad C_m = \frac{(m!)^4}{(2m+1)((2m)!)^3}.$$

Sommando i contributi dei N intervalli, con $NH = 1$, otteniamo

$$|E_{G_m}| \le \frac{C_m e}{N^{2m}}, \qquad N_{G_m} = \left\lceil \left(\frac{C_m e}{\mathrm{tol}} \right)^{1/(2m)} \right\rceil.$$

Applicando le formule e arrotondando per eccesso si trova:

Regola	Soglia reale per N	Intervalli N	Valutazioni di f	Maggiorazione dell'errore
Trapezi	4759.4483	4760	4761	$9.998 \cdot 10^{-9}$
Simpson	17.5277	18	37	$8.991 \cdot 10^{-9}$
Gauss-Legendre, 2 nodi	15.8381	16	32	$9.601 \cdot 10^{-9}$
Gauss-Legendre, 3 nodi	2.2645	3	9	$1.850 \cdot 10^{-9}$
Gauss-Legendre, 4 nodi	0.7908	1	4	$1.529 \cdot 10^{-9}$
Gauss-Legendre, 5 nodi	0.4009	1	5	$1.072 \cdot 10^{-12}$

Regola	Soglia reale per N	Intervalli N	Valutazioni di f	Maggiorazione dell'errore

Le valutazioni sono $N+1$ per i trapezi e $2N+1$ per Simpson, riutilizzando i valori agli estremi comuni; per Gauss-Legendre sono mN, perché i nodi sono interni a ciascun intervallo.

In questo esempio, aumentando il numero di nodi di Gauss diminuisce il numero di intervalli richiesto. Tuttavia, il numero di intervalli da solo non misura il costo: Gauss a 4 e a 5 nodi richiedono entrambi un solo intervallo, ma la regola a 4 nodi soddisfa già la tolleranza con una valutazione in meno. Le quantità in tabella sono maggiorazioni teoriche, non errori effettivi; il confronto dipende dalla funzione e dalla tolleranza scelte.



12. Formule di quadratura

12.1 Principali formule di quadratura

- $\mathcal{E}(g; -1, 1)$ errore su intervallo di riferimento $[-1, 1]$
- $\mathcal{E}(f; a, b)$ errore formula composita su $[a, b]$

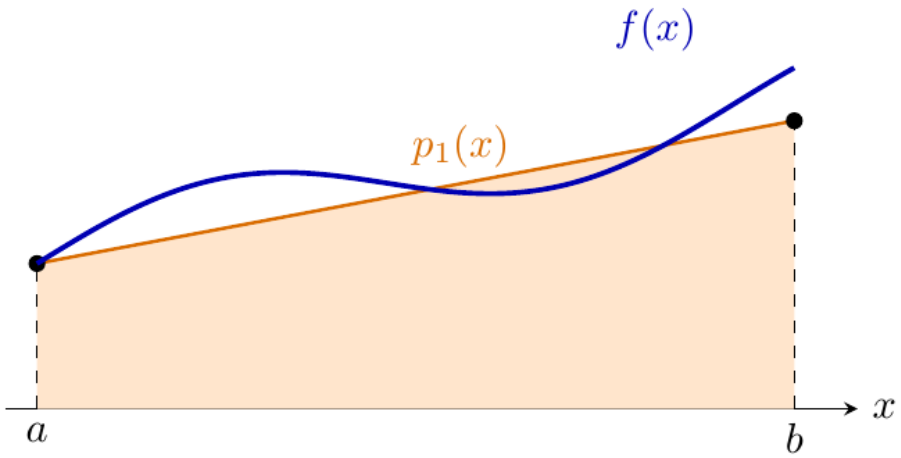


Figure 12.1: Schema geometrico di quadratura: quadratura_06

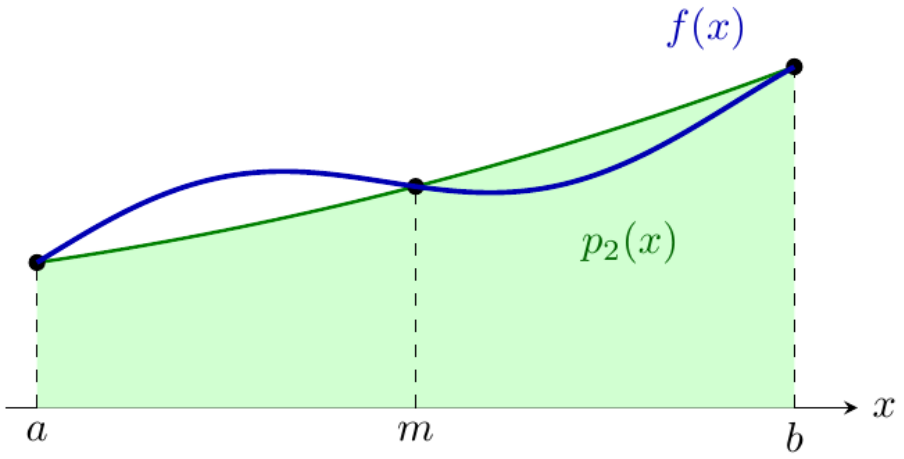


Figure 12.2: Schema geometrico di quadratura: quadratura_07

12.1.1 Newton-Cotes chiuse

Regola dei trapezi

Nodi e pesi

x	w
-1	1
1	1

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{2}{3} g^{(2)}(\omega)$$

$$E_{[a,b]} = -\frac{1}{12}(b-a)H^2 f^{(2)}(\omega)$$

Regola di Simpson

Nodi e pesi

x	w
-1	$1/3$
0	$4/3$
1	$1/3$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{1}{90} g^{(4)}(\omega)$$

$$E_{[a,b]} = -\frac{1}{2880}(b-a)H^4 f^{(4)}(\omega)$$

Regola 3/8

Nodi e pesi	
x	w
± 1	$1/4$
$\pm 1/3$	$3/4$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{2}{405} g^{(4)}(\omega)$$

$$E_{[a,b]} = -\frac{1}{6480}(b-a)H^4 f^{(4)}(\omega)$$

Regola di Boole

Nodi e pesi	
x	w
± 1	$7/45$
$\pm 1/2$	$32/45$
0	$12/45$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{1}{15120} g^{(6)}(\omega)$$

$$E_{[a,b]} = -\frac{1}{1935360}(b-a)H^6 f^{(6)}(\omega)$$

Regola 6 punti

Nodi e pesi	
x	w
± 1	$\frac{19}{144}$
$\pm 3/5$	$\frac{75}{144} = \frac{25}{48}$
$\pm 1/5$	$\frac{50}{144} = \frac{25}{72}$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{22}{590625} g^{(6)}(\omega)$$

$$E_{[a,b]} = -\frac{11}{37800000}(b-a)H^6 f^{(6)}(\omega)$$

Regola 7 punti

Nodi e pesi	
x	w
± 1	$41/420$
$\pm 2/3$	$54/105$
$\pm 1/3$	$9/140$
0	$68/105$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{1}{3061800} g^{(8)}(\omega)$$

$$E_{[a,b]} = -\frac{1}{1567641600}(b-a)H^8 f^{(8)}(\omega)$$

12.1.2 Newton-Cotes aperte

Regola Midpoint

Nodi e pesi

x	w
0	2

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{1}{3} g^{(2)}(\omega)$$

$$E_{[a,b]} = \frac{1}{24}(b-a)H^2 f^{(2)}(\omega)$$

Trapezoid

Nodi e pesi

x	w
$\pm 1/3$	1

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{2}{9} g^{(2)}(\omega)$$

$$E_{[a,b]} = \frac{1}{36}(b-a)H^2 f^{(2)}(\omega)$$

Regola di Milne

Nodi e pesi

x	w
$\pm 1/2$	$4/3$
0	$-2/3$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{7}{720} g^{(4)}(\omega)$$

$$E_{[a,b]} = \frac{7}{23040}(b-a)H^4 f^{(4)}(\omega)$$

12.1.3 Gauss-Legendre

Formula generale per l'errore locale con m nodi:

$$\mathcal{E}(g;-1,1) = C_q g^{(2m)}(\omega) \qquad C_q = \frac{2^{2m+1}\cdot(m!)^4}{(2m+1)\cdot((2m)!)^3}$$

Gauss-Legendre 1 punto (midpoint)

Nodi e pesi

x	w
0	2

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{1}{3} g^{(2)}(\omega)$$

$$E_{[a,b]} = \frac{1}{24}(b-a)H^2 f^{(2)}(\omega)$$

Gauss-Legendre 2 punti

Nodi e pesi	
x	w
$\pm 1/\sqrt{3}$	1

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{1}{135} g^{(4)}(\omega)$$

$$E_{[a,b]} = \frac{1}{4320} (b-a) H^4 f^{(4)}(\omega)$$

Gauss-Legendre 3 punti

Nodi e pesi	
x	w
0	$8/9$
$\pm\sqrt{3/5}$	$5/9$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{1}{15750} g^{(6)}(\omega)$$

$$E_{[a,b]} = \frac{1}{2016000} (b-a) H^6 f^{(6)}(\omega)$$

Gauss-Legendre 4 punti

Nodi e pesi	
x	w
$\pm\frac{\sqrt{525-70\sqrt{30}}}{35}$	$\frac{18+\sqrt{30}}{36}$
$\pm\frac{\sqrt{525+70\sqrt{30}}}{35}$	$\frac{18-\sqrt{30}}{36}$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{1}{3472875} g^{(8)}(\omega)$$

$$E_{[a,b]} = \frac{1}{1778112000} (b-a) H^8 f^{(8)}(\omega)$$

Gauss-Legendre 5 punti

Nodi e pesi	
x	w
0	$128/225$
$\pm\frac{\sqrt{245-14\sqrt{70}}}{21}$	$\frac{322+13\sqrt{70}}{900}$
$\pm\frac{\sqrt{245+14\sqrt{70}}}{21}$	$\frac{322-13\sqrt{70}}{900}$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{1}{1237732650} g^{(10)}(\omega)$$

$$E_{[a,b]} = \frac{1}{2534876467200} (b-a) H^{10} f^{(10)}(\omega)$$

12.1.4 Gauss-Lobatto

Questa quadratura fissa il primo nodo a -1 e l'ultimo a $+1$. Formula generale per l'errore locale con m nodi:

$$\mathcal{E}(g;-1,1) = C_q g^{(2m-2)}(\omega) \qquad C_q = -\frac{m\cdot(m-1)^3\cdot 2^{2m-1}\cdot((m-2)!)^4}{(2m-1)\cdot((2m-2)!)^3}$$

- 2 punti = regola trapezi.
- 3 punti = regola di Simpson[1].

Gauss-Lobatto 4 punti

Nodi e pesi	
x	w
± 1	$1/6$
$\pm 1/\sqrt{5}$	$5/6$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{2}{23625}\, g^{(6)}(\omega)$$

$$E_{[a,b]} = -\frac{1}{1512000}\,(b-a)H^6\, f^{(6)}(\omega)$$

Gauss-Lobatto 6 punti

Nodi e pesi	
x	w
± 1	$\frac{1}{15}$
$\pm\sqrt{\frac{1}{3}-\frac{2}{3\sqrt{7}}}$	$\frac{1}{30}\left(14+\sqrt{7}\right)$
$\pm\sqrt{\frac{1}{3}+\frac{2}{3\sqrt{7}}}$	$\frac{1}{30}\left(14-\sqrt{7}\right)$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = -\frac{1}{1031443875}\, g^{(10)}(\omega)$$

$$E_{[a,b]} = -\frac{1}{2112397056000}\,(b-a)H^{10}\, f^{(10)}(\omega)$$

12.1.5 Gauss-Radau

Questa quadratura fissa il primo nodo a -1. Formula generale per l'errore locale con m nodi:

$$\mathcal{E}(g;-1,1) = C_q g^{(2m-1)}(\omega) \qquad C_q = \frac{2^{2m-1}\cdot m\cdot((m-1)!)^4}{((2m-1)!)^3}$$

Gauss-Radau 2 punti

Nodi e pesi	
x	w
-1	$1/2$
$1/3$	$3/2$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{2}{27}\, g^{(3)}(\omega)$$

$$E_{[a,b]} = \frac{1}{216}\,(b-a)H^3\, f^{(3)}(\omega)$$

[1] Thomas Simpson (1710-1761)

Gauss-Radau 3 punti

Nodi e pesi

x	w
-1	$\frac{2}{9}$
$\frac{1-\sqrt{6}}{5}$	$\frac{16+\sqrt{6}}{18}$
$\frac{1+\sqrt{6}}{5}$	$\frac{16-\sqrt{6}}{18}$

Errore sull'intervallo di riferimento $E_{[-1,1]}$ ed errore della formula composita $E_{[a,b]}$:

$$E_{[-1,1]} = \frac{1}{1125} g^{(5)}(\omega)$$

$$E_{[a,b]} = \frac{1}{72000} (b-a) H^5 f^{(5)}(\omega)$$



13. Stime a posteriori

Una formula di quadratura ha una formula per l'errore del tipo

$$\mathcal{E}(f; a, b) = \frac{C_q}{2^{p+1}}(b-a)H^p f^{(p)}(z) \tag{13.1}$$

$f^{(p)}(z)$ è una media campionata della derivata p-esima. Se $[a, b]$ è un intervallo piccolo possiamo approssimare

$$f^{(p)}(z) \approx \frac{1}{b-a}\int_a^b f^{(p)}(x)\mathrm{d}x := F_p$$

da cui

$$\mathcal{E}(f; a, b) \approx \frac{C_q}{2^{p+1}}(b-a)H^p F_p$$

Usando un solo intervallo nella formula di quadratura abbiamo

$$\int_a^b f(x)\mathrm{d}x \approx I_1(f; a, b) + \frac{C_q}{2^{p+1}}(b-a)^{p+1}F_p \tag{13.2}$$

dove $I_1(f; a, b)$ è il valore della formula di quadratura applicata alla funzione $f(x)$ sull'intervallo $[a, b]$ senza suddividerlo ($H = b - a$). Usando due intervalli nella formula di quadratura abbiamo

$$\int_a^b f(x)\mathrm{d}x \approx I_2(f; a, b) + \frac{C_q}{2^{p+1}}\frac{(b-a)^{p+1}}{2^p}F_p \tag{13.3}$$

dove $I_2(f; a, b)$ è il valore della stessa formula di quadratura applicata alla funzione $f(x)$ sull'intervallo $[a, b]$, suddiviso in due parti ($H = (b-a)/2$). Sottraendo 13.2 da 13.3 si ottiene

$$I_2(f; a, b) - I_1(f; a, b) \approx \frac{C_q}{2^{p+1}}(b-a)^{p+1}F_p\left(1 - \frac{1}{2^p}\right)$$

da cui

$$F_p \approx \frac{2^{2p+1}}{C_q(b-a)^{p+1}(2^p-1)}\left(I_2(f; a, b) - I_1(f; a, b)\right)$$

Sostituendo questa stima in 13.1, con $H = (b-a)/2$, si ottiene una stima dell'errore

$$\begin{aligned}\mathcal{E}(f;a,b) &= \frac{C_q}{2^{2p+1}}(b-a)^{p+1}f^{(p)}(z)\\ &\approx \frac{C_q}{2^{2p+1}}(b-a)^{p+1}F_p\\ &\approx \frac{C_q}{2^{2p+1}}(b-a)^{p+1}\frac{2^{2p+1}}{C_q(b-a)^{p+1}(2^p-1)}\left(I_2(f;a,b)-I_1(f;a,b)\right)\\ &\approx \frac{I_2(f;a,b)-I_1(f;a,b)}{2^p-1}\end{aligned}$$

Importante

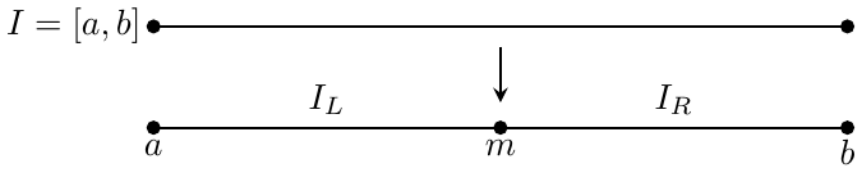


Figure 13.1: Schema geometrico di quadratura: quadratura_08

Stima a posteriori

Data una formula di quadratura di ordine p, si calcoli l'integrale su $[a,b]$ usando un solo intervallo, ottenendo $I_1(f;a,b)$, e poi suddividendo l'intervallo in due, ottenendo $I_2(f;a,b)$. L'errore si può stimare con la formula

$$\mathcal{E}(f;a,b) \approx \frac{I_2(f;a,b)-I_1(f;a,b)}{2^p-1}$$

Si può usare questa stima per costruire formule di quadratura adattative.

13.1 Dalla stima a posteriori a un algoritmo adattativo

La stima ottenuta sopra è una **stima asintotica**, non un limite rigoroso in assenza di ulteriori ipotesi. Sia $Q_1(I)$ la formula su $I=[a,b]$, e sia $Q_2(I)$ la somma delle due formule sui sottointervalli $I_L=[a,m]$ e $I_R=[m,b]$, dove $m=(a+b)/2$. Per una regola composita di ordine globale p (trapezi: $p=2$; Simpson: $p=4$), assumendo che il termine principale dell'errore sia regolare e non si annulli per cancellazione, vale

$$I-Q_1 \simeq Ch^{p+1}, \qquad I-Q_2 \simeq \frac{Ch^{p+1}}{2^p}, \qquad \boxed{\widehat{E}_I = \frac{|Q_2-Q_1|}{2^p-1}}.$$

In particolare, l'errore stimato della **formula raffinata** Q_2 è

$$\widehat{E}_{T,I} = \frac{|T_2-T_1|}{3}, \qquad \widehat{E}_{S,I} = \frac{|S_2-S_1|}{15}.$$

Il denominatore 15 riguarda l'errore di S_2, **non** quello di S_1. L'eventuale correzione di Richardson $Q_{\text{corr}} = Q_2 + (Q_2-Q_1)/(2^p-1)$ è un'approssimazione ulteriore, ma non è automaticamente certificata dalla stessa stima.

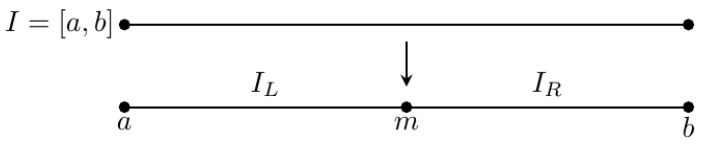


Figure 13.2: Un intervallo e la sua suddivisione in due metà.

13.1.1 Algoritmo ricorsivo con budget locale

Data una tolleranza assoluta globale tol, all'intervallo iniziale si assegna il budget $\tau_{[a,b]} = \text{tol}$. Su ogni intervallo I si calcolano $Q_1(I)$, $Q_2(I)$ e $\widehat{E}_I$. Se $\widehat{E}_I \le \tau_I$ si accetta $Q_2(I)$; altrimenti si divide I in due metà e si assegna a ciascuna $\tau_I/2$. L'algoritmo termina quando tutte le foglie sono accettate (o si raggiunge un limite di sicurezza).

```
adapt(f, a, b, tau):
    Q1 = quadratura(f, a, b)
    m  = (a+b)/2
    Q2 = quadratura(f, a, m) + quadratura(f, m, b)
    est = abs(Q2-Q1)/(2^p-1)
    if est <= tau:
        return Q2, est
    else:
        L = adapt(f, a, m, tau/2)
        R = adapt(f, m, b, tau/2)
        return L.valore + R.valore, L.stima + R.stima
```

Se $\mathcal{L}$ è l'insieme degli intervalli terminali, la costruzione dei budget assicura $\sum_{I\in\mathcal{L}} \tau_I = \text{tol}$. Per la disuguaglianza triangolare, **se** ciascuna stima locale fosse un maggiorante rigoroso, allora

$$\left| \int_a^b f(x)\,\mathrm{d}x - \sum_{I\in\mathcal{L}} Q_2(I) \right| \le \sum_{I\in\mathcal{L}} |E_I| \le \sum_{I\in\mathcal{L}} \tau_I = \text{tol}.$$

Con lo stimatore per differenza di quadrature si ottiene invece un **controllo euristico/asintotico**: errori con segno opposto possono compensarsi, e in presenza di oscillazioni non risolte Q_1 e Q_2 possono essere vicini pur essendo entrambi inaccurati. Sono utili limiti sulla profondità, sul numero di valutazioni e test contro la stagnazione numerica.

13.2 Perché cercare di equidistribuire l'errore

Sia h_i l'ampiezza dell'intervallo I_i e supponiamo che, localmente, l'errore abbia modulo $|E_i| \simeq c(x_i)h_i^{p+1}$, con $c(x) > 0$ lentamente variabile. Un reticolo efficiente usa intervalli piccoli dove $c(x)$ è grande e grandi dove $c(x)$ è piccolo.

Theorem 13.1 — Equidistribuzione asintotica per una partizione ottimale. Nel modello continuo, tra le partizioni che minimizzano il **numero di intervalli** per un budget globale fissato $\int_a^b c(x)h(x)^p\,\mathrm{d}x = \text{tol}$, la condizione ottimale è

$$\boxed{c(x)h(x)^{p+1} = \text{costante}.}$$

Pertanto gli errori locali (in modulo, nel modello asintotico) sono approssimativamente uguali.

Dimostrazione. La densità di intervalli è $1/h(x)$, per cui il costo è $N \simeq \int_a^b 1/h(x)\,\mathrm{d}x$. L'errore globale stimato è $\int_a^b c(x)h(x)^p\,\mathrm{d}x$ perché ciascun intervallo contribuisce $c(x)h^{p+1}$ e ce

ne sono circa $1/h$ per unità di lunghezza. Minimizzando il funzionale

$$\int_a^b \left[\frac{1}{h(x)} + \lambda c(x) h(x)^p \right] \, \mathrm{d}x$$

rispetto a $h(x)$, la condizione di stazionarietà dà

$$-\frac{1}{h(x)^2} + \lambda p c(x) h(x)^{p-1} = 0 \quad \Longrightarrow \quad c(x) h(x)^{p+1} = \frac{1}{\lambda p}.$$

L'equidistribuzione è dunque una **proprietà dell'ottimo nel modello continuo**, non una condizione che ogni algoritmo a bisezione soddisfa esattamente: l'algoritmo con budget dimezzato garantisce un criterio di accettazione locale, ma può produrre errori terminali molto diversi. Per Simpson $p = 4$, dunque $h(x) \propto c(x)^{-1/5}$; per i trapezi $p = 2$, $h(x) \propto c(x)^{-1/3}$. In prossimità di zeri del coefficiente principale o di singolarità servono precauzioni ulteriori.

13.3 Esperimenti numerici: stima ed errore effettivo

Per ciascun esperimento si usa l'algoritmo sopra descritto. Nelle tabelle, $Q_{2,i}$ è la somma delle due quadrature sui semintervalli della foglia I_i, $\widehat{E}_i$ è lo stimatore per differenza e $|E_i| = |\int_{I_i} f - Q_{2,i}|$ è calcolato confrontando con un integrale di riferimento ad alta accuratezza. Le colonne τ_i e $\widehat{E}_i/|E_i|$ permettono di verificare la decisione di accettazione e la qualità dello stimatore. I valori numerici sono arrotondati; l'algoritmo usa valori non arrotondati.

13.3.1 Esempio 1: picco localizzato

Intervallo $[0, 5]$, tolleranza tol $= 2.0e-04$, metodo **Simpson**. La partizione adattativa contiene **16 intervalli terminali**, con 15 suddivisioni. I colori alternati distinguono quattro regioni iniziali dell'intervallo; le linee verticali identificano i sottointervalli terminali.

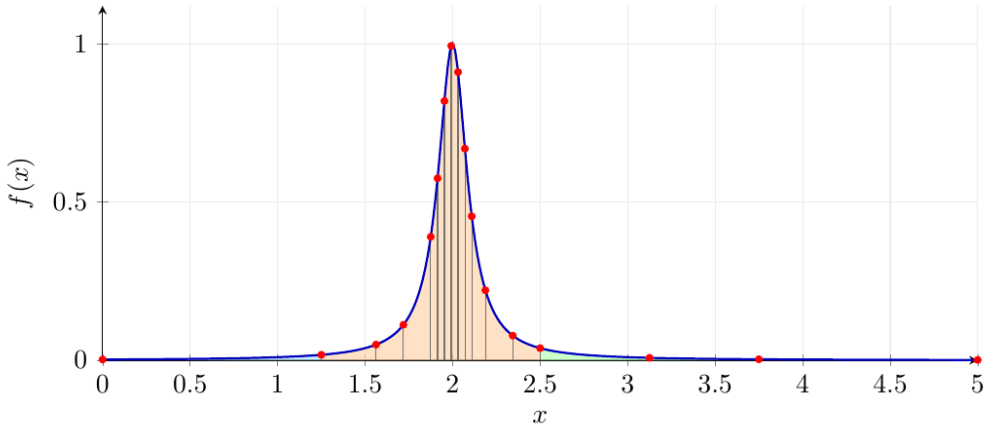


Figure 13.3: Partizione adattativa e nodi per l'esempio 1; colori alternati per le regioni iniziali.

$I_i = [a_i, b_i]$	h_i	$Q_{2,i}$	$\widehat{E}_i$	$\lvert E_i \rvert$	τ_i
[0.00000, 1.25000]	1.25000	0.0082919552	1.67e-05	3.26e-05	5.00e-05
[1.25000, 1.56250]	0.31250	0.0092179833	1.62e-06	2.02e-06	1.25e-05
[1.56250, 1.71875]	0.15625	0.011691209	6.92e-07	7.77e-07	6.25e-06
[1.71875, 1.87500]	0.15625	0.033317659	5.93e-06	5.11e-06	6.25e-06
[1.87500, 1.91406]	0.03906	0.018614328	4.57e-08	4.87e-08	1.56e-06

$I_i = [a_i, b_i]$	h_i	$Q_{2,i}$	$\widehat{E}_i$	$\lvert E_i \rvert$	τ_i
[1.91406, 1.95312]	0.03906	0.027157332	1.82e-07	1.74e-07	1.56e-06
[1.95312, 1.99219]	0.03906	0.0360371	9.73e-08	1.07e-07	1.56e-06
[1.99219, 2.03125]	0.03906	0.038085486	3.68e-07	3.36e-07	1.56e-06
[2.03125, 2.07031]	0.03906	0.030993378	1.78e-07	1.55e-07	1.56e-06
[2.07031, 2.10938]	0.03906	0.021732322	9.48e-08	9.71e-08	1.56e-06
[2.10938, 2.18750]	0.07812	0.025069459	1.02e-07	1.34e-09	3.13e-06
[2.18750, 2.34375]	0.15625	0.020689176	2.71e-06	3.02e-06	6.25e-06
[2.34375, 2.50000]	0.15625	0.0085703575	3.04e-07	3.35e-07	6.25e-06
[2.50000, 3.12500]	0.62500	0.010890084	1.00e-05	1.61e-05	2.50e-05
[3.12500, 3.75000]	0.62500	0.003157862	2.95e-07	3.52e-07	2.50e-05
[3.75000, 5.00000]	1.25000	0.0023766176	4.95e-07	6.39e-07	5.00e-05

Bilancio globale: $\sum_i \widehat{E}_i = 3.9875e-05$; errore effettivo $|I - Q| = 6.0983e-05$; integrale di riferimento $I \approx 0.305831326199$. La stima a posteriori è asintotica, non una maggiorazione rigorosa.

13.3.2 Esempio 2: funzione regolare

Intervallo $[0, 1]$, tolleranza tol $= 1.0e-06$, metodo **Simpson**. La partizione adattativa contiene **4 intervalli terminali**, con 3 suddivisioni. I colori alternati distinguono quattro regioni iniziali dell'intervallo; le linee verticali identificano i sottointervalli terminali.

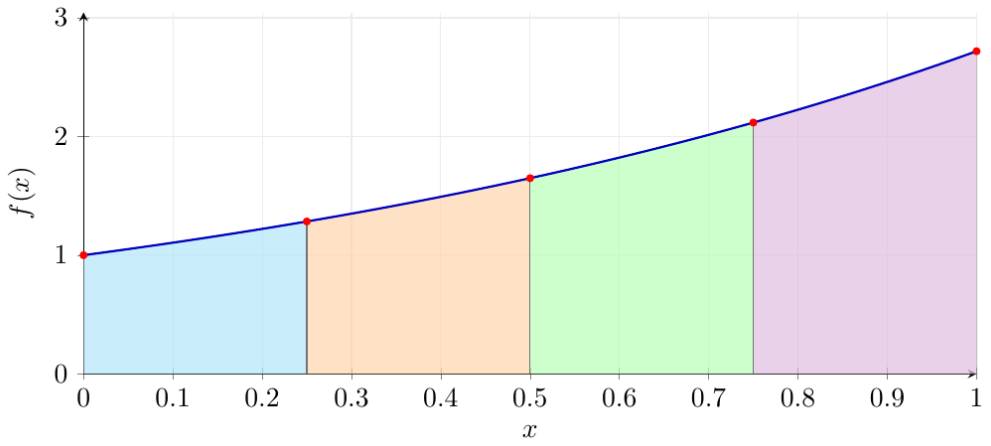


Figure 13.4: Partizione adattativa e nodi per l'esempio 2; colori alternati per le regioni iniziali.

$I_i = [a_i, b_i]$	h_i	$Q_{2,i}$	$\widehat{E}_i$	$\lvert E_i \rvert$	τ_i
[0.00000, 0.25000]	0.25000	0.28402544	2.40e-08	2.41e-08	2.50e-07
[0.25000, 0.50000]	0.25000	0.36469588	3.09e-08	3.09e-08	2.50e-07
[0.50000, 0.75000]	0.25000	0.46827879	3.96e-08	3.97e-08	2.50e-07
[0.75000, 1.00000]	0.25000	0.60128186	5.09e-08	5.09e-08	2.50e-07

Bilancio globale: $\sum_i \widehat{E}_i = 1.4538e-07$; errore effettivo $|I - Q| = 1.4559e-07$; integrale di riferimento $I \approx 1.71828182846$. La stima a posteriori è asintotica, non una maggiorazione rigorosa.

13.3.3 Esempio 3: oscillazioni localizzate

Intervallo $[0, 3]$, tolleranza tol $= 2.0e-02$, metodo **trapezi**. La partizione adattativa contiene **10 intervalli terminali**, con 9 suddivisioni. I colori alternati distinguono quattro regioni iniziali dell'intervallo; le linee verticali identificano i sottointervalli terminali.

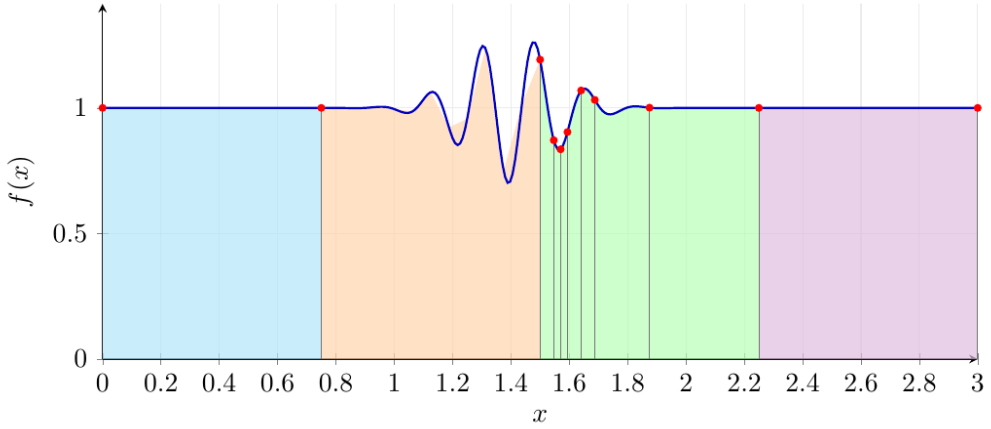


Figure 13.5: Partizione adattativa e nodi per l'esempio 3; colori alternati per le regioni iniziali.

$I_i = [a_i, b_i]$	h_i	$Q_{2,i}$	$\widehat{E}_i$	$\lvert E_i \rvert$	τ_i
[0.00000, 0.75000]	0.75000	0.75000818	2.73e-06	7.94e-06	5.00e-03
[0.75000, 1.50000]	0.75000	0.80949697	4.20e-03	5.58e-02	5.00e-03
[1.50000, 1.54688]	0.04688	0.048064812	1.00e-04	8.71e-05	3.13e-04
[1.54688, 1.57031]	0.02344	0.019815627	6.14e-05	6.06e-05	1.56e-04
[1.57031, 1.59375]	0.02344	0.020272781	3.68e-05	3.65e-05	1.56e-04
[1.59375, 1.64062]	0.04688	0.046705993	1.50e-04	1.36e-04	3.13e-04
[1.64062, 1.68750]	0.04688	0.049716074	1.54e-04	1.51e-04	3.13e-04
[1.68750, 1.87500]	0.18750	0.18838794	7.32e-04	1.69e-03	1.25e-03
[1.87500, 2.25000]	0.37500	0.3750912	3.01e-05	1.37e-04	2.50e-03
[2.25000, 3.00000]	0.75000	0.75	1.29e-09	2.21e-09	5.00e-03

Bilancio globale: $\sum_i \widehat{E}_i = 5.4660e-03$; errore effettivo $|I - Q| = 5.7560e-02$; integrale di riferimento $I \approx 2.99999995924$. La stima a posteriori è asintotica, non una maggiorazione rigorosa.

V Polinomi ortogonali

14 Da Gram–Schmidt ai polinomi di Legendre 157

14.1 Il prodotto scalare pesato sui polinomi 157
14.2 Esistenza della famiglia ortogonale 158
14.3 I polinomi di Legendre 161
14.4 Esercizi 167

15 Peso di Jacobi e famiglie classiche . . 169

15.1 Il peso di Jacobi 169
15.2 Le famiglie classiche come casi particolari . . . 170
15.3 Il peso decide dove si concentra l'errore 175
15.4 La ricorrenza dei polinomi di Jacobi 176
15.5 Perché l'ortogonalità forza tre termini 177
15.6 Esercizi 178

16 Ricorrenze, zeri e stabilità 179

16.1 La ricorrenza a tre termini 179
16.2 Gli zeri dei polinomi ortogonali 182
16.3 Polinomi del nucleo e minimo vincolato 185
16.4 Stabilità della ricorrenza nel calcolo 187
16.5 Esercizi 188

17 Minimi quadrati con basi ortogonali . 189

17.1 Il problema, e tre domande 189
17.2 Impostazione del problema 189
17.3 Un esempio svolto a mano: cinque punti, una parabola 191
17.4 Tre esempi con l'algoritmo 197
17.5 Come si risolve davvero: la fattorizzazione QR 203
17.6 Prima domanda: quanto si guadagna cambiando base 205
17.7 Seconda domanda: lo spettro dei coefficienti e il plateau 208
17.8 Terza domanda: il fitting incrementale 212
17.9 Il problema continuo 215
17.10 Esercizi 219

18 Quadrature di Gauss e zeri dei polinomi 221

18.1 Esattezza di grado $2n-1$ 221
18.2 Gauss–Legendre 223
18.3 Gauss–Jacobi e cambio di intervallo 227
18.4 Errore e varianti con nodi vincolati 229
18.5 Esercizi 229

19 Approssimazione spettrale e altre applicazioni 231

19.1 Approssimazione spettrale 231
19.2 I limiti: discontinuità e fenomeno di Gibbs 232
19.3 Metodi spettrali 234
19.4 Altre applicazioni 235
19.5 Esercizi 236

Che cosa sono

In $\mathbb{R}^3$ è opportuno evitare basi costituite da vettori quasi paralleli: le coordinate di un vettore rispetto a tali basi possono assumere valori molto elevati ed essere fortemente sensibili a piccole perturbazioni dei dati. Quando possibile, si preferisce quindi una base ortogonale, rispetto alla quale ciascuna coordinata si determina mediante un prodotto scalare.

I polinomi ortogonali consentono di costruire basi polinomiali ortogonali rispetto a un opportuno prodotto scalare.

Consideriamo lo spazio $\mathcal{P}_n$ dei polinomi di grado al più n. Fissando un prodotto scalare, questo spazio assume la struttura di uno spazio euclideo di dimensione $n+1$; ad esempio:

$$\langle f, g\rangle_w = \int_a^b f(x)\, g(x)\, w(x)\, \mathrm{d}x, \qquad w > 0.$$

Se si adottasse la base canonica dei monomi $1, x, x^2, \ldots$, rispetto a questo prodotto scalare le funzioni x^8 e x^{10} sarebbero quasi indistinguibili: l'angolo fra loro tende a zero al crescere del grado. Questa quasi-parallelità è una delle cause del cattivo condizionamento della matrice di Vandermonde introdotta in Capitolo 6.

I **polinomi ortogonali** si ottengono ortogonalizzando la base dei monomi rispetto al prodotto scalare considerato. Una successione $p_0(x)$, $p_1(x)$, $p_2(x), \ldots$, con $\deg p_k(x) = k$, costituisce una famiglia di polinomi ortogonali se i suoi elementi sono mutuamente ortogonali rispetto a $\langle \cdot, \cdot \rangle_w$. Pesi diversi danno famiglie diverse — Legendre, Chebyshev, Jacobi, Hermite — perché *ortogonale* ha senso solo quando si specifica rispetto a quale prodotto scalare.

L'importanza di questi polinomi emerge dalle seguenti proprietà e applicazioni:

- i coefficienti della migliore approssimazione si calcolano uno alla volta, un integrale ciascuno, senza risolvere alcun sistema lineare;
- l'intera famiglia si genera con una ricorrenza a tre soli termini, di costo $O(n)$ e memoria costante;
- gli zeri di p_n sono n, reali, semplici e interni all'intervallo, e quelli di p_{n+1} li separano; (sono una successione di Sturm[1])
- utilizzati come nodi di quadratura, consentono di costruire formule gaussiane con il massimo grado di esattezza possibile a parità di numero di nodi.

Perché sono nati (nota storica)

Osservazione

Date e attribuzioni di questa nota storica provengono da:

- J.-B. Pecot, *Le problème de l'ellipsoïde et l'analyse harmonique: la controverse entre Legendre et Laplace*, Cahiers du séminaire d'histoire des mathématiques, 2ª serie, **3** (1993), 113–157, per le memorie di Legendre e Laplace e per le rispettive date di lettura e di pubblicazione;
- A. Sommariva, note del corso sulla quadratura gaussiana, Università di Padova, per la cronologia da Gauss a Golub–Welsch;
- *The life, work and legacy of P. L. Chebyshev*, arXiv:2111.12551, e la raccolta di traduzioni curata da R. Pulskamp, per Chebyshev e la sua scuola;
- *Euler and the Legendre Polynomials*, arXiv:2308.16318, per le anticipazioni di Eulero.

I riferimenti in bibliografia sono presi dalle fonti digitalizzate (NUMDAM, EuDML, AMS). Due avvertenze. L'anno in cui l'aggettivo *orthogonal* viene applicato a questi polinomi poggia su una sola fonte, ed è quindi da prendere con cautela. Per la formula di Christoffel–

[1] Jacques Charles François Sturm (1803-1855)

> Darboux le fonti danno due date diverse (1858 e 1877): per questo la formula compare senza data.

I polinomi ortogonali nascono da un problema di meccanica celeste. Nelle memorie sull'attrazione degli sferoidi omogenei presentate all'Académie des sciences — quella di Legendre[2], letta nel 1783 e pubblicata nel 1785 [28], e quella di Laplace[3], letta nell'agosto 1784 e stampata nel volume dell'Accademia relativo al 1782 — occorreva sviluppare l'inverso della distanza fra due punti,

$$\frac{1}{\sqrt{1 - 2xt + t^2}} = \sum_{n \geq 0} P_n(x)\, t^n,$$

per esprimere il potenziale gravitazionale di un pianeta non sferico. I coefficienti P_n di quello sviluppo sono i polinomi di Legendre: nati come coefficienti di una serie, non come base di uno spazio. Che fossero ortogonali fra loro era una proprietà osservata, non la definizione.

La paternità, a rigore, è più sfumata: Eulero[4] aveva già incontrato sia la funzione generatrice sia la ricorrenza a tre termini in lavori del 1763 e del 1772, senza però isolare la famiglia di polinomi né darle un nome. Fra Legendre e Laplace ci fu poi una disputa di priorità, ricostruita nell'articolo di Pecot citato sopra.

Trent'anni dopo nel 1814 Gauss[5], in *Methodus nova integralium valores per approximationem inveniendi* [15], si chiede fin dove si possa spingere l'accuratezza di una formula di quadratura scegliendo bene non solo i pesi ma anche i nodi, e trova — per via di frazioni continue — che n nodi ben scelti integrano esattamente i polinomi fino al grado $2n - 1$.

Jacobi[6] nel 1826 in *Ueber Gauss' neue Methode, die Werthe der Integrale näherungsweise zu finden* [26] rifà la dimostrazione con un argomento di divisibilità fra polinomi e mette al centro l'ortogonalità. I nodi ottimali di Gauss per la formula di quadratura (vedi Capitolo 18) sono gli zeri del polinomio di Legendre di grado n.

Una volta introdotti i polinomi ortogonali seguono le generalizzazioni.

- Chebyshev[7], a partire dalla memoria *Sur les fractions continues* — presentata in russo nel 1855 e pubblicata in francese nel 1858, nella traduzione di Bienaymé[8] [40] — costruisce la teoria per un peso qualunque anziché per famiglie particolari;
- Charles Hermite[9] (1864), Edmond Laguerre[10] (1879) e Leopold Gegenbauer[11] (1884) introducono le famiglie di polinomi ortogonali che portano il loro nome.
- Christoffel[12] nel 1877 dimostra che gli zeri dei polinomi ortogonali sono sempre reali, semplici e interni all'intervallo (a, b) su cui è definito il prodotto scalare, qualunque esso sia e qualunque sia il peso.
- Stieltjes[13], con le *Recherches sur les fractions continues*, a partire dal 1894 [37], collega tutto al problema dei momenti.
- Erhard Schmidt[14] applica a questi polinomi l'aggettivo *orthogonal*, pare attorno al 1905;

[2] Adrien-Marie Legendre (1752-1833)
[3] Pierre-Simon Laplace (1749-1827)
[4] Leonhard Euler (1707-1783)
[5] Carl Friedrich Gauss (1777-1855)
[6] Carl Gustav Jacob Jacobi (1804-1851)
[7] Pafnutij L'vovič Čebyšëv (1821-1894)
[8] Irénée-Jules Bienaymé (1796-1878)
[9] Charles Hermite (1822-1901)
[10] Edmond Nicolas Laguerre (1834-1886)
[11] Leopold Gegenbauer (1849-1903)
[12] Elwin Bruno Christoffel (1829-1900)
[13] Thomas Joannes Stieltjes (1856-1894)
[14] Erhard Schmidt (1876-1959)

- l'espressione *orthogonal polynomials* diventa definitiva con Szegő[15] attorno al 1918, e la sua monografia del 1939 resta il riferimento classico.
- Nel 1969 Gene H. Golub[16] e John H. Welsch[17] [17] osservano che nodi e pesi di una formula gaussiana sono autovalori e prime componenti degli autovettori di una matrice tridiagonale simmetrica, trasformando una ricerca di radici in un problema (più semplice) di algebra lineare.

Il capitolo presuppone nota

- la fattorizzazione QR e i minimi quadrati di Capitolo 3,
- il condizionamento della matrice di Vandermonde e il fenomeno di Runge di Capitolo 9
- le formule di quadratura interpolatorie di Capitolo 10

vengono date risposte a domande rimaste in sospeso nei capitoli precedenti:

- perché la base monomiale è una cattiva base per l'approssimazione, e quale usare al suo posto (Capitolo 17);
- da dove vengono i nodi e i pesi delle formule di Gauss–Legendre usate in Capitolo 11, e perché i loro pesi sono sempre positivi (Capitolo 18);
- perché i nodi di Chebyshev di Capitolo 9 sono proprio quelli, e non altri (Capitolo 15).

L'ortogonalità rispetto a un peso produce contemporaneamente una ricorrenza a tre termini stabile, zeri reali e separati, la migliore approssimazione in norma pesata e una formula di quadratura di esattezza massima (Capitolo 16).

Le note sono organizzate come segue

1. Legendre
2. Jacobi
3. proprietà strutturali
4. minimi quadrati
5. quadratura di Gauss
6. altre applicazioni.

[15] Gábor Szegő (1895-1985)
[16] Gene Howard Golub (1932-2007)
[17] John H. Welsch



14. Da Gram–Schmidt ai polinomi di Legendre

14.1 Il prodotto scalare pesato sui polinomi

Sia $w : [a, b] \to [0, +\infty)$ una funzione peso integrabile. Definiamo

$$\langle f, g\rangle_w = \int_a^b f(x)g(x)w(x)\,\mathrm{d}x, \qquad \|f\|_w = \sqrt{\langle f, f\rangle_w}.$$

Affinché questa espressione definisca un prodotto scalare pesato definito positivo sullo spazio dei polinomi, sono necessarie due condizioni.

I momenti devono essere finiti. Il k-esimo *momento* del peso è

$$\mu_k = \langle x^k, 1\rangle_w = \int_a^b x^k\, w(x)\,\mathrm{d}x, \tag{14.1}$$

ossia l'integrale del monomio x^k rispetto al peso. La denominazione deriva dalla meccanica: se w è la densità di un'asta, μ_0 ne è la massa, μ_1 il momento statico e μ_2 il momento d'inerzia rispetto all'origine. I momenti devono essere finiti affinché $\langle f, g\rangle_w$ sia finito per ogni coppia di polinomi: sviluppando i due fattori, il prodotto scalare risulta una combinazione lineare di momenti. Su un intervallo limitato, se w è integrabile, questa condizione è automaticamente soddisfatta, poiché $|x|^k \le \max(|a|, |b|)^k$. La condizione è rilevante soprattutto sugli intervalli illimitati, dove il peso deve decadere abbastanza rapidamente. È il caso dei polinomi ortogonali di Hermite e di Laguerre (si veda Capitolo 15.)

Il peso non deve essere nullo. Basta chiedere

$$\int_a^b w(x)\,\mathrm{d}x > 0. \tag{14.2}$$

Serve per avere $\|p\|_w > 0$ per ogni polinomio $p \neq 0$, cioè perché $\|\cdot\|_w$ sia una norma e non solo una seminorma. Che basti si vede così: un polinomio non nullo si annulla al più in un numero finito di punti, quindi $p(x)^2 w(x)$ è positivo ovunque lo sia w, tranne al più in quei pochi punti; se $\int p^2 w$ fosse nullo dovrebbe esserlo anche $\int w$.

> **Nota**
>
> In letteratura si trova spesso la richiesta, più forte, che w sia *positiva quasi ovunque*. Senza scomodare la teoria della misura: vuol dire che w può annullarsi, ma solo su un insieme di punti così piccolo da non cambiare il valore di nessun integrale. Per tutti i pesi di questo

capitolo si tratta di un numero finito di punti, in genere gli estremi dell'intervallo — si pensi a $w(x) = \sqrt{1-x^2}$, nulla in ± 1. Ai fini di quanto segue 14.2 è tutto ciò che occorre.

Una famiglia $p_0(x), p_1(x), \dots$, con $\deg p_n(x) = n$, è **ortogonale** se

$$\langle p_n, p_m \rangle_w = 0 \quad (n \neq m),$$

ed è **ortonormale** se inoltre $\|p_n\|_w = 1$. Il segno e la scala restano liberi finché non si fissa una normalizzazione, per esempio polinomio monico oppure $p_n(1) = 1$.

14.2 Esistenza della famiglia ortogonale

Il prodotto scalare c'è; resta da capire se una famiglia ortogonale esista davvero, come la si costruisce e perché la via più ovvia — ortogonalizzare i monomi — non sia praticabile al calcolatore.

14.2.1 Ortogonalizzazione di Gram–Schmidt

Applicando Gram–Schmidt[1][2] ai monomi $1, x, x^2, \dots$ si ottiene

$$q_0 = \frac{1}{\|1\|_w}, \qquad \widetilde{q}_n = x^n - \sum_{k=0}^{n-1} \langle x^n, q_k \rangle_w \, q_k, \qquad q_n = \frac{\widetilde{q}_n}{\|\widetilde{q}_n\|_w}.$$

Il coefficiente non ha denominatore perché i q_k sono già ortonormali; lavorando con una famiglia ortogonale ma non normalizzata p_k esso diventa $\langle x^n, p_k \rangle_w / \|p_k\|_w^2$.

La formula è una definizione costruttiva e dimostra l'esistenza e l'unicità della famiglia ortonormale una volta fissato il segno del coefficiente principale. Per calcoli con polinomi di grado elevato non si applica direttamente Gram–Schmidt ai monomi: i momenti e le potenze possono essere mal condizionati. Le famiglie classiche evitano questo problema con formule e ricorrenze stabili. Si noti che il rimedio abituale in algebra lineare — Gram–Schmidt modificato o riflessioni di Householder, Capitolo 4 — attenua la perdita di ortogonalità ma non il cattivo condizionamento della *base di partenza*: quindi è necessario cambiare base, non solo algoritmo. Questo è spiegato in maniera precisa in Teorema 14.2 mostrando che il condizionamento della base monomiale cresce esponenzialmente nel grado qualunque sia il peso scelto.

14.2.2 La matrice di Gram e la matrice di Hankel

Che cosa esattamente renda mal condizionata la base monomiale si deduce dalla sua matrice di Gram.

Definition 14.1 — Matrice di Gram. Dati $v_0, v_1, \dots, v_n$ in uno spazio vettoriale con prodotto scalare $\langle \cdot, \cdot \rangle$, la loro **matrice di Gram** è la matrice $G \in \mathfrak{R}^{(n+1)\times(n+1)}$ di elementi

$$G_{ij} = \langle v_i, v_j \rangle, \qquad i, j = 0, 1, \dots, n.$$

$\boldsymbol{G}$ è sempre simmetrica e semidefinita positiva, perché per ogni $c \in \mathfrak{R}^{n+1}$

$$\boldsymbol{c}^T \boldsymbol{G} \boldsymbol{c} = \Big\| \sum_{i=0}^{n} c_i v_i \Big\|^2 \;\geq\; 0,$$

ed è definita positiva se e solo se i v_i sono linearmente indipendenti (si veda per esempio [23]). Il suo condizionamento misura quanto la base è *storta*: $\boldsymbol{G} = \boldsymbol{I}$ esattamente per una base ortonormale, mentre $\kappa_2(\boldsymbol{G})$ grande segnala vettori quasi dipendenti.

[1] Jørgen Pedersen Gram (1850-1916)
[2] Erhard Schmidt (1876-1959)

Per i monomi $v_i = x^i$ accade qualcosa di particolare. Dai momenti 14.1 si ha $\langle x^i, x^j\rangle_w = \mu_{i+j}$: la matrice di Gram dipende solo dalla *somma* degli indici, ed è quindi di un tipo speciale.

Definition 14.2 — Matrice di Hankel. Una matrice $\boldsymbol{H}$ si dice **di Hankel** se H_{ij} dipende solo da $i + j$, cioè se è costante lungo le antidiagonali: esiste una successione $h_0, h_1, \dots, h_{2n}$ con $H_{ij} = h_{i+j}$.

La matrice di Gram dei monomi è dunque la matrice di Hankel dei momenti,

$$H_n = \big(\mu_{i+j}\big)_{i,j=0}^{n}. \tag{14.3}$$

Theorem 14.1 — Esistenza tramite i momenti. Esiste una famiglia di polinomi ortogonali per il peso w se e solo se

$$\det H_n \neq 0 \qquad \text{per ogni } n \geq 0,$$

e in tal caso la famiglia è unica a meno della normalizzazione.

Proof. Diamo l'argomento nel verso che ci serve, cioè costruendo la famiglia; il verso opposto si ottiene ripercorrendo gli stessi passaggi all'indietro (la trattazione completa è in [13]).

Cerchiamo il polinomio di grado n in forma **monica**,

$$\pi_n(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0,$$

e imponiamo che sia ortogonale a tutti i polinomi di grado minore. Non serve imporlo per ognuno di essi: basta farlo sui monomi $1, x, \dots, x^{n-1}$, che ne sono una base. Le condizioni sono dunque

$$\langle \pi_n, x^i\rangle_w = 0, \qquad i = 0, 1, \dots, n-1.$$

Sviluppiamo una di queste condizioni usando la definizione di momento 14.1, cioè $\langle x^i, x^j\rangle_w = \mu_{i+j}$:

$$0 = \langle \pi_n, x^i\rangle_w = \mu_{i+n} + \sum_{j=0}^{n-1} c_j\, \mu_{i+j},$$

ossia, portando a destra il termine noto,

$$\sum_{j=0}^{n-1} \mu_{i+j}\, c_j = -\mu_{i+n}, \qquad i = 0, 1, \dots, n-1. \tag{14.4}$$

Questo è un sistema lineare di n equazioni nelle n incognite $c_0, \dots, c_{n-1}$, e la sua matrice dei coefficienti ha elemento di posto (i, j) uguale a μ_{i+j}: è esattamente H_{n-1}, la matrice di Hankel 14.3. Per l'algebra lineare elementare il sistema ha una e una sola soluzione se e solo se $\det H_{n-1} \neq 0$. Richiedendolo per ogni n si ottengono, uno dopo l'altro, tutti i π_n; ogni altra famiglia ortogonale si ottiene moltiplicando i π_n per costanti non nulle, da cui l'unicità a meno di normalizzazione. ∎

Per un peso positivo la condizione è automaticamente soddisfatta, perché per Definizione 14.1 H_n è la matrice di Gram dei monomi $1, x, \dots, x^n$, che sono linearmente indipendenti: H_n è quindi definita positiva e in particolare invertibile. **L'esistenza non è mai in discussione.**

14.2.3 Il condizionamento della base monomiale

Il problema vero è il condizionamento, e 14.3 lo rende quantitativo: su $[0,1]$ con $w \equiv 1$ si ha $\mu_k = \int_0^1 x^k\,\mathrm{d}x = 1/(k+1)$, cioè

$$(H_n)_{ij} = \frac{1}{i+j+1},$$

che è la matrice di Hilbert, l'esempio canonico di matrice mal condizionata [22]. Risolvere 14.4 — cioè ortogonalizzare i monomi — equivale a risolvere un sistema con quella matrice, ed è per questo che il procedimento degenera già a gradi modesti.

Si potrebbe sperare che il guaio dipenda dal peso scelto e che un altro peso dia una Hankel più docile. Non è così, come mostra il seguente teorema.

Theorem 14.2 — Disuguaglianza di Beckermann. Per ogni matrice di Hankel $\boldsymbol{H}$ definita positiva di ordine $n \geq 3$ vale

$$\kappa_2(\boldsymbol{H}) \;\geq\; \frac{\gamma^{\,n-1}}{16\,n}, \qquad \gamma = e^{4G/\pi} \simeq 3{,}210, \tag{14.5}$$

dove $\kappa_2(\boldsymbol{H}) = \|\boldsymbol{H}\|_2\|\boldsymbol{H}^{-1}\|_2$ è il numero di condizionamento in norma 2 e $G = \sum_{k\geq 0}(-1)^k/(2k+1)^2 \simeq 0{,}916$ è la costante di Catalan. La minorazione è ottimale a meno di un fattore $8\gamma n$.

La dimostrazione, che copre anche le matrici di Vandermonde reali, usa strumenti di teoria del potenziale ed è in [4]. Quello che conta qui è leggerne l'enunciato: $\gamma > 1$, quindi γ^{n-1} cresce **esponenzialmente** in n mentre il denominatore $16n$ cresce solo linearmente. Già per $n = 20$ il membro destro di 14.5 vale circa $1{,}3 \cdot 10^{7}$, e per $n = 30$ supera 10^{12}.

Poiché la matrice di Gram dei monomi è di Hankel qualunque sia il peso, il teorema dice che **nessuna scelta del peso** rende i monomi una base ben condizionata: il condizionamento cresce almeno esponenzialmente nel grado. È la forma precisa dell'avvertimento dato sopra: contro un difetto della base non c'è algoritmo che tenga, bisogna cambiare base.

> **Nota**
>
> La tabella si ferma a $n = 10$ non per brevità: oltre quel grado $\kappa_2(H_n)$ supera 10^{16} e il valore *calcolato* in doppia precisione non è più affidabile. Per $n = 14$ R restituisce $7{,}5 \cdot 10^{17}$, cioè meno che per $n = 12$: il condizionamento della matrice è diventato troppo grande perché lo si possa misurare con la stessa aritmetica che ne soffre.

Table 14.1: Condizionamento della matrice di Hankel dei momenti del peso unitario su [0,1], cioè della matrice di Hilbert: circa un fattore 30 per ogni grado in più, ben oltre il minimo garantito da Teorema 14.2.

n	condizionamento
2	5.24e+02
4	4.77e+05
6	4.75e+08
8	4.93e+11
10	5.22e+14

14.2.4 Esempio svolto: costruire P_2 a mano

Lavoriamo su $[-1,1]$ con peso $w \equiv 1$. I primi due passi sono immediati:

$$\widetilde{q}_0(x) = 1, \qquad \widetilde{q}_1(x) = x,$$

dove $\widetilde{q}_1$ non va corretto perché è già ortogonale a $\widetilde{q}_0$: l'integrando x è dispari su un intervallo simmetrico. Per il grado due Gram–Schmidt richiede invece di sottrarre due componenti:

$$\widetilde{q}_2(x) = x^2 - \frac{\langle x^2, 1\rangle_w}{\langle 1, 1\rangle_w}\cdot 1 - \frac{\langle x^2, x\rangle_w}{\langle x, x\rangle_w}\cdot x.$$

I prodotti scalari che vi compaiono sono tre integrali elementari:

$$\langle x^2, 1\rangle_w = \int_{-1}^{1} x^2\,\mathrm{d}x = \frac{2}{3},$$

$$\langle 1, 1\rangle_w = \int_{-1}^{1} \mathrm{d}x = 2,$$

$$\langle x^2, x\rangle_w = \int_{-1}^{1} x^3\,\mathrm{d}x = 0 \qquad \text{(integrando dispari)}.$$

L'ultimo è nullo, quindi la componente lungo x non c'è e resta

$$\widetilde{q}_2(x) = x^2 - \frac{2/3}{2}\cdot 1 = x^2 - \frac{1}{3}.$$

Manca solo la normalizzazione. La convenzione dei Legendre è $P_2(1) = 1$, mentre

$$\widetilde{q}_2(1) = 1 - \frac{1}{3} = \frac{2}{3},$$

quindi si moltiplica per $3/2$:

$$P_2(x) = \frac{3}{2}\left(x^2 - \frac{1}{3}\right) = \frac{1}{2}\left(3x^2 - 1\right).$$

È lo stesso polinomio che la formula di Rodrigues produce per derivazione.

14.3 I polinomi di Legendre

Fissiamo ora il caso più semplice e più usato: intervallo $[-1, 1]$ e peso costante. La famiglia che ne risulta ha un nome, una formula chiusa, una ricorrenza e un'equazione differenziale, e le incontriamo tutte in questa sezione.

14.3.1 Definizione e primi polinomi

Su $[-1, 1]$ scegliamo il peso costante $w(x) = 1$. I polinomi di Legendre[3] $P_n(x)$ sono normalizzati da $P_n(1) = 1$ e soddisfano

$$\int_{-1}^{1} P_n(x) P_m(x)\,\mathrm{d}x = \frac{2}{2n+1}\,\delta_{nm}.$$

Una definizione esplicita è la formula di Rodrigues[4]:

$$P_n(x) = \frac{1}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}(x^2 - 1)^n. \tag{14.6}$$

I primi sei polinomi, quelli disegnati in Figura 14.1, sono

[3] Adrien-Marie Legendre (1752-1833)
[4] Benjamin Olinde Rodrigues (1795-1851)

$$
\begin{aligned}
P_0(x) &= 1,\\
P_1(x) &= x,\\
P_2(x) &= \frac{1}{2}\left(3x^2 - 1\right),\\
P_3(x) &= \frac{1}{2}\left(5x^3 - 3x\right),\\
P_4(x) &= \frac{1}{8}\left(35x^4 - 30x^2 + 3\right),\\
P_5(x) &= \frac{1}{8}\left(63x^5 - 70x^3 + 15x\right).
\end{aligned}
$$

14.3.2 La ricorrenza a tre termini

Rodrigues è una definizione, non un algoritmo: derivare n volte un polinomio di grado $2n$ è costoso e numericamente pessimo. Per il calcolo si usa la *ricorrenza a tre termini*

$$
\begin{aligned}
P_0(x) &= 1,\\
P_1(x) &= x,\\
(k+1)\,P_{k+1}(x) &= (2k+1)\,x\,P_k(x) - k\,P_{k-1}(x).
\end{aligned}
\tag{14.7}
$$

Che una ricorrenza di questo tipo esista non è un caso fortunato: in Capitolo 15 si vede che l'ortogonalità *obbliga* $x\,p_n$ ad avere componenti non nulle solo su p_{n-1}, p_n, p_{n+1}, e in Capitolo 16 se ne ricava la forma generale. I coefficienti di 14.7 sono il caso $\alpha = \beta = 0$ della ricorrenza di Jacobi.

Una verifica immediata. Per $k = 1$ la ricorrenza dà

$$
2P_2(x) = 3x \cdot x - 1, \qquad \text{cioè} \qquad P_2(x) = \frac{1}{2}\left(3x^2 - 1\right),
$$

e per $k = 2$

$$
3P_3(x) = 5x \cdot \frac{1}{2}\left(3x^2 - 1\right) - 2x, \qquad \text{cioè} \qquad P_3(x) = \frac{1}{2}\left(5x^3 - 3x\right),
$$

che sono i polinomi elencati sopra.

Tradotta in codice, 14.7 costa $O(n)$ operazioni e memoria costante, e non richiede mai i coefficienti monomiali. La funzione seguente è definita in `_common.R` ed è usata da tutti gli esempi del capitolo:

```
legendre <- function(n, x) {
  if (n == 0) return(rep(1, length(x)))
  p0 <- rep(1, length(x)); p1 <- x
  if (n == 1) return(p1)
  for (k in 1:(n - 1)) {
    p2 <- ((2*k + 1)*x*p1 - k*p0)/(k + 1)
    p0 <- p1; p1 <- p2
  }
  p1
}
```

Il comportamento in virgola mobile di questa ricorrenza è discusso in Capitolo 16.

14.3.3 Rodrigues e la dimostrazione dell'ortogonalità

Theorem 14.3 — Ortogonalità e norma dei polinomi di Legendre. Il polinomio P_n definito dalla formula di Rodrigues è ortogonale a ogni polinomio di grado minore di n e soddisfa

$$\int_{-1}^{1} P_n(x)^2 \, \mathrm{d}x = \frac{2}{2n+1}.$$

Proof. Conviene dare un nome alla funzione che si deriva:

$$u(x) = (x^2 - 1)^n = (x-1)^n (x+1)^n,$$

cosicché

$$P_n(x) = \frac{1}{2^n \, n!} \, u^{(n)}(x).$$

Passo 1: $u(x)$ e le sue derivate fino all'ordine $n-1$ si annullano in ± 1. Applichiamo la formula di Leibniz per la derivata di un prodotto:

$$u^{(j)}(x) = \sum_{i=0}^{j} \binom{j}{i} \frac{\mathrm{d}^i}{\mathrm{d}x^i}\left[(x-1)^n\right] \cdot \frac{\mathrm{d}^{j-i}}{\mathrm{d}x^{j-i}}\left[(x+1)^n\right].$$

Derivare i volte $(x-1)^n$ dà

$$\frac{\mathrm{d}^i}{\mathrm{d}x^i}(x-1)^n = \frac{n!}{(n-i)!}\,(x-1)^{n-i},$$

che contiene ancora il fattore $(x-1)$ ogni volta che $i < n$. Se $j \le n-1$, in ogni addendo della somma si ha $i \le j < n$: *tutti* gli addendi contengono $(x-1)$ e quindi si annullano in $x = 1$. Scambiando il ruolo dei due fattori si ragiona allo stesso modo in $x = -1$. Dunque

$$u^{(j)}(\pm 1) = 0, \qquad j = 0, 1, \dots, n-1. \tag{14.8}$$

Passo 2: integrare per parti n volte. Sia $r(x)$ un polinomio qualunque. Un singolo passo di integrazione per parti dà

$$\int_{-1}^{1} r\, u^{(n)} \, \mathrm{d}x = \Big[\, r\, u^{(n-1)} \Big]_{-1}^{1} - \int_{-1}^{1} r'\, u^{(n-1)} \, \mathrm{d}x = -\int_{-1}^{1} r'\, u^{(n-1)} \, \mathrm{d}x,$$

dove il termine di bordo è nullo per 14.8. L'integrale che resta ha la stessa forma di quello di partenza, con una derivata in meno su u e una in più su r. Ripetendo il procedimento n volte — e a ogni passo il termine di bordo è nullo, sempre per 14.8 — si ottiene

$$\int_{-1}^{1} r(x)\, u^{(n)}(x) \, \mathrm{d}x = (-1)^n \int_{-1}^{1} r^{(n)}(x)\, u(x) \, \mathrm{d}x, \tag{14.9}$$

dove il segno $(-1)^n$ raccoglie il meno che compare a ciascuno degli n passi.

Passo 3: l'ortogonalità. Se $\deg r < n$, derivando r per n volte si ottiene la funzione identicamente nulla, $r^{(n)} \equiv 0$, e quindi il secondo membro di 14.9 è zero. Dividendo per la costante $2^n \, n!$,

$$\int_{-1}^{1} r(x)\, P_n(x) \, \mathrm{d}x = 0 \qquad \text{per ogni } r \text{ di grado } < n.$$

In particolare, scegliendo $r(x) = P_m(x)$ con $m < n$, si ha $\langle P_n, P_m \rangle = 0$: i polinomi di Legendre sono a due a due ortogonali.

Passo 4: il coefficiente direttore di $P_n(x)$. Serve per la norma. Scriviamo $u(x) = (x^2-1)^n = x^{2n} + \cdots$, dove i puntini indicano termini di grado minore di $2n$. Derivando n volte, il termine di grado massimo diventa

$$\frac{\mathrm{d}^n}{\mathrm{d}x^n} x^{2n} = 2n\,(2n-1)\cdots(n+1)\,x^n = \frac{(2n)!}{n!}\,x^n,$$

mentre tutti gli altri danno potenze di grado minore. Dividendo per $2^n n!$, il coefficiente direttore di $P_n(x)$ è

$$a_n = \frac{(2n)!}{2^n\,(n!)^2}.$$

Poiché $P_n(x)$ ha grado esattamente n, derivandolo n volte resta la costante

$$P_n^{(n)}(x) = n!\,a_n = \frac{(2n)!}{2^n\,n!}.$$

Passo 5: un integrale ausiliario. Poniamo

$$I_n = \int_{-1}^{1} (1-x^2)^n\,\mathrm{d}x.$$

Integriamo per parti derivando $(1-x^2)^n$ e integrando il fattore 1:

$$I_n = \Big[x\,(1-x^2)^n\Big]_{-1}^{1} + 2n\int_{-1}^{1} x^2(1-x^2)^{n-1}\,\mathrm{d}x = 2n\int_{-1}^{1} x^2(1-x^2)^{n-1}\,\mathrm{d}x,$$

perché $(1-x^2)^n$ si annulla in ± 1. Scrivendo ora $x^2 = 1-(1-x^2)$, l'ultimo integrale si spezza in $I_{n-1} - I_n$, e si ottiene $I_n = 2n\,(I_{n-1} - I_n)$, cioè la ricorrenza

$$I_n = \frac{2n}{2n+1}\,I_{n-1}, \qquad I_0 = \int_{-1}^{1}\mathrm{d}x = 2.$$

Moltiplicando fra loro queste uguaglianze per $k = 1, 2, \dots, n$,

$$I_n = 2\cdot\frac{2\cdot 4\cdots(2n)}{3\cdot 5\cdots(2n+1)} = \frac{2^{2n+1}\,(n!)^2}{(2n+1)!}.$$

Passo 6: la norma. Applichiamo 14.9 con $r(x) = P_n(x)$, che ha grado n:

$$\int_{-1}^{1} P_n(x)^2\,\mathrm{d}x = \frac{1}{2^n n!}\int_{-1}^{1} P_n\,u^{(n)}\,\mathrm{d}x = \frac{(-1)^n}{2^n n!}\int_{-1}^{1} P_n^{(n)}\,u\,\mathrm{d}x.$$

Per il Passo 4 la quantità $P_n^{(n)}$ è una costante e si porta fuori dall'integrale; inoltre $(-1)^n u(x) = (1-x^2)^n$, quindi ciò che resta è esattamente I_n:

$$\int_{-1}^{1} P_n^2\,\mathrm{d}x = \frac{1}{2^n n!}\cdot\frac{(2n)!}{2^n n!}\cdot I_n = \frac{(2n)!}{2^{2n}(n!)^2}\cdot\frac{2^{2n+1}(n!)^2}{(2n+1)!} = \frac{2\,(2n)!}{(2n+1)!} = \frac{2}{2n+1},$$

dove nell'ultimo passaggio si è usato $(2n+1)! = (2n+1)\,(2n)!$. ∎

Di conseguenza

$$\phi_n(x) = \sqrt{\frac{2n+1}{2}}\,P_n(x)$$

è la versione ortonormale.

14.3.4 Equazione differenziale e funzione generatrice

I polinomi di Legendre soddisfano l'equazione:

$$\frac{\mathrm{d}}{\mathrm{d}x}\left[(1-x^2)P_n'(x)\right] + n(n+1)P_n(x) = 0.$$

Questa equazione segue anche in senso inverso: la soluzione polinomiale di grado n, normalizzata a 1 in $x = 1$, è P_n. La funzione generatrice compatta tutti i gradi in un'unica identità:

$$\frac{1}{\sqrt{1-2xt+t^2}} = \sum_{n=0}^{\infty} P_n(x)\,t^n, \qquad |t| < 1, \quad -1 \le x \le 1.$$

Due conseguenze utili sono la simmetria $P_n(-x) = (-1)^n P_n(x)$ e la formula ai capi $P_n(1) = 1$, $P_n(-1) = (-1)^n$.

14.3.5 La famiglia in un grafico

La Figura 14.1 sovrappone i primi polinomi: si vede la parità alternata, il numero crescente di oscillazioni e il valore unitario comune all'estremo destro. Gli zeri sono interni e distinti; la loro separazione sarà dimostrata in Sezione 16.2.

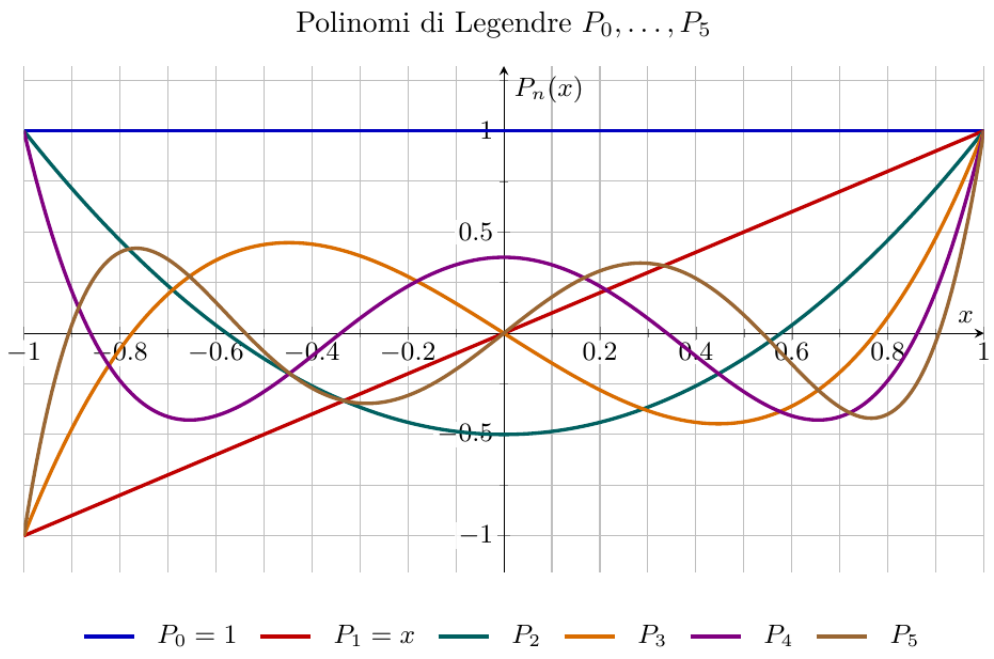


Figure 14.1: Polinomi di Legendre di grado 0–5 su $[-1, 1]$. La figura evidenzia simmetria, oscillazioni e comportamento agli estremi.

Una seconda figura verifica numericamente l'ortogonalità calcolando la matrice di Gram con quadratura composta del punto medio. Nella Figura 14.2 ogni cella riporta il valore di $G_{ij} = \langle P_i, P_j \rangle$: sulla diagonale si leggono i valori $2/(2n+1)$, cioè 2, 0.6667, 0.4, 0.2857, 0.2222, 0.1818, mentre fuori diagonale si trovano numeri dell'ordine di 10^{-8} o minori, cioè zero a meno dell'errore di quadratura.

```
m <- 12000
x <- -1 + (seq_len(m) - 0.5)*2/m
P <- sapply(0:5, legendre, x = x)
G <- crossprod(P)*(2/m)
```

```
d <- 5

par(mar = c(0.6, 3.6, 3.4, 0.6))
# scala radice: comprime il salto fra 2.0000 e 0.1818 senza colorare gli zeri
S   <- sqrt(abs(G)/max(abs(G)))
pal <- colorRampPalette(c("white", "#dbe9f6", "#92c5de",
                          "#3182bd", "#08519c"))(40)
image(0:d, 0:d, t(S[(d+1):1, ]), col = pal,
      breaks = seq(0, 1, length.out = 41),
      axes = FALSE, xlab = "", ylab = "")
axis(3, at = 0:d, tick = FALSE, line = -0.8,
     labels = sapply(0:d, function(k) as.expression(bquote(P[.(k)]))))
axis(2, at = 0:d, tick = FALSE, las = 1, line = -0.8,
     labels = sapply(d:0, function(k) as.expression(bquote(P[.(k)]))))
mtext("Matrice di Gram dei polinomi di Legendre", side = 3, line = 1.9,
      cex = 1.05, font = 2)
abline(h = (0:d) + 0.5, v = (0:d) + 0.5, col = "grey82")
box(col = "grey55")
for (i in 0:d) for (j in 0:d) {
  v <- G[i+1, j+1]
  text(j, d - i,
       if (i == j) sprintf("%.4f", v) else sprintf("%.0e", v),
       cex = 0.80, col = if (S[i+1, j+1] > 0.55) "white" else "grey25")
}
```

Matrice di Gram dei polinomi di Legendre

	P_0	P_1	P_2	P_3	P_4	P_5
P_0	2.0000	−3e−17	−7e−09	−3e−17	−2e−08	−4e−17
P_1	−3e−17	0.6667	−4e−17	−2e−08	4e−17	−4e−08
P_2	−7e−09	−4e−17	0.4000	−3e−17	−3e−08	−5e−18
P_3	−3e−17	−2e−08	−3e−17	0.2857	−2e−17	−5e−08
P_4	−2e−08	4e−17	−3e−08	−2e−17	0.2222	3e−19
P_5	−4e−17	−4e−08	−5e−18	−5e−08	3e−19	0.1818

Figure 14.2: Matrice di Gram dei polinomi di Legendre di grado 0–5, calcolata per quadratura. L'intensità del blu cresce con $\sqrt{|G_{ij}|}$: le celle bianche sono numericamente nulle, quindi resta colorata solo la diagonale, che vale $2/(2n+1)$.

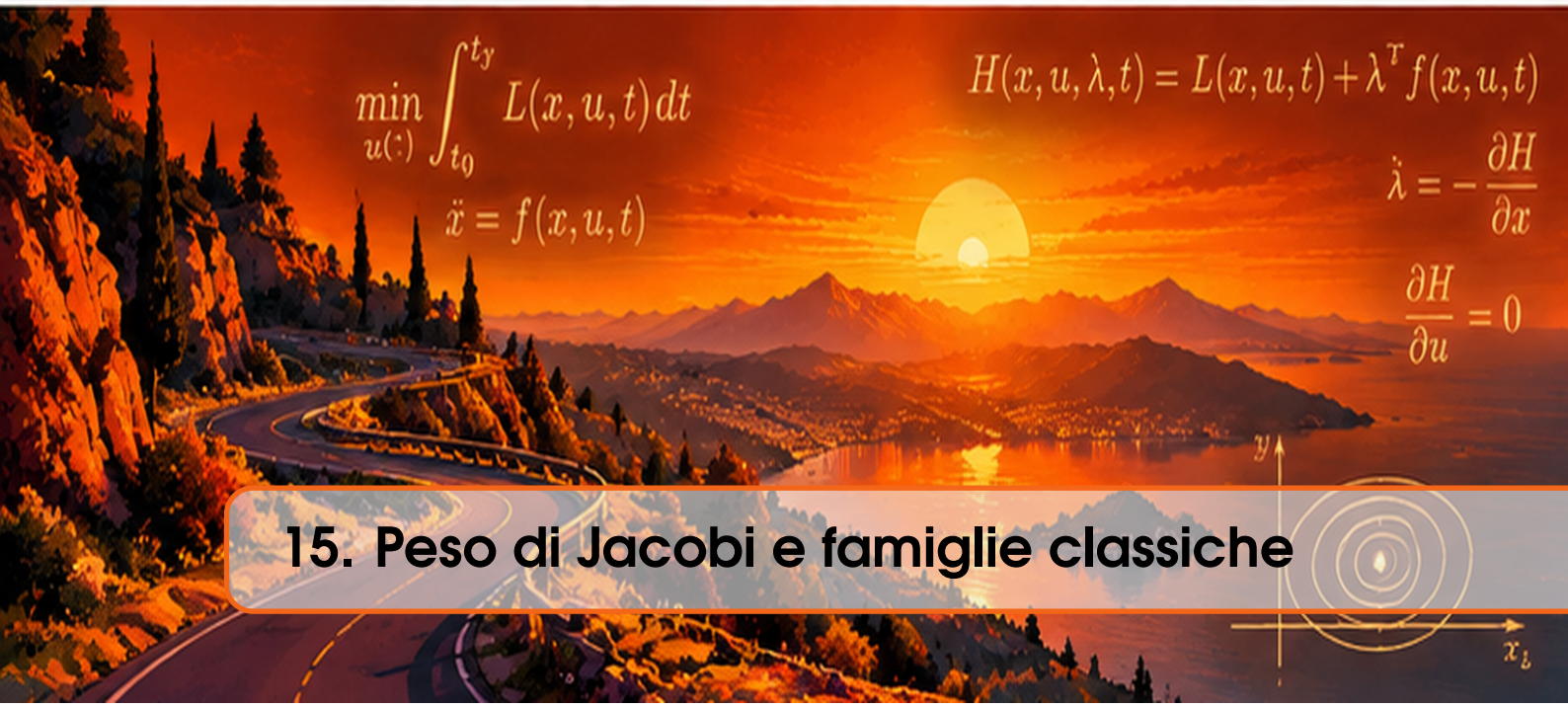
> **Nota**
>
> La figura è un controllo numerico, non una prova dell'ortogonalità: il ragionamento analitico usa Rodrigues. I numeri fuori diagonale non sono esattamente nulli perché l'integrale è stato sostituito da una somma finita: raffinando la quadratura (aumentando m) tendono a zero.

In sintesi

- Un peso positivo w definisce un prodotto scalare sui polinomi; Gram–Schmidt sui monomi dimostra che la famiglia ortogonale esiste ed è unica a meno di normalizzazione, ma non è un buon algoritmo.
- Su $[-1, 1]$ con $w \equiv 1$ si ottengono i polinomi di Legendre, normalizzati da $P_n(1) = 1$, con $\|P_n\|_2^2 = 2/(2n+1)$.
- Rodrigues serve per *dimostrare* l'ortogonalità; la ricorrenza a tre termini 14.7 serve per *calcolare*, in $O(n)$.
- Parità, valori agli estremi e zeri interni seguono dalla simmetria del peso e dall'ortogonalità, non da conti espliciti.

14.4 Esercizi

1. Usando la formula di Rodrigues calcolare $P_0(x)$, $P_1(x)$ e $P_2(x)$, inoltre verificarne i valori in ± 1.
2. Usare la parità per dedurre che $\int_{-1}^{1} x P_{2k}(x)\,\mathrm{d}x = 0$.
3. Spiegare perché la norma dei polinomi ortonormali è uno, mentre $\|P_n\|_2^2 = 2/(2n+1)$.
4. Verificare 14.7 per $k = 3$ confrontando il risultato con l'espressione esplicita di P_4.



15. Peso di Jacobi e famiglie classiche

15.1 Il peso di Jacobi

15.1.1 Definizione e condizione sui parametri

La nozione di ortogonalità non richiede che la funzione peso sia costante. Per $\alpha, \beta > -1$ poniamo

$$w_{\alpha,\beta}(x) = (1-x)^{\alpha}(1+x)^{\beta}, \qquad -1 < x < 1.$$

La condizione che entrambi gli esponenti siano maggiori di -1 garantisce l'integrabilità delle singolarità agli estremi. I polinomi di Jacobi[1] $P_n^{(\alpha,\beta)}$ sono ortogonali per questo peso e adottano la normalizzazione convenzionale $P_n^{(\alpha,\beta)}(1) = \binom{n+\alpha}{n}$. La formula di Rodrigues è

$$P_n^{(\alpha,\beta)}(x) = \frac{(-1)^n}{2^n n!}(1-x)^{-\alpha}(1+x)^{-\beta}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[(1-x)^{n+\alpha}(1+x)^{n+\beta}\right].$$

Mediante n integrazioni per parti si dimostra l'ortogonalità rispetto a ogni polinomio di grado minore di n.

15.1.2 La funzione Gamma

Per scrivere la norma serve una funzione che, con α, β non interi, prende il posto del fattoriale.

Definition 15.1 — Funzione Gamma. La **funzione Gamma** di Eulero[a] è definita per $z > 0$ dall'integrale

$$\Gamma(z) = \int_0^{+\infty} t^{z-1}\mathrm{e}^{-t}\,\mathrm{d}t, \tag{15.1}$$

e si estende a tutti i z reali non interi ≤ 0 richiedendo che continui a valere la proprietà 15.2 qui sotto.

[a] Leonhard Euler (1707-1783)

L'integrale 15.1 converge: in $t = 0$ perché t^{z-1} è integrabile per $z > 0$, all'infinito perché e^{-t} schiaccia qualunque potenza. Servono solo tre proprietà.

- **Ricorrenza.** Integrando per parti 15.1 si ottiene

$$\Gamma(z+1) = z\,\Gamma(z). \tag{15.2}$$

[1] Carl Gustav Jacob Jacobi (1804-1851)

- **Interpolazione del fattoriale.** Poiché $\Gamma(1) = \int_0^{+\infty} \mathrm{e}^{-t}\,\mathrm{d}t = 1$, applicando ripetutamente 15.2 si ricava

$$\Gamma(n+1) = n! \qquad \text{per ogni intero } n \geq 0.$$

  La Gamma è dunque il fattoriale esteso agli argomenti non interi: è esattamente ciò che serve quando α e β sono reali.
- **Un valore non intero.** Con la sostituzione $t = u^2$ in 15.1 si ritrova l'integrale di Gauss e quindi

$$\Gamma\left(\tfrac{1}{2}\right) = \sqrt{\pi},$$

  valore che compare nelle formule dei Chebyshev.

La 15.2, letta da destra a sinistra come $\Gamma(z) = \Gamma(z+1)/z$, definisce Γ per $z < 0$ non intero e mostra che la funzione ha un polo in $z = 0, -1, -2, \ldots$, dove il denominatore si annulla. Non ci sono invece zeri: $1/\Gamma$ è ovunque finita. Per approfondire si veda [2] oppure [1].

15.1.3 La norma dei polinomi di Jacobi

Con questa notazione la norma, nella convenzione scelta sopra, è

$$h_n^{(\alpha,\beta)} = \|P_n^{(\alpha,\beta)}\|^2_{w_{\alpha,\beta}} = \frac{2^{\alpha+\beta+1}}{2n+\alpha+\beta+1}\,\frac{\Gamma(n+\alpha+1)\Gamma(n+\beta+1)}{n!\,\Gamma(n+\alpha+\beta+1)}. \tag{15.3}$$

Gli argomenti $n+\alpha+1$ e $n+\beta+1$ sono positivi perché $\alpha, \beta > -1$, quindi le due Gamma al numeratore sono ben definite. L'unico caso delicato è $n = 0$ con $\alpha+\beta+1 \leq 0$: allora il fattore $1/(2n+\alpha+\beta+1)$ e il fattore $1/\Gamma(\alpha+\beta+1)$ sono uno infinito e l'altro nullo, e la 15.2 scioglie l'indeterminazione raggruppandoli,

$$(\alpha+\beta+1)\,\Gamma(\alpha+\beta+1) = \Gamma(\alpha+\beta+2).$$

Per i Chebyshev di prima specie ($\alpha = \beta = -\tfrac{1}{2}$) si ottiene così

$$h_0^{(-1/2,-1/2)} = \frac{2^0\,\Gamma(1/2)^2}{\Gamma(1)} = \pi,$$

in accordo con il calcolo diretto $\int_{-1}^{1}(1-x^2)^{-1/2}\,\mathrm{d}x = \pi$.

15.2 Le famiglie classiche come casi particolari

15.2.1 La tabella dei casi speciali

I casi simmetrici hanno $\alpha = \beta$ e peso pari. I Chebyshev vengono spesso definiti con normalizzazioni diverse da quella dei Jacobi: la tabella esplicita sia il peso, dato a meno di una costante moltiplicativa, sia il fattore di scala.

Famiglia	Parametri Jacobi	Peso	Relazione
Legendre P_n	$\alpha = \beta = 0$	1	$P_n^{(0,0)} = P_n$
Gegenbauer $C_n^{(\lambda)}$	$\alpha = \beta = \lambda - \tfrac{1}{2}$, $\lambda > -\tfrac{1}{2}$	$(1-x^2)^{\lambda-1/2}$	$C_n^{(\lambda)} = \dfrac{(2\lambda)_n}{(\lambda+1/2)_n} P_n^{(\lambda-1/2,\lambda-1/2)}$
Chebyshev I T_n	$\alpha = \beta = -\tfrac{1}{2}$	$(1-x^2)^{-1/2}$	$T_n = \dfrac{n!}{(1/2)_n} P_n^{(-1/2,-1/2)}$
Chebyshev II U_n	$\alpha = \beta = \tfrac{1}{2}$	$(1-x^2)^{1/2}$	$U_n = \dfrac{(n+1)!}{(3/2)_n} P_n^{(1/2,1/2)}$

Qui $(a)_n = a(a+1)\cdots(a+n-1)$ è il simbolo di Pochhammer.

15.2.2 Chebyshev: coseni travestiti da polinomi

Il caso $\alpha = \beta = -\frac{1}{2}$ merita una nota a parte. Con la sostituzione $x = \cos\theta$ si verifica che

$$T_n(\cos\theta) = \cos(n\theta), \tag{15.4}$$

cioè i Chebyshev[2] sono coseni travestiti da polinomi. Da 15.4 seguono senza conti tre fatti.

- La limitazione uniforme

$$|T_n(x)| \le 1 \qquad \text{per ogni } x \in [-1,1],$$

  immediata perché il coseno non esce da $[-1,1]$.
- Gli zeri, che si leggono dagli zeri del coseno:

$$x_j = \cos\frac{(2j-1)\pi}{2n}, \qquad j = 1, 2, \dots, n.$$

  Sono i nodi di Chebyshev già incontrati in Capitolo 9.
- La possibilità di passare da valori nodali a coefficienti con una trasformata coseno veloce, in $O(n\log n)$ anziché gli $O(n^2)$ di una quadratura generica.

È soprattutto per l'ultimo che, nella pratica numerica, Chebyshev è spesso preferito a Legendre a parità di qualità dell'approssimazione.

15.2.3 La proprietà di minimo dei polinomi di Chebyshev

C'è un secondo motivo, indipendente dall'ortogonalità, per cui i Chebyshev compaiono ovunque: fra tutti i polinomi monici di un dato grado sono quelli che oscillano *il meno possibile*. Prima di enunciare il risultato conviene fissare due parole.

- Un polinomio di grado n è **monico** se il coefficiente di x^n vale 1, cioè se si scrive

$$p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0.$$

- Per una funzione continua f su $[-1,1]$ la **norma uniforme** è

$$\|f\|_\infty = \max_{-1\le x\le 1} |f(x)|,$$

  cioè la massima ampiezza raggiunta sull'intervallo.

Il vincolo «monico» non è un dettaglio tecnico: senza di esso la domanda non avrebbe senso, perché il polinomio $p \equiv 0$ darebbe $\|p\|_\infty = 0$ e vincerebbe sempre. Fissare a 1 il coefficiente di x^n impedisce di abbassare l'ampiezza semplicemente moltiplicando tutto per una costante piccola: la gara si svolge fra polinomi della stessa taglia.

Un esempio concreto con $n = 3$ chiarisce la posta in gioco. Il monico più ovvio è $q(x) = x^3$, che su $[-1,1]$ arriva fino a $\|q\|_\infty = 1$. Il monico di Chebyshev è invece

$$\widetilde{T}_3(x) = \tfrac{1}{4}T_3(x) = \tfrac{1}{4}(4x^3 - 3x) = x^3 - \tfrac{3}{4}x,$$

e la sottrazione del termine $\frac{3}{4}x$ abbassa il massimo a $\|\widetilde{T}_3\|_\infty = \frac{1}{4}$: quattro volte meno, a parità di termine di testa. Il teorema dice che meglio di così non si può fare.

Theorem 15.1 — Minimo della norma uniforme fra i polinomi monici. Sia $n \ge 1$ e sia

$$\widetilde{T}_n(x) = 2^{1-n}\,T_n(x) \tag{15.5}$$

il polinomio di Chebyshev monico di grado n. Allora

[2] Pafnutij L'vovič Čebyšëv (1821-1894)

$$\|\widetilde{T}_n\|_\infty = 2^{1-n},$$

e per **ogni** polinomio monico q di grado n vale

$$\|q\|_\infty \;\geq\; 2^{1-n},$$

con uguaglianza se e solo se $q = \widetilde{T}_n$.

Proof. La dimostrazione si articola in cinque passi: i primi due verificano che $\widetilde{T}_n$ ha davvero le proprietà annunciate, gli ultimi tre mostrano che nessun altro monico fa meglio. Poniamo per brevità

$$M = 2^{1-n}.$$

Passo 1: $\widetilde{T}_n$ è monico. Occorre sapere che il coefficiente di x^n in T_n vale 2^{n-1} per $n \geq 1$. Lo si ottiene dalla formula di addizione del coseno

$$\cos\big((n+1)\theta\big) + \cos\big((n-1)\theta\big) = 2\cos\theta\,\cos(n\theta),$$

che, posto $x = \cos\theta$ e usando 15.4, diventa la ricorrenza a tre termini

$$T_{n+1}(x) = 2x\,T_n(x) - T_{n-1}(x), \qquad T_0(x) = 1, \quad T_1(x) = x. \tag{15.6}$$

Indichiamo con a_n il coefficiente di x^n in T_n. Il prodotto $2x\,T_n$ alza il grado di uno e raddoppia il coefficiente di testa, mentre T_{n-1} ha grado $n-1$ e non contribuisce al termine di grado $n+1$: dunque $a_{n+1} = 2a_n$. Siccome $a_1 = 1$, per induzione

$$a_n = 2^{n-1}, \qquad n \geq 1.$$

Moltiplicando T_n per 2^{1-n}, come in 15.5, il coefficiente di testa diventa $2^{1-n}\cdot 2^{n-1} = 1$: $\widetilde{T}_n$ è monico, come richiesto.

Passo 2: $\widetilde{T}_n$ oscilla fra $-M$ e $+M$ toccando gli estremi $n+1$ volte. Da $|T_n(x)| \leq 1$ su $[-1,1]$ segue subito

$$|\widetilde{T}_n(x)| \leq M \qquad \text{per ogni } x \in [-1,1].$$

Consideriamo ora gli $n+1$ punti

$$y_j = \cos\frac{j\pi}{n}, \qquad j = 0, 1, \dots, n. \tag{15.7}$$

Poiché il coseno è strettamente decrescente su $[0,\pi]$ e gli angoli $j\pi/n$ crescono, i punti sono ordinati in senso stretto:

$$1 = y_0 > y_1 > \cdots > y_n = -1.$$

Per 15.4 si ha $T_n(y_j) = \cos(j\pi) = (-1)^j$, e quindi

$$\widetilde{T}_n(y_j) = (-1)^j\,M, \qquad j = 0, 1, \dots, n. \tag{15.8}$$

Il valore massimo M viene dunque raggiunto in $n+1$ punti, con segni alternati: è il fenomeno di *equioscillazione* illustrato in Figura 15.1. In particolare $\|\widetilde{T}_n\|_\infty = M$, che è la prima affermazione del teorema.

Passo 3: la differenza con un concorrente. Sia ora q un polinomio monico di grado n che soddisfa

$$\|q\|_\infty \le M, \tag{15.9}$$

e poniamo

$$d = \widetilde{T}_n - q.$$

Di d ci servono due proprietà.

Il grado cala. Entrambi i polinomi sono monici di grado n, quindi i loro termini x^n si cancellano nella differenza:

$$\deg d \le n - 1.$$

I segni si alternano. Moltiplichiamo $d(y_j)$ per $(-1)^j$ e procediamo un passaggio alla volta:

$$\begin{aligned}
(-1)^j\, d(y_j) &= (-1)^j\, \widetilde{T}_n(y_j) - (-1)^j\, q(y_j) && \text{definizione di } d,\\
&= M - (-1)^j\, q(y_j) && \text{perché } \widetilde{T}_n(y_j) = (-1)^j M,\\
&\ge M - \bigl|q(y_j)\bigr| && \text{perché } (-1)^j\, q(y_j) \le \bigl|q(y_j)\bigr|,\\
&\ge 0 && \text{perché } \|q\|_\infty \le M.
\end{aligned} \tag{15.10}$$

La seconda riga usa l'equioscillazione 15.8, l'ultima l'ipotesi 15.9. A parole: $d(y_0) \ge 0$, $d(y_1) \le 0$, $d(y_2) \ge 0$, e così via, alternando.

Passo 4: d ha almeno n zeri, contati con molteplicità. Consideriamo gli n intervallini $[y_{j+1}, y_j]$ con $j = 0, 1, \dots, n-1$ determinati dai punti 15.7.

(a) In ciascun intervallino c'è almeno uno zero. Per 15.10 i valori $d(y_j)$ e $d(y_{j+1})$ hanno segni opposti in senso debole. Se uno dei due è nullo, lo zero è l'estremo stesso. Altrimenti i segni sono strettamente opposti e, essendo d continua, il teorema degli zeri fornisce uno zero interno all'intervallino.

(b) Uno zero che cade in un punto interno y_j (cioè con $0 < j < n$) è almeno doppio. Supponiamo $d(y_j) = 0$, ossia $q(y_j) = (-1)^j M$. Allora la funzione q^2, che è derivabile, soddisfa $q^2 \le M^2$ su tutto $[-1, 1]$ e raggiunge il valore M^2 nel punto y_j, che è interno all'intervallo: per il teorema di Fermat la sua derivata vi si annulla,

$$(q^2)'(y_j) = 2\, q(y_j)\, q'(y_j) = 0,$$

e poiché $q(y_j) = \pm M \neq 0$ resta $q'(y_j) = 0$. Lo stesso argomento applicato a $\widetilde{T}_n$, che per 15.8 tocca anch'esso il massimo in y_j, dà $\widetilde{T}_n'(y_j) = 0$. Sottraendo, $d'(y_j) = 0$: in y_j si annullano sia d sia d', cioè y_j è uno zero di molteplicità almeno due.

(c) Il conteggio. Assegniamo a ciascuno degli n intervallini uno zero in esso contenuto, che esiste per il punto (a). Uno stesso zero può essere assegnato a due intervallini soltanto se è un loro estremo comune, cioè un punto y_j con $0 < j < n$; ma per il punto (b) un tale zero ha molteplicità almeno due. Sommando le molteplicità degli zeri distinti così trovati si ottiene dunque almeno n.

Passo 5: conclusione. Un polinomio non identicamente nullo di grado $\le n-1$ ha al più $n-1$ zeri contati con molteplicità. Il Passo 4 ne esibisce almeno n, e l'unica via d'uscita è

$$d \equiv 0, \qquad \text{cioè} \qquad q = \widetilde{T}_n.$$

Abbiamo quindi dimostrato che *l'unico* monico di grado n che soddisfa 15.9 è $\widetilde{T}_n$. Da qui seguono entrambe le affermazioni rimaste:

- nessun monico q può avere $\|q\|_\infty < M$, perché soddisfarebbe a maggior ragione 15.9 e sarebbe allora $\widetilde{T}_n$, la cui norma vale però esattamente M;
- se $\|q\|_\infty = M$ allora 15.9 vale e quindi $q = \widetilde{T}_n$: il minimo è raggiunto da un solo polinomio.

∎

i Perché proprio l'equioscillazione

L'idea geometrica dietro il Passo 4 è questa: un polinomio monico che battesse $\widetilde{T}_n$ dovrebbe restare, in valore assoluto, sotto la quota M su tutto l'intervallo. Ma $\widetilde{T}_n$ tocca alternativamente $+M$ e $-M$ in $n+1$ punti, e quindi la differenza fra i due polinomi sarebbe costretta a cambiare segno n volte. Troppi cambi di segno per un polinomio di grado $n-1$: l'unica possibilità è che la differenza sia nulla. È il caso più semplice del teorema di equioscillazione di Chebyshev, che caratterizza allo stesso modo la migliore approssimazione uniforme di una funzione continua.

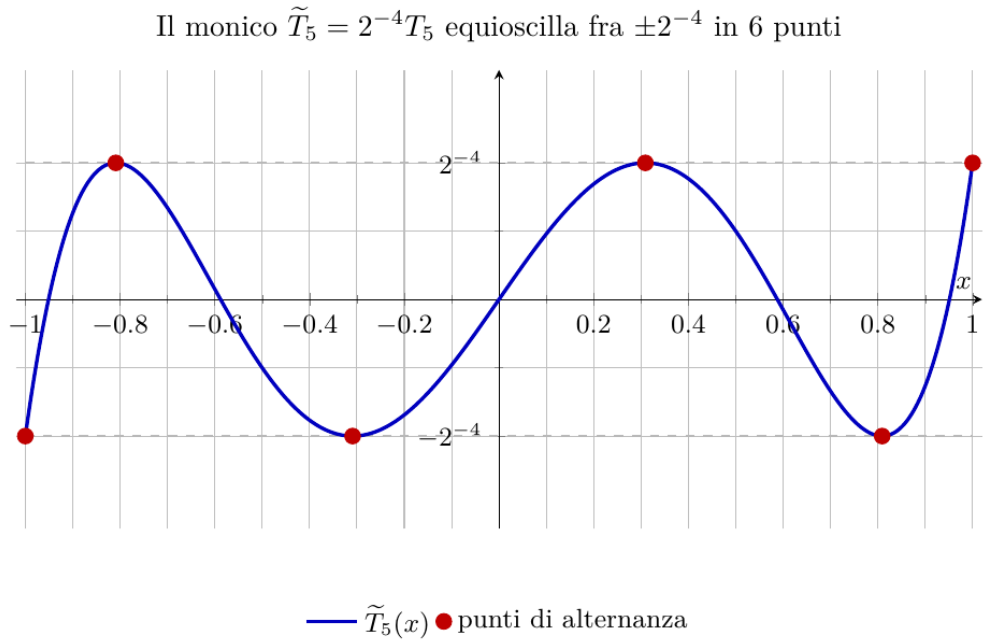


Figure 15.1: Il polinomio di Chebyshev monico di grado 5 tocca $\pm 2^{-4}$ in sei punti alternati: nessun altro monico di grado 5 resta più basso in modulo su tutto l'intervallo.

Il teorema chiude un conto lasciato aperto in Capitolo 9. Il prodotto $(x-x_0)(x-x_1)\cdots(x-x_n)$ che compare nell'errore di interpolazione è monico di grado $n+1$: i nodi che ne minimizzano il massimo modulo sono dunque gli zeri di T_{n+1}, cioè esattamente i nodi di Chebyshev. Là la proprietà era stata enunciata senza dimostrazione.

15.2.4 Laguerre e Hermite: domini illimitati

La tabella copre solo l'intervallo $[-1,1]$. Su domini illimitati le famiglie classiche sono altre due: Laguerre $L_n^{(\alpha)}$, con peso $x^\alpha e^{-x}$ su $[0,+\infty)$, e Hermite H_n (o He_n nella variante probabilistica), con peso e^{-x^2}, rispettivamente $e^{-x^2/2}$, su tutto $\mathbb{R}$. Tutto ciò che segue nel capitolo — ricorrenza a tre termini, zeri reali e semplici, quadratura gaussiana — vale per loro senza modifiche: cambia solo il peso. Gli Hermite ritornano in Capitolo 19. Per $\lambda = 0$ la famiglia di Gegenbauer[3] è degenere: per $n \geq 1$ il limite non normalizzato è

$$\lim_{\lambda \to 0} \frac{C_n^{(\lambda)}(x)}{\lambda} = \frac{2\,T_n(x)}{n},$$

[3] Leopold Gegenbauer (1849-1903)

mentre la famiglia normalizzata $C_n^{(\lambda)}/C_n^{(\lambda)}(1)$ converge a T_n.

15.3 Il peso decide dove si concentra l'errore

Se $\alpha > 0$, il peso smorza il contributo vicino a $x = 1$; se $-1 < \alpha < 0$, lo enfatizza. Il parametro β agisce analogamente in $x = -1$. La Figura 15.2 mostra i pesi classici senza normalizzazione; il grafico R seguente li normalizza ad area unitaria, così le forme si confrontano senza che la scala verticale domini la lettura.

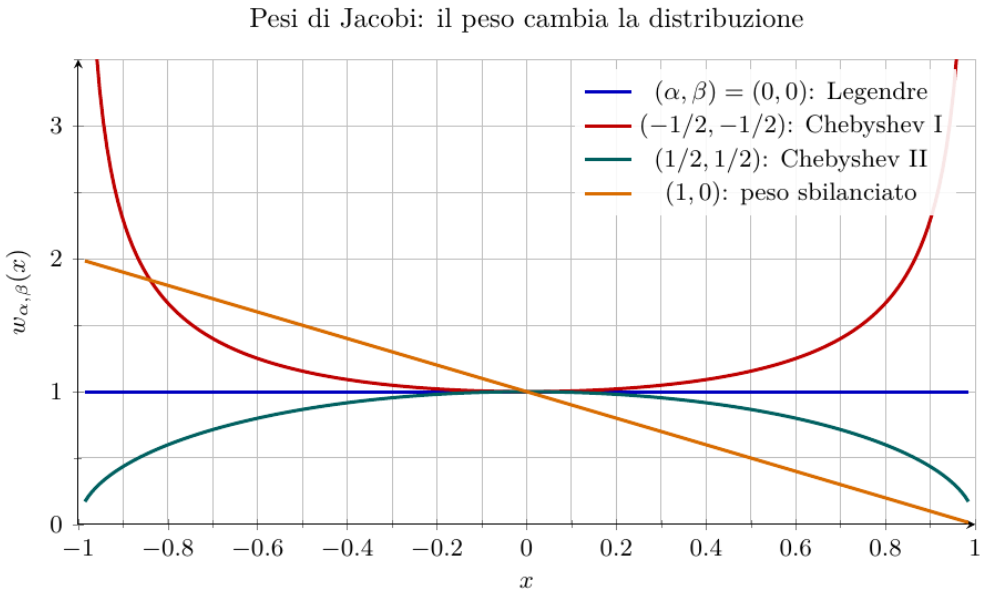


Figure 15.2: Forme dei pesi classici di Jacobi per parametri simmetrici e asimmetrici; le curve TikZ mostrano i pesi senza normalizzazione ad area unitaria.

Possiamo verificare quanto siano diversi i prodotti scalari con un'integrazione numerica che evita di valutare il peso esattamente agli estremi singolari.

```
x <- seq(-0.999, 0.999, length.out=1500)
pars <- list(c(0,0), c(0.5,0.5), c(-0.5,-0.5), c(1.5,-0.4))
cols <- c(2,4,3,6)
labs <- c("Legendre (0,0)", "(0.5,0.5)", "(-0.5,-0.5)", "(1.5,-0.4)")
plot(NA, xlim=c(-1,1), ylim=c(0,5), xlab="x", ylab="peso normalizzato",
     main="Pesi di Jacobi")
for (j in seq_along(pars)) {
  a <- pars[[j]][1]; b <- pars[[j]][2]
  w <- (1-x)^a*(1+x)^b
  w <- w / sum(w)*(length(x)-1)/2
  lines(x, w, col=cols[j], lwd=2)
}
legend("top", legend=labs, col=cols, lwd=2, cex=0.8, bty="n", ncol=2)
```

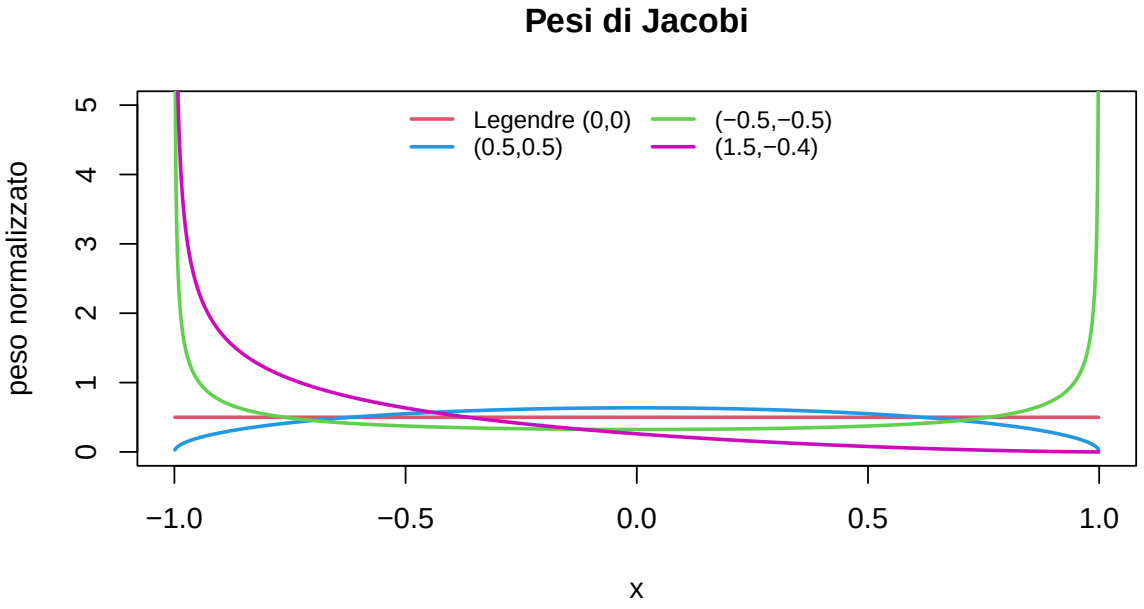


Figure 15.3: Pesi di Jacobi normalizzati ad area unitaria in R: la scelta dei parametri cambia l'importanza relativa degli estremi.

15.4 La ricorrenza dei polinomi di Jacobi

15.4.1 I coefficienti in forma compatta

La normalizzazione standard parte da

$$P_0^{(\alpha,\beta)}(x) = 1, \qquad P_1^{(\alpha,\beta)}(x) = \frac{1}{2}\big[(\alpha-\beta) + (\alpha+\beta+2)\,x\big].$$

Scritta per esteso, la ricorrenza che lega tre gradi consecutivi è ingombrante: i coefficienti sono prodotti di tre o quattro fattori in n, α, β. Diventa però leggibile non appena si dà un nome alla quantità che vi ricorre ovunque,

$$\sigma_n = 2n + \alpha + \beta, \tag{15.11}$$

e ai quattro coefficienti

$$\begin{aligned} A_n &= 2(n+1)(n+\alpha+\beta+1)\,\sigma_n, & B_n &= (\sigma_n+1)(\alpha^2-\beta^2),\\ C_n &= \sigma_n(\sigma_n+1)(\sigma_n+2), & D_n &= 2(n+\alpha)(n+\beta)(\sigma_n+2). \end{aligned} \tag{15.12}$$

Tutti e quattro dipendono anche da α e β attraverso 15.11; l'indice n è lasciato solo per non appesantire la notazione. Con questi simboli, per $n \geq 1$, la ricorrenza si scrive su una riga:

$$A_n\,P_{n+1}^{(\alpha,\beta)}(x) = \big(B_n + C_n\,x\big)\,P_n^{(\alpha,\beta)}(x) - D_n\,P_{n-1}^{(\alpha,\beta)}(x). \tag{15.13}$$

La divisione per A_n è sempre lecita. Infatti $\alpha, \beta > -1$ implica $\alpha+\beta > -2$ e quindi, per $n \geq 1$,

$$\sigma_n = 2n+\alpha+\beta > 2n-2 \geq 0, \qquad n+\alpha+\beta+1 > n-1 \geq 0,$$

cosicché $A_n > 0$: la ricorrenza non degenera mai, per nessuna scelta ammissibile dei parametri.

15.4.2 Due casi particolari

Due letture di 15.13 meritano attenzione.

- **Peso simmetrico.** Se $\alpha = \beta$ allora $B_n = 0$, perché $\alpha^2 - \beta^2 = 0$: il termine costante sparisce e resta $A_n P_{n+1} = C_n\, x\, P_n - D_n P_{n-1}$. È la forma che genera polinomi alternativamente pari e dispari, cioè la parità che ci si aspetta da un peso simmetrico.
- **Caso di Legendre.** Con $\alpha = \beta = 0$ si ha $\sigma_n = 2n$ e

$$\begin{aligned} A_n &= 2(n+1)^2(2n) = 4n(n+1)^2, \\ B_n &= 0, \\ C_n &= 2n(2n+1)(2n+2) = 4n(n+1)(2n+1), \\ D_n &= 2n \cdot n \cdot (2n+2) = 4n^2(n+1). \end{aligned}$$

  Dividendo 15.13 per il fattore comune $4n(n+1)$ resta

$$(n+1)\,P_{n+1}(x) = (2n+1)\,x\,P_n(x) - n\,P_{n-1}(x),$$

  cioè esattamente la ricorrenza dei polinomi di Legendre 14.7.

15.5 Perché l'ortogonalità forza tre termini

La ricorrenza a tre termini non è un'esclusiva dei Jacobi. Sia p_n una famiglia ortogonale con coefficienti principali $\kappa_n > 0$. Il polinomio $x p_n$ ha grado $n+1$, dunque si espande nella base $p_0, \dots, p_{n+1}$. Per $k \le n-2$,

$$\langle x p_n, p_k \rangle_w = \langle p_n, x p_k \rangle_w = 0,$$

perché $x p_k$ ha grado al massimo $n-1$. Restano solo i termini p_{n+1}, p_n, p_{n-1}:

$$x p_n = a_{n+1} p_{n+1} + b_n p_n + c_n p_{n-1}.$$

Confrontando coefficienti principali si trova $a_{n+1} = \kappa_n / \kappa_{n+1}$. Inoltre, per la simmetria del prodotto scalare,

$$c_n = \frac{a_n \|p_n\|_w^2}{\|p_{n-1}\|_w^2} > 0.$$

I simboli minuscoli a_n, b_n, c_n sono volutamente distinti dalle maiuscole di 15.12: i primi valgono per una famiglia ortogonale qualunque, i secondi solo per i Jacobi nella normalizzazione $P_n^{(\alpha,\beta)}(1) = \binom{n+\alpha}{n}$.

Abbiamo dunque dimostrato che *ogni* famiglia ortogonale, per qualunque peso positivo, soddisfa una ricorrenza a tre termini: non è una proprietà dei polinomi di Jacobi, è una conseguenza dell'ortogonalità. Il viceversa — da una ricorrenza di questa forma risalire a un peso rispetto al quale la famiglia è ortogonale — è il **teorema di Favard**, enunciato in Capitolo 16; i due risultati vanno tenuti distinti.

Le formule esplicite dei coefficienti dipendono dunque dalla scala scelta: cambiando normalizzazione cambiano $A_n, \dots, D_n$, non la struttura a tre termini.

In sintesi

- Il peso $(1-x)^\alpha (1+x)^\beta$ è integrabile esattamente per $\alpha, \beta > -1$; ogni scelta dei due parametri dà una famiglia ortogonale diversa.
- Legendre, Gegenbauer e Chebyshev I/II sono casi particolari di Jacobi: cambia solo la normalizzazione, non lo spazio generato.
- α e β decidono *dove* l'errore di approssimazione viene pesato: esponenti negativi enfatizzano gli estremi, positivi li smorzano.

- L'ortogonalità, da sola, impone una ricorrenza a tre termini. Il viceversa è il teorema di Favard.

15.6 Esercizi

1. Per quali valori di α, β il peso di Jacobi è integrabile?
2. Riconoscere i pesi di Legendre e Chebyshev I/II.
3. Quando $\alpha \neq \beta$, quale delle simmetrie $P_n(-x) = (-1)^n P_n(x)$ rimane valida? (In generale nessuna: vale $P_n^{(\alpha,\beta)}(-x) = (-1)^n P_n^{(\beta,\alpha)}(x)$.)
4. Ricavare dalla ricorrenza il polinomio $P_2^{(\alpha,\beta)}(x)$.
5. Dedurre l'ortogonalità dei polinomi $T_n(x)$ rispetto al peso $(1-x^2)^{-1/2}$ dalla 15.4, con la sostituzione $x = \cos\theta$ (la ricorrenza 15.6 è già stata ricavata in questo modo nel Passo 1 di Teorema 15.1).
6. Verificare, per $n = 2, 3, 4$, che $\|\widetilde{T}_n\|_\infty = 2^{1-n}$ confrontandolo con il monico x^n.



16. Ricorrenze, zeri e stabilità

16.1 La ricorrenza a tre termini

16.1.1 Dall'ortogonalità alla ricorrenza

Sia $\{\phi_k\}_{k\geq 0}$ una famiglia ortonormale, con coefficiente principale positivo, rispetto a un peso positivo su un intervallo reale. Ortonormale significa

$$\langle \phi_j, \phi_k \rangle_w = \begin{cases} 1 & \text{se } j = k, \\ 0 & \text{se } j \neq k, \end{cases} \tag{16.1}$$

e $\deg \phi_k = k$, cosicché $\phi_0, \dots, \phi_m$ è una base ortonormale di $\mathcal{P}_m$ per ogni m.
La moltiplicazione per x ha allora una rappresentazione tridiagonale

$$x\,\phi_k(x) = a_{k+1}\phi_{k+1}(x) + b_k\phi_k(x) + a_k\phi_{k-1}(x), \qquad a_k > 0, \tag{16.2}$$

con la convenzione $a_0 = 0$, $\phi_{-1} = 0$. Vale la pena vedere perché, in tre righe. Il polinomio $x\phi_k$ ha grado $k+1$, dunque si scrive in modo unico nella base $\phi_0, \dots, \phi_{k+1}$:

$$x\,\phi_k = \sum_{j=0}^{k+1} c_{kj}\,\phi_j, \qquad c_{kj} = \langle x\phi_k, \phi_j \rangle_w.$$

La chiave è che il fattore x può essere spostato da una parte all'altra del prodotto scalare, perché moltiplicare per x non tocca il peso:

$$\langle x\phi_k, \phi_j \rangle_w = \int_a^b x\,\phi_k(x)\,\phi_j(x)\,w(x)\,\mathrm{d}x = \langle \phi_k, x\phi_j \rangle_w.$$

Se $j \leq k-2$ il polinomio $x\phi_j$ ha grado $j+1 \leq k-1 < k$, quindi è combinazione di $\phi_0, \dots, \phi_{k-1}$ e il prodotto scalare con ϕ_k è nullo: $c_{kj} = 0$. Sopravvivono solo tre coefficienti,

$$b_k = \langle x\phi_k, \phi_k \rangle_w, \qquad a_k = \langle x\phi_{k-1}, \phi_k \rangle_w = \langle x\phi_k, \phi_{k-1} \rangle_w,$$

e lo stesso scambio mostra che il coefficiente di ϕ_{k-1} in $x\phi_k$ coincide con quello di ϕ_{k+1} in $x\phi_{k-1}$: è questo che rende la rappresentazione *simmetrica*. Infine $a_k > 0$ perché, confrontando i coefficienti principali in 16.2, $a_{k+1} = \kappa_k/\kappa_{k+1} > 0$, avendo indicato con $\kappa_k > 0$ il coefficiente principale di ϕ_k.

16.1.2 Il teorema di Favard

A questo punto è importante distinguere l'implicazione diretta da quella inversa: la prima è quanto appena visto, la seconda è un risultato profondo.

Theorem 16.1 — Ricorrenza a tre termini e teorema di Favard. **(Verso diretto.)** Sia $\{\phi_k\}_{k\geq 0}$ ortonormale rispetto a un peso positivo w, con $\deg \phi_k = k$ e coefficiente principale positivo. Allora esistono numeri reali b_k e numeri $a_k > 0$ tali che valga 16.2 per ogni $k \geq 0$.

(Verso inverso, teorema di Favard.)[a] Viceversa, si scelgano in modo del tutto arbitrario numeri reali $b_0, b_1, \ldots$ e numeri positivi $a_1, a_2, \ldots$, e si definiscano i polinomi ϕ_k *per ricorrenza* tramite 16.2, partendo da $\phi_{-1} = 0$ e $\phi_0 = \text{cost} > 0$. Allora esiste una misura positiva sulla retta reale rispetto alla quale la famiglia così costruita è ortonormale.

[a] Jean Favard (1902-1965)

Il verso diretto è stato dimostrato qui sopra (e, in altra normalizzazione, in Capitolo 15): è una conseguenza immediata dell'ortogonalità e dello scambio $\langle x\phi_k, \phi_j\rangle_w = \langle \phi_k, x\phi_j\rangle_w$.

Il verso inverso è il risultato non banale, e la sua dimostrazione — che costruisce la misura a partire dai coefficienti — esula da questo corso; si veda [13]. Il suo significato però è semplice da enunciare: la ricorrenza a tre termini **caratterizza** i polinomi ortogonali. Se una successione di polinomi obbedisce a una relazione di quella forma, con $a_k > 0$, allora *è* una famiglia ortogonale per un qualche peso, anche se il peso non è stato nominato. Non esistono famiglie con quella struttura che non siano ortogonali, e non esistono famiglie ortogonali che non abbiano quella struttura.

16.1.3 La matrice di Jacobi

Vogliamo riscrivere la ricorrenza 16.2 come un prodotto matrice–vettore. Costruiamo la matrice un pezzo alla volta, partendo dalla ricorrenza stessa.

Primo: mettere la x a destra. La ricorrenza, così com'è scritta, esprime $x\phi_k$ in funzione di tre polinomi consecutivi. Riordiniamola mettendo a sinistra i tre polinomi e a destra la moltiplicazione per x:

$$a_k\,\phi_{k-1}(x) + b_k\,\phi_k(x) + a_{k+1}\,\phi_{k+1}(x) \;=\; x\,\phi_k(x). \tag{16.3}$$

Nulla è cambiato, ma ora la 16.3 ha la forma «combinazione lineare di $\phi = x$ per un ϕ»: è una **riga** di un sistema lineare in cui le incognite sono i valori $\phi_0(x), \phi_1(x), \ldots$

Secondo: scrivere le righe una sotto l'altra. Fissiamo n e scriviamo la 16.3 per $k = 0, 1, 2, \ldots, n-1$, ricordando le convenzioni $a_0 = 0$ e $\phi_{-1} = 0$, che fanno sparire il primo termine quando $k = 0$:

$$\begin{aligned}
k = 0 &: \quad b_0\phi_0 + a_1\phi_1 && = x\,\phi_0,\\
k = 1 &: \quad a_1\phi_0 + b_1\phi_1 + a_2\phi_2 && = x\,\phi_1,\\
k = 2 &: \quad a_2\phi_1 + b_2\phi_2 + a_3\phi_3 && = x\,\phi_2,\\
&\qquad\qquad \ddots && \;\;\vdots\\
k = n-2 &: \quad a_{n-2}\phi_{n-3} + b_{n-2}\phi_{n-2} + a_{n-1}\phi_{n-1} && = x\,\phi_{n-2},\\
k = n-1 &: \quad a_{n-1}\phi_{n-2} + b_{n-1}\phi_{n-1} + a_n\phi_n && = x\,\phi_{n-1}.
\end{aligned} \tag{16.4}$$

Si osservino due cose. Le prime $n-1$ righe coinvolgono soltanto $\phi_0, \ldots, \phi_{n-1}$: sono $n-1$ equazioni «chiuse». L'ultima riga, quella con $k = n-1$, fa comparire in più il polinomio ϕ_n, che non è fra le incognite: lo isoliamo portandolo a destra,

$$a_{n-1}\phi_{n-2} + b_{n-1}\phi_{n-1} = x\,\phi_{n-1} - a_n\,\phi_n.$$

Terzo: leggere i coefficienti riga per riga. Raccogliamo le incognite nel vettore colonna

$$\Phi(x) = \begin{pmatrix} \phi_0(x) \\ \phi_1(x) \\ \vdots \\ \phi_{n-1}(x) \end{pmatrix} \in \mathfrak{R}^n,$$

e numeriamo righe e colonne da 0 a $n-1$, come i polinomi. Nella riga k di 16.4 i coefficienti non nulli sono soltanto tre, e occupano le colonne $k-1$, k e $k+1$:

$$\underbrace{a_k}_{\text{colonna } k-1}, \qquad \underbrace{b_k}_{\text{colonna } k}, \qquad \underbrace{a_{k+1}}_{\text{colonna } k+1}.$$

Tutte le altre colonne hanno coefficiente zero. Una matrice i cui elementi non nulli stanno solo sulla diagonale e sulle due diagonali immediatamente adiacenti si dice **tridiagonale**.

Quarto: la matrice. Mettendo insieme i coefficienti delle n righe si ottiene la **matrice di Jacobi**

$$J_n = \begin{pmatrix} b_0 & a_1 & & & \\ a_1 & b_1 & a_2 & & \\ & a_2 & b_2 & \ddots & \\ & & \ddots & \ddots & a_{n-1} \\ & & & a_{n-1} & b_{n-1} \end{pmatrix} \in \mathfrak{R}^{n\times n}, \tag{16.5}$$

dove le caselle lasciate in bianco valgono zero. La matrice è **simmetrica** e non per caso: l'elemento di posto $(k, k-1)$ vale a_k perché è il coefficiente di ϕ_{k-1} nella riga k; l'elemento di posto $(k-1, k)$ vale $a_{(k-1)+1} = a_k$ perché è il coefficiente di ϕ_k nella riga $k-1$. Sono lo stesso numero, ed è questa la ragione per cui conviene lavorare con i polinomi *ortonormali*: in un'altra normalizzazione i due coefficienti sarebbero diversi e la matrice non sarebbe simmetrica.

Con questa notazione le n equazioni 16.4 si riassumono in una sola:

$$J_n\,\Phi(x) = x\,\Phi(x) - a_n\,\phi_n(x)\,e_n, \tag{16.6}$$

dove $e_n = (0, \dots, 0, 1)^T$ è l'ultimo vettore della base canonica: il termine di disturbo $-a_n\phi_n(x)$ compare infatti solo nell'ultima riga. Scritta per esteso con $n = 4$, la 16.6 è

$$\begin{pmatrix} b_0 & a_1 & 0 & 0 \\ a_1 & b_1 & a_2 & 0 \\ 0 & a_2 & b_2 & a_3 \\ 0 & 0 & a_3 & b_3 \end{pmatrix} \begin{pmatrix} \phi_0(x) \\ \phi_1(x) \\ \phi_2(x) \\ \phi_3(x) \end{pmatrix} = x \begin{pmatrix} \phi_0(x) \\ \phi_1(x) \\ \phi_2(x) \\ \phi_3(x) \end{pmatrix} - a_4\,\phi_4(x) \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

Quinto: gli zeri sono autovalori. A questo punto la conclusione si legge da sola. Se x è uno **zero** di ϕ_n, allora $\phi_n(x) = 0$, il termine di disturbo sparisce e 16.6 diventa

$$J_n\,\Phi(x) = x\,\Phi(x).$$

Il vettore $\Phi(x)$ non è nullo, perché la sua prima componente ϕ_0 è una costante diversa da zero. Dunque $\Phi(x)$ è un autovettore di J_n e x è il corrispondente **autovalore**.

Gli n zeri di ϕ_n sono quindi autovalori di J_n; poiché una matrice $n \times n$ ha esattamente n autovalori, sono *tutti* e *soli* gli autovalori di J_n. Da qui si leggono subito due proprietà:

- sono **reali**, perché J_n è simmetrica a elementi reali;
- sono **semplici**, perché una matrice tridiagonale simmetrica con tutti gli elementi extradiagonali non nulli ($a_k > 0$) ha autovalori distinti.

La Sezione 16.2 ridimostra gli stessi fatti, e in più la localizzazione dentro l'intervallo, con un argomento di sola integrazione. Per i polinomi di Legendre ortonormali si ha

$$b_k = 0, \qquad a_k = \frac{k}{\sqrt{(2k-1)(2k+1)}} = \frac{k}{\sqrt{4k^2-1}}, \tag{16.7}$$

i b_k essendo nulli per la simmetria del peso. È su questa matrice che si basa l'algoritmo di Golub–Welsch di Capitolo 18: calcolare i nodi di una formula gaussiana si riduce a calcolare gli autovalori di una matrice tridiagonale simmetrica, problema per cui esistono algoritmi rapidi e stabili.

16.2 Gli zeri dei polinomi ortogonali

16.2.1 Zeri reali, interni e semplici

Theorem 16.2 — Localizzazione degli zeri. Sia w un peso positivo su (a, b) e sia p_n, con $n \geq 1$, il polinomio di grado n della famiglia ortogonale associata. Allora p_n ha n zeri

$$a < x_1 < x_2 < \cdots < x_n < b,$$

cioè n zeri **reali** e **distinti** (quindi tutti **semplici**), tutti **interni** all'intervallo (a, b). In particolare p_n non ha radici complesse, né radici fuori da (a, b), né radici multiple.

Proof. Useremo una sola volta, ma ripetutamente, la proprietà che definisce p_n:

$$\langle p_n, q\rangle_w = \int_a^b p_n(x)\, q(x)\, w(x)\, \mathrm{d}x = 0 \qquad \text{per ogni } q \text{ con } \deg q < n, \tag{16.8}$$

vera perché ogni q di grado minore di n è combinazione lineare di $p_0, \dots, p_{n-1}$, e p_n è ortogonale a ciascuno di essi.

Passo 0: un fatto elementare sugli integrali. Se f è continua su (a, b), non cambia mai segno — diciamo $f \geq 0$ — e non è identicamente nulla, allora

$$\int_a^b f(x)\, w(x)\, \mathrm{d}x > 0. \tag{16.9}$$

Infatti esiste $\bar{x} \in (a, b)$ con $f(\bar{x}) > 0$; per continuità esiste un intervallino $[\bar{x}-\delta, \bar{x}+\delta] \subset (a, b)$ su cui $f > f(\bar{x})/2$, e quindi

$$\int_a^b f\, w \;\geq\; \int_{\bar{x}-\delta}^{\bar{x}+\delta} f\, w \;\geq\; \frac{f(\bar{x})}{2} \int_{\bar{x}-\delta}^{\bar{x}+\delta} w \;>\; 0,$$

perché l'integrale di un peso positivo su un intervallo di lunghezza non nulla è positivo. Se invece $f \leq 0$ si conclude allo stesso modo che l'integrale è strettamente negativo. In breve: *l'integrale pesato di una funzione di segno costante e non nulla non può annullarsi.*

Passo 1: p_n cambia segno almeno una volta in (a, b). Applichiamo 16.8 con $q \equiv 1$, lecito perché $\deg q = 0 < n$:

$$\int_a^b p_n(x)\, w(x)\, \mathrm{d}x = 0.$$

Se p_n avesse segno costante su (a, b), il Passo 0 darebbe un integrale diverso da zero. Dunque p_n assume sia valori positivi sia valori negativi e, essendo continua, cambia segno: per il teorema degli zeri ha almeno una radice in (a, b) in cui il segno si inverte.

Passo 2: i punti di cambio di segno sono almeno n. Siano $x_1 < x_2 < \cdots < x_r$ i punti di (a, b) in cui p_n **cambia segno**. Il Passo 1 garantisce $r \geq 1$. Supponiamo per assurdo $r < n$ e costruiamo il polinomio

$$q(x) = (x - x_1)(x - x_2)\cdots(x - x_r), \qquad \deg q = r < n.$$

Osserviamo che q, prodotto di fattori lineari distinti, cambia segno esattamente negli stessi punti $x_1, \dots, x_r$ e in nessun altro. Allora il prodotto $p_n\, q$ **non cambia mai segno** in (a, b): fra due punti consecutivi x_j e x_{j+1} entrambi i fattori mantengono il loro segno, e attraversando x_{j+1} entrambi lo invertono, sicché il prodotto lo conserva. Gli eventuali altri zeri di p_n interni a (a, b) non creano problemi: per definizione di $x_1, \dots, x_r$ in essi il segno non cambia, il prodotto si annulla e poi riprende il segno di prima.

Il polinomio $p_n q$ non è identicamente nullo (è il prodotto di due polinomi non nulli) e ha segno costante: per il Passo 0

$$\langle p_n, q\rangle_w = \int_a^b p_n\, q\, w \;\neq\; 0.$$

Ma $\deg q = r < n$, e 16.8 impone invece $\langle p_n, q\rangle_w = 0$. Contraddizione. Resta dunque $r \geq n$.

Passo 3: sono esattamente n, e sono tutti gli zeri. Ogni punto di cambio di segno è una radice reale di p_n interna a (a, b), e punti distinti sono radici distinte: abbiamo quindi almeno n radici reali distinte. D'altra parte p_n ha grado n, e un polinomio di grado n non può avere più di n radici distinte. Dunque $r = n$ esattamente.

Per il teorema fondamentale dell'algebra p_n ha n radici contate con la loro molteplicità: le n radici distinte appena trovate le esauriscono tutte. Non ne restano di complesse, non ne restano fuori da (a, b), e nessuna può essere multipla, perché una radice doppia consumerebbe due unità di molteplicità lasciandone meno di n per le altre. ∎

Questo risultato è ciò che rende possibile la quadratura gaussiana di Capitolo 18, perché i nodi di una formula a n punti devono stare dentro l'intervallo di integrazione ed essere distinti, e qui lo sono per costruzione.

16.2.2 Separazione degli zeri di gradi consecutivi

Per i polinomi monici la ricorrenza si scrive

$$\pi_{n+1}(x) = (x - b_n)\pi_n(x) - a_n^2\pi_{n-1}(x), \qquad a_n > 0.$$

Due polinomi consecutivi non hanno zeri comuni: se π_n e π_{n+1} si annullassero entrambi in x, la ricorrenza forzerebbe $\pi_{n-1}(x) = 0$ e, iterando all'indietro, $\pi_0(x) = 0$, impossibile. Inoltre, tutti gli zeri di entrambi sono interni al supporto, come appena dimostrato. La matrice J_n è il minore principale iniziale di J_{n+1}. Il teorema di interlacciamento di Cauchy[1] dà interlacciamento debole; l'assenza di autovalori comuni lo rende stretto: fra due zeri consecutivi di π_{n+1} cade esattamente uno zero di π_n, e inoltre il più piccolo zero di π_{n+1} sta a sinistra del più piccolo di π_n, il più grande a destra del più grande. In altri termini gli $n + 1$ zeri di π_{n+1} separano gli n zeri di π_n e li racchiudono.

La Figura 16.1 illustra il risultato; nella Figura 16.2 si visualizzano le due famiglie di zeri insieme alle curve.

```
nodes_legendre <- function(n) {
  k<-seq_len(n-1); off<-k/sqrt(4*k^2-1); J<-matrix(0,n,n)
  J[cbind(k,k+1)]<-off; J[cbind(k+1,k)]<-off
  sort(eigen(J,symmetric=TRUE,only.values=TRUE)$values)
```

[1] Augustin-Louis Cauchy (1789-1857)

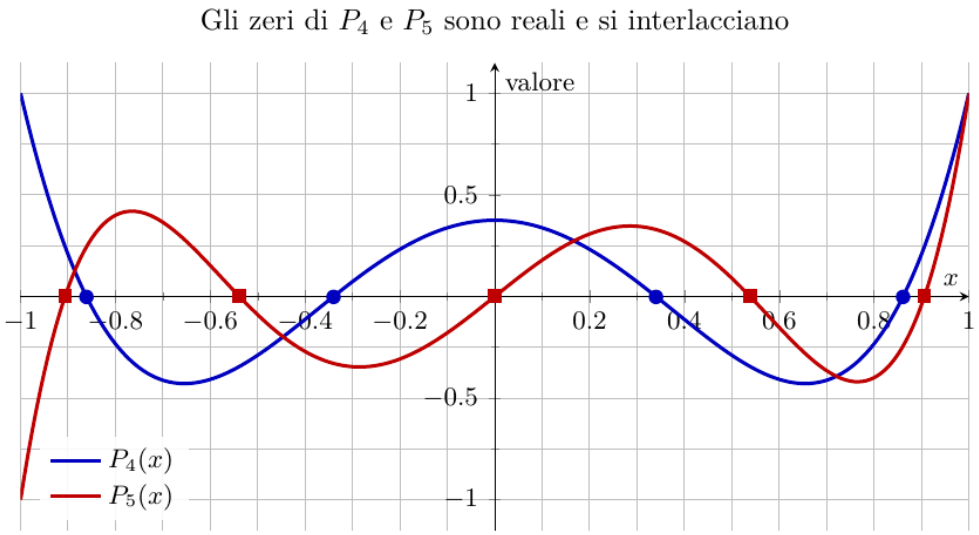


Figure 16.1: Gli zeri dei polinomi di Legendre di grado 4 e 5 si alternano.

```
}
x<-seq(-1,1,length.out=2001)
plot(x,legendre(4,x),type="l",lwd=2,col=4,ylim=c(-0.5,0.7),
     xlab="x",ylab="P_n(x)",main="Zeri e interlacciamento")
lines(x,legendre(5,x),lwd=2,col=2); abline(h=0,col="gray50",lty=2)
points(nodes_legendre(4),rep(0,4),pch=19,col=4,cex=1.1)
points(nodes_legendre(5),rep(0,5),pch=17,col=2,cex=1.1)
legend("topright",c("P4","P5","zeri P4","zeri P5"),
       col=c(4,2,4,2),lty=c(1,1,NA,NA),pch=c(NA,NA,19,17),bty="n")
```

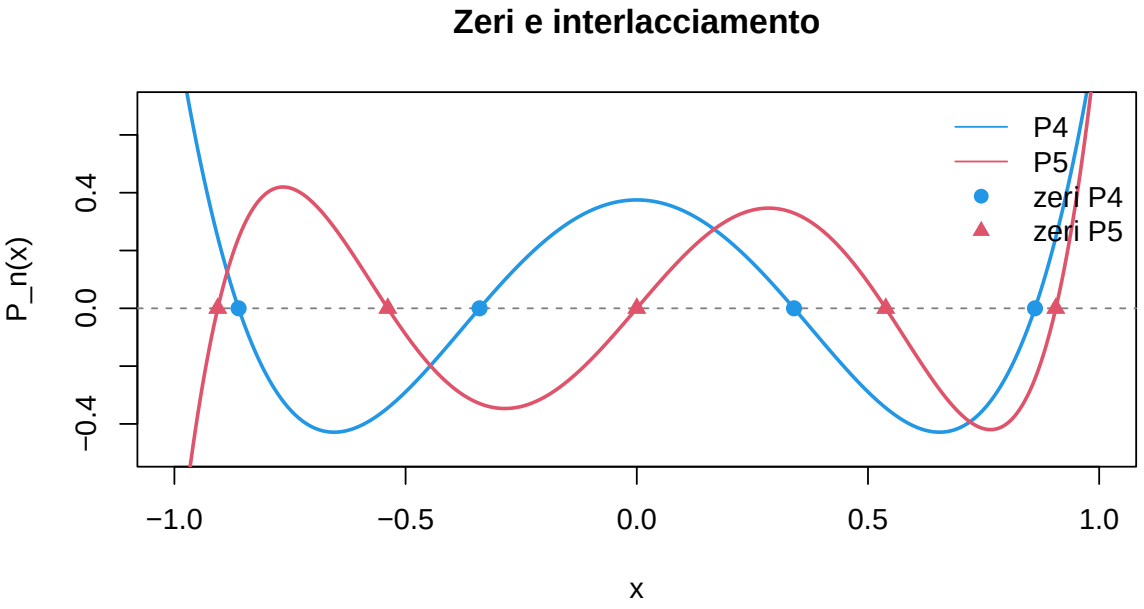


Figure 16.2: P4 e P5 e i rispettivi zeri: tra ogni coppia di zeri di P4 cade uno zero di P5.

16.3 Polinomi del nucleo e minimo vincolato

16.3.1 Il nucleo e la proprietà riproducente

Lavorando con la famiglia **ortonormale** ϕ_k si incontra di continuo una somma che conviene battezzare subito:

$$K_n(x, y) = \sum_{k=0}^{n} \phi_k(x)\, \phi_k(y) \tag{16.10}$$

è il **polinomio del nucleo** di grado n. (Di questa somma esiste anche una forma chiusa, la formula di Christoffel–Darboux, che non serve nel seguito ed è riportata in Sezione A.3.) Il nome viene dalla sua proprietà *riproducente*: per ogni $p \in \mathcal{P}_n$,

$$\langle p, K_n(\cdot, y)\rangle_w = \sum_{k=0}^{n} \langle p, \phi_k\rangle_w\, \phi_k(y) = p(y), \tag{16.11}$$

perché la somma a destra è esattamente lo sviluppo di p nella base ortonormale. In altri termini $K_n(\cdot, y)$ è il vettore che, nel prodotto scalare, *valuta in* y.

16.3.2 Il polinomio di norma minima vincolata

Da quella sola osservazione segue un risultato che useremo due volte.

Theorem 16.3 — Polinomio di norma minima con un vincolo. Siano $n \geq 0$ e $x_0 \in \mathfrak{R}$ un punto qualunque. Fra tutti i polinomi $p \in \mathcal{P}_n$ che soddisfano il vincolo

$$p(x_0) = 1, \tag{16.12}$$

ne esiste uno e uno solo di norma $\|\cdot\|_w$ minima, ed è

$$p^\star(x) = \frac{K_n(x_0, x)}{K_n(x_0, x_0)}, \qquad \text{con} \qquad \|p^\star\|_w = \frac{1}{\sqrt{K_n(x_0, x_0)}}. \tag{16.13}$$

Nota

Il denominatore non si annulla mai. Infatti, per 16.10,

$$K_n(x_0, x_0) = \sum_{k=0}^{n} \phi_k(x_0)^2 \;\geq\; \phi_0(x_0)^2 > 0,$$

perché ϕ_0 è una costante non nulla. La formula 16.13 ha quindi senso per ogni x_0 reale, anche *fuori* dall'intervallo (a, b): è il caso interessante, come mostra la Figura 16.3.

Proof. **Passo 1: il vincolo si legge come un prodotto scalare.** Per la proprietà riproducente 16.11, ogni $p \in \mathcal{P}_n$ soddisfa $p(x_0) = \langle p, K_n(\cdot, x_0)\rangle_w$. Il vincolo 16.12 diventa dunque

$$\langle p,\, K_n(\cdot, x_0)\rangle_w = 1. \tag{16.14}$$

Geometricamente: fra tutti i vettori la cui proiezione sulla direzione fissata $K_n(\cdot, x_0)$ ha lunghezza assegnata, si cerca il più corto. La risposta, in qualunque spazio con prodotto scalare, è il vettore parallelo a quella direzione — ed è esattamente ciò che la dimostrazione formalizza.

Passo 2: la lunghezza della direzione. Il polinomio $K_n(\cdot, x_0)$ appartiene a $\mathcal{P}_n$, quindi la la proprietà riproducente si può applicare al polinomio stesso, con $p = K_n(\cdot, x_0)$ e valutando in $y = x_0$:

$$\|K_n(\cdot, x_0)\|_w^2 = \big\langle K_n(\cdot, x_0), K_n(\cdot, x_0) \big\rangle_w = K_n(x_0, x_0). \tag{16.15}$$

Passo 3: la minorazione. Applichiamo la disuguaglianza di Cauchy–Schwarz $|\langle f, g\rangle_w| \le \|f\|_w \|g\|_w$ a 16.14, e poi 16.15:

$$1 = \big|\langle p, K_n(\cdot, x_0)\rangle_w\big| \le \|p\|_w \, \|K_n(\cdot, x_0)\|_w = \|p\|_w \sqrt{K_n(x_0, x_0)}.$$

Dividendo per $\sqrt{K_n(x_0, x_0)} > 0$ si ottiene, per **ogni** p ammissibile,

$$\|p\|_w \;\ge\; \frac{1}{\sqrt{K_n(x_0, x_0)}}. \tag{16.16}$$

Nessun polinomio ammissibile può dunque scendere sotto quella soglia.

Passo 4: la soglia è raggiunta, e da un solo polinomio. Il candidato $p^\star$ di 16.13 è ammissibile, perché

$$p^\star(x_0) = \frac{K_n(x_0, x_0)}{K_n(x_0, x_0)} = 1,$$

e la sua norma vale, ancora per 16.15,

$$\|p^\star\|_w = \frac{\|K_n(\cdot, x_0)\|_w}{K_n(x_0, x_0)} = \frac{\sqrt{K_n(x_0, x_0)}}{K_n(x_0, x_0)} = \frac{1}{\sqrt{K_n(x_0, x_0)}},$$

cioè esattamente il valore della soglia: il minimo è raggiunto.

Per l'unicità si usa il caso di uguaglianza in Cauchy–Schwarz: in $|\langle f, g\rangle_w| = \|f\|_w \|g\|_w$ vale l'uguale se e solo se f e g sono proporzionali. Se dunque un p ammissibile realizza il minimo, allora $p = c\, K_n(\cdot, x_0)$ per qualche costante c, e il vincolo $p(x_0) = 1$ obbliga $c\, K_n(x_0, x_0) = 1$, cioè $c = 1/K_n(x_0, x_0)$: si ritrova $p = p^\star$ e nessun altro. ∎

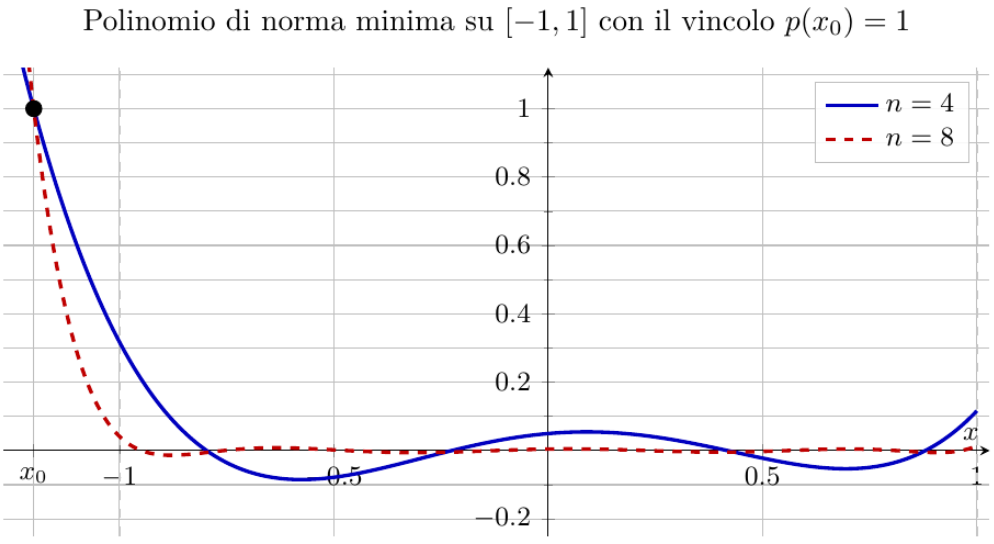


Figure 16.3: Polinomio di norma minima su $[-1, 1]$ con peso unitario, vincolato a valere 1 nel punto esterno $x_0 = -1.2$: già per $n = 4$ oscilla poco dentro l'intervallo, per $n = 8$ pochissimo.

La Figura 16.3 mostra la configurazione tipica: il polinomio vale 1 appena fuori dall'intervallo e dentro è piccolo. È quello che serve quando si vuole un polinomio grande in un punto e piccolo su un intervallo — per esempio un polinomio di precondizionamento, dove $x_0 = 0$ e l'intervallo contiene lo spettro della matrice.

16.3.3 La funzione di Christoffel

La quantità

$$\lambda_n(x) = \frac{1}{K_n(x,x)} = \min\left\{ \|p\|_w^2 \;:\; p \in \mathcal{P}_n,\ p(x) = 1 \right\} \tag{16.17}$$

si chiama **funzione di Christoffel**. Tornerà in Capitolo 18, dove si vedrà che i pesi di una formula gaussiana non sono altro che i suoi valori nei nodi.

16.4 Stabilità della ricorrenza nel calcolo

Per calcolare un singolo $P_n(x)$, la ricorrenza a tre termini costa $O(n)$ operazioni e memoria costante, ed evita i coefficienti monomiali. Il limite analitico $|P_n(x)| \le 1$ per $x \in [-1,1]$ non è una prova di stabilità floating point: gli errori locali di arrotondamento si propagano secondo le due soluzioni indipendenti della ricorrenza e la sensibilità dipende da n e x. Nell'intervallo le soluzioni sono oscillanti e la ricorrenza in avanti è di norma adatta ai polinomi di Legendre, ma vicino a uno zero l'errore relativo resta intrinsecamente grande. Fuori dall'intervallo una componente dominante può sovrastare quella richiesta; quando si cerca la soluzione minimale di una ricorrenza, la direzione in avanti può essere instabile e si ricorre alla ricorrenza all'indietro di Miller, con riscalamento e normalizzazione. I coefficienti di una serie si valutano spesso con Clenshaw. Ogni affermazione di stabilità va quindi qualificata per grado, punto e quantità richiesta.

Per valutare una serie

$$s(x) = \sum_{k=0}^{n} c_k\, \phi_k(x)$$

si usa spesso l'algoritmo all'indietro di Clenshaw[2]. Partendo dalla ricorrenza ortonormale

$$x\phi_k = a_{k+1}\phi_{k+1} + b_k\phi_k + a_k\phi_{k-1},$$

si pongono $d_{n+1} = d_{n+2} = 0$ e si calcola

$$d_k = c_k + \frac{x - b_k}{a_{k+1}}\, d_{k+1} - \frac{a_{k+1}}{a_{k+2}}\, d_{k+2}, \qquad k = n, n-1, \dots, 0, \qquad\qquad s(x) = \phi_0(x) d_0,$$

con la convenzione $d_{n+1} = d_{n+2} = 0$; i coefficienti a_{n+1} e a_{n+2} non intervengono mai, perché moltiplicano termini nulli, e possono essere posti a 1 nell'implementazione. La procedura evita la somma esplicita di termini potenzialmente grandi e spesso migliora l'accuratezza; si devono usare coefficienti coerenti con la normalizzazione scelta. Non rimuove il condizionamento del problema né garantisce accuratezza relativa vicino a cancellazioni.

```
nn<-0:40; xx<-seq(-1,1,length.out=2001)
inside<-sapply(nn,function(n)max(abs(legendre(n,xx))))
outside<-sapply(nn,function(n)abs(legendre(n,1.2)))
plot(nn,inside,type="l",lwd=2,col=4,log="y",ylim=c(0.5,max(outside)),
     xlab="grado n",ylab="ampiezza (scala log)",main="Dominio di valutazione")
lines(nn,outside,lwd=2,col=2)
legend("topleft",c("max su [-1,1]","|P_n(1.2)|"),col=c(4,2),lwd=2,bty="n")
```

[2] Charles William Clenshaw (1926-2004)

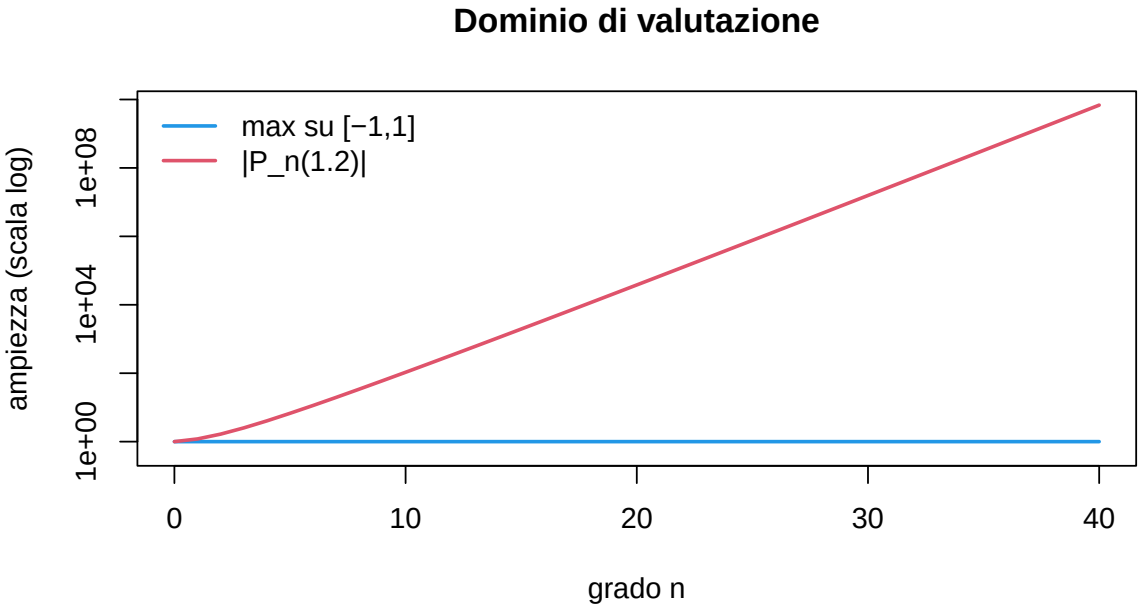


Figure 16.4: Massimo osservato su [-1,1] e valore in 1.2: fuori dal supporto la crescita è rapida.

La Figura 16.4 mostra valori calcolati per gradi finiti; non rappresenta una stima dell'errore di arrotondamento.

In sintesi

- L'ortogonalità equivale alla ricorrenza a tre termini: il verso diretto è elementare, il viceversa è il teorema di Favard.
- La stessa ricorrenza è una matrice tridiagonale simmetrica J_n, i cui autovalori sono gli zeri di p_n: zeri reali e semplici sono un fatto di algebra lineare, non di analisi.
- Gli zeri di p_n e p_{n+1} si separano a vicenda (interlacciamento), conseguenza del teorema di Cauchy applicato al minore principale J_n di J_{n+1}.
- Ricorrenza in avanti e algoritmo di Clenshaw costano $O(n)$ e memoria costante, ma *nessuno dei due* garantisce accuratezza relativa: la stabilità va qualificata per grado, punto e quantità richiesta.
- Il polinomio del nucleo K_n è il vettore che *valuta in un punto*: da questa sola proprietà segue in forma chiusa il polinomio di norma minima vincolato, e con esso i pesi delle formule gaussiane.
- Simmetria, derivate, formula di Christoffel–Darboux e quadro di completezza non servono nei due capitoli che seguono e sono raccolti in Appendice A.

16.5 Esercizi

1. Perché la matrice J_n è simmetrica? Perché gli zeri sono semplici?
2. Spiegare la differenza tra errore assoluto e relativo vicino a uno zero.
3. Dimostrare $P_n(-x) = (-1)^n P_n(x)$ usando unicità e simmetria del peso.
4. Quando è utile Clenshaw rispetto alla valutazione separata della base?
5. Verificare numericamente l'interlacciamento per P_9 e P_{10} calcolando gli autovalori di J_9 e J_{10}.



17. Minimi quadrati con basi ortogonali

17.1 Il problema, e tre domande

Il punto di partenza è una situazione di laboratorio. Un esperimento ha prodotto molti punti (x_i, y_i) — diciamo un centinaio — e le ordinate sono **rumorose**: ogni y_i è il valore vero più un errore di misura. Non si vuole una curva che passi per tutti i punti, perché riprodurrebbe anche il rumore; si vuole il **trend**, cioè la curva liscia che i dati suggeriscono.

Lo strumento è il polinomio dei minimi quadrati di grado n, e il modo corretto di calcolarlo è la fattorizzazione QR. Finito? Non proprio: restano aperte tre domande, ed è attorno a queste che ruota il capitolo.

Prima domanda. La QR funziona con qualunque base. Ma se, invece dei monomi $1, x, x^2, \ldots$, si usa una base ortogonale — i polinomi di Legendre, oppure una costruita sui dati stessi — la matrice del problema diventa diagonale, o quasi, e meglio condizionata. Quanto si guadagna davvero, e in che senso «quasi»?

Seconda domanda. Alzando il grado il residuo scende sempre, quindi il residuo da solo non dice dove fermarsi. Si osserva però che dopo un certo grado la discesa si appiattisce: compare un **plateau**. Che cosa lo provoca, e si può trasformare l'osservazione in un criterio di arresto?

Terza domanda. Si può procedere in modo **incrementale**, cioè aggiungere un grado alla volta riusando tutto il lavoro già fatto, fino a riconoscere il plateau e fermarsi?

La risposta alle tre è affermativa, ma con precisazioni che vale la pena fare bene, perché su questo argomento circolano semplificazioni sbagliate. La più comune è che una base ortogonale «filtri il rumore»: non è così, e la Sezione 17.6 lo chiarisce. Prima però conviene costruire il problema da zero.

17.2 Impostazione del problema

17.2.1 Che cosa si vuole minimizzare

Un esperimento produce m coppie di numeri

$$(x_1, y_1),\ (x_2, y_2),\ \ldots,\ (x_m, y_m),$$

dove x_i è il valore impostato e y_i quello misurato, con il suo errore di misura. Si vuole riassumere i dati con un polinomio

$$p(x) = c_0\varphi_0(x) + c_1\varphi_1(x) + \cdots + c_n\varphi_n(x) \tag{17.1}$$

di grado n, scritto come combinazione di una base $\varphi_0, \dots, \varphi_n$ di $\mathcal{P}_n$ che per ora lasciamo indeterminata.

Il caso interessante è $m > n+1$: i dati sono **più numerosi** delle incognite. Far passare il polinomio esattamente per tutti i punti in generale è impossibile, e sarebbe anche sbagliato, perché i y_i contengono rumore e non vogliamo riprodurlo. Si chiede allora che gli scarti

$$r_i = p(x_i) - y_i, \qquad i = 1, \dots, m,$$

siano complessivamente piccoli, e si misura il «complessivamente» con la somma dei loro quadrati:

$$\min_{c_0,\dots,c_n} \sum_{i=1}^{m} \big[p(x_i) - y_i\big]^2. \tag{17.2}$$

Questo è il problema dei **minimi quadrati** discreti. Si sceglie il quadrato — e non, per esempio, il valore assoluto — perché rende la funzione obiettivo derivabile e, come vedremo, riconduce tutto a un sistema lineare.

17.2.2 La scrittura matriciale

Valutiamo 17.1 in tutti i nodi e raccogliamo i risultati in un vettore. Posto

$$\boldsymbol{A} = \begin{pmatrix} \varphi_0(x_1) & \varphi_1(x_1) & \cdots & \varphi_n(x_1) \\ \varphi_0(x_2) & \varphi_1(x_2) & \cdots & \varphi_n(x_2) \\ \vdots & \vdots & & \vdots \\ \varphi_0(x_m) & \varphi_1(x_m) & \cdots & \varphi_n(x_m) \end{pmatrix} \in \mathfrak{R}^{m\times(n+1)}, \qquad \boldsymbol{c} = \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix}, \qquad \boldsymbol{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix},$$

la i-esima componente del vettore Ac è esattamente $p(x_i)$, e quindi $r = Ac - y$ è il vettore degli scarti. La 17.2 si riscrive

$$\min_{\boldsymbol{c}\in\mathfrak{R}^{n+1}} \|\boldsymbol{A}\boldsymbol{c} - \boldsymbol{y}\|_2^2. \tag{17.3}$$

La matrice $\boldsymbol{A}$, alta e stretta, si chiama **matrice di progetto**. Si noti dove entra la base: una colonna di $\boldsymbol{A}$ è una funzione della base valutata sui nodi. Con i monomi $\varphi_j(x) = x^j$ si ottiene la matrice di Vandermonde[1] $A_{ij} = x_i^j$; con una base ortogonale si ottiene $A_{ij} = \varphi_j(x_i)$. **Lo spazio $\mathcal{P}_n$ è lo stesso**, e quindi anche il polinomio ottimo p è lo stesso: cambiano solo i numeri c_j che lo descrivono e, come vedremo, la difficoltà di calcolarli.

17.2.3 Le equazioni normali

Per trovare il minimo sviluppiamo la funzione obiettivo. Scrivendo la norma come prodotto scalare,

$$\Psi(c) = \|\boldsymbol{A}\boldsymbol{c} - \boldsymbol{y}\|_2^2 = (\boldsymbol{A}\boldsymbol{c} - \boldsymbol{y})^T(\boldsymbol{A}\boldsymbol{c} - \boldsymbol{y}) = \boldsymbol{c}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{c} - 2\,\boldsymbol{c}^T\boldsymbol{A}^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{y}.$$

È un polinomio di secondo grado nelle $n+1$ variabili c_j, con termine quadratico $\boldsymbol{c}^T(\boldsymbol{A}^T\boldsymbol{A})\boldsymbol{c} = \|\boldsymbol{A}\boldsymbol{c}\|_2^2 \geq 0$: la funzione è una «parabola multidimensionale» rivolta verso l'alto, e il suo unico punto di minimo è il punto in cui tutte le derivate parziali si annullano. Imponendo $\nabla\Psi(\boldsymbol{c}) = 0$ si ottiene

$$\boldsymbol{A}^T\boldsymbol{A}\,\boldsymbol{c} = \boldsymbol{A}^T\boldsymbol{y}. \tag{17.4}$$

[1] Alexandre-Théophile Vandermonde (1735-1796)

Sono le **equazioni normali**: un sistema lineare quadrato di ordine $n+1$, dunque piccolo anche se i dati sono tanti. La matrice $\boldsymbol{A}^T A$ è invertibile non appena le colonne di $\boldsymbol{A}$ sono linearmente indipendenti, cosa che accade sempre se i nodi x_i sono almeno $n+1$ e distinti.

Il nome «normali» viene dalla lettura geometrica. L'equazione 17.4 si può riscrivere come

$$\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{c} - \boldsymbol{y}) = 0,$$

cioè: il vettore residuo $\boldsymbol{r} = \boldsymbol{A}\boldsymbol{c} - \boldsymbol{y}$ è **ortogonale a tutte le colonne di $\boldsymbol{A}$**, quindi all'intero sottospazio generato da esse. Il punto $\boldsymbol{A}\boldsymbol{c}$ è la proiezione ortogonale di $\boldsymbol{y}$ su quel sottospazio — il punto del sottospazio più vicino a $\boldsymbol{y}$ — ed è lo stesso quadro geometrico della Figura 17.9, qui in versione discreta.

17.2.4 Che cosa cambia cambiando base

Guardiamo da vicino l'elemento di posto (j,k) della matrice del sistema:

$$(\boldsymbol{A}^T\boldsymbol{A})_{jk} = \sum_{i=1}^{m} \varphi_j(x_i)\,\varphi_k(x_i) = \langle \varphi_j, \varphi_k \rangle_{\text{discreto}}. \tag{17.5}$$

È un prodotto scalare fra due funzioni della base, calcolato **sui nodi** anziché con un integrale. Da qui l'idea che regge tutta la sezione:

> se la base è ortogonale rispetto a questo prodotto scalare discreto, allora $\boldsymbol{A}^T A$ è **diagonale** e il sistema 17.4 si risolve con $n+1$ divisioni, senza eliminazione di Gauss.

L'esempio che segue mostra la cosa in concreto.

17.3 Un esempio svolto a mano: cinque punti, una parabola

Prendiamo cinque dati con ascisse intere, in modo che tutti i conti si possano fare a mente:

x_i	-2	-1	0	1	2
y_i	2	1	0	1	4

(17.6)

e cerchiamo la **parabola** ($n=2$, quindi tre incognite) che minimizza la somma dei quadrati degli scarti. I dati sono cinque, le incognite tre: siamo nel caso $m > n+1$.

17.3.1 Prima versione: la base monomiale

Con $\varphi_0 = 1$, $\varphi_1 = x$, $\varphi_2 = x^2$ la matrice di progetto e i dati sono

$$\boldsymbol{A} = \begin{pmatrix} 1 & -2 & 4 \\ 1 & -1 & 1 \\ 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \end{pmatrix}, \qquad \boldsymbol{y} = \begin{pmatrix} 2 \\ 1 \\ 0 \\ 1 \\ 4 \end{pmatrix}.$$

Costruiamo $\boldsymbol{A}^T\boldsymbol{A}$ usando 17.5. Servono solo cinque somme di potenze:

$$\textstyle\sum_i 1 = 5, \quad \sum_i x_i = 0, \quad \sum_i x_i^2 = 10, \quad \sum_i x_i^3 = 0, \quad \sum_i x_i^4 = 34,$$

(le somme di potenze dispari sono nulle perché i nodi sono simmetrici rispetto all'origine), e analogamente $\sum_i y_i = 8$, $\sum_i x_i y_i = 4$, $\sum_i x_i^2 y_i = 26$. Le equazioni normali sono dunque

$$\begin{pmatrix} 5 & 0 & 10 \\ 0 & 10 & 0 \\ 10 & 0 & 34 \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 8 \\ 4 \\ 26 \end{pmatrix}. \tag{17.7}$$

La seconda equazione è isolata e dà subito $c_1 = 2/5$; la prima e la terza sono invece **accoppiate**, perché $(\boldsymbol{A}^T A)_{02} = \sum x_i^2 = 10 \neq 0$: le funzioni 1 e x^2 non sono ortogonali sui nodi. Bisogna risolvere il sistema 2×2

$$\begin{cases} 5\,c_0 + 10\,c_2 = 8, \\ 10\,c_0 + 34\,c_2 = 26, \end{cases}$$

che per sostituzione dà $14\,c_2 = 10$, cioè $c_2 = 5/7$, e quindi $c_0 = 6/35$. La parabola cercata è

$$p(x) = \frac{6}{35} + \frac{2}{5}\,x + \frac{5}{7}\,x^2. \tag{17.8}$$

17.3.2 Seconda versione: una base ortogonale sui nodi

Cambiamo base, tenendo lo stesso spazio $\mathcal{P}_2$. Vogliamo tre polinomi q_0, q_1, q_2, di gradi $0, 1, 2$, ortogonali rispetto al prodotto scalare discreto $\langle u, v\rangle = \sum_{i=1}^{5} u(x_i)v(x_i)$. Li costruiamo con Gram–Schmidt, esattamente come nel Capitolo 14 ma con la somma al posto dell'integrale.

$$q_0(x) = 1.$$

$$q_1(x) = x - \frac{\langle x, q_0\rangle}{\langle q_0, q_0\rangle}\,q_0(x) = x - \frac{0}{5} = x,$$

perché $\sum_i x_i = 0$.

$$q_2(x) = x^2 - \frac{\langle x^2, q_0\rangle}{\langle q_0, q_0\rangle}\,q_0(x) - \frac{\langle x^2, q_1\rangle}{\langle q_1, q_1\rangle}\,q_1(x) = x^2 - \frac{10}{5}\cdot 1 - \frac{0}{10}\cdot x = x^2 - 2.$$

Si noti che q_2 **dipende dai nodi**: con cinque nodi diversi verrebbe un altro polinomio. Non è il polinomio di Legendre di grado 2, che è ortogonale rispetto a un integrale, non a questa somma.

Con la nuova base la matrice di progetto è

$$\boldsymbol{Q} = \begin{pmatrix} 1 & -2 & 2 \\ 1 & -1 & -1 \\ 1 & 0 & -2 \\ 1 & 1 & -1 \\ 1 & 2 & 2 \end{pmatrix},$$

e la matrice del sistema diventa **diagonale**:

$$\boldsymbol{Q}^T\boldsymbol{Q} = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 14 \end{pmatrix}, \qquad \boldsymbol{Q}^T\boldsymbol{y} = \begin{pmatrix} 8 \\ 4 \\ 10 \end{pmatrix}.$$

Non c'è più nessun sistema da risolvere: ogni coefficiente è un rapporto fra due numeri,

$$d_k = \frac{\langle y, q_k\rangle}{\langle q_k, q_k\rangle} \qquad \Longrightarrow \qquad d_0 = \frac{8}{5}, \quad d_1 = \frac{4}{10} = \frac{2}{5}, \quad d_2 = \frac{10}{14} = \frac{5}{7}. \tag{17.9}$$

Il polinomio è

$$p(x) = \frac{8}{5} + \frac{2}{5}\,x + \frac{5}{7}\,(x^2 - 2),$$

e sviluppando il termine $\frac{5}{7}(x^2-2) = \frac{5}{7}x^2 - \frac{10}{7}$ con $\frac{8}{5} - \frac{10}{7} = \frac{56-50}{35} = \frac{6}{35}$ si ritrova **esattamente** la 17.8. Due strade, stesso polinomio: cambiare base non cambia il risultato, cambia solo la fatica.

Il residuo, uguale nelle due versioni, vale

$$\sum_{i=1}^{5} r_i^2 = \frac{16}{35} \simeq 0{,}4571.$$

> **Due vantaggi, non uno**
>
> Oltre a eliminare il sistema lineare, 17.9 ha una proprietà che le equazioni normali piene non hanno: **i coefficienti non si ricalcolano**. Se dopo aver trovato d_0, d_1, d_2 si decide di aggiungere il termine di grado 3, basta calcolare d_3: i primi tre restano quelli. Con i monomi, invece, aggiungere una colonna cambia tutte le componenti della soluzione, perché il sistema è accoppiato.

17.3.3 Terza versione: costruire la base dai dati

Con cinque nodi e grado due Gram–Schmidt si fa a mano. Con m nodi qualunque e grado n qualunque serve un procedimento sistematico, e soprattutto serve capire **che cosa** si sta ortogonalizzando.

17.3.3.1 Il prodotto scalare discreto

Fissati i nodi $x_1, \dots, x_m$ e, se si vuole, dei pesi $w_i > 0$ che dicono quanto ciascun dato è affidabile, poniamo

$$\langle u, v \rangle_d = \sum_{i=1}^{m} w_i \, u(x_i)\, v(x_i), \qquad \|u\|_d^2 = \langle u, u \rangle_d = \sum_{i=1}^{m} w_i \, u(x_i)^2. \tag{17.10}$$

Il pedice d sta per «discreto». Questa è, parola per parola, la costruzione dei capitoli precedenti con la **somma al posto dell'integrale**: tutto quello che è stato dimostrato lì continua a valere, purché $\langle \cdot, \cdot \rangle_d$ sia davvero un prodotto scalare.

> **L'unica differenza rispetto al caso continuo**
>
> Un prodotto scalare deve soddisfare $\|u\|_d = 0 \Rightarrow u = 0$. Qui $\|u\|_d = 0$ significa solo $u(x_i) = 0$ per ogni i: il polinomio si annulla **nei nodi**, non ovunque. Un polinomio di grado m può annullarsi in tutti gli m nodi senza essere nullo — per esempio $(x - x_1) \cdots (x - x_m)$.
> Dunque 17.10 è un prodotto scalare solo su $\mathcal{P}_{m-1}$, e la costruzione che segue funziona finché
>
> $$n \le m - 1,$$
>
> cioè finché i dati sono almeno quanti i coefficienti. È esattamente l'ipotesi del Passo 1, e non è una coincidenza.

17.3.3.2 Perché non Gram–Schmidt sui monomi

Si potrebbe ortogonalizzare la lista $1, x, x^2, \dots, x^n$ con Gram–Schmidt, come abbiamo fatto a mano. Per n piccolo va bene, ma è la strada sbagliata per due motivi, entrambi già incontrati nel Capitolo 14: costa $O(mn^2)$ operazioni, e parte da una base pessimamente condizionata, con tutti i problemi di cancellazione che ne derivano. Esiste un'alternativa che costa $O(mn)$ e non nomina mai i monomi.

17.3.3.3 La ricorrenza a tre termini, versione discreta

I polinomi q_k che cerchiamo siano **monici** di grado k e ortogonali fra loro rispetto a 17.10. Ripetiamo il ragionamento del Capitolo 16.

Il polinomio $x\,q_k$ è monico di grado $k+1$, quindi la differenza $x\,q_k - q_{k+1}$ ha grado al più k e si scrive nella base $q_0, \dots, q_k$:

$$x\,q_k = q_{k+1} + \sum_{j=0}^{k} \alpha_j\, q_j, \qquad \alpha_j = \frac{\langle x\,q_k, q_j\rangle_d}{\langle q_j, q_j\rangle_d}.$$

Come nel caso continuo, il fattore x si sposta da una parte all'altra del prodotto scalare, $\langle x\,q_k, q_j\rangle_d = \langle q_k, x\,q_j\rangle_d$, e se $j \le k-2$ il polinomio $x\,q_j$ ha grado $\le k-1$: il prodotto scalare è nullo. **Sopravvivono solo $j = k$ e $j = k-1$.** Isolando q_{k+1} si ottiene

$$q_{k+1}(x) = (x - \beta_k)\,q_k(x) - \gamma_k\,q_{k-1}(x), \qquad q_{-1} = 0, \quad q_0 = 1, \tag{17.11}$$

con

$$\beta_k = \frac{\langle x\,q_k,\ q_k\rangle_d}{\langle q_k, q_k\rangle_d} = \frac{\sum_i w_i\,x_i\,q_k(x_i)^2}{\sum_i w_i\,q_k(x_i)^2}, \qquad \gamma_k = \frac{\langle q_k, q_k\rangle_d}{\langle q_{k-1}, q_{k-1}\rangle_d} = \frac{\|q_k\|_d^2}{\|q_{k-1}\|_d^2}. \tag{17.12}$$

La formula di γ_k merita una riga: per definizione $\gamma_k = \langle x\,q_k, q_{k-1}\rangle_d / \|q_{k-1}\|_d^2$, e spostando il fattore x si ottiene $\langle q_k,\ x\,q_{k-1}\rangle_d$; ma $x\,q_{k-1}$ è monico di grado k, dunque uguale a q_k più termini di grado inferiore, che sono ortogonali a q_k. Resta $\langle x q_k, q_{k-1}\rangle_d = \|q_k\|_d^2$.

Si noti che β_k e γ_k si calcolano **dai dati**: sono medie pesate di quantità note al passo k, non richiedono di conoscere i polinomi in forma monomiale.

17.3.3.4 L'algoritmo

Mettendo insieme 17.11 e la formula dei coefficienti 17.9 si ottiene un procedimento che in **una sola passata** costruisce la base e il fit, senza mai nominare i monomi e senza mai formare una matrice.

Tre osservazioni sul costo. Ogni grado richiede un numero fisso di passate sugli m dati, quindi il costo totale è $O(mn)$ operazioni: niente matrici, niente sistemi lineari, nessuna $\boldsymbol{A}^T A$ da formare. La memoria necessaria sono due soli vettori di lunghezza m — i valori q_{k-1} e q_k nei nodi — più una manciata di scalari. E per **valutare** p_n in un punto nuovo non serve convertire in forma monomiale: basta rieseguire 17.11 in quel punto, accumulando i contributi $d_k q_k(t)$, come nel secondo riquadro della Figura 17.1.

17.3.3.5 L'algoritmo in R

```
base_ortogonale <- function(x, n, w = rep(1, length(x))) {
  Q <- matrix(0, length(x), n + 1); Q[, 1] <- 1
  h <- numeric(n + 1); b <- numeric(n + 1); g <- numeric(n + 1)
  h[1] <- sum(w)
  for (k in seq_len(n) - 1) {              # costruisce q_{k+1}
    qk <- Q[, k + 1]
    b[k + 1] <- sum(w * x * qk^2) / h[k + 1]
    Q[, k + 2] <- if (k == 0) (x - b[1]) * qk else {
      g[k + 1] <- h[k + 1] / h[k]
      (x - b[k + 1]) * qk - g[k + 1] * Q[, k]
    }
    h[k + 2] <- sum(w * Q[, k + 2]^2)
  }
  list(Q = Q, h = h, beta = b, gamma = g)
}

fit_ortogonale <- function(x, y, n, w = rep(1, length(x))) {
  B <- base_ortogonale(x, n, w)
  d <- as.vector(crossprod(B$Q, w * y)) / B$h        # una divisione per grado
  c(B, list(d = d, fitted = as.vector(B$Q %*% d)))
}
```

Base ortogonale sui dati e fit ai minimi quadrati di grado n

Dati: nodi $x_1, \dots, x_m$; dati $y_1, \dots, y_m$; pesi $w_i > 0$; grado $n \le m-1$
Restituisce: coefficienti $d_0, \dots, d_n$ e, per la valutazione, β_k, γ_k

$q^- \leftarrow (0, \dots, 0)$, $q \leftarrow (1, \dots, 1)$ ▷ q_{k-1} *e* q_k *nei nodi*
$h \leftarrow \sum_i w_i$, $h^- \leftarrow 1$ ▷ $h_k = \|q_k\|_d^2$
$d_0 \leftarrow \left(\sum_i w_i\, y_i\right)/h$
per $k = 0, 1, \dots, n-1$ **fai**
  $\beta_k \leftarrow \dfrac{1}{h} \sum_i w_i\, x_i\, q_i^2$ ▷ *media pesata delle ascisse*
  $\gamma_k \leftarrow h/h^-$ (con $\gamma_0 = 0$)
  $q_i^+ \leftarrow (x_i - \beta_k)\, q_i - \gamma_k\, q_i^-$, $i = 1, \dots, m$ ▷ *nuovo grado*
  $q^- \leftarrow q$, $q \leftarrow q^+$, $h^- \leftarrow h$
  $h \leftarrow \sum_i w_i\, q_i^2$
  $d_{k+1} \leftarrow \left(\sum_i w_i\, y_i\, q_i\right)/h$ ▷ *una divisione, nessun sistema*
restituisci $(d_k)_{k \le n}$, $(\beta_k, \gamma_k)_{k < n}$

Valutazione di p_n in un punto t

Dati: punto t; coefficienti d_k; ricorrenza β_k, γ_k
Restituisce: $p_n(t) = \sum_{k=0}^{n} d_k\, q_k(t)$

$u^- \leftarrow 0$, $u \leftarrow 1$, $s \leftarrow d_0$ ▷ $u = q_k(t)$
per $k = 0, 1, \dots, n-1$ **fai**
  $u^+ \leftarrow (t - \beta_k)\, u - \gamma_k\, u^-$ ▷ *la stessa ricorrenza, in* t
  $u^- \leftarrow u$, $u \leftarrow u^+$
  $s \leftarrow s + d_{k+1}\, u$
restituisci s

Figure 17.1: Costruzione della base ortogonale sui dati e calcolo dei coefficienti del fit (riquadro in alto); valutazione del polinomio risultante in un punto qualunque, rieseguendo la stessa ricorrenza (riquadro in basso).

```
valuta <- function(t, f) {                  # rifa` la ricorrenza nei punti t
  qp <- rep(0, length(t)); q <- rep(1, length(t)); s <- f$d[1] * q
  for (k in seq_along(f$d)[-1] - 1) {
    qn <- (t - f$beta[k]) * q - f$gamma[k] * qp
    qp <- q; q <- qn; s <- s + f$d[k + 1] * q
  }
  s
}
```

Tre funzioni di poche righe, nessuna libreria di algebra lineare. Proviamole sui cinque dati dell'esempio fatto a mano.

```
x <- c(-2, -1, 0, 1, 2)
y <- c( 2,  1, 0, 1, 4)
f <- fit_ortogonale(x, y, 2)

knitr::kable(data.frame(
  k        = 0:2,
  beta_k   = f$beta[1:3],
  gamma_k  = f$gamma[1:3],
  h_k      = f$h,
  d_k      = f$d,
  frazione = c("8/5", "2/5", "5/7")))
```

Table 17.1: L'algoritmo sui cinque dati fatti a mano: ritrova esattamente beta, gamma, le norme e i coefficienti calcolati sopra.

k	beta_k	gamma_k	h_k	d_k	frazione
0	0	0	5	1.6000000	8/5
1	0	2	10	0.4000000	2/5
2	0	0	14	0.7142857	5/7

I valori coincidono con quelli trovati a mano: $h = (5, 10, 14)$, $\beta_k = \gamma_k$ nulli o pari a 2 per la simmetria dei nodi, e $d = (8/5,\ 2/5,\ 5/7)$.

17.3.4 Le due basi a confronto, in R

```
x <- c(-2, -1, 0, 1, 2)
y <- c( 2,  1, 0, 1, 4)

A <- cbind(1, x, x^2)        # base monomiale
Q <- cbind(1, x, x^2 - 2)    # base ortogonale sui cinque nodi

cA <- solve(crossprod(A), crossprod(A, y))   # risolve un sistema 3x3
cQ <- crossprod(Q, y) / diag(crossprod(Q))   # tre sole divisioni

knitr::kable(data.frame(
  base         = c("monomi", "ortogonale"),
  residuo_RSS  = c(sum((A %*% cA - y)^2), sum((Q %*% cQ - y)^2)),
  kappa2_A     = c(kappa(A, exact = TRUE), kappa(Q, exact = TRUE)),
  kappa2_AtA   = c(kappa(crossprod(A), exact = TRUE),
                   kappa(crossprod(Q), exact = TRUE))))
```

Table 17.2: I cinque dati dell'esempio svolto, nelle due basi: stesso polinomio, stesso residuo, matrici del sistema molto diverse.

base	residuo_RSS	kappa2_A	kappa2_AtA
monomi	0.4571429	4.435961	19.67775
ortogonale	0.4571429	1.673320	2.80000

```
xx <- seq(-2.3, 2.3, length.out = 400)
p  <- function(t) 6/35 + (2/5)*t + (5/7)*t^2
plot(x, y, pch = 19, cex = 1.3, xlab = "x", ylab = "y",
     xlim = c(-2.4, 2.4), ylim = c(-0.4, 4.4),
     main = "Minimi quadrati: cinque dati, una parabola")
segments(x, y, x, p(x), col = "grey45", lwd = 2, lty = 2)
lines(xx, p(xx), col = 4, lwd = 2)
points(x, p(x), pch = 1, cex = 1.3, col = 4)
legend("top", c("dati", "parabola ottima", "scarti"),
       pch = c(19, NA, NA), lty = c(NA, 1, 2), lwd = c(NA, 2, 2),
       col = c(1, 4, "grey45"), bty = "n", horiz = TRUE)
```

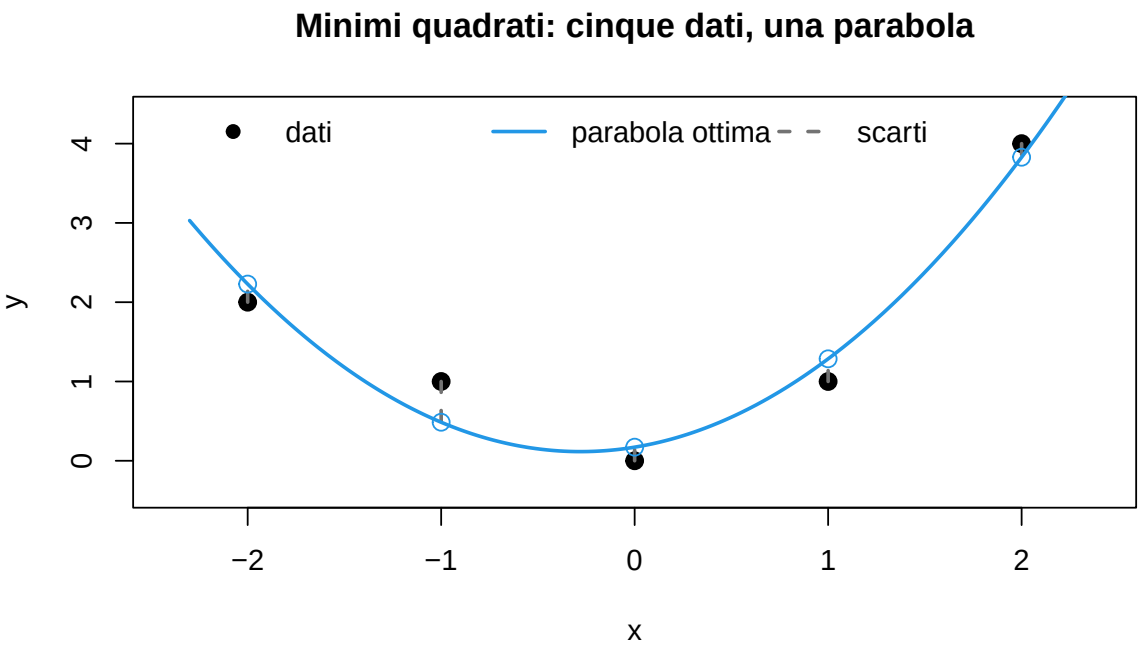


Figure 17.2: I cinque dati e la parabola dei minimi quadrati. I segmenti verticali sono gli scarti: il metodo rende minima la somma dei loro quadrati.

Le due basi danno lo stesso residuo a precisione macchina, come previsto. I due numeri di condizionamento, invece, differiscono già in questo esempio minuscolo: $\kappa_2(\boldsymbol{A}) \simeq 4{,}44$ con i monomi contro $\kappa_2(\boldsymbol{Q}) = \sqrt{14/5} \simeq 1{,}67$ con la base ortogonale. L'ultima colonna mostra il fenomeno che studiamo nel prossimo paragrafo: il condizionamento di $\boldsymbol{A}^T A$ è il **quadrato** di quello di $\boldsymbol{A}$.

17.4 Tre esempi con l'algoritmo

I cinque dati di prima erano simmetrici rispetto all'origine, e questo faceva sparire quasi tutti i coefficienti della ricorrenza. Vediamo ora tre situazioni in cui non sparisce niente. Tutti e tre gli esempi usano gli stessi sei dati, che simulano una misura di laboratorio con ascisse **non**

equispaziate — tipicamente perché i campionamenti iniziali sono fitti e quelli finali radi:

x_i	0	1	2	5	10	20
y_i	1,02	1,95	2,61	4,15	5,71	7,42

(17.13)

17.4.1 Esempio A: la base che i dati si costruiscono addosso

Applichiamo l'algoritmo con $n = 2$ e pesi tutti uguali.

```
xs <- c(0, 1, 2, 5, 10, 20)
ys <- c(1.02, 1.95, 2.61, 4.15, 5.71, 7.42)
fA <- fit_ortogonale(xs, ys, 2)

knitr::kable(data.frame(k = 0:2, beta_k = fA$beta[1:3],
                        gamma_k = fA$gamma[1:3],
                        h_k = fA$h, d_k = fA$d), digits = 5)
```

Table 17.3: Esempio A: le quantita‘ prodotte dalla ricorrenza sui sei dati non equispaziati.

k	beta_k	gamma_k	h_k	d_k
0	6.33333	0.00000	6.0000	3.81000
1	13.63441	48.22222	289.3333	0.30636
2	0.00000	0.00000	8465.0323	-0.01584

Leggiamo la tabella riga per riga, perché ogni numero ha un significato.

- $\beta_0 = \sum_i x_i/6 = 6{,}333$ è la **media delle ascisse**. Di conseguenza

$$q_1(x) = x - \beta_0 = x - 6{,}333$$

  è la variabile *centrata*: il primo polinomio ortogonale non fa altro che spostare l'origine nel baricentro dei dati. Nell'esempio precedente i nodi erano già centrati e infatti era $\beta_0 = 0$.
- $h_0 = 6$ è il numero dei dati, $h_1 = 289{,}3$ è la loro dispersione $\sum_i (x_i - \bar{x})^2$.
- $\gamma_1 = h_1/h_0 = 48{,}22$ e $\beta_1 = 13{,}63$ producono

$$q_2(x) = (x - 13{,}63)\, q_1(x) - 48{,}22,$$

  un polinomio di secondo grado **costruito su misura per questi sei nodi**: non è né x^2, né il polinomio di Legendre di grado due, né il $x^2 - 2$ dell'esempio precedente.
- $d_0 = 3{,}81$ è la media delle ordinate: il fit di grado zero è, come deve essere, la media dei dati.

La verifica che la base sia davvero ortogonale è immediata: la matrice $\boldsymbol{Q}^T Q$ deve risultare diagonale.

```
G <- round(crossprod(fA$Q), 10)
dimnames(G) <- list(c("q0", "q1", "q2"), c("q0", "q1", "q2"))
knitr::kable(G)
```

Table 17.4: Esempio A: la matrice Q^T Q e‘ diagonale a meno dell'errore di arrotondamento; sulla diagonale ci sono le norme h_k.

	q0	q1	q2
q0	6	0.0000	0.000
q1	0	289.3333	0.000

Table 17.4: Esempio A: la matrice Q^T Q e' diagonale a meno dell'errore di arrotondamento; sulla diagonale ci sono le norme h_k.

	q0	q1	q2
q2	0	0.0000	8465.032

La Figura 17.3 mostra i tre polinomi della base — ciascuno riscalato per poterli disegnare insieme — e il fit risultante.

```
tt <- seq(-1, 21, length.out = 400)
B  <- base_ortogonale(tt, 2)$Q
op <- par(mfrow = c(1, 2), mar = c(4, 4, 2.5, 0.8))

Bs <- sweep(B, 2, apply(abs(B), 2, max), "/")   # riscalati per il disegno
matplot(tt, Bs, type = "l", lty = 1, lwd = 2, col = c(1, 4, 2),
        ylim = c(-1.15, 1.45),
        xlab = "x", ylab = "q_k (riscalato)", main = "La base ortogonale")
abline(h = 0, col = "grey70"); rug(xs, lwd = 2)
legend("top", c("q0", "q1", "q2"), col = c(1, 4, 2),
       lwd = 2, bty = "n", horiz = TRUE)

plot(xs, ys, pch = 19, cex = 1.2, xlab = "x", ylab = "y",
     main = "Fit di grado 2")
lines(tt, valuta(tt, fA), col = 4, lwd = 2)
segments(xs, ys, xs, fA$fitted, col = "grey45", lty = 2, lwd = 2)
par(op)
```

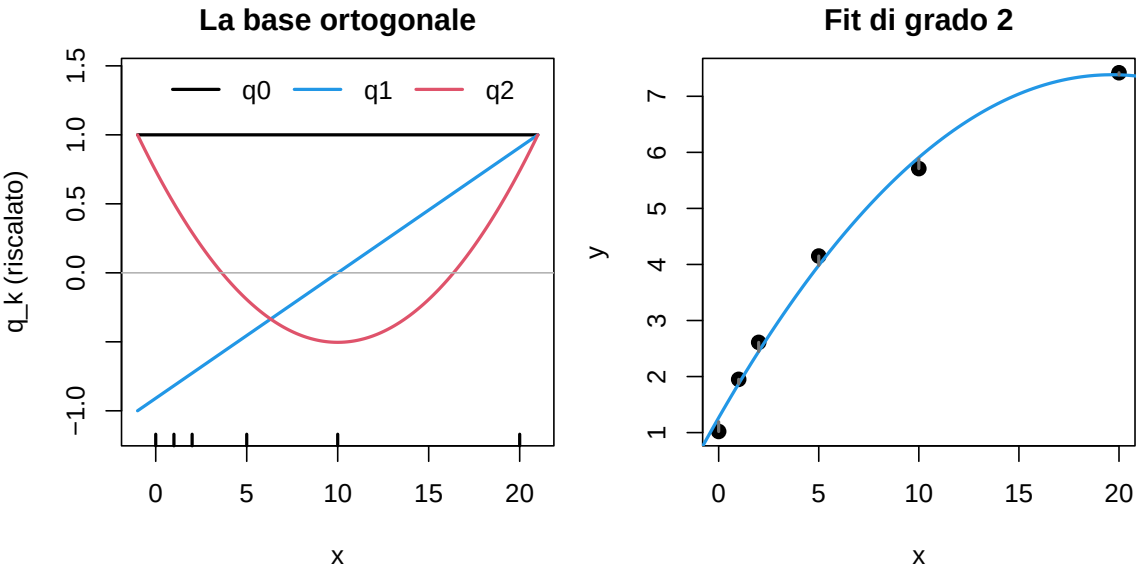


Figure 17.3: Esempio A. A sinistra la base ortogonale generata dai sei nodi: q0 costante, q1 lineare che si annulla nel baricentro, q2 quadratico con due zeri interni. A destra il fit di grado 2.

Si noti, nel pannello di sinistra, che q_1 si annulla nel baricentro 6,33 e che q_2 ha due zeri, entrambi interni all'intervallo dei dati: è il Teorema 16.2, valido anche nella versione discreta.

Infine il confronto con i monomi. Qui le ascisse arrivano a 20, dunque la colonna x^2 arriva a 400: le tre colonne della Vandermonde hanno ordini di grandezza molto diversi, e il condizionamento ne risente.

```
AA <- cbind(1, xs, xs^2)
cA2 <- solve(crossprod(AA), crossprod(AA, ys))
knitr::kable(data.frame(
  base        = c("monomi", "ortogonale"),
  residuo_RSS = c(sum((AA %*% cA2 - ys)^2), sum((fA$fitted - ys)^2)),
  kappa2_A    = c(kappa(AA, exact = TRUE), kappa(fA$Q, exact = TRUE)),
  kappa2_AtA  = c(kappa(crossprod(AA), exact = TRUE),
                  kappa(crossprod(fA$Q), exact = TRUE))), digits = 8)
```

Table 17.5: Esempio A: stesso polinomio, stesso residuo, condizionamento diverso.

base	residuo_RSS	kappa2_A	kappa2_AtA
monomi	0.1613141	293.87043	86359.832
ortogonale	0.1613141	37.56113	1410.839

Il residuo è lo stesso fino all'ultima cifra, come deve essere. Il condizionamento della matrice di progetto passa invece da circa 294 a circa 38, e quello di $A^T A$ da $8{,}6 \cdot 10^4$ a $1{,}4 \cdot 10^3$.

> **Ortogonale non vuol dire ortonormale**
>
> Perché $\kappa_2(Q) \simeq 38$ e non 1? Perché le colonne di Q sono sì perpendicolari, ma hanno lunghezze molto diverse: $h_0 = 6$ contro $h_2 = 8465$. Il condizionamento di una matrice a colonne ortogonali è
>
> $$\kappa_2(Q) = \sqrt{\frac{\max_k h_k}{\min_k h_k}} = \sqrt{\frac{8465}{6}} \simeq 37{,}6.$$
>
> Dividendo ogni colonna per $\sqrt{h_k}$ — cioè passando alla base **ortonormale** $\phi_k = q_k / \|q_k\|_d$ — si ottiene $\kappa_2 = 1$, il minimo possibile. È una riga di codice in più e non cambia il polinomio: cambia solo la scala dei coefficienti.

17.4.2 Esempio B: alzare il grado non ricalcola nulla

Questa è la proprietà annunciata nel callout «Due vantaggi, non uno», ed è il motivo pratico per cui in laboratorio si usa questa base. Rifacciamo il fit degli stessi sei dati con $n = 1$, $n = 2$ e $n = 3$, e confrontiamo i coefficienti nelle due basi.

```
righe <- lapply(1:3, function(n) {
  fo <- fit_ortogonale(xs, ys, n)
  Am <- outer(xs, 0:n, `^`)
  cm <- qr.solve(Am, ys)
  data.frame(grado = n, base = c("ortogonale", "monomi"),
             coef0 = c(fo$d[1], cm[1]), coef1 = c(fo$d[2], cm[2]),
             coef2 = c(if (n >= 2) fo$d[3] else NA, if (n >= 2) cm[3] else NA),
             coef3 = c(if (n >= 3) fo$d[4] else NA, if (n >= 3) cm[4] else NA))
})
knitr::kable(do.call(rbind, righe), digits = 5)
```

Table 17.6: Esempio B: coefficienti per gradi crescenti. Nella base ortogonale le colonne gia' calcolate non cambiano; con i monomi cambiano tutte.

grado	base	coef0	coef1	coef2	coef3
1	ortogonale	3.81000	0.30636	NA	NA
1	monomi	1.86972	0.30636	NA	NA
2	ortogonale	3.81000	0.30636	-0.01584	NA
2	monomi	1.26569	0.62268	-0.01584	NA
3	ortogonale	3.81000	0.30636	-0.01584	0.00121
3	monomi	1.08011	0.84849	-0.05086	0.00121

Si guardino le righe «ortogonale»: la colonna `coef0` vale 3,81 per tutti e tre i gradi, `coef1` vale 0,30636 per tutti e tre. Alzare il grado **aggiunge** un coefficiente e lascia stare gli altri, perché la nuova direzione q_{n+1} è perpendicolare a tutte le precedenti: la proiezione su quelle non cambia.

Nelle righe «monomi» invece c_0 passa da 1,87 a 1,27 a 1,08 e c_1 da 0,31 a 0,62 a 0,85: tutto si rimescola, perché le colonne $1, x, x^2, x^3$ sono fra loro correlate.

> **Consiglio**
>
> Un'eccezione apparente: il coefficiente di grado massimo coincide sempre nelle due basi ($-0{,}01584$ per $n = 2$, 0,00121 per $n = 3$). Non è un caso: i q_k sono monici, quindi d_n e c_n sono entrambi il coefficiente di x^n del medesimo polinomio.

Questo rende naturale la scelta del grado: si costruisce la base una volta sola fino a un grado massimo, si guarda come decrescono i contributi $d_k^2 h_k$ — che sono esattamente le riduzioni del residuo apportate da ciascun grado — e ci si ferma quando smettono di decrescere in modo significativo.

```
fB <- fit_ortogonale(xs, ys, 4)
contrib <- fB$d^2 * fB$h
knitr::kable(data.frame(
  grado_k         = 0:4,
  contributo      = contrib,
  residuo_dopo_k  = sum(ys^2) - cumsum(contrib)), digits = 6)
```

Table 17.7: Esempio B: contributo di ciascun grado alla riduzione del residuo e residuo rimanente.

grado_k	contributo	residuo_dopo_k
0	87.096600	29.441400
1	27.155701	2.285699
2	2.124384	0.161314
3	0.148873	0.012441
4	0.010254	0.002187

La tabella è la traduzione numerica dell'identità di Pitagora: poiché i q_k sono ortogonali,

$$\sum_{i=1}^{m} y_i^2 = \sum_{k=0}^{n} d_k^2 \, h_k + \text{residuo}, \tag{17.14}$$

cioè l'energia dei dati si ripartisce fra i gradi, e ogni grado aggiunto ne sottrae una quota $d_k^2 h_k \geq 0$ al residuo. Con i monomi una decomposizione del genere non esiste.

17.4.3 Esempio C: dati di qualità diversa

Supponiamo ora che i primi tre dati provengano da uno strumento accurato, con incertezza $\sigma = 0{,}05$, e gli ultimi tre da uno strumento grossolano, con $\sigma = 0{,}4$. Trattarli alla pari sarebbe sbagliato: il criterio standard è pesare ciascun dato con l'inverso della varianza,

$$w_i = \frac{1}{\sigma_i^2},$$

cosicché il problema diventa $\min \sum_i w_i \, [p(x_i) - y_i]^2$. L'algoritmo non cambia di una riga: i pesi entrano in 17.10 e si propagano in $\beta_k, \gamma_k, h_k, d_k$.

```
sigma <- c(0.05, 0.05, 0.05, 0.4, 0.4, 0.4)
tt <- seq(-1, 21, length.out = 400)
fU <- fit_ortogonale(xs, ys, 2)                       # pesi uguali
fW <- fit_ortogonale(xs, ys, 2, w = 1/sigma^2)        # pesi 1/sigma^2

plot(xs, ys, pch = 19, cex = 1.2, ylim = c(0.5, 8), xlab = "x", ylab = "y",
     main = "Minimi quadrati pesati")
arrows(xs, ys - sigma, xs, ys + sigma, angle = 90, code = 3,
       length = 0.05, lwd = 2, col = "grey40")
lines(tt, valuta(tt, fU), col = 2, lwd = 2, lty = 2)
lines(tt, valuta(tt, fW), col = 4, lwd = 2)
legend("bottomright", c("pesi uguali", "pesi 1/sigma^2", "barre d'errore"),
       col = c(2, 4, "grey40"), lty = c(2, 1, 1), lwd = 2, bty = "n")
```

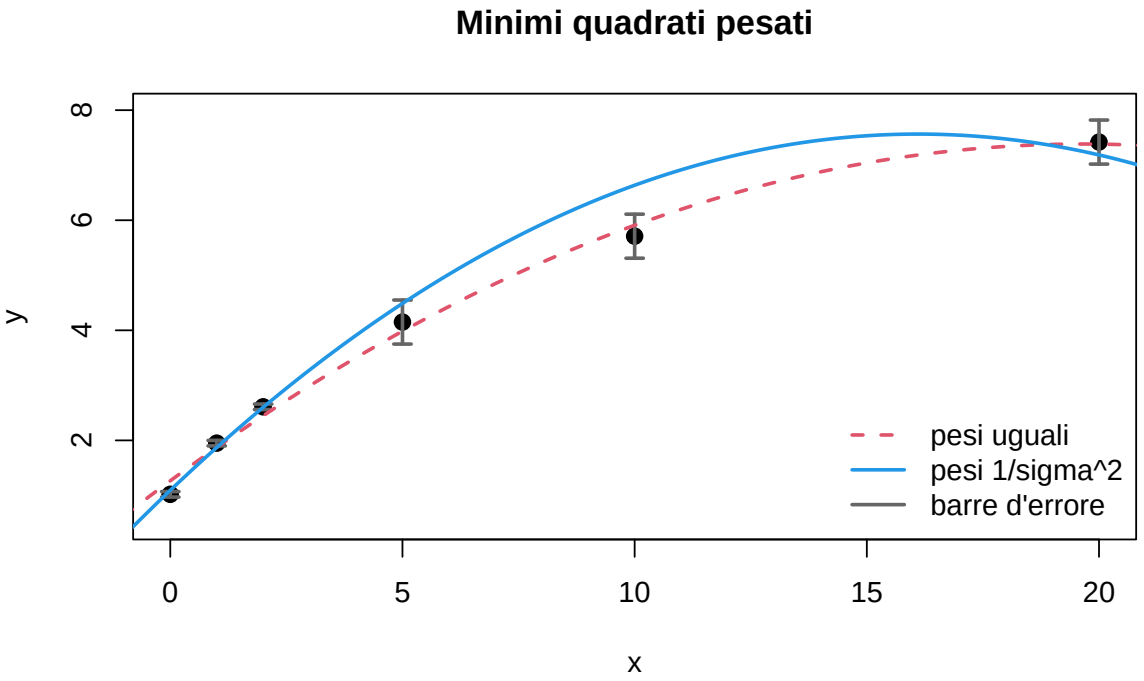


Figure 17.4: Esempio C: con pesi uguali il fit distribuisce l'errore su tutti i punti; pesando con 1/sigma^2 passa vicino ai tre dati accurati e si allontana dagli altri tre.

```
knitr::kable(data.frame(
  x              = xs,
  sigma          = sigma,
  scarto_uniforme = fU$fitted - ys,
  scarto_pesato   = fW$fitted - ys,
  scarto_su_sigma = (fW$fitted - ys)/sigma), digits = 4)
```

Table 17.8: Esempio C: scarti punto per punto. Il fit pesato sacrifica i dati imprecisi per aderire ai tre accurati.

x	sigma	scarto_uniforme	scarto_pesato	scarto_su_sigma
0	0.05	0.2457	0.0723	1.4452
1	0.05	-0.0775	-0.0787	-1.5744
2	0.05	-0.1623	-0.0096	-0.1927
5	0.40	-0.1669	0.3381	0.8452
10	0.40	0.1984	0.9257	2.3141
20	0.40	-0.0373	-0.2338	-0.5845

Nei primi tre punti lo scarto del fit pesato è molto più piccolo di quello uniforme; negli ultimi tre è più grande. È il comportamento voluto: un errore di 0,3 su un dato con $\sigma = 0{,}4$ è irrilevante, uno di 0,25 su un dato con $\sigma = 0{,}05$ vale cinque deviazioni standard. L'ultima colonna, che riporta gli scarti **in unità di** σ, è la quantità che il metodo pesato rende uniformemente piccola, e quella da guardare per giudicare se il modello è adeguato.

17.5 Come si risolve davvero: la fattorizzazione QR

17.5.1 Perché le equazioni normali sono una cattiva idea

Le equazioni normali sono il modo più rapido per *capire* il problema, ma non il modo giusto per *risolverlo* in aritmetica finita. Il motivo sta in un'identità: se $\boldsymbol{A}$ ha colonne indipendenti,

$$\kappa_2(\boldsymbol{A}^T\boldsymbol{A}) = \kappa_2(\boldsymbol{A})^2. \tag{17.15}$$

Infatti i valori singolari di $\boldsymbol{A}^T\boldsymbol{A}$ sono i quadrati di quelli di $\boldsymbol{A}$, e κ_2 è il rapporto fra il più grande e il più piccolo. Formare $\boldsymbol{A}^T\boldsymbol{A}$ **raddoppia le cifre perse**: se $\kappa_2(\boldsymbol{A}) = 10^6$, già accettabile, allora $\kappa_2(\boldsymbol{A}^T\boldsymbol{A}) = 10^{12}$ e in doppia precisione (circa 16 cifre) ne restano quattro. Nell'esempio precedente si passa da 4,44 a $19{,}7 \simeq 4{,}44^2$: innocuo lì, rovinoso al crescere del grado, come mostra la Figura 17.5.

C'è di più: il prodotto $\boldsymbol{A}^T\boldsymbol{A}$ va calcolato, e sommare numeri di ordini di grandezza molto diversi distrugge informazione. Può accadere che $\boldsymbol{A}$ abbia colonne indipendenti ma che $\boldsymbol{A}^T\boldsymbol{A}$, *una volta calcolata in virgola mobile*, risulti numericamente singolare.

La conclusione è che le equazioni normali servono per ragionare, non per calcolare. Il metodo di riferimento è un altro.

17.5.2 Che cosa sono Q e R

Ogni matrice $\boldsymbol{A} \in \Re^{m\times(n+1)}$ con $m \geq n+1$ e colonne indipendenti si fattorizza come

$$\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}, \tag{17.16}$$

dove $\boldsymbol{Q} \in \Re^{m\times(n+1)}$ ha **colonne ortonormali** ($\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$) e $\boldsymbol{R} \in \Re^{(n+1)\times(n+1)}$ è **triangolare superiore** con elementi diagonali positivi. La costruzione è quella del Capitolo 4; concettualmente $\boldsymbol{Q}$ contiene la base ortonormale ottenuta ortogonalizzando le colonne di $\boldsymbol{A}$, e $\boldsymbol{R}$ registra i coefficienti usati per ottenerla.

17.5.3 Perché la QR risolve il problema

La proprietà che rende utile $\boldsymbol{Q}$ è che **moltiplicare per una matrice ortogonale non cambia le lunghezze**: se $\boldsymbol{U}$ è ortogonale ($\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}$) allora $\|\boldsymbol{U}\boldsymbol{v}\|_2 = \|\boldsymbol{v}\|_2$, perché $\|\boldsymbol{U}\boldsymbol{v}\|_2^2 = \boldsymbol{v}^T\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{v} = \boldsymbol{v}^T\boldsymbol{v}$. Una rotazione o una riflessione non allungano né accorciano i vettori.

Completiamo le colonne di $\boldsymbol{Q}$ con altre $m-(n+1)$ colonne ortonormali, ottenendo una matrice quadrata ortogonale $\boldsymbol{U} = \begin{pmatrix} \boldsymbol{Q} & \widetilde{\boldsymbol{Q}} \end{pmatrix}$. Allora, per ogni c,

$$\|\boldsymbol{A}\boldsymbol{c}-\boldsymbol{y}\|_2 = \|\boldsymbol{U}^T(\boldsymbol{A}\boldsymbol{c}-\boldsymbol{y})\|_2 = \left\| \begin{pmatrix} \boldsymbol{Q}^T \\ \widetilde{\boldsymbol{Q}}^T \end{pmatrix} (\boldsymbol{Q}\boldsymbol{R}\boldsymbol{c}-\boldsymbol{y}) \right\|_2 = \left\| \begin{pmatrix} \boldsymbol{R}\boldsymbol{c}-\boldsymbol{Q}^T\boldsymbol{y} \\ -\widetilde{\boldsymbol{Q}}^T\boldsymbol{y} \end{pmatrix} \right\|_2,$$

dove si è usato $\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$ e $\widetilde{\boldsymbol{Q}}^T\boldsymbol{Q} = \boldsymbol{0}$. Elevando al quadrato, le due parti si sommano:

$$\|\boldsymbol{A}\boldsymbol{c}-\boldsymbol{y}\|_2^2 = \underbrace{\|\boldsymbol{R}\boldsymbol{c}-\boldsymbol{Q}^T\boldsymbol{y}\|_2^2}_{\text{dipende da } c} + \underbrace{\|\widetilde{\boldsymbol{Q}}^T\boldsymbol{y}\|_2^2}_{\text{non dipende da } \boldsymbol{c}}. \tag{17.17}$$

Il secondo addendo non si può toccare: è il residuo minimo, quello che resta anche nella migliore delle ipotesi. Il primo si può invece **azzerare**, scegliendo $\boldsymbol{c}$ che risolve

$$\boldsymbol{R}\,\boldsymbol{c} = \boldsymbol{Q}^T\boldsymbol{y}. \tag{17.18}$$

17.5.4 Il sistema triangolare e il suo costo

Il sistema 17.18 è triangolare superiore, e si risolve dall'ultima equazione verso la prima (*sostituzione all'indietro*): l'ultima contiene una sola incognita, la penultima due di cui una già nota, e così via. Costa $\mathcal{O}(n^2)$ operazioni, contro le $\mathcal{O}(n^3)$ di un sistema pieno, e soprattutto non si è mai formata la matrice $\boldsymbol{A}^T\boldsymbol{A}$: si lavora con $\kappa_2(\boldsymbol{A})$, non con il suo quadrato.

In R tutto questo è `qr.solve(A, y)`, che esegue la fattorizzazione, calcola $\boldsymbol{Q}^T\boldsymbol{y}$ e risolve il sistema triangolare, segnalando se il rango numerico è insufficiente.

17.5.5 La QR sull'esempio dei cinque punti

Riprendiamo i dati 17.6. Con la base monomiale la fattorizzazione dà

$$\boldsymbol{R} = \begin{pmatrix} \sqrt{5} & 0 & 2\sqrt{5} \\ 0 & \sqrt{10} & 0 \\ 0 & 0 & \sqrt{14} \end{pmatrix},$$

a meno dei segni, che dipendono dalla convenzione. L'elemento $R_{02} = 2\sqrt{5} \neq 0$ è la traccia dell'accoppiamento fra le colonne 1 e x^2: è il motivo per cui prima abbiamo dovuto risolvere un sistema 2×2.

Con la base ortogonale, invece, le colonne di $\boldsymbol{Q}$ sono già fra loro perpendicolari e la fattorizzazione non ha nulla da ortogonalizzare: $\boldsymbol{R}$ risulta **diagonale**,

$$\boldsymbol{R} = \begin{pmatrix} \sqrt{5} & 0 & 0 \\ 0 & \sqrt{10} & 0 \\ 0 & 0 & \sqrt{14} \end{pmatrix},$$

con sulla diagonale proprio le norme $\|q_k\| = \sqrt{\langle q_k, q_k\rangle}$ delle colonne. Il sistema triangolare 17.18 degenera in tre divisioni, e si ritrovano i coefficienti 17.9.

Table 17.9: Fattore R nelle due basi. Con la base ortogonale R e‘ diagonale: la QR non ha nulla da fare.

riga	monomi_col1	monomi_col2	monomi_col3	ortog_col1	ortog_col2	ortog_col3
1	2.2361	0.0000	4.4721	2.2361	0.0000	0.0000
2	0.0000	3.1623	0.0000	0.0000	3.1623	0.0000
3	0.0000	0.0000	3.7417	0.0000	0.0000	3.7417

> **!** Il punto della sezione
>
> Usare una base ortogonale **non sostituisce** la fattorizzazione QR: ne anticipa il lavoro. La QR ortogonalizza le colonne di $\boldsymbol{A}$ numericamente, al momento del calcolo; una base ortogonale le consegna già ortogonali, perché l'ortogonalizzazione è stata fatta una volta per tutte in forma analitica (o con la ricorrenza 17.11). Nei due casi il risultato è lo stesso, ma nel secondo il fattore $\boldsymbol{R}$ è diagonale, il condizionamento di $\boldsymbol{A}$ è minimo e gli errori di arrotondamento hanno molto meno spazio per accumularsi.

17.6 Prima domanda: quanto si guadagna cambiando base

17.6.1 Le tre basi a confronto

Abbiamo ora tre candidate per le colonne della matrice di progetto.

Base	Ortogonale rispetto a	$\boldsymbol{A}^T A$ sui nodi
monomi $1, x, x^2, \dots$	niente	piena
Legendre $P_0, P_1, \dots$	l'integrale su $[-1, 1]$	*quasi* diagonale
$q_0, q_1, \dots$ dai dati	la somma sui nodi	**esattamente** diagonale

La riga centrale è quella che merita attenzione, perché è la più usata e la più fraintesa. I polinomi di Legendre soddisfano

$$\int_{-1}^{1} P_j(x) P_k(x)\, \mathrm{d}x = 0, \qquad j \neq k,$$

mentre il problema dei minimi quadrati discreti coinvolge

$$(\boldsymbol{A}^T A)_{jk} = \sum_{i=1}^{m} P_j(x_i)\, P_k(x_i).$$

La somma è una **quadratura** dell'integrale: se i nodi sono numerosi e ragionevolmente distribuiti, la somma vale circa l'integrale (a meno del fattore $m/2$) e quindi gli elementi fuori diagonale sono piccoli ma non nulli. Di qui il «quasi». E il «quasi» peggiora al crescere del grado, perché il prodotto $P_j P_k$ ha grado $j + k$ e una quadratura fissata integra bene solo fino a un certo grado.

L'esperimento seguente misura le due cose insieme: quanto $\boldsymbol{A}^T A$ si discosta dalla diagonale e quanto vale $\kappa_2(\boldsymbol{A})$, su $m = 120$ nodi equispaziati in $[-1, 1]$ e gradi crescenti.

```
xg <- seq(-1, 1, length.out = 120)

fuori_diag <- function(A) {                  # 0 = colonne ortogonali
  G <- crossprod(A); G <- G / sqrt(outer(diag(G), diag(G)))
  max(abs(G - diag(ncol(A))))
}

riga <- function(n) {
  M <- outer(xg, 0:n, `^`)                           # monomi
  L <- sapply(0:n, legendre, x = xg)                 # Legendre
  B <- base_ortogonale(xg, n)
  D <- sweep(B$Q, 2, sqrt(B$h), "/")                 # dai dati, ortonormale
  data.frame(grado = n,
             kappa_monomi    = kappa(M, exact = TRUE),
             kappa_legendre = kappa(L, exact = TRUE),
             kappa_dati      = kappa(D, exact = TRUE),
```

```
            fuoridiag_monomi    = fuori_diag(M),
            fuoridiag_legendre = fuori_diag(L),
            fuoridiag_dati      = fuori_diag(D))
}
knitr::kable(do.call(rbind, lapply(c(3, 6, 10, 15, 20), riga)), digits = 4)
```

Table 17.10: Tre basi a confronto su 120 nodi equispaziati. ‘fuori diagonale’ `eil massimo elemento extradiagonale della matrice di correlazione delle colonne: 0 significa ortogonalita` esatta.

grado	kappa_monomi	kappa_legendre	kappa_dati	fuoridiag_monomi	fuoridiag_legendre	fuoridiag_dati
3	8.076500e+00	2.5830	1	0.9166	0.0376	0
6	9.818410e+01	3.4735	1	0.9834	0.0897	0
10	2.945329e+03	4.4337	1	0.9945	0.1655	0
15	2.193304e+05	5.4536	1	0.9977	0.2727	0
20	1.716883e+07	6.3600	1	0.9987	0.3722	0

I numeri dicono tre cose.

- **I monomi collassano.** Il condizionamento passa da 8 a $1{,}7 \cdot 10^7$ fra grado 3 e grado 20, e le colonne diventano quasi parallele (correlazione fuori diagonale vicinissima a 1). È la disuguaglianza di Beckermann 14.5 in azione.
- **Legendre regge benissimo.** Il condizionamento resta sotto 7 anche a grado 20: un guadagno di sei ordini di grandezza. Gli elementi fuori diagonale crescono da 0,04 a 0,37: la matrice è davvero quasi diagonale a gradi bassi, un po’ meno a gradi alti, ma il condizionamento non ne risente quasi.
- **La base costruita sui dati è perfetta**, con $\kappa_2 = 1$ e fuori diagonale nullo a tutti i gradi — purché sia **normalizzata**.

⚠ Ortogonale non basta: serve ortonormale

I polinomi q_k prodotti dalla ricorrenza 17.11 sono monici, e le loro norme $h_k = \|q_k\|_d^2$ cambiano di ordini di grandezza. Le colonne sono perpendicolari, ma lunghissime alcune e cortissime altre, e il condizionamento di una matrice a colonne ortogonali vale

$$\kappa_2(\boldsymbol{Q}) = \sqrt{\frac{\max_k h_k}{\min_k h_k}},$$

che in questo esperimento arriva a $8 \cdot 10^5$ a grado 20: peggio di Legendre. Il rimedio è una riga di codice — dividere ogni colonna per $\sqrt{h_k}$, cioè usare

$$\phi_k = \frac{q_k}{\|q_k\|_d} \tag{17.19}$$

— dopo la quale $\kappa_2 = 1$ esattamente. Da qui in avanti useremo sempre i ϕ_k.

❗ Che cosa la base non fa

Girano affermazioni del tipo «i polinomi ortogonali filtrano il rumore» o «sono meno soggetti a sovradattamento». Sono **false**, e conviene dire perché.
Fissati i nodi e fissato il grado n, il polinomio dei minimi quadrati è **uno solo**: è il punto di $\mathcal{P}_n$ più vicino ai dati, e questo non dipende da come si scrive $\mathcal{P}_n$. Monomi, Legendre e base

dai dati producono lo stesso identico polinomio e lo stesso identico residuo — l'abbiamo verificato numericamente più volte in questo capitolo.
Quello che cambia è **con quanta precisione quel polinomio viene calcolato** in aritmetica finita. Con i monomi a grado 20 si perdono sette cifre prima ancora di cominciare; con una base ortogonale non se ne perde nessuna. Il confronto che a volte si trova — «fit ortogonale di grado 3 contro fit monomiale di grado 12» — non confronta due basi, confronta due **gradi**: il sovradattamento che si vede è colpa del grado 12, e si vedrebbe identico con una base ortogonale.
A filtrare il rumore è la scelta del grado, ed è l'argomento della Sezione 17.7.

17.6.2 Riscalare l'intervallo

Resta un'avvertenza pratica. I polinomi di Legendre vivono su $[-1, 1]$: usarli su dati con ascisse in $[0, 20]$ senza riscalare non solo non aiuta, peggiora le cose. Con la trasformazione affine

$$t = \frac{2x - (x_{\min} + x_{\max})}{x_{\max} - x_{\min}} \in [-1, 1] \tag{17.20}$$

si recupera tutto il guadagno; senza, κ_2 con Legendre su $[0, 20]$ a grado 5 vale $5 \cdot 10^7$ contro $6 \cdot 10^6$ dei monomi — peggio del punto di partenza. La base costruita sui dati non ha questo problema, perché il riscalamento lo fa da sé: β_0 è il baricentro dei nodi e le h_k aggiustano la scala.

```
gradi <- 1:20
kM <- kL <- kD <- numeric(length(gradi))
for (j in seq_along(gradi)) {
  n <- gradi[j]
  kM[j] <- kappa(outer(xg, 0:n, `^`), exact = TRUE)
  kL[j] <- kappa(sapply(0:n, legendre, x = xg), exact = TRUE)
  B <- base_ortogonale(xg, n)
  kD[j] <- kappa(sweep(B$Q, 2, sqrt(B$h), "/"), exact = TRUE)
}
plot(gradi, kM, type = "b", pch = 1, log = "y", lwd = 2, col = 2,
     ylim = range(c(kM, kL, kD)), xlab = "grado n",
     ylab = expression(kappa[2](A)),
     main = "Condizionamento della matrice di progetto")
lines(gradi, kL, type = "b", pch = 16, lwd = 2, col = 4)
lines(gradi, kD, type = "b", pch = 17, lwd = 2, col = 3)
abline(h = 1/.Machine$double.eps, lty = 3, col = "grey40")
legend("topleft", c("monomi", "Legendre", "ortonormale dai dati",
                    "soglia 1/eps"),
       col = c(2, 4, 3, "grey40"), lty = c(1, 1, 1, 3),
       pch = c(1, 16, 17, NA), lwd = 2, bty = "n")
```

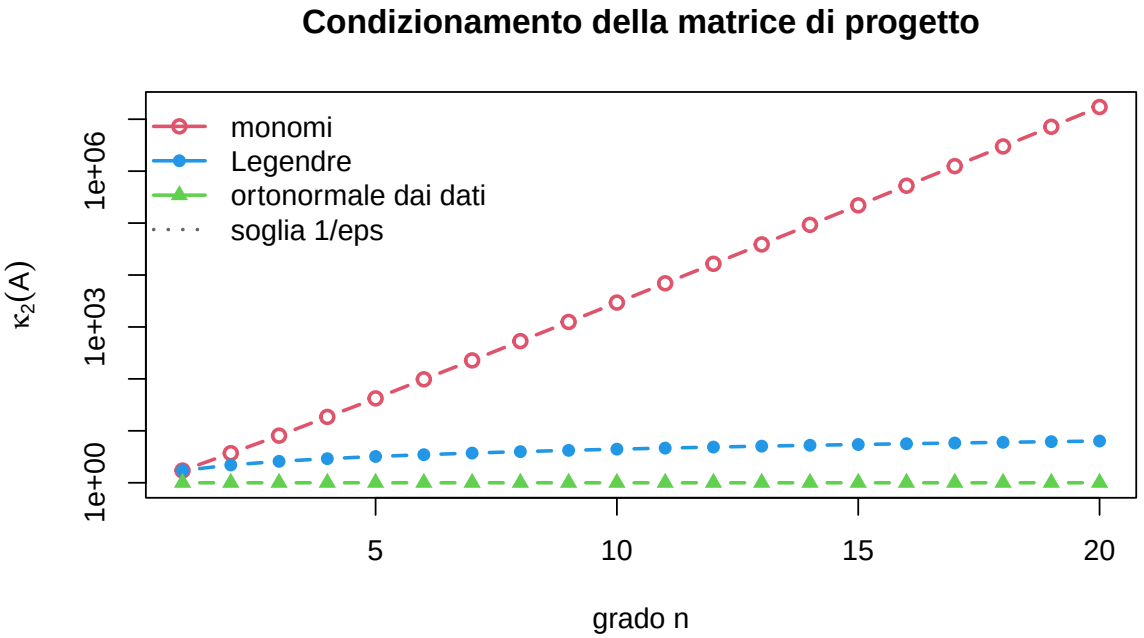


Figure 17.5: Condizionamento della matrice di progetto in funzione del grado, su 120 nodi equispaziati. Scala logaritmica: i monomi crescono esponenzialmente, Legendre quasi per niente, la base ortonormale dai dati resta a 1.

La linea tratteggiata è $1/\varepsilon_{\text{macchina}} \simeq 4{,}5{\cdot}10^{15}$: quando $\kappa_2(\boldsymbol{A})$ la raggiunge, la soluzione calcolata non ha più nessuna cifra significativa. Con i monomi accade attorno al grado 30; con le altre due basi non accade mai.

Risposta alla prima domanda. Sì, conviene, e il guadagno è nel condizionamento, non nell'approssimazione. Se i nodi sono ragionevoli e l'intervallo è riscalato, Legendre basta e ha il vantaggio di essere una base fissa, tabulata, indipendente dai dati. Se si vuole l'ortogonalità *esatta* — e i due regali che ne derivano, i coefficienti annidati e il criterio di arresto della prossima sezione — serve la base costruita sui dati.

17.7 Seconda domanda: lo spettro dei coefficienti e il plateau

Alzando il grado il residuo non può che diminuire: lo spazio $\mathcal{P}_n$ contiene $\mathcal{P}_{n-1}$, quindi il nuovo ottimo è almeno buono quanto il vecchio. Guardare il residuo per scegliere il grado è dunque inutile — porterebbe sempre a $n = m - 1$, cioè all'interpolazione di tutti i punti, rumore compreso.

La base ortonormale ϕ_k di 17.19 rende visibile l'informazione che serve.

17.7.1 I coefficienti come spettro

Con base ortonormale i coefficienti sono semplici prodotti scalari,

$$\widehat{c}_k = \langle y, \phi_k \rangle_d = \sum_{i=1}^{m} \phi_k(x_i)\, y_i, \tag{17.21}$$

e il polinomio di grado n è $p_n = \widehat{c}_0\phi_0 + \cdots + \widehat{c}_n\phi_n$. Per l'identità di Pitagora 17.14 il residuo vale

$$\mathrm{RSS}(n) = \sum_{i=1}^{m} y_i^2 - \sum_{k=0}^{n} \widehat{c}_k^{\,2}, \qquad \text{dunque} \qquad \mathrm{RSS}(n-1) - \mathrm{RSS}(n) = \widehat{c}_n^{\,2}. \tag{17.22}$$

Ogni grado aggiunto riduce il residuo esattamente del quadrato del suo coefficiente. La successione $|\widehat{c}_0|, |\widehat{c}_1|, \ldots$ si chiama **spettro** dei dati: dice quanta parte dei dati vive su ciascun

grado. Si noti che $\widehat{c}_k^2$, essendo una differenza di residui, **non dipende dalla base**: è una quantità intrinseca dei dati e dei nodi. La base ortonormale non la crea, la rende calcolabile gratis.

17.7.2 Perché compare un plateau

Supponiamo ora che i dati siano

$$y_i = f(x_i) + \varepsilon_i,$$

con f il trend vero e ε_i errori di misura indipendenti, di media nulla e deviazione standard σ. Sostituendo in 17.21 e usando la linearità,

$$\widehat{c}_k = \underbrace{\langle f, \phi_k \rangle_d}_{= c_k,\ \text{segnale}} + \underbrace{\sum_{i=1}^{m} \phi_k(x_i)\, \varepsilon_i}_{\text{rumore}}.$$

Il termine di rumore ha media nulla, e la sua varianza si calcola in una riga, perché gli errori sono indipendenti:

$$\operatorname{Var}\Big(\sum_i \phi_k(x_i)\varepsilon_i \Big) = \sigma^2 \sum_{i=1}^{m} \phi_k(x_i)^2 = \sigma^2, \tag{17.23}$$

dove l'ultimo passaggio è esattamente la **normalizzazione** $\|\phi_k\|_d = 1$. Lo stesso conto con due indici diversi dà covarianza nulla, per l'**ortogonalità**. Ne segue il fatto centrale:

> ogni coefficiente stimato è il coefficiente vero più un errore di deviazione standard σ, **la stessa per tutti i gradi**, e gli errori sui diversi coefficienti sono scorrelati.

Il rumore si spalma dunque in modo uniforme su tutto lo spettro, con ampiezza σ, mentre il segnale si concentra sui primi gradi: se f è liscia, i suoi coefficienti c_k decadono rapidamente. Il risultato è lo scenario tipico:

- per k piccolo, $|c_k| \gg \sigma$ e $|\widehat{c}_k|$ misura il segnale;
- da un certo $k^\star$ in poi $|c_k| \ll \sigma$ e $|\widehat{c}_k|$ oscilla attorno a σ, senza più decrescere.

È il **plateau**: lo spettro smette di scendere e si appiattisce sul livello del rumore. Per 17.22 il residuo, da lì in poi, scende di circa σ^2 per ogni grado aggiunto, cioè linearmente:

$$\mathbb{E}\big[\mathrm{RSS}(n)\big] \simeq \underbrace{\sum_i \big[f(x_i) - p_n(x_i)\big]^2}_{\simeq 0 \text{ dopo } k^\star} + (m - n - 1)\,\sigma^2. \tag{17.24}$$

Continuare oltre $k^\star$ significa spendere un grado di libertà per spiegare rumore: il residuo *sui dati* scende, ma la distanza dal trend vero **aumenta**. È il sovradattamento, lo stesso meccanismo del fenomeno di Runge di Capitolo 9 con il rumore al posto della derivata alta.

17.7.3 Il criterio di arresto

17.24 suggerisce anche come stimare σ senza conoscerlo:

$$\widehat{\sigma}^2 = \frac{\mathrm{RSS}(n)}{m - n - 1} \qquad \text{per un } n \text{ nel plateau}, \tag{17.25}$$

oppure, equivalentemente, come radice della media dei $\widehat{c}_k^2$ sui gradi alti. Il criterio diventa allora:

> si alza il grado finché $|\widehat{c}_k| > 2\widehat{\sigma}$; ci si ferma quando **due o tre coefficienti consecutivi** scendono sotto la soglia.

Due precisazioni, entrambe necessarie.

- *Perché consecutivi.* Un coefficiente può essere nullo per ragioni di simmetria: se f è pari, tutti i coefficienti dispari sono nulli, e fermarsi al primo sotto soglia troncherebbe il fit troppo presto.
- *Perché non il massimo.* Con 25 coefficienti nel plateau, per puro caso ci si aspetta che circa uno superi 2σ. Non è segnale: è statistica. Si guarda dove lo spettro **smette di decrescere**, non l'ultimo valore che sporge.

Il criterio è una regola pratica, non un teorema. Resta valido il controllo definitivo, cioè la validazione incrociata: si toglie una parte dei dati, si calcola il fit sul resto e si misura l'errore sui dati esclusi, che — a differenza del residuo — **non** è monotono nel grado e ha un minimo vicino a $k^\star$.

17.7.4 L'esperimento

Generiamo $m = 120$ dati da un trend liscio con rumore di deviazione standard $\sigma = 0{,}15$ e guardiamo lo spettro.

```
set.seed(42)
m     <- 120
xr    <- seq(-1, 1, length.out = m)
trend <- function(t) exp(0.6*t) * sin(pi*t) + 0.3*t
sigma <- 0.15
yr    <- trend(xr) + rnorm(m, sd = sigma)

nmax <- 24
Br   <- base_ortogonale(xr, nmax)
chat <- as.vector(crossprod(Br$Q, yr)) / sqrt(Br$h)    # coefficienti ortonormali
rss  <- sum(yr^2) - cumsum(chat^2)                     # residuo grado per grado
shat <- sqrt(mean(chat[9:(nmax + 1)]^2))               # sigma stimato dal plateau
```

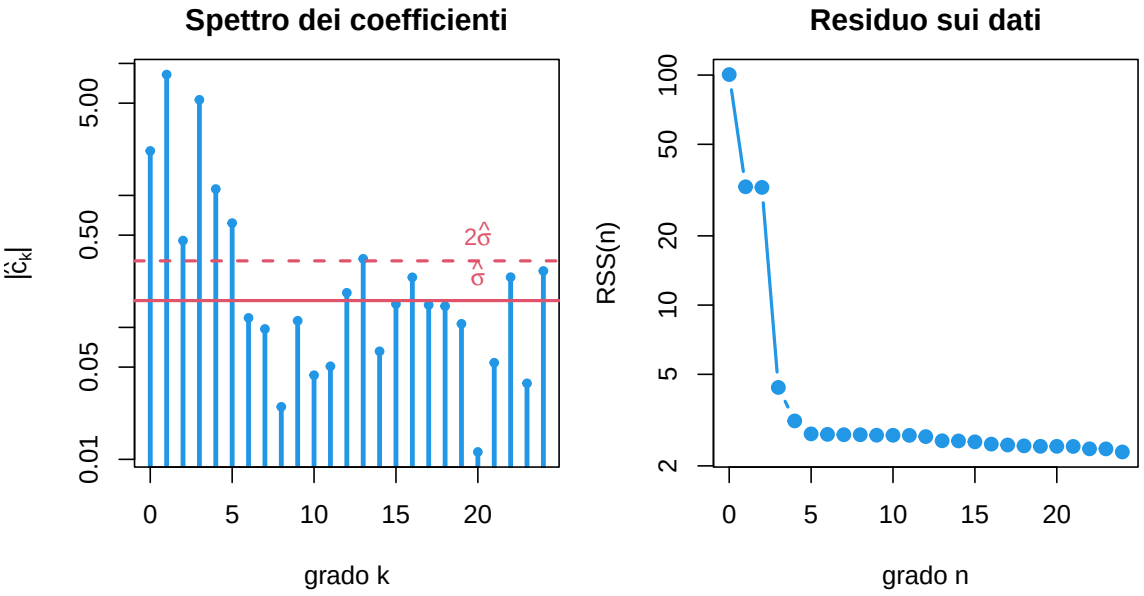


Figure 17.6: A sinistra lo spettro |c_k| dei dati rumorosi, in scala logaritmica: sei coefficienti emergono sopra il livello del rumore, poi comincia il plateau. A destra il residuo, che continua a scendere e non segnala nulla.

Lo spettro parla da solo: i coefficienti di grado 0–5 valgono 2,2, 8,2, 0,45, 5,3, 1,1, 0,62, tutti ben sopra la soglia $2\widehat{\sigma} \simeq 0{,}32$; dal grado 6 in poi non si scende più sotto e non si risale sopra in modo sistematico. Il valore stimato $\widehat{\sigma} = 0{,}16$ è vicino al $\sigma = 0{,}15$ usato per generare i dati, che

però in un caso reale non si conosce.

Il pannello di destra mostra perché il residuo non serve: scende in modo monotono e non ha ginocchi riconoscibili dopo il grado 5.

La tabella seguente chiude il discorso confrontando, grado per grado, il residuo **sui dati** con l'errore rispetto al **trend vero** — che nella realtà non si può misurare, ma qui sì, perché il trend lo abbiamo scelto noi.

```
valuta_fino <- function(t, n) {
  d <- as.vector(crossprod(Br$Q[, 1:(n+1), drop = FALSE], yr)) / Br$h[1:(n+1)]
  valuta(t, list(d = d, beta = Br$beta, gamma = Br$gamma))
}
tg <- seq(-1, 1, length.out = 801)
gr <- c(1, 3, 5, 6, 8, 12, 16, 24)
knitr::kable(data.frame(
  n              = gr,
  c_hat_n        = abs(chat[gr + 1]),
  sopra_soglia   = ifelse(abs(chat[gr + 1]) > 2*shat, "si", "no"),
  RSS_dati       = rss[gr + 1],
  errore_vs_trend = sapply(gr, function(n) max(abs(valuta_fino(tg, n) - trend(tg)))),
  digits = 4)
```

Table 17.11: Il residuo sui dati scende sempre; l'errore rispetto al trend vero ha un minimo al grado 5 e poi peggiora. Il criterio sullo spettro individua il grado giusto senza conoscere il trend.

n	c_hat_n	sopra_soglia	RSS_dati	errore_vs_trend
1	8.2361	si	32.6847	1.1897
3	5.3003	si	4.3852	0.5151
5	0.6172	si	2.7552	0.0833
6	0.1180	no	2.7413	0.1011
8	0.0250	no	2.7312	0.0877
12	0.1829	no	2.6807	0.1046
16	0.2395	no	2.4869	0.1697
24	0.2675	no	2.2990	0.2117

Il minimo dell'errore rispetto al trend cade al grado 5, esattamente dove il criterio dello spettro dice di fermarsi. Dal grado 12 in poi l'errore raddoppia mentre il residuo continua tranquillamente a scendere: il fit sta imparando il rumore.

```
plot(xr, yr, pch = 19, cex = 0.55, col = "grey45", xlab = "x", ylab = "y",
     main = "Grado scelto dallo spettro contro grado eccessivo")
lines(tg, trend(tg), lwd = 3, col = 1, lty = 2)
lines(tg, valuta_fino(tg, 5),  lwd = 2.5, col = 4)
lines(tg, valuta_fino(tg, 24), lwd = 2,   col = 2)
legend("bottom", c("dati", "trend vero", "grado 5", "grado 24"),
       pch = c(19, NA, NA, NA), lty = c(NA, 2, 1, 1),
       lwd = c(NA, 3, 2.5, 2), col = c("grey45", 1, 4, 2),
       bty = "n", cex = 0.9, ncol = 2)
```

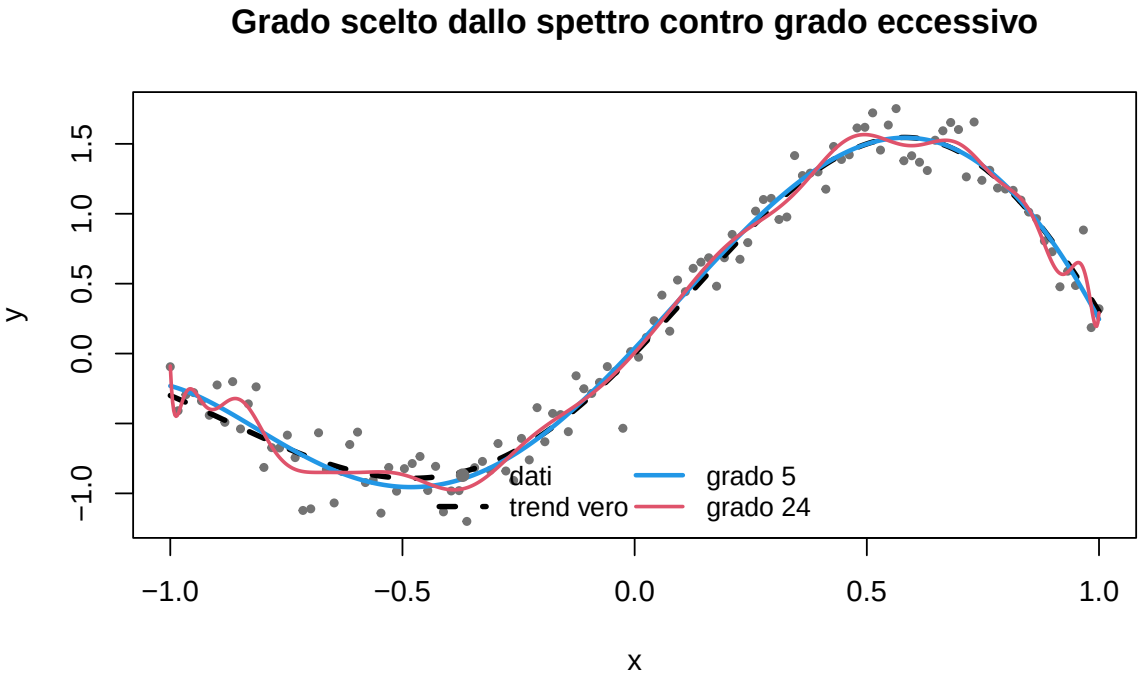


Figure 17.7: Lo stesso set di dati con due gradi diversi. Il grado 5, scelto dallo spettro, segue il trend; il grado 24 insegue il rumore, soprattutto agli estremi.

Risposta alla seconda domanda. Il plateau esiste, ha una causa precisa — il rumore si distribuisce uniformemente sullo spettro con ampiezza σ, il segnale no — e si legge nei coefficienti, non nel residuo. È uno dei motivi per cui la base ortonormale **sui dati** è preferibile a Legendre: la 17.23, da cui tutto discende, vale esattamente solo se i ϕ_k sono ortonormali rispetto al prodotto scalare discreto. Con Legendre su nodi qualunque i coefficienti stimati sono leggermente correlati e lo spettro è un po' più confuso, pur restando leggibile.

17.8 Terza domanda: il fitting incrementale

Mettiamo insieme i pezzi. La ricorrenza 17.11 produce i q_k **uno alla volta**, ciascuno da due predecessori; i coefficienti sono indipendenti fra loro (17.9) e non vanno ricalcolati quando se ne aggiunge uno nuovo; il residuo si aggiorna per sottrazione (17.22) e lo spettro dice quando fermarsi (Sezione 17.7). Ne esce un algoritmo che costruisce il fit per gradi crescenti e si arresta da solo.

17.8.1 L'algoritmo

La Figura 17.8 lo riassume. Rispetto alla Figura 17.1 cambiano tre cose: i coefficienti sono normalizzati ($\widehat{c}_k$ anziché d_k, cioè divisione per $\sqrt{h_k}$ anziché per h_k), il residuo viene aggiornato per sottrazione a ogni grado, e un test di arresto interrompe il ciclo appena lo spettro entra nel plateau.

La riga che fa il lavoro concettuale è l'aggiornamento $R \leftarrow R - \widehat{c}^2_{k+1}$: è la 17.22, cioè l'identità di Pitagora applicata un grado alla volta. Da essa discende anche la stima del rumore, perché per 17.24 il residuo nel plateau vale circa $(m - k - 2)\,\sigma^2$.

Tre proprietà rendono l'algoritmo adatto anche a dataset molto grandi.

- **Costo.** Ogni grado richiede un numero fisso di passate sui dati, quindi $O(m)$ operazioni per grado e $O(mn)$ in totale — contro gli $O(mn^2)$ di una QR rifatta da capo per ogni grado.
- **Memoria.** Servono solo due vettori di lunghezza m (q_k e q_{k-1} nei nodi) più gli scalari. Non si memorizza nessuna matrice $m \times (n+1)$.

Fit polinomiale incrementale, con arresto sul plateau

Dati: nodi $x_1, \dots, x_m$; dati $y_1, \dots, y_m$; pesi $w_i > 0$;
grado massimo $n_{\max}$; soglia τ (tipicamente $\tau = 2$)
Restituisce: grado n; coefficienti ortonormali $\widehat{c}_0, \dots, \widehat{c}_n$;
coefficienti della ricorrenza β_k, γ_k; stima del rumore $\widehat{\sigma}$

$q^- \leftarrow (0, \dots, 0), \quad q \leftarrow (1, \dots, 1)$ ▷ q_{k-1} e q_k nei nodi
$h^- \leftarrow 1, \quad h \leftarrow \sum_i w_i$ ▷ $h = \|q_k\|_d^2$
$\widehat{c}_0 \leftarrow \big(\sum_i w_i y_i\big)/\sqrt{h}$
$R \leftarrow \sum_i w_i y_i^2 - \widehat{c}_0^2$ ▷ residuo
$\widehat{\sigma} \leftarrow \sqrt{R/(m-2)}$
per $k = 0, 1, \dots, n_{\max} - 1$ **fai**
  $\beta_k \leftarrow \dfrac{1}{h} \sum_i w_i \, x_i \, q_i^2$
  $\gamma_k \leftarrow h/h^- \quad (\text{con } \gamma_0 = 0)$
  $q_i^+ \leftarrow (x_i - \beta_k)\, q_i - \gamma_k \, q_i^-, \quad i = 1, \dots, m$ ▷ ricorrenza a tre termini
  $q^- \leftarrow q, \quad q \leftarrow q^+, \quad h^- \leftarrow h, \quad h \leftarrow \sum_i w_i \, q_i^2$
  se $h \le 0$ **allora**
    esci dal ciclo ▷ grado oltre $m - 1$
  $\widehat{c}_{k+1} \leftarrow \big(\sum_i w_i \, y_i \, q_i\big)/\sqrt{h}$
  $R \leftarrow R - \widehat{c}_{k+1}^2$ ▷ il residuo cala del quadrato del coefficiente
  $\widehat{\sigma} \leftarrow \sqrt{R/(m-k-3)}$
  se $k \ge 3$ **e** $|\widehat{c}_j| < \tau\widehat{\sigma}$ per $j = k-1, k, k+1$ **allora**
    esci dal ciclo ▷ plateau raggiunto
$n \leftarrow \max\{\, k : \ |\widehat{c}_k| \ge \tau\widehat{\sigma} \,\}$ ▷ ultimo grado sopra la soglia
restituisci $n, \ (\widehat{c}_k)_{k \le n}, \ (\beta_k, \gamma_k)_{k < n}, \ \widehat{\sigma}$

Figure 17.8: L'algoritmo incrementale: costruisce la base un grado alla volta, aggiorna residuo e stima del rumore strada facendo, e si ferma da solo quando tre coefficienti consecutivi scendono sotto la soglia.

- **Niente lavoro buttato.** Fermandosi al grado n si è speso esattamente quanto serviva per il grado n: non c'è un $n_{\max}$ da indovinare in anticipo e nessun calcolo viene scartato.

17.8.2 L'implementazione in R

```
fit_incrementale <- function(x, y, nmax = 30, tau = 2,
                             w = rep(1, length(x))) {
  m  <- length(x)
  qp <- rep(0, m); q <- rep(1, m)          # q_{k-1} e q_k valutati nei nodi
  hp <- 1;         h <- sum(w)             # norme al quadrato
  ch <- hk <- rss <- sig <- beta <- gamma <- numeric(nmax + 1)

  hk[1]  <- h
  ch[1]  <- sum(w * y) / sqrt(h)           # primo coefficiente
  rss[1] <- sum(w * y^2) - ch[1]^2
  sig[1] <- sqrt(max(rss[1], 0) / max(m - 2, 1))

  K <- 0
  for (k in seq_len(nmax) - 1) {
    beta[k + 1]  <- sum(w * x * q^2) / h
    gamma[k + 1] <- if (k == 0) 0 else h / hp
    qn <- (x - beta[k + 1]) * q - gamma[k + 1] * qp   # nuovo grado
    qp <- q; q <- qn; hp <- h; h <- sum(w * q^2)
    if (h <= 0) break                      # grado oltre il consentito
    hk[k + 2]  <- h
    ch[k + 2]  <- sum(w * y * q) / sqrt(h)
    rss[k + 2] <- rss[k + 1] - ch[k + 2]^2            # il residuo cala
    sig[k + 2] <- sqrt(max(rss[k + 2], 0) / max(m - k - 3, 1))
    K <- k + 1
    if (K >= 3 && all(abs(ch[(K - 1):(K + 1)]) < tau * sig[K + 1])) break
  }

  i     <- 0:K
  sopra <- which(abs(ch[i + 1]) >= tau * sig[K + 1])
  g     <- if (length(sopra)) max(sopra) - 1 else 0
  list(storia = data.frame(k = i, c_hat = ch[i + 1],
                           RSS = rss[i + 1], sigma = sig[i + 1]),
       sigma  = sig[K + 1],
       grado  = g,
       d      = (ch[i + 1] / sqrt(hk[i + 1]))[1:(g + 1)],  # coeff. monici
       beta   = beta, gamma = gamma)
}
```

Il campo `storia` registra, grado per grado, coefficiente, residuo e stima del rumore: è la tabella da guardare per capire che cosa ha fatto l'algoritmo.

17.8.3 L'algoritmo sui dati rumorosi

```
inc <- fit_incrementale(xr, yr, nmax = 24, tau = 2)
st  <- inc$storia
st$sopra_soglia <- ifelse(abs(st$c_hat) >= 2 * inc$sigma, "si", "no")
knitr::kable(st, digits = 4, row.names = FALSE)
```

Table 17.12: Storia dell'algoritmo incrementale sui 120 dati rumorosi: si ferma da solo dopo aver visto tre coefficienti consecutivi sotto la soglia.

k	c_hat	RSS	sigma	sopra_soglia
0	2.1722	100.5181	0.9230	si

Table 17.12: Storia dell'algoritmo incrementale sui 120 dati rumorosi: si ferma da solo dopo aver visto tre coefficienti consecutivi sotto la soglia.

k	c_hat	RSS	sigma	sopra_soglia
1	8.2361	32.6847	0.5285	si
2	0.4546	32.4780	0.5291	si
3	-5.3003	4.3852	0.1953	si
4	-1.1176	3.1362	0.1659	si
5	0.6172	2.7552	0.1561	si
6	0.1180	2.7413	0.1564	no
7	0.0972	2.7319	0.1569	no
8	0.0250	2.7312	0.1576	no

L'algoritmo si ferma dopo aver calcolato pochi gradi oltre quello utile, e restituisce `grado` $= 5$, con una stima del rumore $\widehat{\sigma} = 0.1576$ contro il valore vero 0,15. Non ha mai costruito una matrice, non ha mai risolto un sistema, e il costo è stato di una manciata di passate sui 120 dati.

> **Il legame con QR**
>
> Vale la pena vedere la stessa cosa con gli occhi dell'algebra lineare. Costruire la base ortonormale $\phi_0, \dots, \phi_n$ sui nodi **è** calcolare il fattore $\boldsymbol{Q}$ della fattorizzazione $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$ della matrice di Vandermonde: le colonne di $\boldsymbol{Q}$ sono proprio i $\phi_k(x_i)$, e $\boldsymbol{R}$ contiene i coefficienti che esprimono i monomi nella base ortogonale.
> La differenza è che la QR classica ortogonalizza le colonne *numericamente*, con riflessioni di Householder, costa $O(mn^2)$ e va rifatta da capo se cambia il grado; la ricorrenza a tre termini le ortogonalizza *analiticamente*, costa $O(mn)$ ed è incrementale. Il prezzo è che la ricorrenza funziona solo per il caso polinomiale: la QR funziona per qualunque matrice.

Risposta alla terza domanda. Sì, ed è il modo naturale di procedere: la ricorrenza aggiunge un grado alla volta, il coefficiente nuovo non disturba i vecchi, il residuo si aggiorna per sottrazione e lo spettro dice quando smettere. Il lavoro complessivo è lo stesso che servirebbe per calcolare il solo fit del grado finale.

17.9 Il problema continuo

17.9.1 La proiezione in norma pesata

Per approssimare f in $L^2_w(-1, 1)$, con base ortogonale ϕ_k, la minimizzazione

$$\min_{c_0,\dots,c_n} \int_{-1}^{1} \left| f(x) - \sum_{k=0}^{n} c_k \phi_k(x) \right|^2 w(x)\, \mathrm{d}x$$

si disaccoppia: $c_k = \langle f, \phi_k \rangle_w$ nella base ortonormale. Con base ortogonale non normalizzata, $c_k = \langle f, p_k \rangle_w / \|p_k\|^2_w$. La proiezione è il minimizzatore unico, come dimostrato in Sezione A.4; la Figura 17.9 ne riassume il contenuto geometrico.

17.9.2 Esempio: proiezione di e^x e dipendenza dal peso

Approssimiamo e^x con Legendre ortonormali. I coefficienti integrali sono stimati con una regola del punto medio fine; il grafico mostra la funzione, le proiezioni per tre gradi e l'errore puntuale. Non si tratta di un bound teorico: riportiamo anche l'errore massimo misurato su una griglia indipendente.

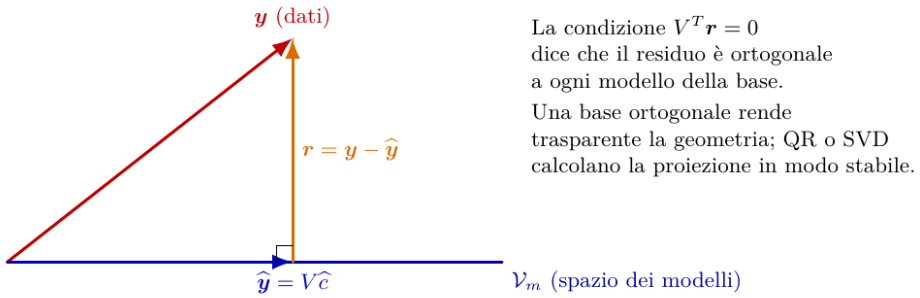


Figure 17.9: La proiezione è il punto più vicino a f nel prodotto scalare pesato; il residuo è ortogonale allo spazio approssimante.

```
m<-20000; xq<- -1+(seq_len(m)-0.5)*2/m; wq<-rep(2/m,m)
xx<-seq(-1,1,length.out=1001); truth<-exp(xx)
degrees<-c(2,4,8); cols<-c(2,4,3)
plot(xx,truth,type="l",lwd=3,col=1,xlab="x",ylab="f(x)",main="Proiezione ortogonale di
    ↪ exp(x)")
errs<-numeric(length(degrees))
for(j in seq_along(degrees)){
  d<-degrees[j]; Pq<-sapply(0:d,legendre,x=xq)
  phiQ<-sweep(Pq,2,sqrt((2*(0:d)+1)/2),`*`)
  coef<-colSums(sweep(phiQ,1,wq*exp(xq),`*`))
  Px<-sapply(0:d,legendre,x=xx)
  phiX<-sweep(Px,2,sqrt((2*(0:d)+1)/2),`*`)
  approx<-drop(phiX%*%coef); lines(xx,approx,col=cols[j],lwd=2)
  errs[j]<-max(abs(truth-approx))
}
legend("topleft",c("exp(x)",paste0("grado
    ↪ ",degrees)),col=c(1,cols),lwd=c(3,2,2,2),bty="n")
```

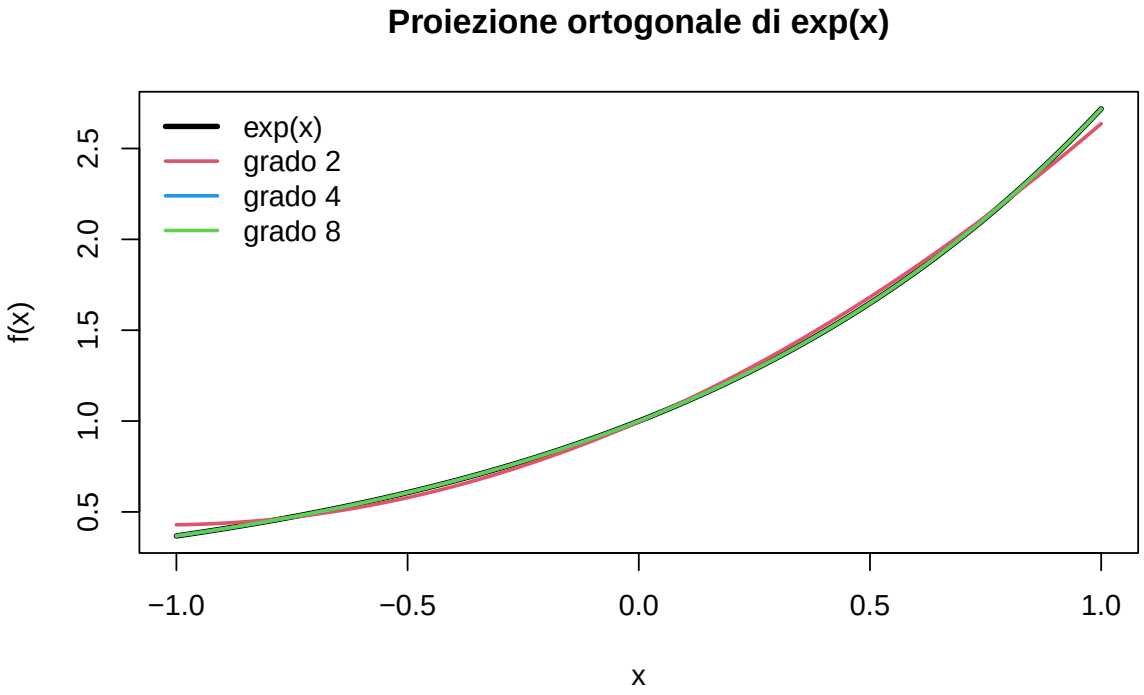


Figure 17.10: Proiezioni di eˆx su Legendre ortonormali per tre gradi e relativo errore puntuale.

Table 17.13: Errore massimo misurato su una griglia indipendente per le tre proiezioni.

grado	errore_massimo_griglia
2	0.0816280
4	0.0012073
8	0.0000010

Per un peso diverso, cambiano sia i coefficienti sia la funzione obiettivo. L'errore massimo non coincide con l'errore L^2_w; il primo è utile per visualizzare, il secondo è quello minimizzato dalla proiezione.

17.9.3 Esempio: cambiare il peso cambia la proiezione

Il prossimo esempio risolve il problema continuo pesato con quadratura del punto medio. Usiamo la stessa base polinomiale, ma due prodotti scalari: uniforme e Jacobi con $\alpha = 0.6$, $\beta = 0.2$. La minimizzazione discreta approssima l'integrale; aumentando `m` si raffina la quadratura.

```
m<-10000; xq<- -1+(seq_len(m)-0.5)*2/m; dx<-2/m
degree<-4; A<-sapply(0:degree,legendre,x=xq); y<-exp(xq)
w_uniform<-rep(dx,m)
w_jacobi<-dx*(1-xq)^0.6*(1+xq)^0.2
fit_weighted<-function(w){Aw<-sweep(A,1,sqrt(w),`*`)
  yw<-sqrt(w)*y; cc<-qr.solve(Aw,yw,tol=1e-12); list(c=cc,rss=sum(w*(A%*%cc-y)^2))}
u<-fit_weighted(w_uniform); j<-fit_weighted(w_jacobi)
xx<-seq(-1,1,length.out=1001); B<-sapply(0:degree,legendre,x=xx)
pu<-drop(B%*%u$c); pj<-drop(B%*%j$c)
plot(xx,exp(xx),type="l",lwd=3,col=1,xlab="x",ylab="funzione",
     main="Proiezione pesata: stesso grado, pesi diversi")
lines(xx,pu,col=4,lwd=2);lines(xx,pj,col=2,lwd=2,lty=2)
legend("topleft",c("exp(x)","peso uniforme","peso Jacobi (0.6,0.2)"),
       col=c(1,4,2),lty=c(1,1,2),lwd=c(3,2,2),bty="n")
```

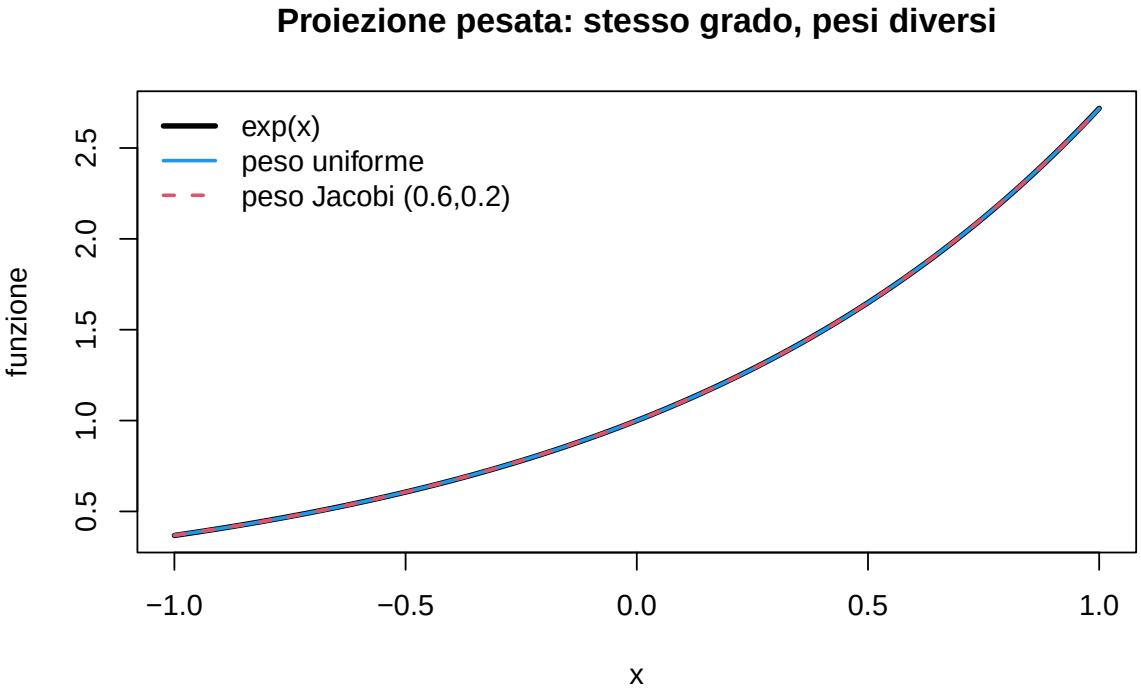


Figure 17.11: Proiezioni di grado 4 rispetto a due pesi: minimizzano errori integrali diversi e quindi non coincidono.

Table 17.14: Errore quadratico ciascuno rispetto al proprio peso: i due valori non misurano la stessa quantità.

peso	errore_L2_pesato_discreto
uniforme	2e-07
Jacobi alpha=0.6, beta=0.2	1e-07

Si tratta di errori quadratici rispetto ai rispettivi pesi, quindi i valori delle due righe non misurano la stessa quantità. Per un confronto equo fra approssimazioni, si possono valutare entrambe in una terza norma comune.

In sintesi

Le tre domande poste all'inizio hanno ora una risposta.

La base. Cambiare base non cambia lo spazio approssimante, non cambia il polinomio ottimo e non cambia il residuo: cambia il condizionamento della matrice di progetto, e quindi l'accuratezza con cui quel polinomio viene calcolato. Il guadagno è grande — sei ordini di grandezza a grado 20 — ma è tutto lì. In particolare una base ortogonale **non** filtra il rumore e **non** riduce il sovradattamento.

- I monomi sono una base pessima: κ_2 cresce esponenzialmente nel grado (14.5), e nessuna scelta del peso lo evita.
- I polinomi di Legendre sono una base fissa ed eccellente, purché i dati siano riscalati su $[-1, 1]$ con 17.20. Danno una $\boldsymbol{A}^T A$ *quasi* diagonale, perché la somma sui nodi è una quadratura dell'integrale rispetto al quale sono ortogonali.
- La base ϕ_k costruita **sui dati** con la ricorrenza 17.11 dà ortogonalità esatta, $\boldsymbol{A}^T A$ diagonale e $\kappa_2 = 1$ — a patto di normalizzare, perché i q_k monici hanno norme di ordini di grandezza diversi.

Il grado. Con base ortonormale ogni coefficiente stimato è il coefficiente vero più un errore di deviazione standard σ, uguale per tutti i gradi e scorrelato fra gradi diversi (17.23). Il rumore si spalma quindi uniformemente sullo spettro, il segnale no: da un certo grado in poi $|\widehat{c}_k|$ smette di decrescere e si appiattisce su σ. È il plateau, e dà il criterio di arresto: fermarsi quando due o tre coefficienti consecutivi scendono sotto $2\widehat{\sigma}$. Il residuo, che decresce sempre, non fornisce invece nessuna indicazione.

L'incrementalità. La ricorrenza produce un grado alla volta, i coefficienti sono indipendenti, il residuo si aggiorna per sottrazione (17.22) e il criterio di arresto si valuta strada facendo. Il costo è $O(m)$ per grado e la memoria $O(m)$ in tutto, contro $O(mn^2)$ e $O(mn)$ di una QR rifatta per ogni grado. La ricorrenza a tre termini è, in sostanza, la fattorizzazione QR della Vandermonde eseguita analiticamente invece che numericamente.

Il peso. Nel problema continuo il peso decide *quale* errore viene minimizzato: proiezioni rispetto a pesi diversi sono approssimazioni diverse e non vanno confrontate con il residuo di una sola delle due norme. Nel problema discreto i pesi $w_i = 1/\sigma_i^2$ sono il modo corretto di combinare misure di precisione diversa.

Un parente stretto. La variante *vincolata* dello stesso problema — minimizzare la norma fra i polinomi che valgono 1 in un punto assegnato — ha soluzione in forma chiusa tramite il polinomio del nucleo (Teorema 16.3), e tornerà nei pesi della quadratura gaussiana.

> **Consiglio**
>
> Gli integrali $\langle f, \phi_k \rangle_w$ del problema continuo sono qui approssimati con una regola del punto medio molto fine, per non anticipare risultati non ancora dimostrati. In Capitolo 18 si vedrà che poche decine di nodi di Gauss bastano a ottenere gli stessi coefficienti a precisione macchina.

17.10 Esercizi

1. Generare $m = 200$ dati da $f(x) = \cos(4x)$ su $[-1, 1]$ con rumore $\sigma = 0{,}05$ e applicare `fit_incrementale`. Verificare che il grado scelto cresce al diminuire di σ e spiegare perché: quali coefficienti emergono sopra il livello del rumore?
2. Ripetere l'esperimento della Tabella 17.10 con nodi **non** equispaziati, per esempio $x_i = \cos\big((2i-1)\pi/(2m)\big)$ (nodi di Chebyshev). Che cosa succede agli elementi fuori diagonale della base di Legendre, e perché?
3. Verificare numericamente 17.23: fissati nodi e grado, generare mille realizzazioni di solo rumore ($y_i = \varepsilon_i$) e controllare che la deviazione standard campionaria di ciascun $\widehat{c}_k$ valga σ e che le correlazioni fra coefficienti diversi siano trascurabili.
4. Implementare la validazione incrociata *leave-one-out* sul fit polinomiale e confrontare il grado da essa scelto con quello del criterio dello spettro, sugli stessi dati della
    1.
5. Modificare `fit_incrementale` per accettare pesi $w_i = 1/\sigma_i^2$ e verificare, su dati con incertezze diverse, che il plateau si forma al livello 1 anziché σ (gli scarti sono ora misurati in unità di deviazione standard).
6. Costruire un fit pesato discreto con $w_i = 1 + x_i$ e risolverlo anche con `qr.solve` su $\min \|W^{1/2}(Ac - y)\|_2$; confrontare i coefficienti ottenuti per le due strade.
7. Usare la SVD della matrice di progetto nella base monomiale e scartare i valori singolari minori di $10^{-10}\sigma_{\max}$; spiegare il legame fra questo troncamento e il criterio dello spettro.



18. Quadrature di Gauss e zeri dei polinomi

18.1 Esattezza di grado $2n-1$

18.1.1 Dall'ortogonalità all'esattezza

Sia $w > 0$ su $[a,b]$, e siano $x_1, \dots, x_n$ gli zeri del polinomio ortogonale p_n: per Teorema 16.2 sono reali, semplici e interni a (a,b), dunque utilizzabili come nodi. La formula di quadratura gaussiana[1] è quella interpolatoria su tali nodi,

$$Q_n(f) = \sum_{i=1}^{n} \omega_i f(x_i), \qquad \omega_i = \int_a^b \ell_i(x)\, w(x)\, \mathrm{d}x, \tag{18.1}$$

dove ℓ_i è l'i-esimo polinomio cardinale di Lagrange[2] sui nodi $x_1, \dots, x_n$. Definendo i pesi come in 18.1, la formula integra per costruzione esattamente ogni polinomio di $\mathcal{P}_{n-1}$: è quanto fa una qualsiasi formula interpolatoria, come quelle di Newton-Cotes di Capitolo 10. La proprietà fondamentale è che questa particolare scelta dei nodi consente di raggiungere un grado di esattezza quasi doppio.

Theorem 18.1 — Esattezza delle formule gaussiane. Siano $x_1, \dots, x_n$ gli zeri del polinomio ortogonale p_n e siano $\omega_1, \dots, \omega_n$ i pesi definiti da 18.1. Allora la formula

$$Q_n(f) = \sum_{i=1}^{n} \omega_i\, f(x_i) \tag{18.2}$$

soddisfa

$$Q_n(r) = \int_a^b r(x)\, w(x)\, \mathrm{d}x \qquad \text{per ogni } r \in \mathcal{P}_{2n-1},$$

e questo grado di esattezza è massimo: **nessuna** formula a n nodi, comunque se ne scelgano nodi e pesi, può essere esatta su tutto $\mathcal{P}_{2n}$.

Proof. **Passo 1: dividere per p_n.** Sia r un polinomio di grado al più $2n-1$. La divisione euclidea fra polinomi — la stessa che si fa fra numeri interi — fornisce quoziente q e resto s con

$$r = q\, p_n + s, \qquad \deg s < \deg p_n = n. \tag{18.3}$$

[1] Carl Friedrich Gauss (1777-1855)
[2] Joseph-Louis Lagrange (1736-1813)

Il grado del quoziente si conta sottraendo: $\deg q = \deg r - n \le n - 1$. Dunque entrambi, quoziente e resto, stanno in $\mathcal{P}_{n-1}$: è questo il punto in cui si usa l'ipotesi $\deg r \le 2n-1$, ed è il motivo per cui la soglia è proprio $2n-1$.

Passo 2: l'integrale esatto non vede il quoziente. Integrando 18.3 contro il peso,

$$\int_a^b r\,w = \underbrace{\int_a^b q\,p_n\,w}_{=0} + \int_a^b s\,w = \int_a^b s\,w.$$

Il primo integrale è nullo perché è $\langle p_n, q\rangle_w$ con $\deg q \le n-1 < n$: è esattamente l'ortogonalità di p_n ai polinomi di grado inferiore.

Passo 3: nemmeno la formula vede il quoziente. Qui entra la scelta dei nodi. Valutando 18.3 in un nodo x_i e usando $p_n(x_i) = 0$,

$$r(x_i) = q(x_i)\underbrace{p_n(x_i)}_{=0} + s(x_i) = s(x_i),$$

cioè r e s hanno gli **stessi valori nei nodi**. Poiché Q_n usa solo quei valori,

$$Q_n(r) = \sum_{i=1}^{n} \omega_i\, r(x_i) = \sum_{i=1}^{n} \omega_i\, s(x_i) = Q_n(s).$$

Passo 4: sul resto la formula è esatta per costruzione. Il resto s ha grado al più $n-1$, e una formula interpolatoria a n nodi con i pesi 18.1 integra esattamente tutto $\mathcal{P}_{n-1}$. Quindi $Q_n(s) = \int_a^b s\,w$.

Conclusione. Incatenando i tre passi,

$$Q_n(r) \overset{\text{Passo 3}}{=} Q_n(s) \overset{\text{Passo 4}}{=} \int_a^b s\,w \overset{\text{Passo 2}}{=} \int_a^b r\,w.$$

Passo 5: $2n-1$ non si può superare. L'affermazione di massimalità riguarda una formula a n nodi qualunque, non solo quella gaussiana. Siano dunque $y_1, \dots, y_n$ nodi arbitrari e $\nu_1, \dots, \nu_n$ pesi arbitrari, e consideriamo

$$\Omega(x) = \big[(x-y_1)(x-y_2)\cdots(x-y_n)\big]^2,$$

polinomio di grado esattamente $2n$. Da un lato $\Omega \ge 0$ e non è identicamente nullo, quindi $\int_a^b \Omega\,w > 0$. Dall'altro $\Omega(y_i) = 0$ per ogni i, quindi la formula restituisce

$$\sum_{i=1}^{n} \nu_i\,\Omega(y_i) = 0 \;\neq\; \int_a^b \Omega\,w.$$

Nessuna formula a n nodi è dunque esatta su $\mathcal{P}_{2n}$. Nel caso gaussiano il controesempio si scrive in modo ancora più suggestivo: $\Omega = p_n^2$, a meno del fattore di normalizzazione. ∎

18.1.2 Positività dei pesi

Proposition 18.1 — Positività dei pesi. I pesi $\omega_1, \dots, \omega_n$ di una formula gaussiana sono tutti strettamente positivi.

Proof. Fissiamo un indice i e consideriamo il polinomio cardinale di Lagrange ℓ_i, caratterizzato da

$$\deg \ell_i = n-1, \qquad \ell_i(x_j) = \begin{cases} 1 & j = i, \\ 0 & j \neq i. \end{cases} \tag{18.4}$$

Il suo quadrato ℓ_i^2 ha grado $2n-2$, che è minore o uguale a $2n-1$: per Teorema 18.1 la formula lo integra **esattamente**. Calcoliamo allora lo stesso numero in due modi.

Con la formula, usando 18.4, sopravvive un solo addendo:

$$Q_n(\ell_i^2) = \sum_{j=1}^{n} \omega_j \, \ell_i(x_j)^2 = \omega_i \cdot 1^2 = \omega_i.$$

Con l'integrale, invece, si tratta dell'integrale pesato di un quadrato:

$$\int_a^b \ell_i(x)^2 \, w(x)\,\mathrm{d}x \;>\; 0,$$

strettamente positivo perché $\ell_i^2 \geq 0$, perché ℓ_i^2 non è identicamente nullo (vale 1 in x_i) e perché $w > 0$.

Essendo i due numeri uguali, $\omega_i > 0$. L'indice i era arbitrario. ∎

La positività non è un dettaglio estetico. Poiché

$$\sum_{i=1}^{n} |\omega_i| = \sum_{i=1}^{n} \omega_i = \int_a^b w$$

indipendentemente da n, la formula non amplifica gli errori sui valori di f per quanto si alzi l'ordine. Le formule di Newton-Cotes di Capitolo 10 perdono questa proprietà già per n dell'ordine di otto: è la ragione tecnica per cui non si usano ad alto ordine, mentre le gaussiane sì.

18.2 Gauss–Legendre

18.2.1 Nodi e pesi

Per $w = 1$ su $[-1, 1]$ si ottengono le formule di Gauss–Legendre[3]: i nodi sono gli zeri di P_n e i pesi si possono scrivere

$$\omega_i = \frac{2}{(1 - x_i^2)[P_n'(x_i)]^2}.$$

Entrambe le formule sono casi particolari di un fatto generale, che chiude il discorso aperto in Sezione 16.3.

18.2.2 I pesi sono la funzione di Christoffel

Proposition 18.2 — I pesi sono la funzione di Christoffel. Qualunque sia il peso w, i pesi della formula gaussiana a n nodi valgono

$$\omega_i = \lambda_{n-1}(x_i) = \frac{1}{K_{n-1}(x_i, x_i)} = \frac{1}{\sum_{k=0}^{n-1} \phi_k(x_i)^2}, \qquad i = 1, \dots, n, \tag{18.5}$$

dove λ_{n-1} è la funzione di Christoffel 16.17 e i ϕ_k sono i polinomi ortonormali.

Proof. L'idea è calcolare **due volte**, in due modi diversi, la stessa quantità: il minimo di $\|p\|_w^2$ fra i polinomi $p \in \mathcal{P}_{n-1}$ vincolati da $p(x_i) = 1$. Fissiamo dunque l'indice i e chiamiamo

$$m_i = \min \left\{ \|p\|_w^2 \;:\; p \in \mathcal{P}_{n-1},\; p(x_i) = 1 \right\}.$$

Primo calcolo: $m_i = \omega_i$. Sia $p \in \mathcal{P}_{n-1}$ con $p(x_i) = 1$. Il quadrato p^2 ha grado al più $2n-2 \leq 2n-1$, dunque per Teorema 18.1 la formula gaussiana lo integra esattamente:

[3] Adrien-Marie Legendre (1752-1833)

$$\|p\|_w^2 = \int_a^b p^2\, w = Q_n(p^2) = \sum_{j=1}^{n} \omega_j\, p(x_j)^2. \tag{18.6}$$

Tutti gli addendi della somma sono ≥ 0, perché i pesi sono positivi per Proposizione 18.1 e i $p(x_j)^2$ sono quadrati. Trattenendo il solo addendo $j = i$ e usando il vincolo $p(x_i) = 1$,

$$\|p\|_w^2 \;\geq\; \omega_i\, p(x_i)^2 = \omega_i.$$

Vale quindi $m_i \geq \omega_i$. Per vedere che il valore ω_i è effettivamente raggiunto basta scegliere $p = \ell_i$, il polinomio cardinale 18.4: è di grado $n-1$, soddisfa $\ell_i(x_i) = 1$ e, per 18.6, ha norma al quadrato pari a ω_i. Dunque

$$m_i = \omega_i,$$

e il minimo è raggiunto solo da ℓ_i: perché in 18.6 valga l'uguaglianza occorre che tutti gli addendi con $j \neq i$ siano nulli, cioè $p(x_j) = 0$ per $j \neq i$; ma un polinomio di $\mathcal{P}_{n-1}$ è determinato dai suoi valori negli n nodi, e quei valori sono proprio quelli di ℓ_i.

Secondo calcolo: $m_i = 1/K_{n-1}(x_i, x_i)$. Il minimo m_i è esattamente quello descritto da Teorema 16.3, applicato con grado $n-1$ al posto di n e con $x_0 = x_i$. Quel teorema dice che il minimo vale $1/K_{n-1}(x_i, x_i)$ ed è raggiunto da $K_{n-1}(x_i, \cdot)/K_{n-1}(x_i, x_i)$.

Conclusione. I due calcoli riguardano lo stesso numero, quindi

$$\omega_i = m_i = \frac{1}{K_{n-1}(x_i, x_i)} = \lambda_{n-1}(x_i).$$

Per inciso, confrontando anche i due minimizzanti si ottiene gratis l'identità $\ell_i(x) = K_{n-1}(x_i, x)/K_{n-1}(x_i, x_i)$: i polinomi cardinali di Lagrange sui nodi gaussiani sono polinomi del nucleo normalizzati. ∎

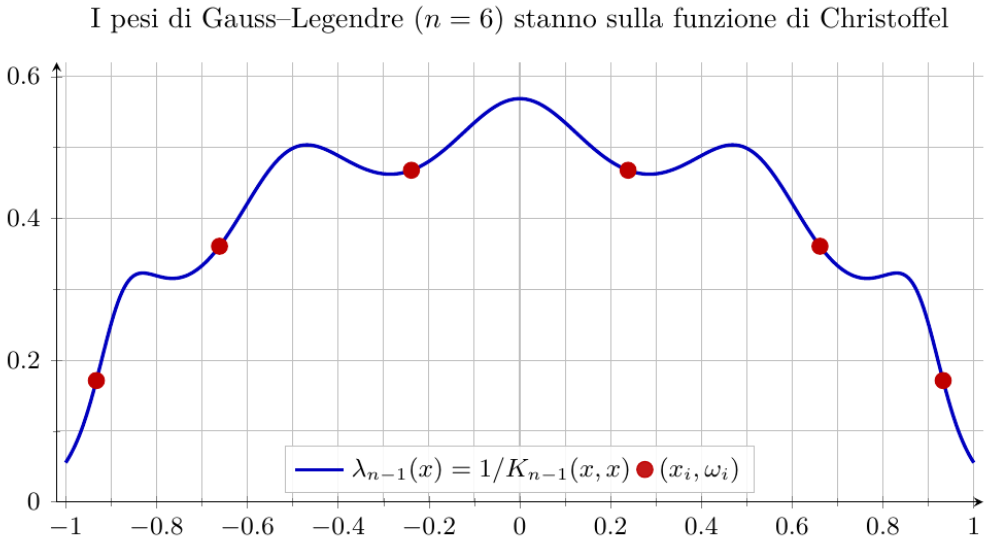


Figure 18.1: I sei pesi di Gauss–Legendre non sono numeri slegati: sono i valori della funzione di Christoffel nei nodi.

La Figura 18.1 lo mostra per $n = 6$: i punti (x_i, ω_i) cadono sulla curva λ_{n-1}. A differenza della formula per i soli Legendre scritta sopra, 18.5 vale per qualunque peso, e dà una seconda dimostrazione del fatto che i pesi sono positivi: sono reciproci di somme di quadrati.

In pratica è spesso preferibile la procedura di Golub–Welsch[45] [17]: si costruisce la matrice di Jacobi J_n dalla ricorrenza ortonormale, si calcolano autovalori e autovettori, e si pone

$$x_i = \lambda_i(J_n), \qquad \omega_i = \mu_0 |v_{1i}|^2, \qquad \mu_0 = \int_{-1}^{1} w(x)\,\mathrm{d}x.$$

Nel caso Legendre $\mu_0 = 2$. Questo metodo produce nodi reali e pesi positivi per costruzione e scala bene a ordini alti. La Figura 18.2 mostra la configurazione per $n = 4$.

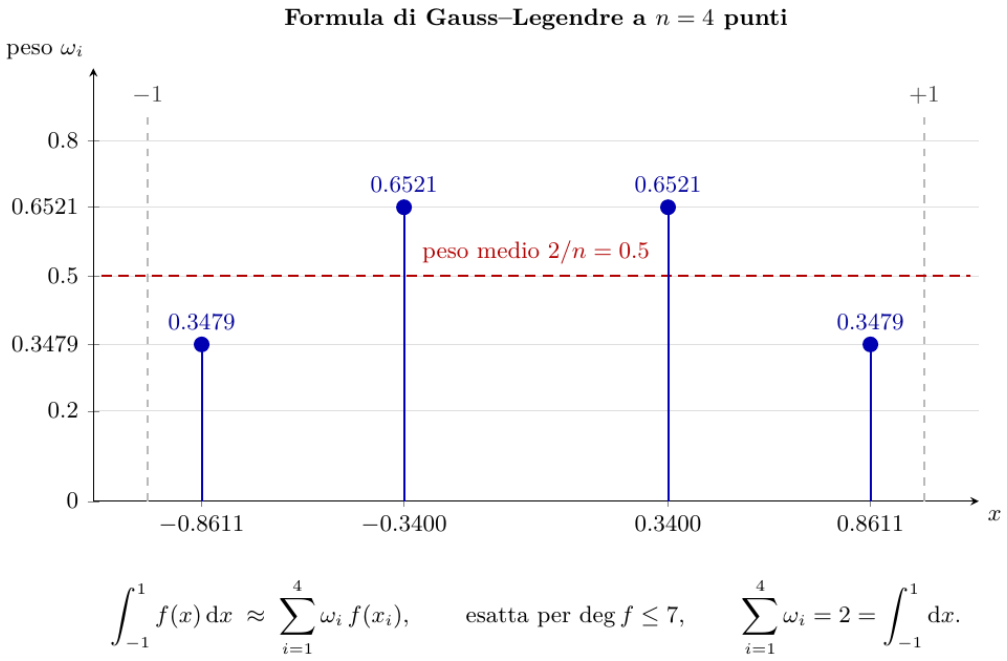


Figure 18.2: Nodi e pesi della formula di Gauss–Legendre a quattro punti. I nodi sono gli zeri di P_4, interni e simmetrici; i pesi sono tutti positivi, più grandi al centro e più piccoli vicino agli estremi, e la loro somma vale 2, cioè l'integrale del peso $w \equiv 1$.

18.2.3 Esempio: costruzione e verifica in R base

```
gauss_legendre <- function(n) {
  k<-seq_len(n-1); off<-k/sqrt(4*k^2-1); J<-matrix(0,n,n)
  J[cbind(k,k+1)]<-off; J[cbind(k+1,k)]<-off
  e<-eigen(J,symmetric=TRUE); o<-order(e$values)
  list(x=e$values[o],w=2*e$vectors[1,o]^2)
}
g2<-gauss_legendre(2); g4<-gauss_legendre(4)
plot(g4$x,g4$w,pch=19,cex=1.4,xlim=c(-1,1),ylim=c(0,max(g4$w)*1.25),
     xlab="nodo x_i",ylab="peso omega_i",main="Gauss--Legendre, n=4")
segments(g4$x,0,g4$x,g4$w,col=4,lwd=2)
points(g2$x,g2$w,pch=17,col=2,cex=1.3)
legend("topleft",c("n=4","n=2"),pch=c(19,17),col=c(4,2),bty="n")
moments<-sapply(0:7,function(k)sum(g4$w*g4$x^k))
exact<-sapply(0:7,function(k)if(k%%2==0)2/(k+1) else 0)
```

[4] Gene Howard Golub (1932-2007)
[5] John H. Welsch

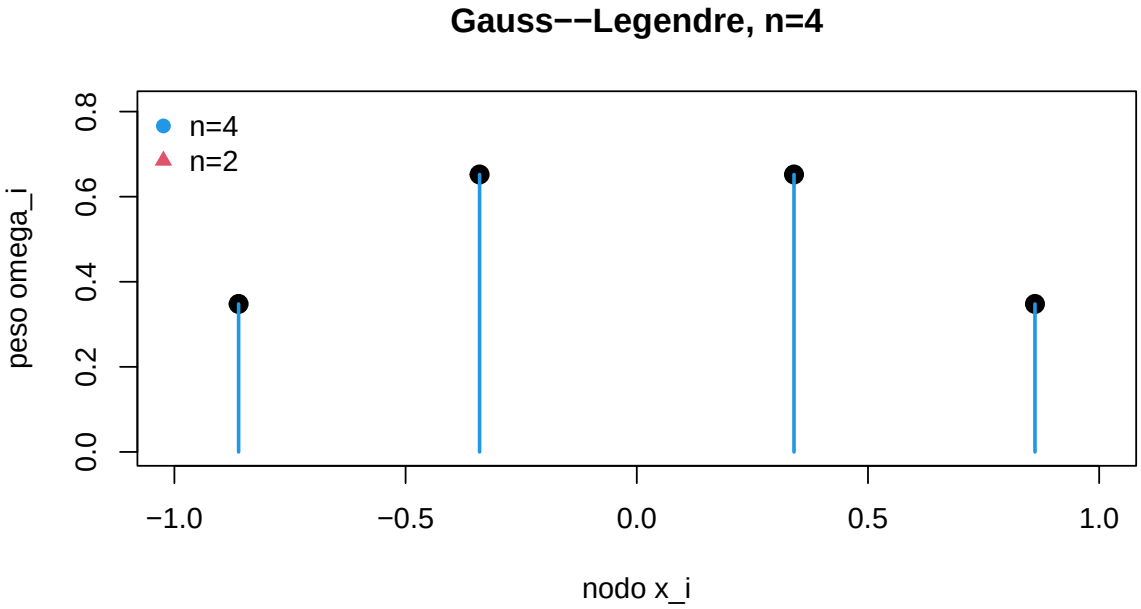


Figure 18.3: Nodi e pesi della formula di Gauss–Legendre a quattro punti, calcolati dagli autovalori della matrice di Jacobi.

Table 18.1: Quattro nodi integrano esattamente i monomi fino al grado 7.

grado	integrale_Gauss	esatto	errore
0	2.0000000	2.0000000	0
1	0.0000000	0.0000000	0
2	0.6666667	0.6666667	0
3	0.0000000	0.0000000	0
4	0.4000000	0.4000000	0
5	0.0000000	0.0000000	0
6	0.2857143	0.2857143	0
7	0.0000000	0.0000000	0

Quattro nodi devono integrare esattamente i monomi fino al grado 7, a precisione floating point. Il test non dimostra il teorema, ma controlla una conseguenza della costruzione.

Gli stessi nodi e pesi verificano 18.5: i pesi coincidono con i reciproci della funzione di Christoffel calcolata nei nodi.

Table 18.2: Verifica di 18.5 per Gauss–Legendre con n=6.

nodo	peso	reciproco_di_K	differenza
-0.9324695	0.1713245	0.1713245	0
-0.6612094	0.3607616	0.3607616	0
-0.2386192	0.4679139	0.4679139	0
0.2386192	0.4679139	0.4679139	0
0.6612094	0.3607616	0.3607616	0
0.9324695	0.1713245	0.1713245	0

18.3 Gauss–Jacobi e cambio di intervallo

18.3.1 Peso di Jacobi e mappa affine

Per il peso $(1-x)^{\alpha}(1+x)^{\beta}$, i nodi di Gauss–Jacobi sono gli zeri di $P_n^{(\alpha,\beta)}$. Anche qui Golub–Welsch si applica alla matrice tridiagonale simmetrica della ricorrenza ortonormale. Per la ricorrenza monica, i coefficienti sono, ponendo $s = \alpha + \beta$,

$$b_0 = \frac{\beta - \alpha}{s+2}, \qquad b_k = \frac{\beta^2 - \alpha^2}{(2k+s)(2k+s+2)} \ (k \geq 1),$$

$$a_1^2 = \frac{4(1+\alpha)(1+\beta)}{(s+2)^2(s+3)}, \qquad a_k^2 = \frac{4k(k+\alpha)(k+\beta)(k+s)}{(2k+s)^2(2k+s+1)(2k+s-1)}, \quad k \geq 2.$$

Il caso $k = 1$ è scritto a parte non per distrazione: sostituendo $k = 1$ nella formula generale i fattori $(k+s)$ e $(2k+s-1)$ si semplificano e si ritrova esattamente a_1^2, **tranne** quando $s = \alpha + \beta = -1$, dove entrambi si annullano e la forma generale diventa $0/0$. È precisamente il caso di Chebyshev di prima specie ($\alpha = \beta = -\tfrac{1}{2}$), e la formula separata serve a coprirlo.

I pesi finali sono $\mu_0 |v_{1i}|^2$, con $\mu_0 = 2^{s+1} B(\alpha+1, \beta+1)$. La trasformazione a $[a,b]$ è $t_i = (a+b)/2 + (b-a)x_i/2$ e moltiplica i pesi per $(b-a)/2$; se l'integrale originario ha un suo peso, anche questo si trasforma.

18.3.2 Esempio: Gauss–Jacobi asimmetrica

Il codice usa la formula generale di Golub–Welsch e confronta i momenti pesati fino al grado $2n-1$. I momenti esatti si calcolano espandendo $x^k = (2u-1)^k$, $u = (x+1)/2$, e usando la funzione Beta.

```
beta_fn <- function(a,b) exp(lgamma(a)+lgamma(b)-lgamma(a+b))
gauss_jacobi <- function(n,alpha,beta) {
  s<-alpha+beta
  b<-numeric(n); b[1]<-(beta-alpha)/(s+2)
  if(n>1) for(j in 1:(n-1))
    b[j+1]<-(beta^2-alpha^2)/((2*j+s)*(2*j+s+2))
  off<-numeric(max(0,n-1))
  if(n>1) {
    off[1]<-sqrt(4*(1+alpha)*(1+beta)/((s+2)^2*(s+3)))
    if(n>2) {
      k<-2:(n-1)
      off[k]<-sqrt(4*k*(k+alpha)*(k+beta)*(k+s)/
        ((2*k+s)^2*(2*k+s+1)*(2*k+s-1)))
    }
  }
  J<-diag(b,n,n)
  if(n>1){idx<-seq_len(n-1);J[cbind(idx,idx+1)]<-off;J[cbind(idx+1,idx)]<-off}
  e<-eigen(J,symmetric=TRUE);o<-order(e$values)
  mu0<-2^(s+1)*beta_fn(alpha+1,beta+1)
  list(x=e$values[o],w=mu0*e$vectors[1,o]^2)
}
alpha<-0.4; beta<-1.1; n<-4; gj<-gauss_jacobi(n,alpha,beta)
moment_jacobi <- function(k,alpha,beta) {
  sum(sapply(0:k,function(j)choose(k,j)*2^j*(-1)^(k-j)*
    beta_fn(beta+j+1,alpha+1))) * 2^(alpha+beta+1)
}
approx<-sapply(0:(2*n-1),function(k)sum(gj$w*gj$x^k))
exact<-sapply(0:(2*n-1),moment_jacobi,alpha=alpha,beta=beta)
knitr::kable(data.frame(k=0:(2*n-1),gauss_jacobi=approx,integrale_esatto=exact,
           errore_assoluto=abs(approx-exact)))
```

Table 18.3: Verifica dei momenti di Gauss–Jacobi con alpha=0.4, beta=1.1 e n=4.

k	gauss_jacobi	integrale_esatto	errore_assoluto
0	1.5804651	1.5804651	0
1	0.3160930	0.3160930	0
2	0.4003845	0.4003845	0
3	0.1659009	0.1659009	0
4	0.2026591	0.2026591	0
5	0.1073954	0.1073954	0
6	0.1280556	0.1280556	0
7	0.0772643	0.0772643	0

`beta_fn` calcola la funzione Beta $B(a,b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$ (la Gamma è la Definizione 15.1) passando per i logaritmi: la forma diretta `gamma(a)*gamma(b)/gamma(a+b)` va in overflow già per $a, b \gtrsim 170$, mentre `exp(lgamma(a)+lgamma(b)-lgamma(a+b))` resta stabile. In R si può usare direttamente `beta()` o `lbeta()`.

18.3.3 Esempio: i coefficienti di una serie ortogonale

In Capitolo 17 i coefficienti $c_k = \langle f, \phi_k \rangle_w$ sono stati calcolati con una regola del punto medio su decine di migliaia di punti, per non anticipare risultati non ancora dimostrati. Ora possiamo fare molto meglio. La funzione $f\phi_k$ con $f = e^x$ non è un polinomio, quindi il teorema di esattezza non si applica alla lettera, ma per una funzione analitica l'accuratezza della formula gaussiana resta eccellente.

```
d <- 8
coef_gauss <- function(n) {
  g <- gauss_legendre(n)
  Phi <- legendre_matrix_orthonormal(d, g$x)
  colSums(sweep(Phi, 1, g$w*exp(g$x), `*`))
}
coef_midpoint <- function(m) {
  xq <- -1 + (seq_len(m) - 0.5)*2/m
  Phi <- legendre_matrix_orthonormal(d, xq)
  colSums(sweep(Phi, 1, (2/m)*exp(xq), `*`))
}
riferimento <- coef_gauss(80)
knitr::kable(data.frame(
  regola = c("punto medio, m=1000", "punto medio, m=20000",
             "Gauss--Legendre, n=10", "Gauss--Legendre, n=20"),
  valutazioni_di_f = c(1000, 20000, 10, 20),
  errore_massimo = c(max(abs(coef_midpoint(1000) - riferimento)),
                     max(abs(coef_midpoint(20000) - riferimento)),
                     max(abs(coef_gauss(10)     - riferimento)),
                     max(abs(coef_gauss(20)     - riferimento)))))
```

Table 18.4: Coefficienti di Legendre di exp(x) fino al grado 8: errore massimo del punto medio composito e della quadratura di Gauss, a parità di coefficienti calcolati.

regola	valutazioni_di_f	errore_massimo
punto medio, m=1000	1000	5.51e-05
punto medio, m=20000	20000	1.00e-07
Gauss–Legendre, n=10	10	0.00e+00

Table 18.4: Coefficienti di Legendre di exp(x) fino al grado 8: errore massimo del punto medio composito e della quadratura di Gauss, a parità di coefficienti calcolati.

regola	valutazioni_di_f	errore_massimo
Gauss–Legendre, n=20	20	0.00e+00

Dieci nodi di Gauss battono ventimila valutazioni del punto medio, e con venti si arriva alla precisione macchina. La ragione è il teorema di esattezza: il punto medio composito converge come $O(m^{-2})$, mentre su una funzione analitica la formula gaussiana converge in modo spettrale. Il prezzo è che i nodi cambiano con n e non si possono riutilizzare quando si raffina.

18.4 Errore e varianti con nodi vincolati

Se $f \in C^{2n}[-1,1]$, l'errore di Gauss–Legendre ammette la forma

$$\int_{-1}^{1} f(x)\,\mathrm{d}x - Q_n(f) = \frac{f^{(2n)}(\xi)}{(2n)!}\int_{-1}^{1}\left(\prod_{i=1}^{n}(x-x_i)\right)^2 \mathrm{d}x, \qquad \xi \in (-1,1).$$

Per Jacobi vale una forma analoga integrando il quadrato del polinomio monico d'errore contro il peso. La formula spiega l'esattezza massima; la dimensione concreta dell'errore dipende dalla derivata e dalla regolarità della funzione.

Se è necessario includere un estremo, Gauss–Radau[6] fissa un nodo in uno dei capi e raggiunge esattezza $2n-2$ con n nodi. Se si fissano entrambi gli estremi, Gauss–Lobatto[7] raggiunge $2n-3$. La ragione è ancora ortogonalità: i nodi interni sono gli zeri dei polinomi di grado $n-1$ o $n-2$ ortogonali rispetto ai pesi modificati $(1\mp x)w(x)$ o $(1-x^2)w(x)$.

In sintesi

- Scegliere come nodi gli zeri del polinomio ortogonale raddoppia il grado di esattezza, da $n-1$ a $2n-1$, e $2n-1$ è il massimo possibile con n nodi.
- I pesi gaussiani sono sempre positivi, quindi la formula è stabile a ogni ordine: è la differenza sostanziale rispetto a Newton-Cotes.
- In pratica nodi e pesi non si calcolano dagli zeri di p_n, ma come autovalori e prime componenti degli autovettori della matrice di Jacobi (Golub–Welsch).
- Cambiare peso significa cambiare famiglia ortogonale: Gauss–Jacobi integra esattamente le singolarità $(1\mp x)^{\alpha,\beta}$ agli estremi, che una formula a peso costante risolverebbe male.
- Radau e Lobatto scambiano grado di esattezza con il controllo di uno o due nodi agli estremi, utile quando servono i valori al bordo.

18.5 Esercizi

1. Dimostrare che n nodi non possono dare esattezza oltre $2n-1$.
2. Perché i pesi di Gauss sono positivi?
3. Confrontare Gauss, Radau e Lobatto per lo stesso numero di nodi.
4. Applicare i nodi a $\int_0^1 e^x\,\mathrm{d}x$ dopo una mappa lineare; confrontare con l'integrale esatto $e-1$.
5. Riprendere l'esercizio 5 di Capitolo 11 e verificare numericamente le stime a priori là ottenute per Gauss–Legendre con $2,3,4,5$ nodi.

[6] Rodolphe Radau (1835-1911)
[7] Rehuel Lobatto (1797-1866)

6. Calcolare $\int_{-1}^{1}(1-x)^{-1/2}\cos x\,\mathrm{d}x$ con Gauss–Legendre e con Gauss–Jacobi ($\alpha=-1/2,\beta=0$) a parità di nodi, e spiegare la differenza.



19. Approssimazione spettrale e altre applicazioni

19.1 Approssimazione spettrale

19.1.1 Quanto velocemente decadono i coefficienti

Una funzione sufficientemente regolare può essere approssimata mediante un numero limitato di coefficienti in una base ortogonale. La rapidità del loro decadimento dipende dalla regolarità di f, non dal grado scelto. Se $f \in C^m$, i coefficienti di Legendre decadono algebricamente come $O(k^{-m-1/2})$. Se f è analitica e prolungabile all'ellisse di Bernstein E_ρ — l'ellisse di fuochi ± 1 e semiassi di somma $\rho > 1$ — il decadimento è geometrico,

$$|c_k| \le C\,\rho^{-k}, \tag{19.1}$$

e l'errore di troncamento al grado N si comporta come ρ^{-N}: è la *convergenza spettrale*. Una singolarità vicina all'intervallo riduce ρ e quindi rallenta la convergenza senza distruggerla; una discontinuità *sull'* intervallo la distrugge, e si ricade nel caso algebrico lento discusso più sotto. Nella pratica i coefficienti si calcolano con quadratura, FFT per Chebyshev oppure ricorrenze e trasformazioni dedicate. ### Una mappa del capitolo

La Figura 19.1 illustra le relazioni tra ortogonalità, approssimazione, quadratura, distribuzione degli zeri e metodi spettrali.

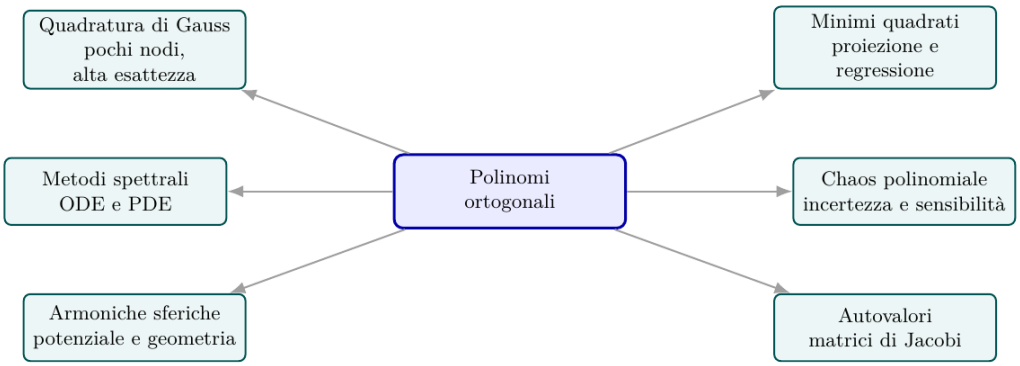


Figure 19.1: Mappa delle relazioni tra famiglie ortogonali, approssimazione, quadratura e metodi spettrali.

19.1.2 Esempio: convergenza dello sviluppo di Legendre di e^x

Calcoliamo i coefficienti della serie di Legendre con una griglia uniforme fine come quadratura composita e misuriamo l'errore massimo su una griglia di validazione. Aumentando il numero di punti di integrazione e di valutazione si può verificare la robustezza della misura. I risultati sono errori osservati, non una maggiorazione analitica.

```
m<-30000; xq<- -1+(seq_len(m)-0.5)*2/m; wq<-rep(2/m,m)
xx<-seq(-1,1,length.out=2001); truth<-exp(xx); Nmax<-30
Pq<-sapply(0:Nmax,legendre,x=xq)
coef<-colSums(sweep(Pq,1,wq*exp(xq),`*`))*(2*(0:Nmax)+1)/2
errs<-numeric(Nmax+1)
for(n in 0:Nmax){Px<-sapply(0:n,legendre,x=xx)
  vals<-drop(Px%*%coef[1:(n+1)]); errs[n+1]<-max(abs(truth-vals))}
plot(0:Nmax,errs,type="b",pch=16,log="y",lwd=2,col=4,
     xlab="grado massimo N",ylab="max_x |f-p_N|",
     main="Convergenza osservata: espansione di Legendre")
```

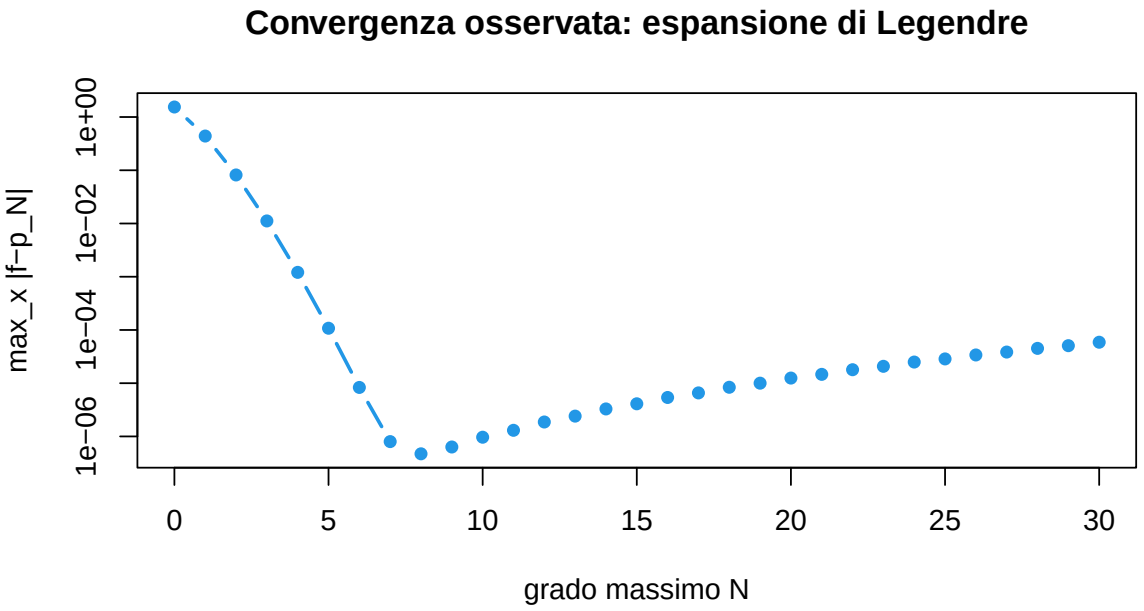


Figure 19.2: Errore massimo misurato per lo sviluppo troncato di Legendre di exp(x).

Il decadimento iniziale rapido è compatibile con l'analiticità di e^x. Quando l'errore si avvicina alla precisione numerica della quadratura e dell'aritmetica, l'andamento può appiattirsi o oscillare.

19.2 I limiti: discontinuità e fenomeno di Gibbs

Un salto produce coefficienti che decadono molto più lentamente e innesca il fenomeno di Gibbs[1]. L'aumento del grado restringe la zona oscillante, ma non elimina l'overshoot vicino alla discontinuità. Un filtro sui coefficienti attenua oscillazioni ad alta frequenza, con il costo di allargare la transizione. La Figura 19.3 confronta somme troncate della funzione segno.

```
m<-40000; xq<- -1+(seq_len(m)-0.5)*2/m; wq<-rep(2/m,m)
f<-sign(xq); xx<-seq(-1,1,length.out=2001); N<-c(12,40)
cols<-c(2,4)
plot(xx,sign(xx),type="s",col="gray50",lty=2,lwd=2,
     xlab="x",ylab="approssimazione",main="Gibbs con polinomi di Legendre",
```

[1] Josiah Willard Gibbs (1839-1903)

```
     ylim=c(-1.35,1.35))
for(j in seq_along(N)){
  n<-N[j]; Pq<-sapply(0:n,legendre,x=xq)
  cf<-colSums(sweep(Pq,1,wq*f,`*`))*(2*(0:n)+1)/2
  Px<-sapply(0:n,legendre,x=xx); lines(xx,drop(Px%*%cf),col=cols[j],lwd=2)
}
legend("bottomright",c("segno(x)",paste0("N=",N)),
       col=c("gray50",cols),lty=c(2,1,1),lwd=2,bty="n")
```

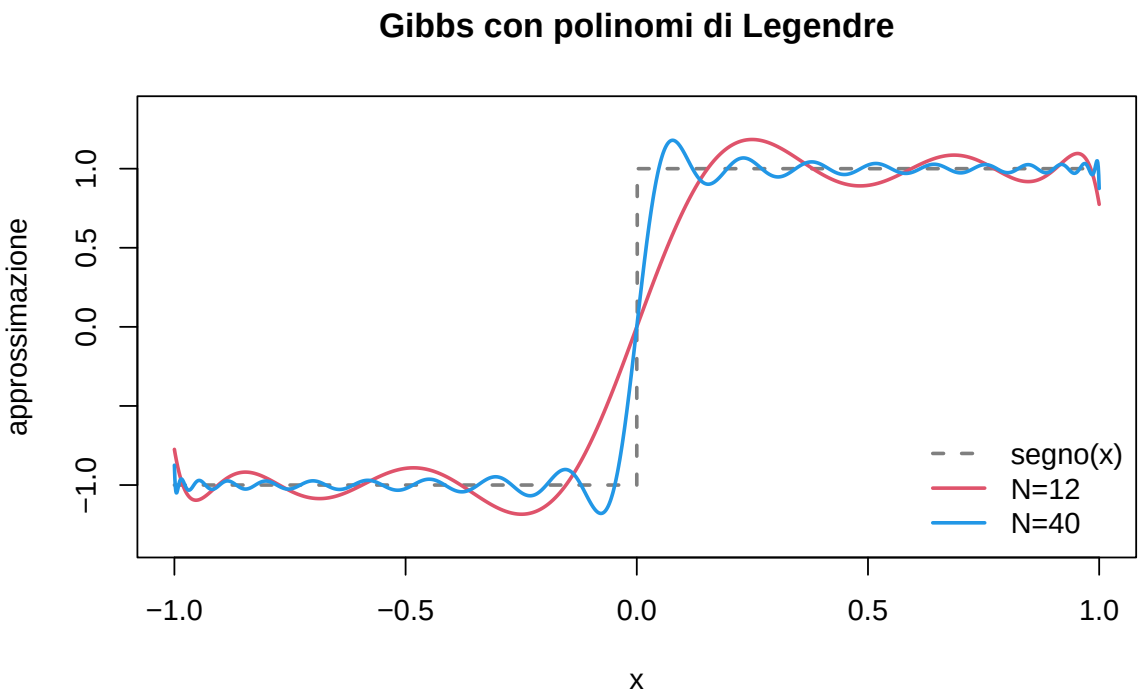


Figure 19.3: Sviluppo di Legendre della funzione segno: l'aumento del grado restringe ma non annulla le oscillazioni presso il salto.

Un filtro esponenziale smorza i coefficienti con $\sigma_k = \exp[-\gamma(k/N)^q]$ prima di risommare la serie. La scelta dei due parametri non è arbitraria: q è l'*ordine* del filtro e decide quanto bruscamente si passa dai modi conservati a quelli soppressi, mentre γ si prende di norma pari a $-\log \varepsilon_{\text{mach}} \approx 36$ in doppia precisione, così che $\sigma_N = \varepsilon_{\text{mach}}$: l'ultimo modo viene azzerato esattamente al livello del rumore di arrotondamento, e non oltre. Con $q = 8$ e $\gamma = 36$ si ottiene il filtro usato qui sotto:

```
m<-40000;xq<- -1+(seq_len(m)-0.5)*2/m;wq<-rep(2/m,m);f<-sign(xq)
xx<-seq(-1,1,length.out=2001);n<-40;Pq<-sapply(0:n,legendre,x=xq)
cf<-colSums(sweep(Pq,1,wq*f,`*`))*(2*(0:n)+1)/2
Px<-sapply(0:n,legendre,x=xx); raw<-drop(Px%*%cf)
sigma<-exp(-36*((0:n)/n)^8); filtered<-drop(Px%*%(cf*sigma))
plot(xx,raw,type="l",col=2,lwd=1.5,xlab="x",ylab="valore",
     main="Gibbs: somma filtrata e non filtrata",ylim=c(-1.35,1.35))
lines(xx,filtered,col=4,lwd=2);lines(xx,sign(xx),col="gray40",lty=2)
legend("bottomright",c("senza filtro","filtro esponenziale","segno(x)"),
       col=c(2,4,"gray40"),lty=c(1,1,2),lwd=c(1.5,2,1),bty="n")
```

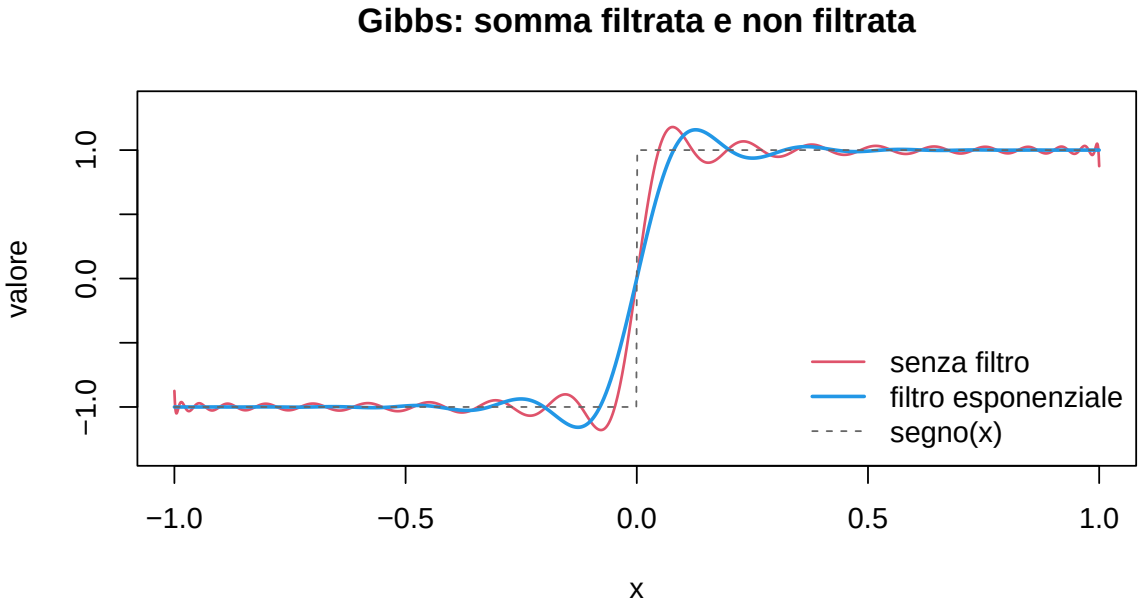


Figure 19.4: Filtro esponenziale: attenua le oscillazioni, rendendo più larga la transizione del salto.

19.3 Metodi spettrali

19.3.1 Collocazione per un problema ai limiti

In un metodo di collocazione spettrale si cerca un polinomio u_N e si impone che il residuo dell'equazione differenziale si annulli in nodi scelti. La matrice di derivazione si costruisce valutando le derivate della base nei nodi; condizioni al bordo e nodi di Chebyshev–Lobatto sono spesso comodi perché includono gli estremi. Per soluzioni lisce, l'errore spettrale può diminuire rapidamente con N, mentre discontinuità e strati limite rallentano la convergenza.

Esempio: per $-u'' = \pi^2 \sin(\pi x)$, $u(-1) = u(1) = 0$, la soluzione è $u(x) = \sin(\pi x)$. La seguente figura mostra la soluzione calcolata imponendo l'equazione ai nodi interni di Chebyshev–Lobatto. Per ordini alti si usano matrici di derivazione con scalature e formulazioni che controllano il condizionamento.

```
N<-12; k<-0:N; xn<-cos(pi*k/N)
cvec<-c(2,rep(1,N-1),2)*(-1)^k
DX<-outer(xn,xn,"-"); D<-matrix(0,N+1,N+1)
for(i in 1:(N+1)) for(j in 1:(N+1)) if(i!=j)
  D[i,j]<-(cvec[i]/cvec[j])/DX[i,j]
diag(D)<- -rowSums(D)
D2<-D%*%D; interior<-2:N
rhs<-pi^2*sin(pi*xn[interior])
u<-numeric(N+1)
u[interior]<-solve(-D2[interior,interior],rhs)
xx<-seq(-1,1,length.out=1001); exact<-sin(pi*xx)
lambda<-(-1)^k; lambda[c(1,N+1)]<-lambda[c(1,N+1)]/2
barycentric_eval<-function(z,xn,yn,lambda) {
  vapply(z,function(x) {
    d<-x-xn; hit<-which.min(abs(d))
    if(abs(d[hit])<=8*.Machine$double.eps*max(1,abs(x))) return(yn[hit])
    t<-lambda/d
    sum(t*yn)/sum(t)
  },numeric(1))
}
u_grid<-barycentric_eval(xx,xn,u,lambda)
plot(xx,exact,type="l",lwd=2,col=1,xlab="x",ylab="u(x)",
```

```
     main="Collocazione spettrale di -u'' = pi^2 sin(pi x)")
lines(xx,u_grid,lwd=2,lty=2,col=4);points(xn,u,pch=19,col=2,cex=0.6)
legend("topleft",c("soluzione esatta","collocazione","nodi Lobatto"),
       col=c(1,4,2),lty=c(1,2,NA),pch=c(NA,NA,19),bty="n")
cat("errore massimo sulla griglia =",max(abs(u_grid-exact)),"\n")
```

```
errore massimo sulla griglia = 2.125982e-07
```

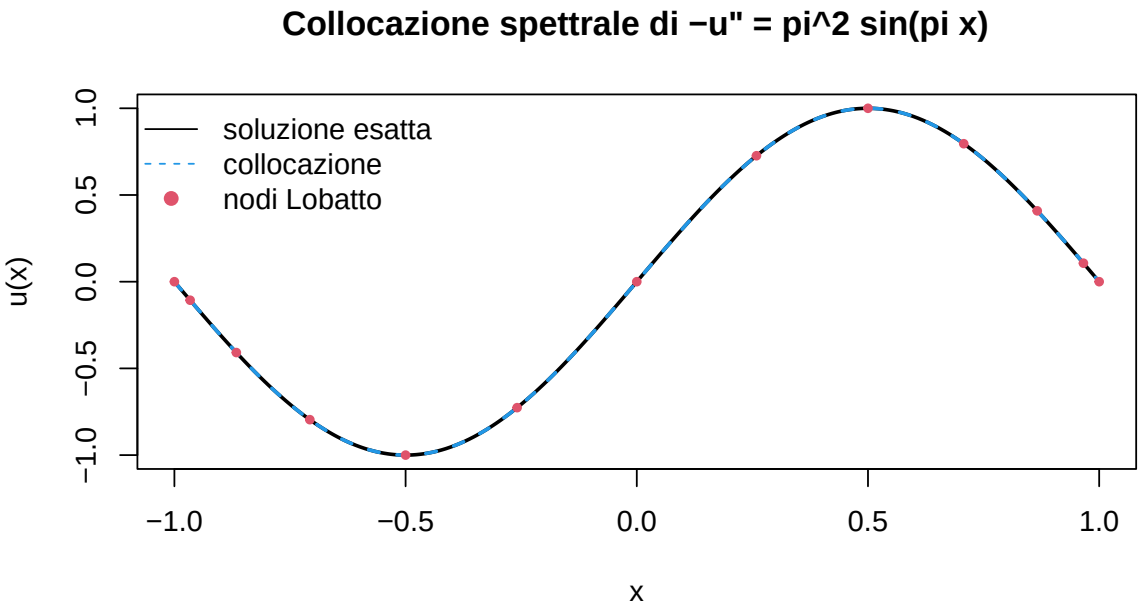


Figure 19.5: Collocazione Chebyshev–Lobatto della soluzione liscia sin(pi x); indicato l'errore massimo misurato sulla griglia.

19.4 Altre applicazioni

19.4.1 Probabilità, chaos polinomiale e armoniche sferiche

La scelta della misura è centrale anche fuori dall'intervallo finito:

- per una variabile uniforme su $[-1,1]$, Legendre è la base naturale;
- per una variabile gaussiana standard, i polinomi di Hermite[2] sono ortogonali rispetto alla densità normale e costituiscono la base del *polynomial chaos* per propagare incertezza in modelli non lineari;
- per coordinate sferiche, i Legendre associati compaiono nelle armoniche sferiche e nella separazione dell'equazione di Laplace[3];
- i Chebyshev sono usati nell'approssimazione, nell'interpolazione e nei metodi spettrali grazie alla loro relazione con FFT e coseni;
- la tridiagonale di Jacobi collega coefficienti, zeri e quadratura, oltre a fornire approssimazioni spettrali di operatori di moltiplicazione.

19.4.2 Gli Hermite e la misura gaussiana

Per la gaussiana standard, la ricorrenza monica di Hermite è $He_{n+1}(x) = xHe_n(x) - nHe_{n-1}(x)$, con $\mathbb{E}[He_n(X)He_m(X)] = n!\,\delta_{nm}$. La normalizzazione $He_n/\sqrt{n!}$ rende le basi ortonormali. Ciò consente di espandere una risposta casuale in coefficienti e stimare media e varianza dai coefficienti,

[2] Charles Hermite (1822-1901)
[3] Pierre-Simon Laplace (1749-1827)

senza simulare ogni possibile ingresso.

In sintesi

La famiglia si sceglie in funzione del dominio e della misura: peso uniforme su un intervallo porta a Legendre, peso singolare agli estremi a Jacobi/Chebyshev, densità gaussiana a Hermite. In ogni caso si controllano condizionamento, quadratura usata per i coefficienti, risoluzione delle singolarità e convergenza osservata.

- La convergenza spettrale è una proprietà della *funzione*, non del metodo: 19.1 vale finché f è analitica in un intorno ellittico dell'intervallo.
- Una discontinuità produce il fenomeno di Gibbs: l'overshoot non si riduce aumentando N, si restringe soltanto. I filtri lo attenuano al prezzo di allargare la transizione.
- Nella collocazione spettrale l'equazione differenziale viene imposta nei nodi; la scelta di nodi di Chebyshev–Lobatto, già incontrati in Capitolo 9, include gli estremi e rende immediate le condizioni al bordo.
- Fuori dall'intervallo finito cambia solo la misura: Hermite per la gaussiana, Laguerre per la semiretta. Ricorrenza a tre termini, zeri, quadratura gaussiana valgono identicamente.

19.5 Esercizi

1. Confrontare gli errori di Legendre e Chebyshev per $|x|$ e per $1/(1+25x^2)$; per la seconda, stimare ρ in 19.1 a partire dalla posizione dei poli $\pm i/5$ e confrontare con la pendenza misurata.
2. Ripetere l'esperimento di Gibbs con una discontinuità spostata in $x = 0.3$ e descrivere l'effetto della base pari.
3. Per il chaos polinomiale, espandere $f(X) = \exp(X)$ fino al grado 4 nella base di Hermite ortonormale e confrontare media e varianza approssimate con i valori analitici.
4. Usare nodi di Gauss–Legendre per calcolare $\int_{-1}^{1} e^{x^2}\,\mathrm{d}x$ al crescere di n e separare l'errore di quadratura dall'errore roundoff.

VI Zeri di funzione

20 Zeri di funzione 239
20.1 Classificazione dei metodi 239
20.2 Notazione 240
20.3 Metodo di bisezione (o dicotomico) 240
20.4 Criteri di arresto e accuratezza raggiungibile . 242
20.5 Dalla bisezione ai metodi aperti 243
20.6 Metodo di Newton-Raphson 244
20.7 Metodo delle secanti 247
20.8 Metodo delle false posizioni (*regula falsi*) 251
20.9 Metodo delle corde (derivata congelata) . . . 253
20.10 Metodo di Müller: interpolazione parabolica su tre punti 254
20.11 Metodo della parabola tangente (Taylor quadratico) 255
20.12 Metodo di Halley 257
20.13 Confronto numerico dei metodi 259

21 Ordine di convergenza e radici multiple 265
21.1 Velocità di convergenza 265
21.2 Verifica numerica dell'ordine 268
21.3 Radici multiple e rallentamento della convergenza 274
21.4 Esercizi 277

22 Metodi ibridi e algoritmi con salvaguardia 279
22.1 Lo schema generale 279
22.2 Metodo di Dekker 280
22.3 Metodo di Brent–Dekker 282
22.4 Metodo di Ridders 285
22.5 Metodo TOMS 748 (Alefeld–Potra–Shi) 286
22.6 Metodo di Chandrupatla 290
22.7 Metodo ITP (Interpolate–Truncate–Project) . . . 292
22.8 Gli ibridi con derivate: uno schema comune . 294
22.9 Metodo Newton–Bisection 296
22.10 Metodo Halley–Bisection 297
22.11 Metodo Chebyshev–Bisection 297
22.12 Metodo Householder–Bisection 298
22.13 Confronto sperimentale 299
22.14 Esercizi 300

23 Iterazioni di punto fisso 303
23.1 Elenco metodi numerici per zeri di funzione . . 307

24 Zeri di polinomi 309
24.1 Il metodo di Sturm 309
24.2 Limitazione delle radici 315



20. Zeri di funzione

In molte applicazioni si presenta il problema di trovare una soluzione x tale che:

$$f(x) = 0,$$

dove $f : \mathbb{R} \to \mathbb{R}$ è una funzione continua. Un numero reale α che soddisfa $f(\alpha) = 0$ si chiama *radice* della funzione. Se $f(x)$ è lineare, ossia ha la forma:

$$f(x) = ax + b,$$

dove $a \neq 0$ e b sono costanti assegnate, la radice si calcola facilmente come $\alpha = -b/a$.
Se invece $f(x)$ è non lineare, per esempio:

$$f(x) = ax^n - b,$$

e se a e b hanno lo stesso segno oppure n è dispari, si può determinare una radice reale ponendo $\alpha = \sqrt[n]{b/a}$. Nei problemi pratici, tuttavia, la funzione $f(x)$ è spesso non lineare e non è disponibile una formula esplicita per determinarne facilmente una radice. In questi casi occorre ricorrere a metodi numerici per approssimarla.

20.1 Classificazione dei metodi

I metodi numerici per approssimare le soluzioni di $f(x) = 0$ si suddividono in tre famiglie, che corrispondono all'organizzazione di questo capitolo e del successivo.

Metodi con intervallo di confinamento (*bracketing*). Mantengono a ogni passo un intervallo $[a_k, b_k]$ tale che $f(a_k)f(b_k) \leq 0$ e che, per il teorema degli zeri, contiene una radice. Appartengono a questa famiglia il metodo di bisezione (o dicotomia, Sezione 20.3) e la *regula falsi* (Sezione 20.8). La convergenza è garantita, ma in generale è lenta.

Metodi aperti locali. Generano una successione $x_0, x_1, x_2, \ldots$ sostituendo localmente f con un modello semplice (una retta tangente, una secante o una parabola) e assumendo come iterata successiva uno zero del modello. Appartengono a questa famiglia il metodo di Newton[1]-Raphson[2] (Sezione 20.6), il metodo delle secanti (Sezione 20.7), il metodo delle corde (Sezione 20.9), il metodo di Müller (Sezione 20.10), la parabola tangente (Sezione 20.11) e il metodo di Halley (Sezione 20.12). Alla stessa famiglia appartengono anche le iterazioni di punto fisso (Capitolo 23). La convergenza locale è rapida, ma **non è garantita globalmente**.

[1] Isaac Newton (1643-1727)
[2] Joseph Raphson (1648-1715)

Metodi ibridi con salvaguardia. Propongono a ogni passo il candidato di un metodo aperto e lo accettano solo se supera i test di ammissibilità; in caso contrario, ricorrono a un passo di bisezione. Combinano così la rapidità dei metodi aperti con la garanzia dei metodi di confinamento. Sono trattati nel capitolo sui metodi ibridi.

In alcuni casi particolari, come nel caso di polinomi, esistono tecniche speciali che non rientrano nelle famiglie sopra descritte (Capitolo 24).

> ⚠ Avvertenza terminologica
>
> In letteratura il nome *metodo delle corde* è usato con due significati diversi: in alcuni testi indica la *regula falsi*, in altri il metodo di Newton con derivata congelata. In questi appunti il nome è riservato alla seconda costruzione (Sezione 20.9), mentre la prima è sempre chiamata *regula falsi* o *metodo delle false posizioni*.

20.2 Notazione

Per tutto il capitolo adottiamo le seguenti convenzioni uniformi:

Table 20.1: Notazione usata nel capitolo

Simbolo	Significato
α	la radice cercata, $f(\alpha) = 0$
x_k	iterata k-esima; l'indice di iterazione è sempre k
$e_k = x_k - \alpha$	errore alla k-esima iterazione
τ	tolleranza richiesta dall'utente (criterio d'arresto)
ρ	raggio dell'intorno di α in cui vale la convergenza locale
$M_j = \sup \lvert f^{(j)} \rvert$	estremo superiore della derivata j-esima sull'intervallo considerato
$\varepsilon_{\text{mach}}$	precisione di macchina ($2^{-52} \approx 2.22 \cdot 10^{-16}$ in doppia precisione)
$I[a, b, c, \dots]$	il più piccolo intervallo chiuso contenente $a, b, c, \dots$

20.3 Metodo di bisezione (o dicotomico)

Il metodo di bisezione è una semplice applicazione del seguente teorema, che enunciamo senza dimostrazione.

Theorem 20.1 — Teorema: esistenza di uno zero per cambio di segno. Sia $f : [a, b] \to \mathbb{R}$ una funzione continua su $[a, b]$ e tale che $f(a)f(b) < 0$. Allora esiste almeno un punto $\alpha \in (a, b)$ tale che $f(\alpha) = 0$.

Se $f(x)$ è continua e a_0 e b_0 sono due punti tali che $f(a_0)f(b_0) < 0$, significa che $f(a_0)$ e $f(b_0)$ hanno segno opposto. Il Teorema 20.1 garantisce l'esistenza di almeno una radice α nell'intervallo (a_0, b_0).

Ora consideriamo il punto medio dell'intervallo:

$$c_0 = \frac{a_0 + b_0}{2}.$$

Si possono verificare solo tre possibili situazioni:

- **Caso 1:** $f(c_0) = 0$; in tal caso, c_0 è una radice esatta.
- **Caso 2:** $f(c_0)f(a_0) < 0$; in questo caso, esiste almeno una radice nell'intervallo (a_0, c_0).
- **Caso 3:** $f(c_0)f(b_0) < 0$; in questo caso, esiste almeno una radice nell'intervallo (c_0, b_0).

Escludendo il primo caso in cui la radice è già stata trovata, è sempre possibile ridurre l'intervallo di ricerca dimezzandolo. Ripetendo questo procedimento, il metodo dicotomico consente, data una tolleranza τ, di trovare un intervallo di lunghezza inferiore a τ che contiene una radice.

L'algoritmo può essere descritto come segue:

Metodo di bisezione

Dati: a, b con $f(a)f(b) < 0$; tolleranza τ
Restituisce: approssimazione della radice con errore minore di $\tau/2$

```
mentre b − a ≥ τ fai
    c ← a + (b − a)/2          ▷ non (a + b)/2: rischio di overflow
    se f(c) = 0 allora
        restituisci c          ▷ radice esatta
    se sign f(c) ≠ sign f(a) allora
        b ← c                  ▷ la radice sta in (a, c)
    altrimenti
        a ← c                  ▷ la radice sta in (c, b)
restituisci c
```

Due dettagli di implementazione

1. Il punto medio va calcolato come $a_k + (b_k - a_k)/2$ e non come $(a_k + b_k)/2$: la seconda forma può produrre *overflow* quando a_k e b_k sono grandi, e per estremi di segno concorde non è garantito che il risultato cada nell'intervallo.
2. Il test di cambio di segno va fatto confrontando i segni, non calcolando il prodotto $f(c)f(a_k)$: il prodotto di due numeri piccoli può andare in *underflow* a zero (facendo credere di aver trovato una radice esatta) e quello di due numeri grandi in *overflow*.

È possibile calcolare il numero minimo di passi necessari per approssimare una radice con l'accuratezza desiderata. Consideriamo che, ad ogni iterazione, la lunghezza dell'intervallo si dimezza. Dunque, per l'intervallo all'iterazione k abbiamo:

$$\begin{aligned} b_k - a_k &= \frac{1}{2}(b_{k-1} - a_{k-1}), \\ &\vdots \\ &= 2^{-k}(b_0 - a_0), \\ &= 2^{-k}(b - a), \end{aligned}$$

Imponendo che $b_k - a_k < \tau$, si ottiene:

$$2^{-k}(b-a) < \tau, \qquad \Rightarrow \qquad \frac{b-a}{\tau} < 2^k.$$

Applicando il logaritmo base 2 ad entrambi i membri:

$$k > \log_2\left(\frac{b-a}{\tau}\right) = \log_2(b-a) - \log_2(\tau).$$

Il più piccolo intero che soddisfa questa disuguaglianza, cioè

$$k_{\min} = \left\lceil \log_2 \left(\frac{b-a}{\tau} \right) \right\rceil,$$

fornisce il numero minimo di iterazioni necessarie per garantire che l'ampiezza dell'intervallo sia minore di τ. Poiché come approssimazione si assume il punto medio, l'errore commesso è allora minore di $\tau/2$. Si noti che questo numero **non dipende da** f: è il principale pregio del metodo, insieme al fatto che l'errore è maggiorato in modo rigoroso a ogni passo.

In termini di velocità di convergenza (Sezione 21.1) la bisezione è R-lineare con fattore $1/2$: ogni iterazione guadagna esattamente un bit di accuratezza, quindi servono circa 3.3 iterazioni per ogni cifra decimale e circa 52 iterazioni per esaurire la doppia precisione.

20.4 Criteri di arresto e accuratezza raggiungibile

Prima di introdurre metodi più rapidi è bene chiarire fino a dove ha senso spingersi. In aritmetica finita la funzione non viene valutata esattamente: il valore calcolato è

$$\hat{f}(x) = f(x) + \delta(x), \qquad |\delta(x)| \le \epsilon_f,$$

dove ϵ_f raccoglie gli errori di arrotondamento commessi nella valutazione. Una stima realistica è $\epsilon_f \approx \varepsilon_{\text{mach}}\, \mu(x)$, dove $\mu(x)$ è l'ordine di grandezza dei **termini intermedi** del calcolo, che può essere molto maggiore di $|f(x)|$ in presenza di cancellazione.

20.4.1 Intervallo di incertezza e condizionamento della radice

Tutti i punti x per cui $|f(x)| \le \epsilon_f$ sono **indistinguibili da radici**: nessun algoritmo che usi solo i valori calcolati di f può separarli. Se α è una radice semplice, sviluppando $f(x) \approx f'(\alpha)(x-\alpha)$ si ottiene che l'intervallo di incertezza ha semiampiezza

$$\boxed{|x-\alpha| \lesssim \frac{\epsilon_f}{|f'(\alpha)|}} \tag{20.1}$$

La quantità $1/|f'(\alpha)|$ è dunque il **numero di condizionamento della radice**: una radice con derivata piccola è intrinsecamente mal determinata, indipendentemente dal metodo usato.

Se invece α ha molteplicità m, il primo termine non nullo dello sviluppo è $f^{(m)}(\alpha)(x-\alpha)^m/m!$ e si ottiene

$$|x-\alpha| \lesssim \left(\frac{m!\,\epsilon_f}{|f^{(m)}(\alpha)|} \right)^{1/m}, \tag{20.2}$$

cioè un intervallo di incertezza di ampiezza $O(\epsilon_f^{1/m})$: per una radice doppia in doppia precisione si arriva a circa 10^{-8}, per una tripla a circa 10^{-5}. Nessun metodo può fare meglio.

> **Esempio: la radice tripla di $(x-1)^3$**
>
> Valutiamo il polinomio $p(x) = (x-1)^3$ nella sua forma espansa $x^3 - 3x^2 + 3x - 1$ in un intorno di $x = 1$. Il valore calcolato è puro rumore di arrotondamento e il segno è erratico.

```
p_espanso <- function(x) x^3 - 3*x^2 + 3*x - 1
x <- 1 + (-8:8)*1e-6
data.frame(delta = x - 1, p = p_espanso(x), segno = sign(p_espanso(x)))
```

```
    delta             p segno
1  -8e-06 -4.440892e-16    -1
2  -7e-06 -2.220446e-16    -1
3  -6e-06  0.000000e+00     0
4  -5e-06 -4.440892e-16    -1
5  -4e-06  0.000000e+00     0
6  -3e-06 -2.220446e-16    -1
7  -2e-06  0.000000e+00     0
8  -1e-06  0.000000e+00     0
9   0e+00  0.000000e+00     0
10  1e-06 -4.440892e-16    -1
11  2e-06  0.000000e+00     0
12  3e-06  4.440892e-16     1
13  4e-06  0.000000e+00     0
14  5e-06  4.440892e-16     1
15  6e-06  0.000000e+00     0
16  7e-06  4.440892e-16     1
17  8e-06  4.440892e-16     1
```

I termini intermedi sono di ordine 3, quindi $\epsilon_f \approx 3\varepsilon_{\text{mach}} \approx 6.7 \cdot 10^{-16}$ e la 20.2 con $m = 3$, $f''' = 6$ prevede un'incertezza di circa $(6.7 \cdot 10^{-16})^{1/3} \approx 8.8 \cdot 10^{-6}$: esattamente l'ampiezza della zona di rumore osservata.

20.4.2 Criteri pratici

Alla luce di quanto sopra, i criteri d'arresto usati in pratica sono i seguenti.

- **Test sull'incremento (misto assoluto/relativo).** Si arresta quando

$$|x_{k+1} - x_k| \le \tau_r\,|x_{k+1}| + \tau_a.$$

  È il criterio più usato per i metodi aperti; è affidabile quando la convergenza è almeno lineare con fattore nettamente minore di 1, perché in tal caso $|x_{k+1} - x_k|$ è una buona stima di $|e_k|$. Non lo è per convergenza lenta: con fattore $1 - 1/m$ l'incremento sottostima l'errore di un fattore m.
- **Test sul residuo.** Si arresta quando $|f(x_k)| \le \epsilon_f$. Dipende dalla scala di f (moltiplicare f per 10^6 cambia il risultato) e, per la 20.1, si attiva lontano dalla radice quando $|f'(\alpha)|$ è piccolo. Usato da solo è il criterio peggiore; è però l'unico utilizzabile quando si vuole una garanzia sul residuo e non sull'argomento.
- **Test sull'ampiezza del bracket.** Per i metodi di confinamento si arresta quando

$$b_k - a_k \le 2\varepsilon_{\text{mach}}\,|b_k| + \tau_a,$$

  che è la forma usata nelle implementazioni di tipo *zeroin*. È l'unico criterio che fornisce una **maggiorazione rigorosa** dell'errore, perché la radice è certamente nell'intervallo.
- **Limite sul numero di iterazioni.** In ogni caso va imposto un numero massimo di iterazioni: un metodo aperto può non convergere affatto (Sezione 20.6.4).

Una buona implementazione combina i primi due test in `or` con il limite sulle iterazioni, e per i metodi con bracket usa il terzo.

20.5 Dalla bisezione ai metodi aperti

La bisezione conserva un intervallo con cambio di segno e garantisce la riduzione della sua ampiezza. I metodi aperti sostituiscono invece localmente f con un modello semplice: una retta tangente, una secante o una parabola. La rapidità locale può migliorare di molto, ma senza salvaguardie non è garantita la convergenza globale.

20.6 Metodo di Newton-Raphson

20.6.1 Derivazione e interpretazione geometrica

Sia $f : [a, b] \to \mathbb{R}$ continua e differenziabile e sia x_k l'approssimazione corrente della radice. Ponendo $s = x_{k+1} - x_k$, lo sviluppo di Taylor al primo ordine fornisce

$$f(x_k + s) = f(x_k) + f'(x_k)s + O(s^2).$$

Trascurando il resto e imponendo lo zero del modello lineare, $f(x_k) + f'(x_k)s = 0$, si ottiene $s = -f(x_k)/f'(x_k)$ e quindi lo schema iterativo

$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad k = 0, 1, 2, \ldots} \tag{20.3}$$

noto come **metodo di Newton-Raphson**.

La stessa formula ha una lettura geometrica immediata: il modello lineare non è altro che la retta tangente al grafico di f nel punto $(x_k, f(x_k))$,

$$y = f(x_k) + (x - x_k)f'(x_k), \tag{20.4}$$

e x_{k+1} è l'ascissa in cui tale tangente incontra l'asse x (Figura 20.1). Ogni iterazione sostituisce dunque la curva con la sua tangente e assume come nuova stima lo zero della tangente.

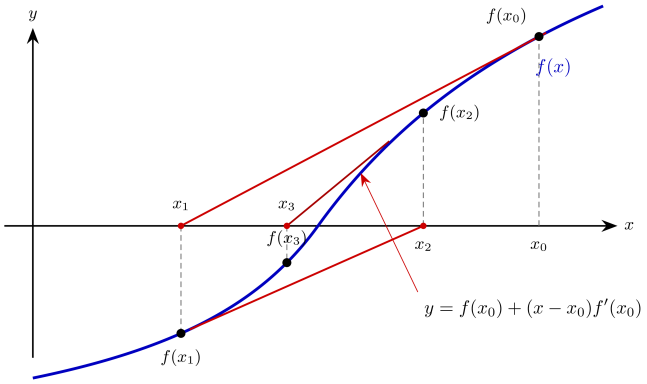


Figure 20.1: Costruzione grafica del metodo di Newton-Raphson

Lo schema richiede $f'(x_k) \neq 0$ e un punto iniziale adeguato: una derivata piccola produce passi molto grandi e può far divergere il procedimento (Sezione 20.6.4). Il costo è di **due valutazioni per iterazione** (f e f').

20.6.2 Convergenza locale

Theorem 20.2 — Teorema: convergenza del metodo di Newton per radici semplici. Sia $f \in C^2([a, b])$ e α una radice semplice di $f(x)$, cioè tale che $f'(\alpha) \neq 0$. Allora esiste $\rho > 0$ tale che, se $x_0 \in (\alpha - \rho, \alpha + \rho)$, la successione generata dal metodo di Newton-Raphson 20.3 soddisfa:

- $x_k \in (\alpha - \rho, \alpha + \rho)$ per $k = 1, 2, 3, \ldots$;
- la convergenza è **quadratica**, ossia $|e_{k+1}| \leq C\,|e_k|^2$ per $k = 0, 1, 2, \ldots$, dove C è una costante positiva.

Proof. Consideriamo lo sviluppo di $f(x)$ attorno a x_k, valutato nella radice α, tramite la serie di Taylor[3]:

$$f(\alpha) = f(x_k) + f'(x_k)(\alpha - x_k) + \frac{f''(\eta_k)}{2}(x_k - \alpha)^2,$$

dove $\eta_k \in I[x_k, \alpha]$. Assumendo $f'(x_k) \neq 0$ e ricordando che $f(\alpha) = 0$, dividendo per $f'(x_k)$ possiamo scrivere

$$0 = \frac{f(x_k)}{f'(x_k)} + \alpha - x_k + \frac{f''(\eta_k)}{2f'(x_k)}(x_k - \alpha)^2. \tag{20.5}$$

Poiché dallo schema 20.3 segue $\dfrac{f(x_k)}{f'(x_k)} = x_k - x_{k+1}$, sostituendo nella 20.5 i primi due addendi si semplificano e si ottiene

$$e_{k+1} = x_{k+1} - \alpha = \frac{f''(\eta_k)}{2f'(x_k)}\, e_k^2. \tag{20.6}$$

Prendendo il valore assoluto:

$$|e_{k+1}| \leq \frac{M_2}{2\,|f'(x_k)|}\,|e_k|^2\,, \qquad M_2 = \sup_{\eta \in (\alpha-\rho, \alpha+\rho)} \left|f''(\eta)\right|.$$

Poiché α è una radice semplice e f' è continua, esiste $\rho > 0$ tale che $2\,|f'(x)| > |f'(\alpha)|$ per ogni $x \in (\alpha - \rho, \alpha + \rho)$. Posto

$$C = \frac{M_2}{|f'(\alpha)|},$$

si ha dunque

$$|e_{k+1}| \leq C\,|e_k|^2\,. \tag{20.7}$$

Riducendo eventualmente ρ possiamo supporre $C\rho < 1$: infatti M_2, e quindi C, non cresce al diminuire di ρ, mentre il prodotto $C\rho$ tende a zero. Allora, se $x_k \in (\alpha - \rho, \alpha + \rho)$,

$$|e_{k+1}| \leq C\rho\,|e_k| < |e_k| < \rho,$$

cosicché anche x_{k+1} appartiene all'intorno. Per induzione la successione $\{x_k\}$ vi rimane e, poiché l'errore è ridotto a ogni passo almeno del fattore $C\rho < 1$, converge ad α. La 20.7 mostra infine che la convergenza è quadratica. ∎

Dalla 20.6 si ricava anche la costante asintotica esatta:

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k^2} = \frac{f''(\alpha)}{2f'(\alpha)},$$

che si annulla se $f''(\alpha) = 0$: in tal caso la convergenza è addirittura **cubica**. Accade quando la radice coincide con un punto di flesso, come per $f(x) = \sin x$ o $f(x) = \arctan x$ con $\alpha = 0$, oppure per $f(x) = x^3 + x$.

[3] Brook Taylor (1685-1731)

20.6.3 Convergenza globale per funzioni convesse

La convergenza del Teorema 20.2 è soltanto *locale*. Esiste però una classe di problemi in cui la convergenza è garantita a partire da un estremo dell'intervallo: è la classica **condizione di Fourier**.

Theorem 20.3 — Teorema: convergenza monotona di Newton per funzioni convesse. Sia $f \in C^2([a,b])$ con $f(a) < 0 < f(b)$, $f'(x) > 0$ e $f''(x) \geq 0$ per ogni $x \in [a,b]$. Allora f ha un'unica radice $\alpha \in (a,b)$ e la successione di Newton generata da $x_0 = b$ è monotona decrescente, rimane in $[\alpha, b]$ e converge ad α.

Proof. L'unicità segue dalla stretta monotonia di f. Per la convessità, il grafico di f sta sopra ogni sua tangente:

$$f(x) \geq f(x_k) + f'(x_k)(x - x_k), \qquad \forall x \in [a,b].$$

Procediamo per induzione supponendo $f(x_k) \geq 0$, vero per $k = 0$ essendo $x_0 = b$. Valutando la disuguaglianza in $x = x_{k+1}$, e osservando che per costruzione il membro destro si annulla proprio in x_{k+1}, si ottiene $f(x_{k+1}) \geq 0$; poiché f è crescente e $f(\alpha) = 0$, ciò implica $x_{k+1} \geq \alpha$. Inoltre

$$x_{k+1} - x_k = -\frac{f(x_k)}{f'(x_k)} \leq 0,$$

perché $f(x_k) \geq 0$ e $f'(x_k) > 0$. La successione è dunque decrescente e limitata inferiormente da α, quindi converge a un limite $\mu \geq \alpha$. Passando al limite nello schema si ottiene $f(\mu)/f'(\mu) = 0$, cioè $f(\mu) = 0$ e quindi $\mu = \alpha$. ∎

In pratica la condizione si riassume così: **si parte dall'estremo in cui f e f'' hanno lo stesso segno**. Le altre tre combinazioni di segni di f' e f'' si riconducono a questa cambiando x in $-x$ oppure f in $-f$. Si noti la simmetria con la dimostrazione della convergenza della *regula falsi* (Teorema 20.5), che usa la stessa ipotesi di convessità per ottenere una successione monotona crescente dall'altro estremo: sotto le ipotesi di Fourier, Newton e *regula falsi* partendo dai due estremi stringono la radice da entrambi i lati, fornendo un bracket a costo nullo.

20.6.4 Quando Newton fallisce

Fuori dall'intorno di convergenza il metodo può comportarsi in modo arbitrariamente cattivo. I modi di fallimento tipici sono quattro.

1. Derivata quasi nulla. Se $f'(x_k) \approx 0$ il passo $-f(x_k)/f'(x_k)$ è enorme e l'iterata viene proiettata lontanissimo dalla radice, spesso fuori dal dominio di definizione di f.

2. Ciclo. Per $f(x) = \arctan x$, con radice $\alpha = 0$, lo schema è $x_{k+1} = x_k - (1 + x_k^2)\arctan x_k$. Esiste un valore critico $x^* \approx 1.3917452002707$ per cui $x_1 = -x_0$: il metodo entra in un ciclo di periodo due e non converge mai.

3. Divergenza. Per $|x_0| > x^*$ nell'esempio precedente le iterate divergono con oscillazioni di ampiezza crescente. Un caso ancora più netto è $f(x) = \operatorname{sign}(x)\,|x|^{1/3}$, che ha la sola radice $\alpha = 0$ e per cui $f(x)/f'(x) = 3x$, cioè $x_{k+1} = -2x_k$: il metodo diverge **per qualunque** $x_0 \neq 0$.

4. Radice multipla. La convergenza degrada da quadratica a lineare con fattore $1 - 1/m$ (cfr. Sezione 21.3).

I primi tre modi di fallimento sono ripresi e analizzati in dettaglio in Tre patologie del metodo di Newton, dove si vede che ciclo e divergenza sono due casi della stessa famiglia.

```r
newton_arctan <- function(x0, n = 6) {
  x <- x0
  for (k in 1:n) x <- c(x, x[k] - atan(x[k])*(1 + x[k]^2))
```

```
  x
}
rbind(
  `x0 = 1.0 (converge)` = newton_arctan(1.0),
  `x0 = 1.3917452 (ciclo)` = newton_arctan(1.3917452002707),
  `x0 = 1.5 (diverge)` = newton_arctan(1.5)
)
```

```
                          [,1]       [,2]      [,3]         [,4]         [,5]
x0 = 1.0 (converge)    1.000000 -0.5707963 0.1168599 -0.001061022 7.963096e-10
x0 = 1.3917452 (ciclo) 1.391745 -1.3917452 1.3917452 -1.391745200 1.391745e+00
x0 = 1.5 (diverge)     1.500000 -1.6940796 2.3211270 -5.114087837 3.229568e+01
                             [,6]         [,7]
x0 = 1.0 (converge)      0.000000 0.000000e+00
x0 = 1.3917452 (ciclo)  -1.391745 1.391745e+00
x0 = 1.5 (diverge)     -1575.316951 3.894976e+06
```

Il rimedio non è scegliere meglio il punto iniziale — cosa in generale impossibile — ma **imporre una salvaguardia**: o mantenendo un bracket e rifiutando i passi che ne escono (metodi ibridi, cfr. il capitolo successivo), oppure accorciando il passo (*damping*) finché $|f|$ non diminuisce.

20.7 Metodo delle secanti

Il difetto del metodo di Newton è la necessità di f'. Sostituendo la derivata con il rapporto incrementale costruito sulle ultime due iterate,

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}},$$

si ottiene il **metodo delle secanti**: geometricamente, la nuova iterata è lo zero della retta *secante* per $(x_{k-1}, f(x_{k-1}))$ e $(x_k, f(x_k))$ anziché della tangente. Lo schema si scrive nella forma

$$\boxed{x_{k+1} = \frac{x_{k-1} f(x_k) - x_k f(x_{k-1})}{f(x_k) - f(x_{k-1})}, \qquad k = 1, 2, \ldots} \tag{20.8}$$

A differenza della *regula falsi* (Sezione 20.8), che usa anch'essa una secante ma ne sceglie gli estremi in modo da conservare il cambio di segno, qui si usano sempre le **ultime due iterate**, qualunque sia il segno di f in esse. Si guadagna in velocità e si risparmia il controllo dei segni, ma si perde la garanzia di convergenza.

Il costo è di **una sola valutazione di f per iterazione** (il valore $f(x_{k-1})$ è già disponibile) e nessuna valutazione di derivate: è questa la ragione per cui, a parità di valutazioni, le secanti sono competitive con Newton (Sezione 21.1).

Due avvertenze pratiche:

- quando si raggiunge la convergenza si ha $f(x_k) \approx f(x_{k-1})$, e il denominatore della 20.8 è soggetto a **cancellazione numerica**. Nella *regula falsi* questo non si verifica, poiché è garantito che $f(x_k) f(a_k) < 0$;
- la forma 20.8 è la più compatta, ma vicino alla radice il suo numeratore è la differenza di due prodotti quasi uguali. Conviene allora usare la forma **incrementale** equivalente

$$x_{k+1} = x_k - f(x_k) \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})},$$

che calcola una piccola correzione da sommare a x_k anziché ricostruire da zero un numero di ordine α: è la forma usata in tutte le implementazioni.

20.7.1 Convergenza locale

Se x_k non è sufficientemente vicino alla radice, non è garantito che il metodo converga. Esaminiamo dunque la **convergenza locale** dello schema. Sia α una radice di $f(x)$. Sottraendo α da entrambi i membri della 20.8 otteniamo:

$$\begin{aligned} x_{k+1} - \alpha &= \frac{x_{k-1}f(x_k) - x_k f(x_{k-1})}{f(x_k) - f(x_{k-1})} - \alpha \\ &= \frac{x_{k-1}f(x_k) - x_k f(x_{k-1}) - \alpha(f(x_k) - f(x_{k-1}))}{f(x_k) - f(x_{k-1})} \\ &= \frac{(x_{k-1} - \alpha)f(x_k) - (x_k - \alpha)f(x_{k-1})}{f(x_k) - f(x_{k-1})} \\ &= (x_k - \alpha)(x_{k-1} - \alpha)\frac{\dfrac{f(x_k)}{x_k - \alpha} - \dfrac{f(x_{k-1})}{x_{k-1} - \alpha}}{f(x_k) - f(x_{k-1})}. \end{aligned}$$

Inoltre, poiché $f(\alpha) = 0$, possiamo scrivere:

$$\begin{aligned} \frac{\dfrac{f(x_k)}{x_k - \alpha} - \dfrac{f(x_{k-1})}{x_{k-1} - \alpha}}{f(x_k) - f(x_{k-1})} &= \frac{\dfrac{f(x_k) - f(\alpha)}{x_k - \alpha} - \dfrac{f(x_{k-1}) - f(\alpha)}{x_{k-1} - \alpha}}{f(x_k) - f(x_{k-1})} \\ &= \frac{\dfrac{f(x_k) - f(\alpha)}{x_k - \alpha} - \dfrac{f(x_{k-1}) - f(\alpha)}{x_{k-1} - \alpha}}{x_k - x_{k-1}} \left(\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}\right)^{-1}, \end{aligned} \tag{20.9}$$

dove possiamo applicare il teorema di **Lagrange**[4], che afferma che:

$$\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} = f'(\eta_k), \qquad \eta_k \in I[x_{k-1}, x_k]. \tag{20.10}$$

Lemma 20.1 — Lemma: differenza divisa seconda. Se $f \in C^2$ e $h + k \neq 0$, allora vale la seguente uguaglianza:

$$\frac{\dfrac{f(\alpha + h) - f(\alpha)}{h} - \dfrac{f(\alpha - k) - f(\alpha)}{-k}}{h + k} = \frac{1}{2}f''(\zeta),$$

dove $\zeta \in I[\alpha - k, \alpha, \alpha + h]$.

Proof. Consideriamo la funzione

$$G(t) := \frac{\dfrac{f(\alpha + th) - f(\alpha)}{h} - \dfrac{f(\alpha - tk) - f(\alpha)}{-k}}{h + k}.$$

La funzione

$$H(t) := G(t) - G(1)t^2$$

si annulla per $t = 0$ e $t = 1$. Pertanto, secondo il teorema di **Rolle**[5], esiste un punto $\theta \in (0, 1)$ tale che $H'(\theta) = 0$.

Calcoliamo $H'(t)$:

[4] Joseph-Louis Lagrange (1736-1813)
[5] Michel Rolle (1652-1719)

$$H'(t) = G'(t) - 2G(1)t, \qquad G'(t) = \frac{f'(\alpha + th) - f'(\alpha - tk)}{h + k}.$$

Valutando in $t = \theta$, otteniamo:

$$0 = G'(\theta) - 2G(1)\theta,$$
$$G(1) = \frac{1}{2\theta}G'(\theta) = \frac{f'(\alpha + \theta h) - f'(\alpha - \theta k)}{2\theta(h + k)} = \frac{1}{2}f''(\zeta),$$

dove nell'ultimo passaggio si è applicato il teorema di Lagrange a f' sull'intervallo di estremi $\alpha - \theta k$ e $\alpha + \theta h$. ∎

Ponendo $h = x_k - \alpha$ e $k' = \alpha - x_{k-1}$ nel Lemma 20.1, per cui $\alpha + h = x_k$, $\alpha - k' = x_{k-1}$, $-k' = x_{k-1} - \alpha$ e $h + k' = x_k - x_{k-1}$, otteniamo

$$\frac{\dfrac{f(x_k) - f(\alpha)}{x_k - \alpha} - \dfrac{f(x_{k-1}) - f(\alpha)}{x_{k-1} - \alpha}}{x_k - x_{k-1}} = \frac{1}{2}f''(\zeta_k), \tag{20.11}$$

e sostituendo 20.9 ed 20.10 in 20.11 si ricava la relazione fondamentale

$$e_{k+1} = e_k\, e_{k-1}\, \frac{f''(\zeta_k)}{2f'(\eta_k)}, \qquad \zeta_k, \eta_k \in I[x_{k-1}, x_k, \alpha], \tag{20.12}$$

del tutto analoga alla 20.6 di Newton, con e_k^2 sostituito da $e_k e_{k-1}$. Questa relazione permette di dimostrare il seguente teorema.

Theorem 20.4 — Teorema: convergenza del metodo delle secanti. Se α è uno zero semplice di $f \in C^2([a, b])$ (cioè $f'(\alpha) \neq 0$), allora esiste un intervallo $J = (\alpha - \rho, \alpha + \rho)$ tale che, per ogni scelta di $x_0, x_1 \in J$, il metodo delle secanti converge. Inoltre, si ha:

$$|e_k| \leq Cq^{p^k}, \quad 0 < q < 1, \quad p = \frac{1 + \sqrt{5}}{2} \approx 1.618034,$$

cioè la convergenza è R-convergente di ordine p (il rapporto aureo).

Proof. Poiché α è uno zero semplice e $f'(x)$ è continua, esiste $\rho > 0$ tale che:

$$|f'(x)| > \frac{1}{2}|f'(\alpha)|, \quad \forall x \in (\alpha - \rho, \alpha + \rho),$$

e, per la continuità di f'', esiste M_2 con $|f''(x)| \leq M_2$ sullo stesso intorno.

Definiamo $J = (\alpha - \rho, \alpha + \rho)$ e poniamo $M = \dfrac{M_2}{|f'(\alpha)|}$. Allora, per ogni $\eta, \zeta \in J$:

$$\left|\frac{1}{2}\frac{f''(\zeta)}{f'(\eta)}\right| \leq \left|\frac{f''(\zeta)}{f'(\alpha)}\right| \leq M.$$

Supponendo che $\rho M < 1$ (altrimenti basta ridurre ρ), otteniamo che $x_k \in J$ per ogni k. Infatti, dalla relazione 20.12 segue

$$|e_{k+1}| \leq |e_k|\,|e_{k-1}|\,M \leq \rho^2 M < \rho, \qquad \Longrightarrow \qquad x_{k+1} \in J.$$

Definiamo $\epsilon_k = M\,|e_k|$; moltiplicando per M la relazione precedente otteniamo la ricorrenza

$$\epsilon_{k+1} \leq \epsilon_k \epsilon_{k-1}.$$

Poniamo $E_0 \geq \epsilon_0$, $E_1 \geq \epsilon_1$ e $E_{k+1} = E_k E_{k-1}$. Allora $\epsilon_k \leq E_k$ per ogni k: infatti, se ciò vale fino a k,

$$\epsilon_{k+1} \le \epsilon_k \epsilon_{k-1} \le E_k E_{k-1} = E_{k+1}.$$

Ponendo $E_k = \exp(cz^k)$, otteniamo:

$$\exp(cz^{k+1}) = \exp(cz^k)\exp(cz^{k-1}) = \exp(c(z^k + z^{k-1})).$$

Prendendo il logaritmo e dividendo per cz^{k-1}, arriviamo all'equazione caratteristica:

$$z^2 = z + 1, \quad \Longrightarrow \quad z_{1,2} = \frac{1 \pm \sqrt{5}}{2}.$$

Scegliendo la radice positiva $p = \dfrac{1+\sqrt{5}}{2}$, la soluzione particolare della ricorrenza $E_{k+1} = E_k E_{k-1}$ è

$$E_0 = \exp(c), \quad E_1 = \exp(cp), \quad E_k = \exp(cp^k).$$

Osserviamo ora che

$$\epsilon_k = M\,|e_k| < \rho M < 1,$$

quindi $\log(\epsilon_0)$ e $\log(\epsilon_1)$ sono entrambi negativi ed è possibile scegliere una costante $c < 0$ che soddisfi le condizioni

$$\begin{aligned} \epsilon_0 \le E_0 = \exp(c), &\qquad c \ge \log(\epsilon_0), \\ \epsilon_1 \le E_1 = \exp(cp), &\qquad c \ge \frac{\log(\epsilon_1)}{p}. \end{aligned}$$

Posto $q = \exp(c) < 1$ segue che:

$$\epsilon_k \le \exp(cp^k), \quad \Longrightarrow \quad M\,|e_k| \le q^{p^k},$$

cioè la tesi con $C = 1/M$. ∎

20.7.2 Variante con differenza finita di sicurezza

La secante approssima la derivata di Newton con

$$D_k = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}, \qquad x_{k+1} = x_k - \frac{f(x_k)}{D_k}.$$

Se $|x_k - x_{k-1}| < h$, il denominatore può essere numericamente sfavorevole. Si può allora usare una differenza in avanti, con passo $h > 0$:

$$D_k = \begin{cases} \dfrac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}, & |x_k - x_{k-1}| \ge h, \\[2ex] \dfrac{f(x_k + h) - f(x_k)}{h}, & |x_k - x_{k-1}| < h. \end{cases}$$

Il secondo ramo richiede una valutazione aggiuntiva di f. Per $f \in C^2$, la differenza in avanti ha errore di troncamento $O(h)$; in aritmetica floating point il contributo dell'arrotondamento cresce come $O(\varepsilon_{\text{mach}}/h)$, e il bilanciamento dei due contributi porta alla scelta pratica $h \approx \sqrt{\varepsilon_{\text{mach}}}\max(1, |x_k|)$, adattata alla scala del problema. Se D_k è nullo o troppo piccolo, occorre una salvaguardia. Questa variante non conserva automaticamente l'ordine superlineare della secante quando h resta fisso.

20.8 Metodo delle false posizioni (*regula falsi*)

Sia data una funzione $f : [a, b] \to \mathbb{R}$ continua nell'intervallo $[a, b]$ e tale che $f(a)f(b) < 0$. In queste condizioni, il Teorema 20.1 garantisce che esiste almeno uno zero di $f(x)$ nell'intervallo $[a, b]$.

Per stimare la posizione della radice, costruiamo la retta che interpola i punti:

$$\begin{pmatrix} a \\ f(a) \end{pmatrix}, \qquad \begin{pmatrix} b \\ f(b) \end{pmatrix},$$

ossia l'equazione della retta diventa:

$$y = f(a) + (x - a)\frac{f(b) - f(a)}{b - a}, \tag{20.13}$$

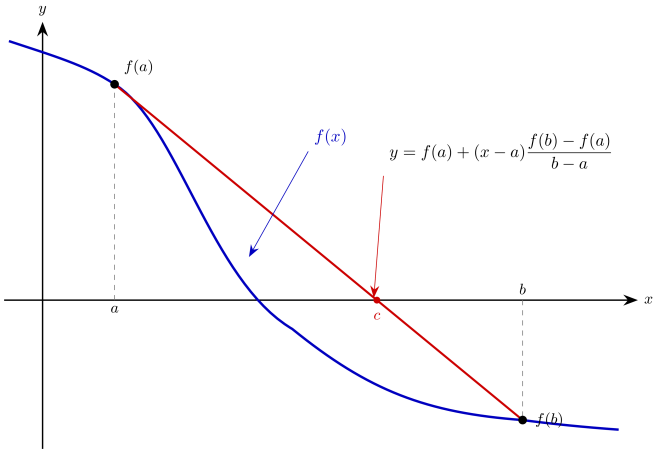


Figure 20.2: Costruzione della *regula falsi*

Lo zero della retta interpolante, ottenuto risolvendo l'20.13, fornisce la seguente stima della radice $\hat{x}$ della funzione $f(x)$:

$$0 = f(a) + (\hat{x} - a)\frac{f(b) - f(a)}{b - a}, \qquad \Rightarrow \qquad \hat{x} = \frac{af(b) - bf(a)}{f(b) - f(a)}.$$

A questo punto, possiamo applicare un procedimento simile a quello del metodo di bisezione, scegliendo come nuovo intervallo $[a, \hat{x}]$ o $[\hat{x}, b]$, a seconda del segno di $f(\hat{x})$. Tuttavia, la *regula falsi* non garantisce che gli intervalli costruiti diventino sempre più piccoli, come mostrato in Figura 20.3.

Infatti, il metodo genera una coppia di punti x_k e a_k, dove x_k è l'approssimazione corrente della radice e a_k è tale che $f(x_k)f(a_k) < 0$; tipicamente solo x_k converge alla radice, mentre a_k resta congelato in uno degli estremi iniziali.

Il metodo delle false posizioni può essere riassunto nel seguente algoritmo.

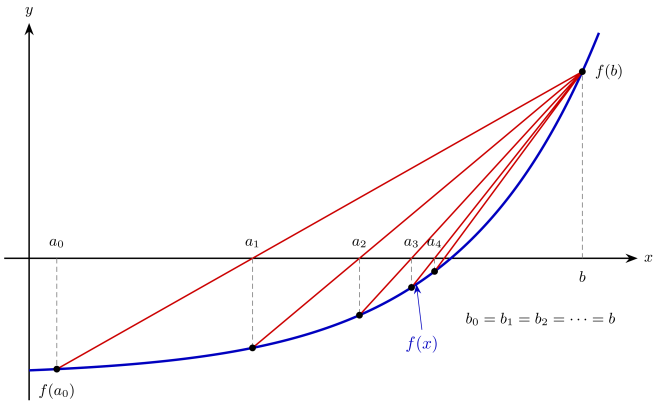


Figure 20.3: Andamento della *regula falsi*

Regula falsi

Dati: a, b con $f(a)f(b) < 0$; tolleranza τ
Restituisce: approssimazione della radice

```
x ← b
mentre |f(x)| ≥ τ fai
    x̄ ← (a f(x) − x f(a)) / (f(x) − f(a))        ▷ secante fra a e x
    se sign f(x) ≠ sign f(x̄) allora               ▷ la radice sta fra x e x̄
        a ← x
    x ← x̄                                        ▷ resta vero f(x)f(a) < 0
restituisci x
```

Si noti che il criterio d'arresto è qui basato sul residuo, con i limiti discussi in Sezione 20.4. Poiché l'ampiezza $|x_k - a_k|$ non tende a zero, non è utilizzabile il test sul bracket.

20.8.1 Convergenza della *regula falsi*

Per discutere la convergenza della *regula falsi*, assumiamo che la funzione $f(x)$ abbia derivata seconda continua e consideriamo le seguenti ipotesi:

a) $x_0 < a_0$;
b) $f(x_0) < 0$ e $f(a_0) > 0$;
c) $f''(x) \geq 0$ per ogni $x \in [x_0, a_0]$.

Con queste condizioni, è possibile dimostrare il seguente teorema.

Theorem 20.5 — Teorema: convergenza della *regula falsi*. Sotto le ipotesi a–c, la *regula falsi* converge, e inoltre:

$$x_k \leq x_{k+1} < a_k = a_0, \qquad \text{per } k = 1, 2, \ldots.$$

Proof. Osserviamo innanzitutto che l'ipotesi $f''(x) \geq 0$ in $[x_0, a_0]$ implica che la funzione $f(x)$ è convessa su $[x_0, a_0]$, ovvero:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y), \qquad \forall x, y \in [x_0, a_0],\ t \in [0, 1].$$

Inoltre, poiché $f(x_0) < 0$ e $f(a_0) > 0$, esiste almeno una radice nell'intervallo $[x_0, a_0]$. Definiamo:

$$\eta = \sup\{x \in [x_0, a_0] \,|\, f(x) < 0\}.$$

Dalla continuità di $f(x)$ segue che $f(\eta) = 0$. Ora, calcoliamo x_1:

$$x_1 = \frac{a_0 f(x_0) - x_0 f(a_0)}{f(x_0) - f(a_0)} = x_0 + (a_0 - x_0)\frac{f(x_0)}{f(x_0) - f(a_0)}.$$

Poiché $f(x_0) < 0$, $f(a_0) > 0$ e $x_0 < a_0$, segue che $x_1 > x_0$. Dalla convessità di $f(x)$, possiamo affermare che:

$$\begin{aligned} f(x_1) &= f\left(x_0 \frac{f(a_0)}{f(a_0) - f(x_0)} + a_0\left(1 - \frac{f(a_0)}{f(a_0) - f(x_0)}\right)\right), \\ &\leq \frac{f(a_0)}{f(a_0) - f(x_0)} f(x_0) + \left(1 - \frac{f(a_0)}{f(a_0) - f(x_0)}\right) f(a_0) = 0, \end{aligned}$$

e quindi $x_1 \leq \eta$. Poiché $f(x_1) \leq 0$ ha lo stesso segno di $f(x_0)$, l'algoritmo lascia $a_1 = a_0$. Procedendo in modo analogo, otteniamo:

$$x_0 \leq x_1 \leq x_2 \leq \ldots \leq x_k \leq \eta < a_0 = a_1 = \cdots = a_k.$$

Di conseguenza, x_k è una successione monotona crescente e limitata, dunque convergente. Indichiamo il limite con μ, per cui

$$\mu = \lim_{k\to\infty} x_k \leq \eta.$$

Passando al limite nello schema deve valere:

$$\mu = \frac{a_0 f(\mu) - \mu f(a_0)}{f(\mu) - f(a_0)}, \qquad \Longrightarrow \qquad (\mu - a_0) f(\mu) = 0.$$

Poiché $\mu - a_0 \leq \eta - a_0 < 0$, ne segue che $f(\mu) = 0$, dimostrando così la convergenza del metodo. ∎

Poiché l'estremo a_k resta congelato in a_0, lo schema è asintoticamente equivalente al metodo delle corde con pendenza $m = \big(f(a_0) - f(\alpha)\big)/(a_0 - \alpha)$: la convergenza della *regula falsi* è dunque in generale soltanto **lineare**, con fattore

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k} = 1 - \frac{f'(\alpha)(a_0 - \alpha)}{f(a_0)},$$

che può essere molto vicino a 1 se f è fortemente convessa. Le varianti *Illinois* e *Pegasus* ovviano al problema dimezzando (o riscalando) il valore $f(a_k)$ dell'estremo congelato ogni volta che esso non viene aggiornato per due passi consecutivi, recuperando convergenza superlineare senza perdere il bracket.

20.9 Metodo delle corde (derivata congelata)

A differenza della falsa posizione, che aggiorna gli estremi di un intervallo, il **metodo delle corde** nel senso di *Newton con pendenza congelata* fissa una derivata $m = f'(x_0) \neq 0$ e ripete

$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{m}}. \tag{20.14}$$

È l'intersezione con l'asse x della retta di pendenza costante m passante per $(x_k, f(x_k))$. Con $e_k = x_k - \alpha$, $f(\alpha) = 0$ e $f'(\alpha) \neq 0$,

$$e_{k+1} = \left(1 - \frac{f'(\alpha)}{m}\right) e_k + O(e_k^2).$$

La convergenza locale è lineare se $|1 - f'(\alpha)/m| < 1$; scegliendo $m = f'(\alpha)$ si annulla il termine lineare. Il metodo è lento ma estremamente economico: una sola valutazione di f per iterazione e **una sola** valutazione di f' in tutto. È il prototipo dei metodi di Newton *inesatti*, in cui la derivata (o la matrice jacobiana, nel caso di sistemi) viene aggiornata solo ogni tanto.

20.10 Metodo di Müller: interpolazione parabolica su tre punti

Il metodo di Müller[6] interpola tre coppie $(x_j, f(x_j))$ con un polinomio quadratico e ne seleziona uno zero. Non richiede derivate ed è distinto dal metodo della **parabola tangente di Taylor** descritto in Sezione 20.11. La convergenza locale generica ha ordine circa 1.84, la radice reale di $z^3 = z^2 + z + 1$.

20.10.1 Costruzione del metodo

Il metodo parte da tre punti distinti x_{k-2}, x_{k-1}, x_k, e costruisce una parabola che interpola la funzione $f(x)$ in questi punti. La parabola ha la seguente forma generale:

$$P(x) = a(x - x_k)^2 + b(x - x_k) + c$$

I coefficienti a, b e c sono determinati imponendo che la parabola passi attraverso i punti $(x_{k-2}, f(x_{k-2}))$, $(x_{k-1}, f(x_{k-1}))$ e $(x_k, f(x_k))$. Posto

$$h_1 = x_{k-1} - x_{k-2}, \qquad h_2 = x_k - x_{k-1},$$
$$\delta_1 = \frac{f(x_{k-1}) - f(x_{k-2})}{h_1}, \qquad \delta_2 = \frac{f(x_k) - f(x_{k-1})}{h_2},$$

si ottengono in forma chiusa

$$a = \frac{\delta_2 - \delta_1}{h_2 + h_1}, \qquad b = a\,h_2 + \delta_2, \qquad c = f(x_k).$$

La nuova iterata x_{k+1} è lo zero di $P(x)$ più vicino a x_k. Scrivendo la formula risolutiva in forma razionalizzata, per evitare la cancellazione numerica, si ottiene

$$\boxed{x_{k+1} = x_k - \frac{2c}{b \pm \sqrt{b^2 - 4ac}}} \tag{20.15}$$

dove il segno va scelto in modo da rendere **massimo** il modulo del denominatore. Se il radicando è negativo la radice quadrata è complessa e il metodo produce iterate complesse: questa è una delle ragioni per cui il metodo è usato nella ricerca delle radici complesse dei polinomi, dove è un ingrediente classico degli algoritmi di *deflazione*.

Avvertenze: se i tre punti sono troppo vicini tra loro, i rapporti δ_1, δ_2 e la loro differenza sono soggetti a cancellazione e il metodo diventa instabile; inoltre se $a = 0$ la parabola degenera in una retta e la 20.15 va sostituita dal passo delle secanti.

20.10.2 Esempio numerico

Cerchiamo uno zero di $f(x) = x^3 - 2x - 5$ (la cubica di Wallis, la cui radice reale vale $\alpha = 2.0945514815423266\ldots$) partendo da tre valori iniziali.

[6] David Eugene Muller (1924-2008)

```r
# Esempio in R
f <- function(x) x^3 - 2*x - 5

# Inizializzazione dei punti
x0 <- 3; x1 <- 4; x2 <- 2

# Iterazione del metodo di Muller
for (i in 1:5) {
  h1 <- x1 - x0
  h2 <- x2 - x1
  delta1 <- (f(x1) - f(x0)) / h1
  delta2 <- (f(x2) - f(x1)) / h2
  a <- (delta2 - delta1) / (h2 + h1)
  b <- a*h2 + delta2
  c <- f(x2)
  radice <- sqrt(b^2 - 4*a*c)

  # Scelta del segno per garantire stabilità
  if (abs(b + radice) > abs(b - radice)) {
    denominatore <- b + radice
  } else {
    denominatore <- b - radice
  }

  # Nuovo punto
  dx <- -2*c / denominatore
  x3 <- x2 + dx

  # Aggiornamento dei punti
  x0 <- x1
  x1 <- x2
  x2 <- x3

  print(x3) # Stampa dell'approssimazione della radice
}
```

```
[1] 2.111111
[1] 2.094815
[1] 2.094552
[1] 2.094551
[1] 2.094551
```

In questo esempio il discriminante $b^2 - 4ac$ resta positivo a ogni passo e le iterate restano reali. In generale non è così: se la parabola interpolante non interseca l'asse reale il radicando diventa negativo e occorre passare all'aritmetica complessa (in R basta `sqrt(as.complex(b^2-4*a*c))`), altrimenti `sqrt` restituisce `NaN` e l'iterazione si arresta.

20.11 Metodo della parabola tangente (Taylor quadratico)

Questa costruzione è comunemente chiamata **metodo di Newton di secondo ordine mediante approssimazione quadratica** o *quadratic Taylor method*; come osservato in Sezione 20.10 non va attribuita a Müller, che usa tre valori della funzione anziché le derivate. Si cerca lo zero del polinomio di Taylor di grado due tangente a f in x_k. La formula razionalizzata che ricaveremo seleziona il ramo che tende al passo di Newton quando $f'' \to 0$.

Consideriamo l'espansione di Taylor attorno alla radice α a partire da un punto x_0, con $h = \alpha - x_0$:

$$0 = f(\alpha) = f(x_0) + h f'(x_0) + \frac{h^2}{2} f''(x_0) + \mathcal{O}(h^3).$$

Trascurando il termine $\mathcal{O}(h^3)$, otteniamo la seguente condizione su h:

$$0 = f(x_0) + hf'(x_0) + \frac{h^2}{2} f''(x_0), \tag{20.16}$$

risolvendo per h, otteniamo:

$$h_{1,2} = \frac{-f'(x_0) \pm \sqrt{f'(x_0)^2 - 2f(x_0)f''(x_0)}}{f''(x_0)}. \tag{20.17}$$

Osserviamo che:

$$\begin{aligned} \sqrt{f'(x_0)^2 - 2f(x_0)f''(x_0)} &= \sqrt{f'(x_0)^2 \left(1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}\right)} \\ &= |f'(x_0)| \sqrt{1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}}. \end{aligned} \tag{20.18}$$

Poiché $|f'(x_0)| = \text{sign}(f'(x_0))f'(x_0)$, possiamo riscrivere l'equazione precedente come:

$$\sqrt{f'(x_0)^2 - 2f(x_0)f''(x_0)} = \text{sign}(f'(x_0))f'(x_0)\sqrt{1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}}. \tag{20.19}$$

Combinando l'20.17 con 20.19, otteniamo:

$$h_{1,2} = -\frac{f'(x_0)}{f''(x_0)} \left(1 \mp \text{sign}(f'(x_0))\sqrt{1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}}\right).$$

Scegliamo la soluzione che produce il valore assoluto di h più piccolo, il che corrisponde a $\mp\text{sign}(f'(x_0)) = -1$. Quindi:

$$h = -\frac{f'(x_0)}{f''(x_0)} \left(1 - \sqrt{1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}}\right). \tag{20.20}$$

Quando $f(x_0) \approx 0$, abbiamo $\sqrt{1 - 2f(x_0)f''(x_0)/f'(x_0)^2} \approx 1$, e la 20.20 calcola h come differenza di due quantità quasi uguali: è il caso tipico di cancellazione numerica. Per evitarla riscriviamo l'espressione usando l'identità:

$$1 - \sqrt{A} = \frac{1 - \sqrt{A}}{1 + \sqrt{A}}(1 + \sqrt{A}) = \frac{1 - A}{1 + \sqrt{A}}.$$

Applicandola otteniamo:

$$1 - \sqrt{1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}} = \frac{2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}}{1 + \sqrt{1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}}},$$

e sostituendo nella 20.20,

$$h = -\frac{1}{f'(x_0)} \frac{2f(x_0)}{1 + \sqrt{1 - 2\frac{f(x_0)f''(x_0)}{f'(x_0)^2}}}.$$

Il che porta al metodo numerico:

$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \left(\frac{2}{1 + \sqrt{1 - 2\dfrac{f(x_k) f''(x_k)}{f'(x_k)^2}}} \right)}. \tag{20.21}$$

Il fattore fra parentesi tende a 1 quando $f'' \to 0$: in tal caso si ritrova esattamente il passo di Newton. Su radici semplici e per $f \in C^3$ il metodo ha **ordine locale tre**, al prezzo di tre valutazioni per iterazione (f, f', f''). Si ribadisce che l'ordine ≈ 1.84 citato in Sezione 20.10 appartiene al metodo di Müller interpolatorio, non a questa costruzione.

20.12 Metodo di Halley

Il metodo presentato in 20.21 presenta una limitazione: se il termine sotto radice è negativo non possiamo applicarlo nel campo reale, poiché la parabola tangente non interseca l'asse x. Per ottenere un passo razionale senza radice quadrata, utilizziamo lo sviluppo di Taylor della funzione $g(z) = \sqrt{1+z}$:

$$g(z) = g(0) + z g'(0) + \mathcal{O}(z^2), \qquad g'(z) = \frac{1}{2\sqrt{1+z}},$$

da cui $g'(0) = 1/2$ e

$$\sqrt{1+z} = 1 + \frac{z}{2} + \mathcal{O}(z^2).$$

Da questo sviluppo otteniamo l'approssimazione:

$$\sqrt{1 - 2\frac{f(x_k) f''(x_k)}{f'(x_k)^2}} \approx 1 - \frac{f(x_k) f''(x_k)}{f'(x_k)^2}.$$

Sostituendo questa approssimazione nel passo della parabola tangente 20.21 e semplificando, giungiamo al seguente metodo numerico:

$$\boxed{x_{k+1} = x_k - \frac{f(x_k) f'(x_k)}{f'(x_k)^2 - \dfrac{1}{2} f(x_k) f''(x_k)}} \tag{20.22}$$

Questo è il metodo di Halley[7]. Rispetto alla parabola tangente il passo è sempre definito (salvo annullamento del denominatore) e non richiede radici quadrate, pur conservando l'ordine tre.

Vale il seguente teorema.

Theorem 20.6 — Teorema: convergenza del metodo di Halley. Sia $f \in C^3([a,b])$ e α una radice semplice di $f(x)$, cioè $f'(\alpha) \neq 0$. Se $x_0 \in (\alpha - \rho, \alpha + \rho)$ con ρ *sufficientemente piccolo*, allora la successione generata dallo schema 20.22 soddisfa:

1. $x_k \in (\alpha - \rho, \alpha + \rho)$ per $k = 1, 2, 3, \ldots$
2. $|e_{k+1}| \leq C\,|e_k|^3$ per $k = 0, 1, 2, \ldots$,

cioè la convergenza è **cubica**.

Proof. Sviluppando con la serie di Taylor con resto di Lagrange ai tre ordini successivi,

[7] Edmond Halley (1656-1742)

$$0 = f(\alpha) = f(x_k) + f'(\zeta_k)(\alpha - x_k),$$
$$0 = f(\alpha) = f(x_k) + f'(x_k)(\alpha - x_k) + f''(\omega_k)\frac{(x_k-\alpha)^2}{2},$$
$$0 = f(\alpha) = f(x_k) + f'(x_k)(\alpha - x_k) + f''(x_k)\frac{(x_k-\alpha)^2}{2} + f'''(\eta_k)\frac{(x_k-\alpha)^3}{6},$$

con $\zeta_k, \omega_k, \eta_k \in I[x_k, \alpha]$, e ricordando che $e_k = x_k - \alpha$,

$$\begin{aligned} f(x_k) &= f'(\zeta_k)e_k, \\ f(x_k) &= f'(x_k)e_k - f''(\omega_k)\frac{e_k^2}{2}, \\ f(x_k) &= f'(x_k)e_k - f''(x_k)\frac{e_k^2}{2} + f'''(\eta_k)\frac{e_k^3}{6}. \end{aligned} \tag{20.23}$$

I tre colori indicano in quale punto del calcolo seguente viene usata ciascuna delle tre espressioni. Utilizzando lo schema 20.22 si ottiene

$$\begin{aligned} e_{k+1} = x_{k+1} - \alpha &= x_k - \alpha - \frac{f(x_k)f'(x_k)}{f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)} \\ &= \frac{(x_k-\alpha)\left(f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)\right) - f(x_k)f'(x_k)}{f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)} \\ &= \frac{e_k\left(f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)\right) - f'(x_k)\left(f'(x_k)e_k - f''(x_k)\frac{e_k^2}{2} + f'''(\eta_k)\frac{e_k^3}{6}\right)}{f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)} \\ &= \frac{-\frac{e_k}{2}f(x_k)f''(x_k) + f'(x_k)f''(x_k)\frac{e_k^2}{2} - f'(x_k)f'''(\eta_k)\frac{e_k^3}{6}}{f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)} \\ &= \frac{-\frac{e_k}{2}\left(f'(x_k)e_k - f''(\omega_k)\frac{e_k^2}{2}\right)f''(x_k) + f'(x_k)f''(x_k)\frac{e_k^2}{2} - f'(x_k)f'''(\eta_k)\frac{e_k^3}{6}}{f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)} \\ &= \frac{\frac{1}{4}f''(x_k)f''(\omega_k) - \frac{1}{6}f'(x_k)f'''(\eta_k)}{f'(x_k)^2 - \frac{1}{2}f'(\zeta_k)f''(x_k)e_k}e_k^3 \end{aligned}$$

da cui, moltiplicando numeratore e denominatore per 12,

$$e_{k+1} = \frac{3f''(x_k)f''(\omega_k) - 2f'(x_k)f'''(\eta_k)}{2f'(x_k)^2 - f'(\zeta_k)f''(x_k)e_k}\frac{e_k^3}{6}. \tag{20.24}$$

Prendendo il valore assoluto di entrambi i membri e ponendo

$$M_1 = \sup_{\eta\in[a,b]}\left|f'(\eta)\right|, \qquad M_2 = \sup_{\eta\in[a,b]}\left|f''(\eta)\right|, \qquad M_3 = \sup_{\eta\in[a,b]}\left|f'''(\eta)\right|,$$
$$M = \frac{1}{6}\left[3M_2^2 + 2M_1M_3\right],$$

il numeratore della 20.24 è maggiorato da $6M$ e otteniamo

$$|e_{k+1}| \le |e_k|^3 \frac{M}{2f'(x_k)^2 - M_1 M_2 |e_k|}.$$

Poiché α è supposto zero semplice si ha $|f'(\alpha)| > 0$. Inoltre, per la continuità di f', esiste ρ tale che $2f'(x)^2 > \frac{3}{2}f'(\alpha)^2$ per ogni $x \in (\alpha-\rho, \alpha+\rho)$ e $M_1 M_2\, \rho < \frac{1}{2}f'(\alpha)^2$. Se $x_k \in (\alpha-\rho, \alpha+\rho)$, il denominatore è allora maggiore di $\frac{3}{2}f'(\alpha)^2 - \frac{1}{2}f'(\alpha)^2 = f'(\alpha)^2$ e quindi

$$|e_{k+1}| \le |e_k|^3 \frac{M}{2f'(x_k)^2 - M_1 M_2 |e_k|} \le |e_k|^3 \frac{M}{f'(\alpha)^2}. \tag{20.25}$$

Eventualmente riducendo ρ possiamo supporre $\rho^2 M / f'(\alpha)^2 < 1$, cosicché dalla 20.25 segue

$$|e_{k+1}| \le \rho^2 \frac{M}{f'(\alpha)^2} |e_k| < |e_k| < \rho,$$

cioè il punto 1:

$$|x_{k+1} - \alpha| \le \rho, \quad \Longrightarrow \quad x_{k+1} \in (\alpha-\rho, \alpha+\rho).$$

Ponendo infine $C = M/f'(\alpha)^2$ si trova verificato il punto 2. ∎

20.13 Confronto numerico dei metodi

Mettiamo alla prova i metodi visti finora sullo stesso problema: la radice reale di

$$f(x) = x^3 - 2x - 5, \qquad \alpha = 2.0945514815423266\ldots$$

Tutti i metodi con bracket partono da $[2, 3]$, quelli aperti da $x_0 = 3$ (e $x_0 = 2$, $x_1 = 3$ per le secanti).

```
alpha <- 2.0945514815423265148   # radice, nota ad alta precisione
f    <- function(x) x^3 - 2*x - 5
df   <- function(x) 3*x^2 - 2
ddf  <- function(x) 6*x
err  <- function(x) abs(x - alpha)

bisezione <- function(a, b, n) {
  e <- numeric(n)
  for (k in 1:n) {
    cm <- a + (b - a)/2
    if (sign(f(a)) != sign(f(cm))) b <- cm else a <- cm
    e[k] <- err(cm)
  }
  e
}

regula_falsi <- function(a, x, n) {
  e <- numeric(n)
  for (k in 1:n) {
    xh <- (a*f(x) - x*f(a)) / (f(x) - f(a))
    if (sign(f(x)) != sign(f(xh))) a <- x
    x <- xh
    e[k] <- err(x)
  }
  e
}

corde <- function(x, n) {
  m <- df(x); e <- numeric(n)
  for (k in 1:n) { x <- x - f(x)/m; e[k] <- err(x) }
```

```
  e
}

secanti <- function(x0, x1, n) {
  e <- numeric(n)
  for (k in 1:n) {
    x2 <- (x0*f(x1) - x1*f(x0)) / (f(x1) - f(x0))
    x0 <- x1; x1 <- x2
    e[k] <- err(x2)
  }
  e
}

newton <- function(x, n) {
  e <- numeric(n)
  for (k in 1:n) { x <- x - f(x)/df(x); e[k] <- err(x) }
  e
}

halley <- function(x, n) {
  e <- numeric(n)
  for (k in 1:n) {
    x <- x - f(x)*df(x) / (df(x)^2 - 0.5*f(x)*ddf(x))
    e[k] <- err(x)
  }
  e
}

muller <- function(x0, x1, x2, n) {
  e <- numeric(n)
  for (k in 1:n) {
    h1 <- x1 - x0; h2 <- x2 - x1
    if (h1 == 0 || h2 == 0) { e[k] <- err(x2); next }   # già convergito
    d1 <- (f(x1) - f(x0))/h1; d2 <- (f(x2) - f(x1))/h2
    a  <- (d2 - d1)/(h2 + h1); b <- a*h2 + d2; cc <- f(x2)
    disc <- b^2 - 4*a*cc
    r   <- if (disc >= 0) sqrt(disc) else 0
    den <- if (abs(b + r) > abs(b - r)) b + r else b - r
    x3  <- if (den != 0) x2 - 2*cc/den else x2
    x0 <- x1; x1 <- x2; x2 <- x3
    e[k] <- err(x3)
  }
  e
}

parabola <- function(x, n) {   # parabola tangente (Taylor quadratico)
  e <- numeric(n)
  for (k in 1:n) {
    A <- 1 - 2*f(x)*ddf(x)/df(x)^2
    x <- if (A >= 0) x - (f(x)/df(x))*(2/(1 + sqrt(A))) else x - f(x)/df(x)
    e[k] <- err(x)
  }
  e
}

n   <- 7
fmt <- function(v) formatC(v, format = "e", digits = 2)

knitr::kable(
  data.frame(k = 1:n,
             bisezione      = fmt(bisezione(2, 3, n)),
             `regula falsi` = fmt(regula_falsi(2, 3, n)),
```

```
            secanti        = fmt(secanti(2, 3, n)),
            Müller         = fmt(muller(2, 2.5, 3, n)),
            check.names    = FALSE),
  align = "r", caption = "Errore $\\lvert x_k-\\alpha\\rvert$: metodi che usano solo
      ↪  $f$")
```

Table 20.2: Errore $|x_k - \alpha|$: metodi che usano solo f

k	bisezione	regula falsi	secanti	Müller
1	4.05e-01	3.57e-02	3.57e-02	3.17e-03
2	1.55e-01	1.33e-02	1.33e-02	1.08e-04
3	3.04e-02	4.91e-03	2.73e-04	2.78e-08
4	3.21e-02	1.81e-03	2.05e-06	8.88e-16
5	8.01e-04	6.68e-04	3.15e-10	0.00e+00
6	1.48e-02	2.46e-04	8.88e-16	0.00e+00
7	7.01e-03	9.06e-05	0.00e+00	0.00e+00

```
knitr::kable(
  data.frame(k = 1:n,
             corde                = fmt(corde(3, n)),
             Newton               = fmt(newton(3, n)),
             `parabola tangente`  = fmt(parabola(3, n)),
             Halley               = fmt(halley(3, n)),
             check.names          = FALSE),
  align = "r", caption = "Errore $\\lvert x_k-\\alpha\\rvert$: metodi che usano le
      ↪  derivate")
```

Table 20.3: Errore $|x_k - \alpha|$: metodi che usano le derivate

k	corde	Newton	parabola tangente	Halley
1	2.65e-01	2.65e-01	9.46e-02	7.38e-02
2	1.28e-01	3.26e-02	7.59e-05	8.49e-05
3	6.69e-02	5.85e-04	3.91e-14	1.39e-13
4	3.59e-02	1.92e-07	0.00e+00	0.00e+00
5	1.95e-02	2.09e-14	0.00e+00	0.00e+00
6	1.07e-02	0.00e+00	0.00e+00	0.00e+00
7	5.90e-03	0.00e+00	0.00e+00	0.00e+00

La lettura delle tabelle conferma la teoria:

- **bisezione**: l'errore non decresce in modo monotono (il punto medio può capitare dalla parte sbagliata), ma è sempre maggiorato da $2^{-k}(b-a)$;
- **regula falsi** e **corde**: decrescita lineare, con un fattore costante per iterazione;
- **secanti**: superlineare, raggiunge la precisione di macchina alla sesta iterazione;
- **Müller**: superlineare con ordine più alto delle secanti, alla quarta — e con una sola valutazione per iterazione, come le secanti;
- **Newton**: quadratico, raddoppia le cifre esatte a ogni passo;
- **parabola tangente** e **Halley**: cubici, sono alla precisione di macchina già alla quarta iterazione.

Il confronto per **iterazione** però non è equo, perché il costo di un'iterazione non è lo stesso:

Table 20.4: Costo e ordine dei metodi; ν è il numero di valutazioni per iterazione

Metodo	f	f'	f''	ν	Ordine p	Efficienza $p^{1/\nu}$
bisezione	1	–	–	1	1 (fattore 1/2)	1
regula falsi	1	–	–	1	1	1
corde	1	–	–	1	1	1
secanti	1	–	–	1	1.618	1.618
Müller	1	–	–	1	1.839	**1.839**
Newton	1	1	–	2	2	1.414
parabola tangente	1	1	1	3	3	1.442
Halley	1	1	1	3	3	1.442

Per i metodi lineari l'indice $p^{1/\nu}$ vale 1 e non discrimina: fra di essi il confronto va fatto sul fattore di riduzione asintotico, che per la bisezione è $1/2$ e per *regula falsi* e corde dipende da f.

Fra i metodi superlineari, invece, l'ordinamento si ribalta rispetto a quello per iterazione: **i metodi senza derivate sono i più efficienti**, perché riutilizzano i valori già calcolati e spendono una sola valutazione per passo. Contando le valutazioni totali necessarie per raggiungere la precisione di macchina su questo problema si ottiene

$$\underbrace{3+4=7}_{\text{Müller}} \;<\; \underbrace{2+6=8}_{\text{secanti}} \;<\; \underbrace{6\times 2=12}_{\text{Newton}} \;=\; \underbrace{4\times 3=12}_{\text{Halley, par. tang.}},$$

dove i primi addendi di Müller e secanti sono le valutazioni nei punti iniziali. Newton e Halley tornano competitivi solo quando f' e f'' si ottengono insieme a f con poco lavoro aggiuntivo — come accade per i polinomi (schema di Horner esteso alle derivate) o con la differenziazione automatica — oppure quando la robustezza conta più della velocità, nel qual caso la scelta corretta è comunque un metodo ibrido.

In sintesi

- **Bisezione**: convergenza **garantita** se $f(a)f(b)<0$, ma solo lineare con fattore $1/2$; è il metodo di sicurezza e l'unico con una maggiorazione rigorosa dell'errore a ogni passo.
- **Newton-Raphson**: convergenza **quadratica** per radici semplici, ma soltanto locale, e richiede f'. Sotto le ipotesi di Fourier (convessità e monotonia) la convergenza è globale e monotona a partire dall'estremo opportuno.
- Il metodo delle **secanti** è la variante di Newton senza derivata, con ordine $\varphi \approx 1.618$ ma una sola valutazione per iterazione. Il metodo di **Müller** fa lo stesso con una parabola interpolante anziché una retta, salendo a $p \approx 1.839$ sempre con una sola valutazione: è il più efficiente dei metodi del capitolo.
- La **regula falsi** conserva il bracket, e dunque la convergenza, ma è in generale soltanto lineare perché un estremo resta congelato; le varianti Illinois e Pegasus correggono il difetto.
- **Parabola tangente** e **Halley** salgono a ordine tre usando la derivata seconda, ma al prezzo di tre valutazioni per iterazione.
- Il confronto fra metodi va fatto a parità di **valutazioni di funzione**, non di iterazioni: l'indice di efficienza $p^{1/\nu}$ è il criterio corretto, e ribalta l'ordinamento a favore dei metodi senza derivate.
- L'accuratezza raggiungibile non dipende dal metodo ma dal **condizionamento della radice** $1/|f'(\alpha)|$ e, per radici multiple, degrada come $\epsilon_f^{1/m}$: nessun metodo può fare meglio

(Sezione 20.4).

- Tutti i metodi aperti possono fallire (cicli, divergenza, derivata nulla). La soluzione pratica è combinarli con un passo di sicurezza, come fanno i metodi ibridi di Capitolo 22.



21. Ordine di convergenza e radici multiple

21.1 Velocità di convergenza

Consideriamo una successione $\{x_0, x_1, x_2, ...\}$ che converge a α, ossia:

$$\alpha = \lim_{k \to \infty} x_k.$$

Un aspetto fondamentale è determinare la *velocità* con cui la successione converge a α. Questa analisi permette di confrontare l'efficacia di diversi metodi numerici nel risolvere lo stesso problema. Come in tutto il capitolo poniamo $e_k = x_k - \alpha$.

21.1.1 Q-convergenza

Data una successione $\{x_k\}$ convergente ad α diremo che:

(A) Q-lineare. La successione converge Q-linearmente se esiste $C < 1$ tale che, per ogni k sufficientemente grande,

$$|e_{k+1}| \leq C\,|e_k|\,, \qquad C < 1.$$

(B) Q-ordine p. La successione converge con Q-ordine $p > 1$ se esiste $C > 0$ tale che, per ogni k sufficientemente grande,

$$|e_{k+1}| \leq C\,|e_k|^p\,.$$

Qui non serve richiedere $C < 1$: è però necessario che l'errore sia abbastanza piccolo, precisamente che $C\,|e_{k_0}|^{p-1} < 1$ per qualche k_0, altrimenti la disuguaglianza non implica affatto la convergenza. È il motivo per cui tutti i teoremi di convergenza quadratica o cubica del capitolo sono **locali**: richiedono x_0 in un intorno sufficientemente piccolo di α.

(A') Q-superlineare. La successione converge Q-superlinearmente se, per ogni k sufficientemente grande,

$$|e_{k+1}| \leq C_k\,|e_k| \qquad \text{con} \qquad \lim_{k \to \infty} C_k = 0.$$

Fra le tre nozioni valgono le implicazioni

$$(B) \implies (A') \implies (A),$$

cioè una successione con Q-ordine $p > 1$ è Q-superlineare, e una successione Q-superlineare è definitivamente Q-lineare per qualunque $C < 1$ fissato. Le implicazioni inverse sono **false**:

ad esempio $x_k = k^{-k}$ converge a zero Q-superlinearmente, perché $x_{k+1}/x_k \to 0$, ma non ha Q-ordine p per nessun $p > 1$. All'interno della classe (B), se $p > p'$ allora il metodo di ordine p è asintoticamente più veloce del metodo di ordine p'.

Una lettura operativa: con Q-ordine p il numero di cifre corrette viene moltiplicato per p a ogni iterazione. Con $p = 2$ (Newton) le cifre raddoppiano, con $p = 3$ (Halley) triplicano; con convergenza lineare di fattore C si guadagnano invece $-\log_{10} C$ cifre **per iterazione**, cioè circa 0.3 cifre per la bisezione.

21.1.2 R-convergenza

La Q-convergenza è un criterio piuttosto restrittivo, che non include le successioni la cui decrescita non è monotona. Prendiamo, ad esempio, la successione $x_k = \dfrac{1}{2^k}$, che converge Q-linearmente a 0. In questo caso, abbiamo:

$$|x_{k+1} - 0| = \frac{1}{2^{k+1}} \leq C\,|x_k - 0| = \frac{C}{2^k},$$

e scegliendo $C = \dfrac{1}{2}$ la disuguaglianza risulta vera per ogni k.

Tuttavia, consideriamo la successione:

$$y_k = \begin{cases} \dfrac{1}{2^k} & \text{se } k \text{ è dispari} \\ 0 & \text{se } k \text{ è pari} \end{cases}$$

Questa successione, pur convergendo a zero almeno altrettanto in fretta della precedente, non è Q-convergente. Infatti, per k pari,

$$\frac{1}{2^{k+1}} = |y_{k+1} - 0| \not\leq C\,|y_k - 0| = C \cdot 0 = 0, \quad \text{per ogni } C > 0.$$

L'idea della R-convergenza è allora di non chiedere nulla al rapporto fra errori consecutivi, ma di limitare l'errore con una **successione maggiorante** che converga a zero con la velocità voluta.

Data una successione $\{x_k\}$ convergente ad α diremo che:

(A) R-lineare. La successione converge R-linearmente se esiste una successione di numeri positivi η_k tale che

$$|e_k| \leq \eta_k \qquad \text{e} \qquad \eta_{k+1} \leq C\,\eta_k, \qquad C < 1,$$

ovvero, equivalentemente, se $|e_k| \leq MC^k$ per opportuni $M > 0$ e $0 < C < 1$.

(B) R-ordine p. La successione converge con R-ordine $p > 1$ se esiste una successione di numeri positivi η_k tale che

$$|e_k| \leq \eta_k \qquad \text{e} \qquad \eta_{k+1} \leq C\,\eta_k^p,$$

ovvero, equivalentemente, se $|e_k| \leq Mq^{p^k}$ per opportuni $M > 0$ e $0 < q < 1$. È esattamente la forma della tesi del Teorema 20.4.

(A') R-superlineare. La successione converge R-superlinearmente se esiste una successione di numeri positivi η_k tale che

$$|e_k| \leq \eta_k, \qquad \eta_{k+1} \leq C_k\,\eta_k \qquad \text{con} \qquad \lim_{k\to\infty} C_k = 0.$$

La successione y_k dell'esempio precedente, che non è Q-convergente, è R-linearmente convergente: basta prendere $\eta_k = 2^{-k}$. In generale la Q-convergenza di un dato ordine implica la R-convergenza dello stesso ordine, ma non viceversa: la R-convergenza descrive l'andamento **medio** dell'errore, la Q-convergenza quello di **ogni singolo passo**.

21.1.3 Ordine ed efficienza dei metodi del capitolo

Con queste definizioni, in base alle discussioni precedenti possiamo dire che:

- il metodo *dicotomico* (Sezione 20.3) è R-lineare con fattore $1/2$ (non è Q-lineare: l'errore del punto medio può occasionalmente crescere);
- il metodo delle *corde* (Sezione 20.9) e la *regula falsi* (Sezione 20.8) sono Q-lineari;
- il metodo di *Newton* (Sezione 20.6) è Q-quadratico, cioè Q-convergente di ordine $p = 2$;
- il metodo delle *secanti* (Sezione 20.7) è R-convergente di ordine $p = (1 + \sqrt{5})/2 = 1.618\ldots$;
- il metodo di *Müller* (Sezione 20.10) è R-convergente di ordine $p \approx 1.84$;
- il metodo della *parabola tangente* (Sezione 20.11) e il metodo di *Halley* (Sezione 20.12) sono Q-cubici, cioè Q-convergenti di ordine $p = 3$.

Ordinando i metodi in base alla velocità di convergenza **per iterazione**:

$$\begin{aligned} & \text{[dicotomico]} \\ \leq\ & \text{[corde]} \simeq \text{[regula falsi]} \\ \leq\ & \text{[secanti]} \\ \leq\ & \text{[Müller]} \\ \leq\ & \text{[Newton]} \\ \leq\ & \text{[parabola tangente]} \simeq \text{[Halley]} \end{aligned}$$

dove i metodi collegati da $\simeq$ hanno lo stesso ordine e vanno confrontati sul fattore asintotico.

La Figura 21.1 riassume questa scala affiancandole le due quantità che l'ordinamento non mostra: il numero ν di valutazioni richieste da una iterazione e l'indice di efficienza $p^{1/\nu}$ discusso qui sotto.

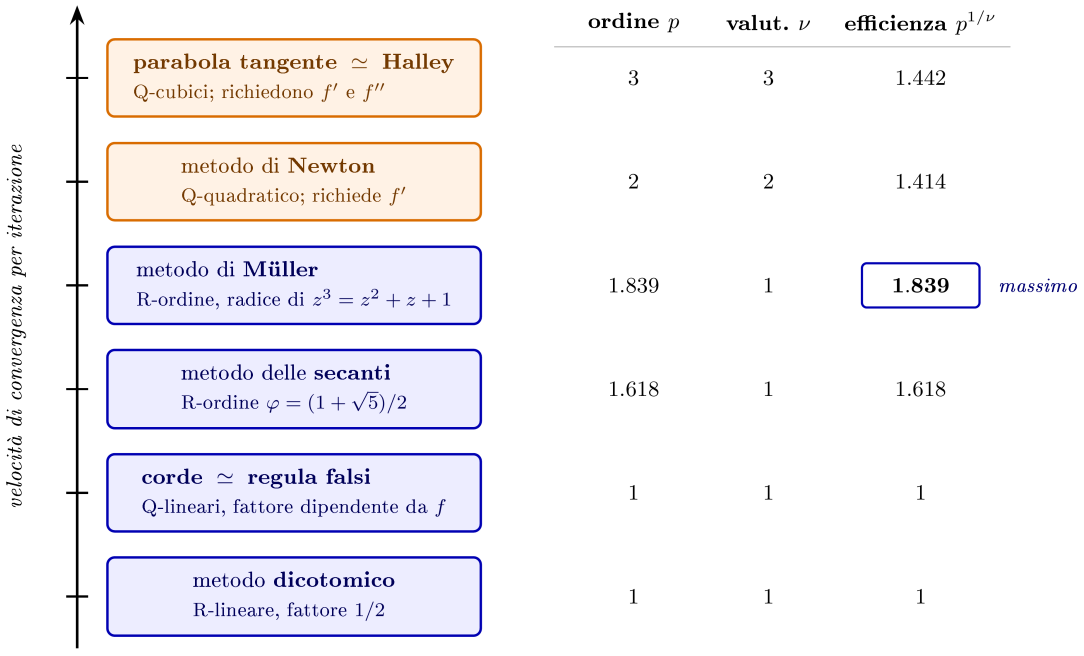


Figure 21.1: I metodi del capitolo ordinati per velocità di convergenza **per iterazione**, con il costo di una iterazione e l'indice di efficienza

! L'ordine non basta: l'indice di efficienza

L'ordinamento precedente confronta l'ordine *per iterazione* e non tiene conto del costo di una iterazione. Il confronto corretto usa l'**indice di efficienza**

$$E = p^{1/\nu},$$

dove ν è il numero di valutazioni di funzione (e derivate) richieste da una iterazione: è il fattore di cui si riduce l'errore per **singola valutazione**. Si ottiene

$$E_{\text{secanti}} = 1.618^{1/1} = 1.618, \qquad E_{\text{Müller}} = 1.839^{1/1} = 1.839,$$
$$E_{\text{Newton}} = 2^{1/2} = 1.414, \qquad E_{\text{Halley}} = E_{\text{par.tang.}} = 3^{1/3} = 1.442,$$

cioè, a parità di valutazioni, l'ordinamento **si ribalta**: i metodi che non usano derivate e riutilizzano i valori già calcolati (secanti, Müller) battono quelli di ordine più alto, e Halley è solo marginalmente migliore di Newton. Si vedano le tabelle di Sezione 20.13.
Per i metodi di ordine $p = 1$ l'indice vale 1 e non discrimina: fra di essi il confronto va fatto sul fattore di riduzione asintotico ($1/2$ per la bisezione, dipendente da f per corde e *regula falsi*).

21.2 Verifica numerica dell'ordine

Per verificare la corretta implementazione di un metodo numerico è utile controllare che le iterazioni convergano con l'ordine teorico atteso: un errore di programmazione (un segno sbagliato, una derivata approssimata male) si manifesta quasi sempre come una **perdita di ordine**, anche quando il metodo continua a convergere. Supponiamo che un metodo converga con ordine p. Vicino alla soluzione possiamo aspettarci che

$$|e_{k+1}| \approx C\,|e_k|^p\,, \qquad \text{e analogamente} \qquad |e_{k+2}| \approx C\,|e_{k+1}|^p\,.$$

Facendo il rapporto tra queste due espressioni la costante C, che non è nota, si elimina:

$$\frac{|e_{k+1}|}{|e_{k+2}|} \approx \frac{|e_k|^p}{|e_{k+1}|^p}.$$

Prendendo il logaritmo di entrambi i membri otteniamo

$$\log\left(\frac{|e_{k+1}|}{|e_{k+2}|}\right) \approx p \log\left(\frac{|e_k|}{|e_{k+1}|}\right),$$

da cui si ricava la stima numerica dell'ordine:

$$\boxed{p \approx \frac{\log\left(|e_{k+1}| \,/\, |e_{k+2}|\right)}{\log\left(|e_k| \,/\, |e_{k+1}|\right)}} \tag{21.1}$$

Come esempio applichiamo il metodo di Newton al calcolo di $\sqrt{2}$, cioè a $f(x) = x^2 - 2$, per cui lo schema si semplifica in

$$x_{k+1} = x_k - \frac{x_k^2 - 2}{2x_k} = \frac{x_k}{2} + \frac{1}{x_k},$$

partendo da $x_1 = 2$. Le iterate sono calcolate in doppia precisione IEEE.

```r
newton_sqrt2 <- function(n) {
  x <- 2
  out <- numeric(n)
  for (k in 1:n) { out[k] <- x; x <- x/2 + 1/x }
  out
}

stima_p <- function(e) {
  n <- length(e)
  p <- rep(NA_real_, n)
  if (n >= 3)
    for (k in 3:n)
      if (e[k] > 0 && e[k-1] > 0 && e[k-2] > 0)
        p[k] <- log(e[k-1]/e[k]) / log(e[k-2]/e[k-1])
  p
}

x <- newton_sqrt2(7)
e <- abs(x - sqrt(2))

knitr::kable(
  data.frame(
    k      = seq_along(x),
    x      = sprintf("%.17g", x),
    `f(x)` = formatC(x^2 - 2, format = "e", digits = 3),
    e      = formatC(e,       format = "e", digits = 4),
    p      = formatC(stima_p(e), format = "f", digits = 3),
    check.names = FALSE),
  align = "r", caption = "Iterate di Newton per $\\sqrt{2}$")
```

Table 21.1: Iterate di Newton per $\sqrt{2}$

k	x	f(x)	e	p
1	2	2.000e+00	5.8579e-01	NA
2	1.5	2.500e-01	8.5786e-02	NA
3	1.4166666666666665	6.944e-03	2.4531e-03	1.850
4	1.4142156862745097	6.007e-06	2.1239e-06	1.984
5	1.4142135623746899	4.511e-12	1.5947e-12	2.000
6	1.4142135623730949	-4.441e-16	2.2204e-16	0.630
7	1.4142135623730949	-4.441e-16	2.2204e-16	0.000

Le prime iterate coincidono, a meno dell'arrotondamento, con i valori razionali esatti

$$2, \quad \frac{3}{2}, \quad \frac{17}{12}, \quad \frac{577}{408}, \quad \frac{665857}{470832},$$

e la stima di p si avvicina rapidamente al valore teorico 2. All'iterata 6, però, l'errore scende al livello della precisione di macchina, l'iterazione diventa **stazionaria** ($x_7 = x_6$) e la stima dell'ordine perde ogni significato: il valore riportato nell'ultima riga non ha alcun contenuto.

Questo comportamento è esattamente quello previsto in Sezione 20.4. Qui $\epsilon_f \approx \varepsilon_{\text{mach}} \cdot 2 \approx 4.4 \cdot 10^{-16}$ e $|f'(\sqrt{2})| = 2\sqrt{2}$, quindi la 20.1 prevede un'incertezza di circa $1.6 \cdot 10^{-16}$, cioè meno di una unità di ultima posizione: è il livello a cui l'iterazione si blocca. Si osservi inoltre che il punto fisso raggiunto, 1.4142135623730949, **non è** il numero macchina più vicino a $\sqrt{2}$, che è 1.4142135623730951: differisce da esso di una unità di ultima posizione. Infine, anche il valore di riferimento usato per calcolare e_k è il numero macchina `sqrt(2)`, a sua volta arrotondato: errori inferiori a un ulp non sono misurabili in questo modo e vanno valutati in precisione estesa.

> **Morale**
>
> Non si deve mai iterare "fino a che l'errore è zero". Un metodo di ordine elevato raggiunge la precisione di macchina in pochissimi passi: ogni iterazione successiva è lavoro sprecato che, nei casi peggiori (si veda l'esempio della radice doppia perturbata più avanti), può addirittura peggiorare il risultato.

21.2.1 Stima dell'ordine senza conoscere la radice

La 21.1 ha un difetto che la rende inservibile proprio quando servirebbe: richiede α. Su un problema di prova la radice è nota e la formula va benissimo, ma la verifica dell'ordine serve soprattutto quando si collauda un'implementazione **nuova** su un problema **nuovo** — cioè esattamente quando α è l'incognita. Serve una stima che usi solo le iterate.

L'idea: sostituire l'errore con l'incremento

L'unica quantità osservabile è l'incremento fra due iterate successive,

$$\Delta_k := x_{k+1} - x_k = e_{k+1} - e_k = -e_k \left(1 - \frac{e_{k+1}}{e_k}\right).$$

Il fattore fra parentesi si controlla facilmente nei due regimi che ci interessano.

- Se la convergenza ha **ordine** $p > 1$, allora $e_{k+1}/e_k \approx C\,|e_k|^{p-1} \to 0$, e quindi
$$\Delta_k = -e_k\big(1 + O(|e_k|^{p-1})\big) \ :$$
l'incremento *è* l'errore, a meno di un fattore che tende a 1.
- Se la convergenza è **lineare** con fattore $C < 1$, allora $e_{k+1}/e_k \to C$ e
$$\Delta_k = -e_k\,(1 - C)\big(1 + o(1)\big) \ :$$
l'incremento non è l'errore, ma gli è **proporzionale**, con costante $1 - C$.

In entrambi i casi $|\Delta_k| = \gamma\,|e_k|\,\big(1 + o(1)\big)$ con γ costante. È questo il punto decisivo: la 21.1 è un rapporto fra logaritmi di **rapporti** di errori, e in un rapporto la costante γ si semplifica. Si può dunque sostituire e con Δ senza alterare il limite:

$$\boxed{p \approx \frac{\log\big(|\Delta_{k+1}| \,/\, |\Delta_{k+2}|\big)}{\log\big(|\Delta_k| \,/\, |\Delta_{k+1}|\big)}} \tag{21.2}$$

ovvero, esplicitando gli incrementi in termini di iterate,

$$p \approx \frac{\log\big(|x_{k+2} - x_{k+1}| \,/\, |x_{k+3} - x_{k+2}|\big)}{\log\big(|x_{k+1} - x_k| \,/\, |x_{k+2} - x_{k+1}|\big)}.$$

Rispetto alla 21.1 serve **una iterata in più** — quattro consecutive anziché tre — e nient'altro: nessuna informazione a priori, nessuna soluzione di riferimento. È la forma usata nei test automatici di validazione delle librerie numeriche, e la stessa idea si ritrova nello studio sperimentale dell'ordine dei metodi per equazioni differenziali, dove la soluzione esatta non è quasi mai disponibile.

Verifica

Mettiamo alla prova la formula su tre metodi applicati alla **stessa** equazione $x^2 - 2 = 0$, scelti in modo da coprire i tre regimi: Newton ($p = 2$), secanti ($p = \varphi \approx 1.618$) e il metodo delle corde con pendenza congelata $m = f'(2) = 4$, che converge linearmente con fattore $|1 - f'(\alpha)/m| = 1 - \sqrt{2}/2 \approx 0.293$. Per ciascuno calcoliamo entrambe le stime, quella che usa α e quella che non la usa.

```
f <- function(x) x^2 - 2;  df <- function(x) 2*x
alpha <- sqrt(2)

itera <- function(passo, x0, n) {
  x <- x0; out <- x
  for (k in 1:n) { x <- passo(x); out <- c(out, x) }
  out
}
secanti <- function(x0, x1, n) {
  out <- c(x0, x1)
  for (k in 1:n) {
    x2 <- x1 - f(x1)*(x1-x0)/(f(x1)-f(x0))
    out <- c(out, x2); x0 <- x1; x1 <- x2
  }
  out
}

# stima che usa la radice esatta: eq. @eq-stima-ordine
p_con <- function(x, alpha) {
  e <- abs(x - alpha); n <- length(e); p <- rep(NA_real_, n)
  for (k in 3:n)
    if (all(e[(k-2):k] > 0)) p[k] <- log(e[k-1]/e[k]) / log(e[k-2]/e[k-1])
  p
}
# stima che usa i soli incrementi: eq. @eq-stima-ordine-delta
p_senza <- function(x) {
  d <- abs(diff(x)); n <- length(x); p <- rep(NA_real_, n)
  if (n >= 4)
    for (k in 3:(n-1))
      if (all(d[(k-2):k] > 0)) p[k] <- log(d[k-1]/d[k]) / log(d[k-2]/d[k-1])
  p
}

confronta <- function(x, didascalia, righe = seq_along(x)) {
  knitr::kable(
    data.frame(k                        = seq_along(x),
               `$x_k$`                  = sprintf("%.15f", x),
               `$p$ con $\\alpha$`      = sprintf("%.4f", p_con(x, alpha)),
               `$p$ dai soli $\\Delta$` = sprintf("%.4f", p_senza(x)),
               check.names = FALSE)[righe, ],
    row.names = FALSE, align = "r", caption = didascalia)
}

confronta(itera(function(x) x - f(x)/df(x), 2, 6), "Newton: ordine atteso $p=2$")
```

Table 21.2: Newton: ordine atteso $p = 2$

k	x_k	p con α	p dai soli Δ
1	2.000000000000000	NA	NA
2	1.500000000000000	NA	NA
3	1.416666666666667	1.8503	1.9681
4	1.414215686274510	1.9839	1.9995
5	1.414213562374690	1.9998	2.0000
6	1.414213562373095	NA	0.6296
7	1.414213562373095	NA	NA

```
confronta(secanti(1, 2, 6), "Secanti: ordine atteso $p=\\varphi\\approx1.618$")
```

Table 21.3: Secanti: ordine atteso $p = \varphi \approx 1.618$

k	x_k	p con α	p dai soli Δ
1	1.000000000000000	NA	NA
2	2.000000000000000	NA	NA
3	1.333333333333333	-5.7130	5.6789
4	1.400000000000000	0.8782	0.6585
5	1.414634146341463	2.0246	2.3375
6	1.414211438474870	1.5023	1.4935
7	1.414213562057320	1.6666	1.6650
8	1.414213562373095	1.6075	NA

```
confronta(itera(function(x) x - f(x)/4, 2, 8),
          "Corde con $m=4$: ordine atteso $p=1$", righe = 1:9)
```

Table 21.4: Corde con $m = 4$: ordine atteso $p = 1$

k	x_k	p con α	p dai soli Δ
1	2.000000000000000	NA	NA
2	1.500000000000000	NA	NA
3	1.437500000000000	0.6788	0.6375
4	1.420898437500000	0.9571	0.9458
5	1.416160345077515	0.9885	0.9852
6	1.414782814334998	0.9967	0.9958
7	1.414380211400584	0.9990	0.9988
8	1.414262365800194	0.9997	0.9996
9	1.414227855970503	0.9999	NA

Le tabelle dicono tre cose.

1. **Le due stime coincidono** non appena il metodo entra nel regime asintotico: per Newton entrambe danno 2.00, per le secanti entrambe si assestano attorno a 1.61, per le corde entrambe danno 1.00. La colonna che richiede α non aggiunge nulla.
2. **La formula funziona anche per $p = 1$.** È la conseguenza della cancellazione della costante $\gamma = 1 - C$ vista sopra, e non è ovvia: la stima resta valida per i metodi lineari, dove misura correttamente $p = 1$ (il *fattore* C, se interessa, si legge direttamente dal rapporto $|\Delta_{k+1}| \,/\, |\Delta_k|$).
3. **Nel transitorio iniziale le stime non significano nulla.** Per le secanti le prime righe danno valori negativi o maggiori di 5: le iterate sono ancora lontane dalla radice e la relazione $|e_{k+1}| \approx C\,|e_k|^p$, che è asintotica, non vale.

⚠ Come leggere una stima dell'ordine

- **Non fidarsi di un singolo numero.** Si cerca un *plateau*: tre o quattro stime consecutive che si stabilizzano sullo stesso valore. Un valore isolato può essere un artefatto del transitorio o dell'arrotondamento.
- **Attenzione all'ultima riga.** Sul plateau di arrotondamento la stima basata sull'errore smette semplicemente di essere definita, perché e_k diventa esattamente zero; quella basata sugli incrementi, invece, continua a produrre un numero. Nella tabella di Newton l'ultima riga ne è l'esempio: 0.63, un valore perfettamente plausibile e del

tutto privo di significato. È il caso più insidioso, perché non dà alcun segnale d'allarme.

- **Finestra utile.** Le stime attendibili sono quelle calcolate quando $|\Delta_k|$ è già piccolo rispetto a $|x_k|$ ma ancora molto maggiore di $\varepsilon_{\text{mach}}\,|x_k|$: fra il transitorio e il plateau di Sezione 20.4. Con un metodo di ordine alto questa finestra può essere larga due o tre iterazioni soltanto, ed è il motivo per cui conviene fare la verifica con una tolleranza *lasca*, non spinta al limite della macchina.

Variante: stimare prima la radice

Un'alternativa è usare le iterate per stimare α, e poi applicare la 21.1 come se la radice fosse nota. Lo strumento naturale è l'estrapolazione Δ^2 di Aitken:

$$\hat{\alpha}_k = x_k - \frac{\Delta_k^2}{\Delta_{k+1} - \Delta_k},$$

che elimina il termine dominante dell'errore e converge ad α più in fretta della successione di partenza.

```
aitken <- function(x) {
  n <- length(x); a <- rep(NA_real_, n)
  for (k in 1:(n-2)) {
    d1 <- x[k+1]-x[k]; d2 <- x[k+2]-x[k+1]; den <- d2 - d1
    if (is.finite(den) && den != 0) a[k] <- x[k] - d1^2/den
  }
  a
}
xn <- itera(function(x) x - f(x)/df(x), 2, 6)
knitr::kable(
  data.frame(k = seq_along(xn),
             `$\\hat\\alpha_k$ (Aitken)` = sprintf("%.15f", aitken(xn)),
             `$\\lvert\\hat\\alpha_k-\\alpha\\rvert$` = sprintf("%.1e",
                  ↪  abs(aitken(xn)-alpha)),
             check.names = FALSE),
  row.names = FALSE, align = "r",
  caption = "Stima della radice per estrapolazione di Aitken")
```

Table 21.5: Stima della radice per estrapolazione di Aitken

k	$\hat{\alpha}_k$ (Aitken)	$\lvert\hat{\alpha}_k - \alpha\rvert$
1	1.400000000000000	1.4e-02
2	1.414141414141414	7.2e-05
3	1.414213560532626	1.8e-09
4	1.414213562373095	0.0e+00
5	1.414213562373095	2.2e-16
6	NA	NA
7	NA	NA

Già alla terza riga $\hat{\alpha}$ ha una decina di cifre corrette, e usandola al posto di α nella 21.1 si riottengono le stesse stime della colonna «p con α». Il prezzo è una maggiore sensibilità all'arrotondamento: il denominatore $\Delta_{k+1} - \Delta_k$ è una differenza di differenze, e si annulla per cancellazione non appena le iterate si avvicinano alla precisione di macchina. Per la sola verifica dell'ordine la 21.2 è più robusta e va preferita; l'estrapolazione di Aitken resta utile quando serve davvero un valore accurato di α a partire da una successione che converge lentamente.

21.3 Radici multiple e rallentamento della convergenza

Quando si risolvono equazioni non lineari $f(x) = 0$ con il **metodo di Newton**, è noto che il metodo ha **convergenza quadratica** se la radice è semplice. Tuttavia, in presenza di **radici multiple**, la convergenza diventa **lineare**, causando un significativo rallentamento. Vediamo come gestire queste situazioni e recuperare un ordine superiore.

21.3.1 Rallentamento della convergenza

Supponiamo che la radice r abbia molteplicità m, cioè che

$$f(x) = (x - r)^m g(x), \qquad g(r) \neq 0.$$

Il fattore di riduzione dell'errore si calcola esplicitamente. Derivando,

$$f'(x) = m(x - r)^{m-1} g(x) + (x - r)^m g'(x),$$

si ottiene

$$\frac{f(x)}{f'(x)} = \frac{(x - r)\, g(x)}{m\, g(x) + (x - r)\, g'(x)},$$

e quindi, posto $e_k = x_k - r$,

$$e_{k+1} = e_k \left(1 - \frac{g(x_k)}{m\, g(x_k) + e_k g'(x_k)}\right) = \left(1 - \frac{1}{m}\right) e_k + O(e_k^2). \tag{21.3}$$

Il metodo di Newton su una radice di molteplicità m ha dunque **convergenza lineare** con fattore di riduzione esattamente $1 - 1/m$: per $m = 2$ l'errore si dimezza a ogni passo, per $m = 3$ si riduce di un terzo, e per m grande il metodo diventa inutilizzabile.

Si noti che il rallentamento si somma al problema di condizionamento discusso in Sezione 20.4. Su una radice di molteplicità m l'accuratezza raggiungibile è comunque limitata a $O(\epsilon_f^{1/m})$, quindi le iterazioni in più non servono a nulla.

21.3.2 Esempio in R

Usiamo un unico *driver* che restituisce la storia delle iterate, con la salvaguardia necessaria quando la derivata si annulla (cosa che su una radice multipla accade **esattamente nella radice**).

```
newton_storia <- function(f, df, x0, n, m = 1) {
  x   <- x0
  out <- numeric(0)
  for (k in seq_len(n)) {
    d <- df(x)
    if (!is.finite(d) || d == 0) break   # derivata nulla: non si può proseguire
    x   <- x - m * f(x) / d
    out <- c(out, x)
  }
  out
}
```

Consideriamo $f(x) = (x - 2)^2$, che ha una radice doppia in $x = 2$. In questo caso il passo di Newton si calcola a mano,

$$x_{k+1} = x_k - \frac{(x_k - 2)^2}{2(x_k - 2)} = \frac{x_k + 2}{2},$$

cioè $e_{k+1} = e_k/2$, in accordo con il fattore $1 - 1/m = 1/2$ previsto dalla 21.3.

```
f  <- function(x) (x - 2)^2
df <- function(x) 2 * (x - 2)

st <- newton_storia(f, df, x0 = 4, n = 10)
data.frame(k        = seq_along(st),
           x        = st,
           e        = abs(st - 2),
           rapporto = c(NA, abs(st[-1] - 2) / abs(st[-length(st)] - 2)))
```

```
    k        x           e rapporto
1   1 3.000000 1.000000000       NA
2   2 2.500000 0.500000000      0.5
3   3 2.250000 0.250000000      0.5
4   4 2.125000 0.125000000      0.5
5   5 2.062500 0.062500000      0.5
6   6 2.031250 0.031250000      0.5
7   7 2.015625 0.015625000      0.5
8   8 2.007812 0.007812500      0.5
9   9 2.003906 0.003906250      0.5
10 10 2.001953 0.001953125      0.5
```

Il rapporto $|e_{k+1}| \,/\, |e_k|$ vale costantemente $1/2$: servono circa 3.3 iterazioni per cifra decimale, come per la bisezione. Su una radice semplice le cifre corrette sarebbero invece raddoppiate a ogni passo.

21.3.3 Una radice doppia perturbata

Una seconda difficoltà, distinta dalla precedente, si presenta quando la radice doppia viene perturbata. La funzione

$$f_\epsilon(x) = (x-2)^2 + \epsilon, \qquad \epsilon > 0,$$

non ha radici reali: il suo minimo vale $\epsilon > 0$. Il metodo di Newton si avvicina comunque a $x = 2$ finché il passo è dominato dal termine $(x-2)/2$, ma appena $(x-2)^2$ scende sotto ϵ il passo diventa $\approx -\epsilon/\big(2(x-2)\big)$, cioè un rapporto fra due quantità piccole, e le iterate si allontanano di nuovo dal minimo, oscillando in modo irregolare.

```
epsilon <- 1e-8
f_eps   <- function(x) (x - 2)^2 + epsilon
df_eps  <- function(x) 2 * (x - 2)

st3 <- newton_storia(f_eps, df_eps, x0 = 4, n = 30)
tail(data.frame(k = seq_along(st3), x = st3, f = f_eps(st3)), 12)
```

```
    k        x            f
19 19 2.000053 1.283323e-08
20 20 1.999933 1.453216e-08
21 21 2.000041 1.164917e-08
22 22 1.999897 2.057141e-08
23 23 1.999997 1.000772e-08
24 24 2.001798 3.242725e-06
25 25 2.000896 8.131889e-07
26 26 2.000443 2.058283e-07
27 27 2.000210 5.408475e-08
28 28 2.000081 1.658828e-08
29 29 1.999979 1.044169e-08
30 30 2.000227 6.171144e-08
```

Le iterate vagano in un intorno di ampiezza $\approx \sqrt{\epsilon} = 10^{-4}$ senza mai stabilizzarsi. Questo

illustra due cose: che su radici multiple o quasi multiple un criterio d'arresto basato su $|f(x)|$ si attiva su un intero intervallo di ampiezza $O(\sqrt{\epsilon})$ (coerentemente con la 20.2), e che **occorre sempre un limite al numero di iterazioni**.

21.3.4 Correzione del metodo di Newton per radici multiple

Se la molteplicità m è nota, basta moltiplicare il passo per m:

$$\boxed{x_{k+1} = x_k - m\,\frac{f(x_k)}{f'(x_k)}} \tag{21.4}$$

Lo schema equivale ad applicare il metodo di Newton a $f^{1/m}$, che ha in r una radice semplice, e recupera quindi la convergenza quadratica. Per $f(x) = (x-2)^2$ la correzione individua la radice **in un solo passo**, dopodiché la derivata si annulla e il driver si arresta:

```
newton_storia(f, df, x0 = 4, n = 10, m = 2)
```

```
[1] 2
```

Se la molteplicità non è nota, si può invece applicare il metodo di Newton alla funzione

$$u(x) = \frac{f(x)}{f'(x)},$$

che ha in r una radice **semplice** qualunque sia m. Lo schema risultante,

$$x_{k+1} = x_k - \frac{f(x_k)f'(x_k)}{f'(x_k)^2 - f(x_k)f''(x_k)}, \tag{21.5}$$

converge quadraticamente senza conoscere m, al prezzo del calcolo di f'' e di una maggiore sensibilità agli errori di arrotondamento (il denominatore è una differenza di quantità che tendono entrambe a zero). Si confronti la 21.5 con il passo di Halley dato dalla 20.22. Le due formule differiscono solo per il fattore $1/2$ davanti a ff''.

Una terza possibilità, quando si sospetta una radice multipla ma non se ne conosce l'ordine, è **stimare** m dal rapporto osservato fra errori consecutivi: se $|e_{k+1}|\,/\,|e_k| \to C$, allora $m \approx 1/(1-C)$.

Riepilogo su ordine e radici multiple

- **Q-convergenza** descrive il comportamento di ogni singolo passo, **R-convergenza** l'andamento medio dell'errore: la prima implica la seconda, non viceversa. La bisezione è R-lineare, le secanti sono R-convergenti di ordine φ, Newton è Q-quadratico, Halley Q-cubico.
- L'ordine teorico va **verificato numericamente** con la 21.1 (o la sua variante senza α): la perdita di ordine è il sintomo più affidabile di un'implementazione sbagliata.
- La stima dell'ordine ha senso solo finché l'errore resta sopra il livello di arrotondamento: al di sotto, i numeri ottenuti sono privi di significato.
- Su una radice di molteplicità m Newton degrada a convergenza **lineare** con fattore $1 - 1/m$; si recupera l'ordine due o con la correzione 21.4, se m è noto, o applicando Newton a f/f' 21.5 altrimenti.
- Su radici multiple l'accuratezza raggiungibile è comunque limitata a $O(\epsilon_f^{1/m})$: nessuna correzione dello schema può superare questo limite, che dipende solo dal problema.

21.4 Esercizi

1. **Bisezione.** Quante iterazioni servono per localizzare con errore minore di 10^{-12} una radice contenuta in $[1, 2]$? E in $[0, 100]$? Verificare sperimentalmente la formula $k_{\min} = \lceil \log_2((b-a)/\tau) \rceil$.
2. **Robustezza della bisezione.** Si implementi la bisezione con il test $f(c)f(a) < 0$ anziché con il confronto dei segni e la si applichi a $f(x) = 10^{-200}(x-1)$ su $[0, 2]$ in doppia precisione. Che cosa succede, e perché?
3. **Ordine di Newton quando $f''(\alpha) = 0$.** Si applichi Newton a $f(x) = x^3 - 8$ partendo da $x_0 = 3$ e poi a $f(x) = x^3 + x$ partendo da $x_0 = 1$, stimando p con la 21.1 in entrambi i casi. Perché nel primo si ottiene $p = 2$ e nel secondo $p = 3$? Si giustifichi con la costante asintotica $f''(\alpha)/(2f'(\alpha))$.
4. **Condizione di Fourier.** Verificare che $f(x) = e^x - 2$ soddisfa le ipotesi del Teorema 20.3 su $[0, 1]$ e che Newton partendo da $x_0 = 1$ genera una successione decrescente. Che cosa accade partendo da $x_0 = 0$?
5. **Fallimento di Newton.** Determinare numericamente, con una bisezione sulla funzione $\phi(x_0) = x_1(x_0) + x_0$, il valore critico x^* per cui Newton applicato a $\arctan x$ entra in un ciclo di periodo due. Confrontare con il valore $1.3917452002707\ldots$ riportato in Sezione 20.6.4.
6. **Secanti contro Newton.** Risolvere $\cos x = x$ con entrambi i metodi, contando le **valutazioni di funzione** necessarie per scendere sotto 10^{-14}. Il risultato conferma l'indice di efficienza $p^{1/\nu}$?
7. **Regula falsi e varianti.** Applicare la *regula falsi* a $f(x) = x^{10} - 1$ su $[0, 1.3]$ e osservare la stagnazione dell'estremo destro. Implementare poi la variante *Illinois* (si dimezza $f(a_k)$ ogni volta che a_k non viene aggiornato) e confrontare il numero di iterazioni.
8. **Radici multiple.** Per $f(x) = (x-1)^3(x-2)$ si stimi numericamente la molteplicità della radice $x = 1$ a partire dal rapporto osservato fra errori consecutivi, usando $m \approx 1/(1-C)$, e si verifichi che 21.4 con il valore stimato recupera la convergenza quadratica.
9. **Accuratezza raggiungibile.** Valutare $p(x) = (x-1)^5$ nella forma espansa in 1 ± 10^{-3} e confrontare l'ampiezza della zona di rumore con la previsione $O(\epsilon_f^{1/5})$ della 20.2.
10. **Parabola tangente e Halley.** Implementare entrambi i metodi per $f(x) = x^3 - 2x - 5$ e verificare numericamente che l'ordine è 3 per entrambi. Quale dei due è più robusto quando f'' è grande, e perché?



22. Metodi ibridi e algoritmi con salvaguardia

Nel capitolo precedente abbiamo visto due famiglie di metodi con difetti complementari: i metodi di confinamento (bisezione, *regula falsi*) **convergono sempre** ma lentamente, mentre i metodi aperti (Newton, secanti, Halley) sono rapidi ma possono ciclare o divergere (Sezione 20.6.4). I metodi di questo capitolo risolvono il dilemma nel modo più semplice possibile: **propongono il candidato di un metodo veloce e lo accettano solo se supera un test di sicurezza**, ripiegando altrimenti su un passo di bisezione.

Tutti mantengono quindi un intervallo $[a, b]$ tale che $f(a)f(b) \le 0$, supponendo f continua, e conservano il cambio di segno dopo ogni valutazione accettata. La differenza fra i metodi riguarda **quale candidato proporre**, **quando rifiutarlo** e **come imporre la riduzione dell'intervallo**.

Nel seguito, quando l'intervallo è descritto da più di due punti, useremo la convenzione usuale: b è il **miglior estremo** (quello con $|f|$ più piccolo, cioè l'iterata corrente), a è il **contropunto** (l'estremo che garantisce il cambio di segno, $f(a)f(b) \le 0$) e c è il valore precedente di b.

22.1 Lo schema generale

Tutti gli algoritmi del capitolo hanno la stessa struttura.

Schema generale di un metodo con salvaguardia

Dati: a, b con $f(a)f(b) \le 0$; tolleranza τ
Restituisce: approssimazione della radice

1 **mentre** *il criterio d'arresto non è soddisfatto* **fai**
2 **proponi** un candidato s con un metodo veloce
3 (secante, IQI, Newton, …)
4 **verifica** che s sia *ammissibile*: interno a (a, b),
5 non troppo vicino agli estremi, con *progresso*
6 sufficiente rispetto ai passi precedenti ▷ *è qui che i metodi differiscono*
7 **se** s *non è ammissibile* **allora**
8 $s \leftarrow m = a + \dfrac{b-a}{2}$ ▷ *ripiego: bisezione*
9 **valuta** $f(s)$ e aggiorna il bracket conservando il cambio di segno

Figure 22.1: Lo schema comune a tutti i metodi del capitolo: un candidato veloce, un test di ammissibilità, e la bisezione come ripiego. Cambiando solo il test del punto 2 si ottengono Dekker, Brent, Ridders, TOMS 748, Chandrupatla e ITP.

Il punto 2 è ciò che distingue un algoritmo dall'altro, ed è anche la parte più delicata: un test troppo permissivo lascia sopravvivere il cattivo comportamento del metodo veloce, uno troppo restrittivo riduce l'algoritmo a una bisezione mascherata.

Il criterio d'arresto è sempre quello discusso in Sezione 20.4, cioè $b - a \le 2\varepsilon_{\text{mach}}\,|b| + \tau$: è l'unico che fornisce una maggiorazione rigorosa dell'errore. Fa eccezione la famiglia che vedremo in Sezione 22.8, dove — come nel metodo di Newton — l'intervallo può non restringersi e ci si arresta sull'incremento $|x_{k+1} - x_k|$.

22.1.1 Il problema di riferimento

Per confrontare i metodi useremo lo stesso problema del capitolo precedente, la cubica di Wallis

$$f(x) = x^3 - 2x - 5, \qquad \alpha = 2.0945514815423266\ldots,$$

sull'intervallo di partenza $[2, 3]$. Ogni metodo sarà accompagnato dalla tabella delle iterate, con il tipo di passo effettivamente eseguito, il valore di f, l'errore e l'ampiezza del bracket. Alla fine del capitolo (Sezione 22.13) i dieci metodi vengono confrontati su cinque problemi diversi, contando le **valutazioni di funzione**.

```
alpha <- 2.0945514815423265148
f     <- function(x) x^3 - 2*x - 5
df    <- function(x) 3*x^2 - 2
ddf   <- function(x) 6*x
dddf  <- function(x) 6
EPS   <- .Machine$double.eps

# formattazione uniforme delle tabelle delle iterate
tabella <- function(st, didascalia) {
  knitr::kable(
    data.frame(
      k                    = st$k,
      passo                = st$tipo,
      `$x_k$`              = formatC(st$x,   format = "f", digits = 12),
      `$f(x_k)$`           = formatC(st$fx,  format = "e", digits = 2),
      `$\\lvert x_k-\\alpha\\rvert$` = formatC(st$err, format = "e", digits = 2),
      `$b-a$`              = formatC(st$amp, format = "e", digits = 2),
      check.names = FALSE),
    align = "r", caption = didascalia)
}
```

22.2 Metodo di Dekker

È il capostipite di tutta la famiglia [14]. L'idea è la più semplice possibile: **usare la secante, ma solo quando cade dalla parte giusta**.

22.2.1 Come funziona

Si mantengono tre punti: il miglior estremo b (l'iterata corrente, quella con $|f|$ minore), il contropunto a con $f(a)f(b) \le 0$, e il valore precedente c di b. A ogni passo si calcolano due candidati:

$$m = \frac{a+b}{2} \qquad \text{(bisezione)}, \qquad s = b - f(b)\,\frac{b-c}{f(b)-f(c)} \qquad \text{(secante)}.$$

La secante è costruita sulle **ultime due iterate** c e b, esattamente come nel metodo delle secanti di Sezione 20.7: non sugli estremi del bracket, come farebbe la *regula falsi*. È questa la ragione della sua rapidità, perché entrambi i punti convergono alla radice invece che restare uno congelato.

Il test di ammissibilità è minimale: si accetta s **solo se cade nell'intervallo aperto fra** b **e** m. In altre parole, la secante è accettata quando si muove nella stessa direzione della bisezione ma senza superarla; in tutti gli altri casi (secante che esce dal bracket, denominatore nullo, passo che supera il punto medio) si usa m. Qualunque sia il candidato scelto, il nuovo punto diventa b, il vecchio b diventa c, e il bracket viene aggiornato; se il nuovo b ha lo stesso segno di a, il contropunto diventa il vecchio b. Infine si scambiano a e b se necessario, per mantenere in b il punto con $|f|$ minore.

Poiché il candidato di ripiego è il punto medio, **l'ampiezza del bracket non può mai aumentare** e la convergenza è sempre garantita. Il difetto, notato dallo stesso Dekker e poi corretto da Brent, è che la garanzia di *velocità* è debole: esistono funzioni per cui la secante viene accettata molte volte di seguito con passi minuscoli, e il bracket si riduce molto più lentamente della bisezione. Nel caso peggiore il numero di valutazioni può superare di molto quello della sola bisezione.

Metodo di Dekker

Dati: a, b con $f(a)f(b) \le 0$; tolleranza τ
Restituisce: approssimazione della radice

se $|f(a)| < |f(b)|$ **allora**
  scambia $a \leftrightarrow b$
$c \leftarrow a$ ▷ *c = iterata precedente*
mentre $|b-a| > \tau$ e $f(b) \neq 0$ **fai**
  $m \leftarrow (a+b)/2$
  se $f(b) \neq f(c)$ **allora**
    $s \leftarrow b - f(b)\dfrac{b-c}{f(b)-f(c)}$ ▷ *secante su c, b*
  altrimenti
    $s \leftarrow m$
  se *s cade strettamente fra b e m* **allora**
    $b_{\text{new}} \leftarrow s$ ▷ *unico test di ammissibilità*
  altrimenti
    $b_{\text{new}} \leftarrow m$
  $c \leftarrow b$, $\quad b \leftarrow b_{\text{new}}$, $\quad$ **valuta** $f(b)$
  se $f(a)f(b) > 0$ **allora**
    $a \leftarrow c$ ▷ *ripristina il cambio di segno*
  se $|f(a)| < |f(b)|$ **allora**
    scambia $a \leftrightarrow b$ ▷ *in b il punto migliore*
restituisci b

Figure 22.2: Il metodo di Dekker: la secante è costruita sulle ultime due iterate c e b, non sugli estremi del bracket, e viene accettata solo se cade fra b e il punto medio.

22.2.2 Esempio numerico

```
dekker <- function(f, a, b, tol = 1e-12, nmax = 60) {
  fa <- f(a); fb <- f(b)
  if (fa*fb > 0) stop("gli estremi non racchiudono una radice")
  if (abs(fa) < abs(fb)) { t<-a;a<-b;b<-t; t<-fa;fa<-fb;fb<-t }
  c <- a; fc <- fa                       # c = iterata precedente
  st <- data.frame()
  for (k in 1:nmax) {
    m <- (a + b)/2
    s <- if (fb != fc) b - fb*(b - c)/(fb - fc) else m
    usa <- if (is.finite(s) && ((s > b && s < m) || (s < b && s > m)))
```

```
            "secante" else "bisezione"
    xn <- if (usa == "secante") s else m
    c <- b; fc <- fb
    b <- xn; fb <- f(b)
    if (fa*fb > 0) { a <- c; fa <- fc }               # ripristina il bracket
    if (abs(fa) < abs(fb)) { t<-a;a<-b;b<-t; t<-fa;fa<-fb;fb<-t }
    st <- rbind(st, data.frame(k=k, tipo=usa, x=b, fx=fb,
                               err=abs(b-alpha), amp=abs(b-a)))
    if (abs(b-a) < tol || fb == 0) break
  }
  st
}

tabella(dekker(f, 2, 3), "Metodo di Dekker su $x^3-2x-5$, $[2,3]$")
```

Table 22.1: Metodo di Dekker su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	secante	2.058823529412	-3.91e-01	3.57e-02	9.41e-01
2	secante	2.096558636819	2.24e-02	2.01e-03	3.77e-02
3	secante	2.094510553557	-4.57e-04	4.09e-05	2.05e-03
4	secante	2.094551435331	-5.16e-07	4.62e-08	2.01e-03
5	secante	2.094551481543	1.19e-11	1.06e-12	4.62e-08
6	secante	2.094551481542	-8.88e-16	0.00e+00	1.06e-12
7	bisezione	2.094551481542	-8.88e-16	0.00e+00	5.32e-13

Sei passi di secante portano alla precisione di macchina; la bisezione interviene solo all'ultimo passo, quando la secante degenera perché $f(b)$ e $f(c)$ coincidono in aritmetica finita. Si osservi la colonna $b - a$: il bracket si restringe molto più lentamente dell'errore, perché il contropunto resta a lungo fermo. È esattamente il motivo per cui l'arresto va controllato su $b - a$ e non su $|f|$.

22.3 Metodo di Brent–Dekker

Brent [8] corregge il punto debole di Dekker in due modi: sostituisce la secante con un'interpolazione di grado più alto, e introduce un test di progresso che impedisce le sequenze di passi inutilmente piccoli. È l'algoritmo alla base di `uniroot` in R, di `brentq` in SciPy e della routine `zeroin` della maggior parte delle librerie numeriche; una variante con garanzie migliori è analizzata in [9].

22.3.1 Come funziona

L'interpolazione quadratica inversa (IQI). Dati tre punti distinti a, b, c con valori di f a due a due distinti, invece di interpolare $y = f(x)$ con una parabola — che potrebbe non intersecare l'asse x — si interpola la **funzione inversa** $x = f^{-1}(y)$, che vicino a una radice semplice è regolare e ben definita:

$$P(y) = \sum_{j \in \{a,b,c\}} x_j \prod_{\substack{\ell \in \{a,b,c\} \\ \ell \neq j}} \frac{y - f(x_\ell)}{f(x_j) - f(x_\ell)}, \qquad s = P(0).$$

Valutare l'interpolante in $y = 0$ dà direttamente una stima di $f^{-1}(0) = \alpha$. Il candidato esiste sempre, senza radici quadrate e senza casi degeneri, purché i tre valori di f siano distinti; se non lo sono si ripiega sulla secante.

Il test di progresso. È la vera novità rispetto a Dekker. Detta e la lunghezza del passo *penultimo* e d quella del passo precedente, il candidato interpolatorio $s = b + p/q$ viene accettato solo se soddisfa entrambe le condizioni

$$2\,|p| < 3\;|m\,q| - |\tau_1 q| \qquad \text{e} \qquad 2\,|p| < |e\,q|\,,$$

dove $m = (c-b)/2$ è il semipasso di bisezione e $\tau_1 = 2\varepsilon_{\text{mach}}\,|b| + \tau/2$. La prima impone di restare ragionevolmente dentro il bracket; la seconda, che è quella decisiva, impone che **ogni passo sia meno della metà del penultimo**. Una successione di passi interpolatori che si accorciano troppo lentamente viene così interrotta da una bisezione, e questo garantisce che l'ampiezza del bracket si dimezzi almeno ogni due iterazioni.

Il risultato è un algoritmo che nel caso peggiore richiede al più circa il quadrato del numero di valutazioni della bisezione, e che su funzioni regolari converge superlinearmente: la combinazione che ne ha fatto lo standard di fatto.

Metodo di Brent–Dekker

Dati: a, b con $f(a)f(b) \le 0$; tolleranza τ
Restituisce: approssimazione della radice

$c \leftarrow b, \quad f_c \leftarrow f_b, \quad d \leftarrow e \leftarrow b - a$
mentre vero **fai**
  se f_b *e* f_c *hanno lo stesso segno* **allora**
    $c \leftarrow a,\ f_c \leftarrow f_a,\ d \leftarrow e \leftarrow b - a$
  se $|f_c| < |f_b|$ **allora**
    ruota $(a, b, c) \leftarrow (b, c, b)$ ▷ *in b il punto migliore*
  $\tau_1 \leftarrow 2\varepsilon_{\text{mach}}\,|b| + \tau/2, \quad m \leftarrow (c-b)/2$
  se $|m| \le \tau_1$ oppure $f_b = 0$ **allora**
    restituisci b
  se $|e| \ge \tau_1$ e $|f_a| > |f_b|$ **allora**
    se $a = c$ **allora**
      candidato ← **secante**
    altrimenti
      candidato ← **IQI**
    scrivi il candidato come $b + p/q$ con $q > 0$
    se $2\,|p| < \min\big(3\,|mq| - |\tau_1 q|\,,\ |eq|\big)$ **allora**
      $e \leftarrow d,\ d \leftarrow p/q$ ▷ *passo meno della metà del penultimo*
    altrimenti
      $d \leftarrow m,\ e \leftarrow d$ ▷ *bisezione*
  altrimenti
    $d \leftarrow m,\ e \leftarrow d$ ▷ *bisezione*
  $a \leftarrow b, \quad f_a \leftarrow f_b$
  se $|d| > \tau_1$ **allora**
    $b \leftarrow b + d$
  altrimenti
    $b \leftarrow b + \operatorname{sign}(m)\,\tau_1$
  valuta f_b

Figure 22.3: Il metodo di Brent–Dekker: all'interpolazione quadratica inversa si aggiunge il test di progresso $2\,|p| < |eq|$, che impone a ogni passo di essere meno della metà del penultimo e impedisce le sequenze di passi inutilmente piccoli di Dekker.

22.3.2 Esempio numerico

```
brent <- function(f, x1, x2, tol = 1e-12, nmax = 60) {
  a <- x1; b <- x2; c <- x2
  fa <- f(a); fb <- f(b); fc <- fb
  if (fa*fb > 0) stop("gli estremi non racchiudono una radice")
  d <- 0; e <- 0; st <- data.frame()
  for (k in 1:nmax) {
    if (fb*fc > 0) { c<-a; fc<-fa; d<-b-a; e<-d }        # c = contropunto
    if (abs(fc) < abs(fb)) { a<-b;b<-c;c<-a; fa<-fb;fb<-fc;fc<-fa }
    tol1 <- 2*EPS*abs(b) + 0.5*tol
    xm   <- 0.5*(c-b)
    if (abs(xm) <= tol1 || fb == 0) {
      st <- rbind(st, data.frame(k=k,tipo="arresto",x=b,fx=fb,
                                 err=abs(b-alpha),amp=abs(c-b))); break
    }
    tipo <- "bisezione"
    if (abs(e) >= tol1 && abs(fa) > abs(fb)) {
      s <- fb/fa
      if (a == c) {                                      # solo due punti: secante
        p <- 2*xm*s; q <- 1 - s; tipo <- "secante"
      } else {                                           # tre punti distinti: IQI
        q <- fa/fc; r <- fb/fc
        p <- s*(2*xm*q*(q-r) - (b-a)*(r-1))
        q <- (q-1)*(r-1)*(s-1); tipo <- "IQI"
      }
      if (p > 0) q <- -q
      p <- abs(p)
      min1 <- 3*xm*q - abs(tol1*q); min2 <- abs(e*q)
      if (2*p < min(min1, min2)) { e <- d; d <- p/q }     # candidato accettato
      else { d <- xm; e <- d; tipo <- "bisezione" }       # test di progresso fallito
    } else { d <- xm; e <- d }
    a <- b; fa <- fb
    b  <- b + if (abs(d) > tol1) d else sign(xm)*tol1
    fb <- f(b)
    st <- rbind(st, data.frame(k=k,tipo=tipo,x=b,fx=fb,
                               err=abs(b-alpha),amp=abs(c-b)))
  }
  st
}

tabella(brent(f, 2, 3), "Metodo di Brent--Dekker su $x^3-2x-5$, $[2,3]$")
```

Table 22.2: Metodo di Brent–Dekker su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	secante	2.058823529412	-3.91e-01	3.57e-02	9.41e-01
2	IQI	2.095658932291	1.24e-02	1.11e-03	9.04e-01
3	secante	2.094528891117	-2.52e-04	2.26e-05	3.57e-02
4	secante	2.094551467464	-1.57e-07	1.41e-08	1.11e-03
5	IQI	2.094551481542	3.55e-15	4.44e-16	1.11e-03
6	secante	2.094551481542	-5.59e-12	5.00e-13	1.41e-08
7	arresto	2.094551481542	3.55e-15	4.44e-16	5.01e-13

Si riconoscono i tre tipi di passo che l'algoritmo alterna. Il primo è necessariamente una secante, perché servono tre punti distinti per l'IQI; da lì in poi Brent usa l'IQI quando ha tre valori distinti e la secante altrimenti, senza mai dover ricorrere alla bisezione su questo problema.

22.4 Metodo di Ridders

Ridders [34] ottiene la salvaguardia in un modo completamente diverso: **non serve nessun test di ammissibilità**, perché il candidato cade *automaticamente* dentro il bracket. È il motivo per cui l'algoritmo sta in poche righe e non ha casi particolari.

22.4.1 Come funziona

L'idea è di trasformare la funzione in modo da renderla localmente lineare. Dato $[a, b]$ si valuta il punto medio $m = (a + b)/2$ e si cerca un fattore e^{Q} tale che i tre punti

$$\big(a,\ f(a)e^{-Q}\big), \qquad \big(m,\ f(m)\big), \qquad \big(b,\ f(b)e^{Q}\big)$$

siano **allineati**. Imponendo l'allineamento si ottiene un'equazione di secondo grado in e^{Q} la cui soluzione, sostituita nello zero della retta, dà il candidato in forma chiusa:

$$\Delta = \sqrt{f(m)^2 - f(a)f(b)}, \qquad r = m + (m - a)\frac{\operatorname{sign}(f(a))\, f(m)}{\Delta}.$$

Due osservazioni rendono il metodo elegante.

1. **Il candidato è sempre interno.** Se $f(a)f(b) < 0$ allora $-f(a)f(b) > 0$ e quindi $\Delta > |f(m)|$: il fattore moltiplicativo $f(m)/\Delta$ ha modulo minore di 1 e r dista da m meno di $|m - a|$, cioè resta in (a, b). Non serve verificare nulla.
2. **Il segno.** Poiché per ipotesi $f(a)$ e $f(b)$ hanno segno opposto, si ha $\operatorname{sign}(f(a)) = \operatorname{sign}(f(a) - f(b))$, che è la forma in cui la regola compare di solito in letteratura.

Dopo aver valutato $f(r)$ si dispone di quattro punti $a < m < b$ e r, e si sceglie come nuovo bracket la coppia **più stretta** con cambio di segno: (m, r) se $f(m)f(r) < 0$, altrimenti (a, r) o (r, b) secondo il segno.

Il prezzo è che ogni iterazione richiede **due nuove valutazioni** di f (in m e in r). In cambio l'ordine per iterazione è 2, cioè $\sqrt{2} \approx 1.414$ per valutazione: la stessa efficienza del metodo di Newton (Sezione 21.1), ma senza derivate e con il bracket sempre conservato.

Una conseguenza visibile nelle tabelle: il bracket di Ridders si dimezza soltanto, perché uno dei due estremi è sempre il punto medio. L'accuratezza non sta nell'ampiezza dell'intervallo ma nell'iterata r, e per questo il criterio d'arresto naturale è $|r_k - r_{k-1}| \le \tau$ e non $b - a \le \tau$.

22.4.2 Esempio numerico

```
ridders <- function(f, a, b, tol = 1e-12, nmax = 60) {
  fa <- f(a); fb <- f(b)
  if (fa*fb > 0) stop("gli estremi non racchiudono una radice")
  ans <- NA; st <- data.frame()
  for (k in 1:nmax) {
    m  <- 0.5*(a+b); fm <- f(m)
    dd <- sqrt(fm^2 - fa*fb)
    if (dd == 0) break
    r <- m + (m-a)*sign(fa-fb)*fm/dd          # sempre interno ad (a,b)
    if (!is.na(ans) && abs(r-ans) <= tol) break
    ans <- r; fr <- f(r)
    if (fm*fr < 0)      { a<-m; fa<-fm; b<-r; fb<-fr }   # coppia piu' stretta
    else if (fa*fr < 0) { b<-r; fb<-fr }
    else                { a<-r; fa<-fr }
    if (a > b) { t<-a;a<-b;b<-t; t<-fa;fa<-fb;fb<-t }
    st <- rbind(st, data.frame(k=k, tipo="esponenziale", x=r, fx=fr,
                               err=abs(r-alpha), amp=b-a))
    if (fr == 0) break
  }
  st
```

Metodo di Ridders

Dati: a, b con $f(a)f(b) < 0$; tolleranza τ
Restituisce: approssimazione della radice

mentre non convergenza **fai**
  $m \leftarrow (a+b)/2$, **valuta** $f(m)$
  $\Delta \leftarrow \sqrt{f(m)^2 - f(a)f(b)}$
  se $\Delta = 0$ **allora**
    restituisci m
  $r \leftarrow m + (m-a)\,\operatorname{sign}(f(a))\,f(m)/\Delta$ ▷ *sempre interno ad* (a, b)
  se $|r - r_{\text{prec}}| \le \tau$ **allora**
    restituisci r
  valuta $f(r)$
  se $f(r) = 0$ **allora**
    restituisci r
  se $f(m)f(r) < 0$ **allora**
    $(a, b) \leftarrow (m, r)$ ▷ *coppia più stretta*
  altrimenti se $f(a)f(r) < 0$ **allora**
    $b \leftarrow r$
  altrimenti
    $a \leftarrow r$
  riordina in modo che $a < b$

Figure 22.4: Il metodo di Ridders: la correzione esponenziale costruita su a, m e b dà un punto sempre interno al bracket, quindi non serve alcun test di ammissibilità.

```
}

tabella(ridders(f, 2, 3), "Metodo di Ridders su $x^3-2x-5$, $[2,3]$")
```

Table 22.3: Metodo di Ridders su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	esponenziale	2.092522337716	-2.26e-02	2.03e-03	4.07e-01
2	esponenziale	2.094540948859	-1.18e-04	1.05e-05	2.02e-01
3	esponenziale	2.094551467368	-1.58e-07	1.42e-08	1.01e-01
4	esponenziale	2.094551481537	-5.43e-11	4.87e-12	5.04e-02
5	esponenziale	2.094551481542	-7.11e-15	4.44e-16	2.52e-02

Cinque iterazioni (dieci valutazioni di f) portano alla precisione di macchina. Si noti la colonna $b - a$: si dimezza esattamente a ogni passo, come previsto, mentre l'errore cala molto più in fretta.

22.5 Metodo TOMS 748 (Alefeld–Potra–Shi)

L'algoritmo 748 della collezione ACM TOMS [3] è, fra quelli senza derivate, quello con le garanzie teoriche più forti. L'idea è di spingere l'interpolazione inversa fino al **grado tre** e di organizzare il lavoro in *cicli* che contengono più valutazioni, al termine dei quali una contrazione protetta garantisce la riduzione dell'intervallo.

22.5.1 Come funziona

Un ciclo dell'algoritmo 4.2 della memoria originale si compone di quattro fasi.

1. **Prima interpolazione.** Con i punti a, b e un terzo punto d (l'estremo scartato al passo precedente) si costruisce la *quadratica di Newton*: il polinomio di secondo grado che interpola f nei tre punti, di cui si cerca lo zero con due passi di Newton partendo dall'estremo opportuno. Se sono disponibili **quattro** punti a, b, d, e con valori di f distinti si usa invece l'interpolazione **cubica inversa**

$$s = \sum_{j=0}^{3} x_j \prod_{\substack{\ell=0 \\ \ell \neq j}}^{3} \frac{-f_\ell}{f_j - f_\ell}.$$

2. **Seconda interpolazione.** Dopo aver aggiornato il bracket si ripete, questa volta con tre passi di Newton sulla quadratica (o di nuovo con la cubica inversa). Due interpolazioni consecutive per ciclo sono ciò che fa salire l'ordine.
3. **Secante di lunghezza doppia.** Detto u l'estremo con $|f|$ minore, si propone

$$c = u - 2f(u)\,\frac{b-a}{f(b)-f(a)},$$

cioè un passo di secante **raddoppiato**, pensato per saltare oltre la radice e ribaltare il bracket; se $|c-u| > (b-a)/2$ si usa il punto medio.
4. **Contrazione garantita.** Se al termine delle tre fasi l'intervallo non si è ridotto almeno di un fattore $\mu = 1/2$ rispetto all'inizio del ciclo, si esegue un passo di bisezione. È questa la salvaguardia che rende l'algoritmo robusto.

Gli autori dimostrano per l'algoritmo 4.2 un **R-ordine almeno** $2 + \sqrt{7} \approx 4.65$ con asintoticamente tre valutazioni per ciclo, cioè un'efficienza di $4.65^{1/3} \approx 1.67$; la variante più semplice 4.1 ha R-ordine $1 + \sqrt{3} \approx 2.73$ con due valutazioni, cioè $2.73^{1/2} \approx 1.65$. Sono i valori più alti fra i metodi senza derivate di questo capitolo. Va però sottolineato che **non basta applicare ripetutamente la sola formula cubica** per ottenere TOMS 748: sono i test di ammissibilità e la contrazione protetta a fare la differenza.

22.5.2 Esempio numerico

```
toms748 <- function(f, a, b, tol = 1e-12, nmax = 20) {
  bracket <- function(a,b,c,fa,fb) {          # valuta f(c) e aggiorna (a,b,d)
    if (!is.finite(c) || c <= a || c >= b) c <- 0.5*(a+b)
    fc <- f(c)
    if (fc == 0) return(list(a=c,b=c,d=c,fa=0,fb=0,fd=0))
    if (fa*fc < 0) list(a=a,b=c,d=b,fa=fa,fb=fc,fd=fb)
    else           list(a=c,b=b,d=a,fa=fc,fb=fb,fd=fa)
  }
  newton_quad <- function(a,b,d,fa,fb,fd,kk) {        # kk passi di Newton
    A1 <- (fb-fa)/(b-a); A2 <- ((fd-fb)/(d-b) - A1)/(d-a)
    if (!is.finite(A1) || A1 == 0) return(0.5*(a+b))
    if (!is.finite(A2) || A2 == 0) {
      r <- a - fa/A1
      return(if (!is.finite(r) || r <= a || r >= b) 0.5*(a+b) else r)
    }
    r <- if (A2*fa > 0) a else b
    for (i in 1:kk) {
      pr  <- fa + A1*(r-a) + A2*(r-a)*(r-b)
      dpr <- A1 + A2*(2*r - a - b)
      if (dpr == 0) return(0.5*(a+b))
      r <- r - pr/dpr
    }
    if (!is.finite(r) || r <= a || r >= b) 0.5*(a+b) else r
  }
  ipzero <- function(a,b,c,d,fa,fb,fc,fd) {           # cubica inversa
```

TOMS 748 (versione 4.2, un ciclo)

Dati: a, b con $f(a)f(b) < 0$; d punto scartato; e punto scartato precedente
Restituisce: bracket aggiornato dopo un giro completo

Fase 1
se a, b, d, e *hanno valori di* f *a due a due distinti* **allora**
    $c \leftarrow$ cubica inversa(a, b, d, e)
altrimenti
    $c \leftarrow$ quadratica di Newton(a, b, d) con 2 passi
se $c \notin (a, b)$ **allora**
    $c \leftarrow$ quadratica di Newton(a, b, d) ▷ *il candidato deve restare nel bracket*
valuta $f(c)$ e aggiorna (a, b, d)

Fase 2
come la fase 1, ma con 3 passi di Newton sulla quadratica;
valuta $f(c)$ e aggiorna (a, b, d)

Fase 3
$u \leftarrow$ estremo con $|f|$ minore
$c \leftarrow u - 2f(u)\dfrac{b-a}{f(b)-f(a)}$ ▷ *doppia secante*
se $|c-u| > (b-a)/2$ **allora**
    $c \leftarrow (a+b)/2$
valuta $f(c)$ e aggiorna (a, b, d)

Fase 4
se $b - a > \mu\,(b-a)_{\text{iniziale}}$ con $\mu = 1/2$ **allora**
    esegui un passo di bisezione ▷ *garantisce il dimezzamento*

Figure 22.5: Un ciclo di TOMS 748: quattro fasi di interpolazione di grado crescente, chiuse da un passo di bisezione se il bracket non si è ridotto abbastanza.

```
    q11<-(c-d)*fc/(fd-fc); q21<-(b-c)*fb/(fc-fb); q31<-(a-b)*fa/(fb-fa)
    d21<-(b-c)*fc/(fc-fb); d31<-(a-b)*fb/(fb-fa)
    q22<-(d21-q11)*fb/(fd-fb); d32<-(d31-q21)*fc/(fc-fa)
    q32<-d32; q33<-(d32-q22)*fa/(fd-fa)
    a + q31 + q32 + q33
  }
  fa <- f(a); fb <- f(b)
  if (fa*fb > 0) stop("gli estremi non racchiudono una radice")
  migliore <- function(a,b,fa,fb) if (abs(fa) < abs(fb)) c(a,fa) else c(b,fb)
  st <- data.frame()
  B <- bracket(a, b, a - fa*(b-a)/(fb-fa), fa, fb)      # passo iniziale: secante
  a<-B$a; b<-B$b; d<-B$d; fa<-B$fa; fb<-B$fb; fd<-B$fd
  mb <- migliore(a,b,fa,fb)
  st <- rbind(st, data.frame(k=1,tipo="secante",x=mb[1],fx=mb[2],
                             err=abs(mb[1]-alpha),amp=b-a))
  e <- NA; fe <- NA
  for (k in 2:nmax) {
    ab0 <- b - a
    for (passi in c(2,3)) {                             # fasi 1 e 2
      quattro <- !is.na(e) && length(unique(c(fa,fb,fd,fe))) == 4
      cc <- if (quattro) ipzero(a,b,d,e,fa,fb,fd,fe)
            else         newton_quad(a,b,d,fa,fb,fd,passi)
      if (!is.finite(cc) || cc <= a || cc >= b)
        cc <- newton_quad(a,b,d,fa,fb,fd,passi)
      if (passi == 2) { e<-d; fe<-fd }
      B <- bracket(a,b,cc,fa,fb); a<-B$a;b<-B$b;d<-B$d;fa<-B$fa;fb<-B$fb;fd<-B$fd
      if (b - a < tol) break
    }
    if (b - a < tol) {
      mb <- migliore(a,b,fa,fb)
      st <- rbind(st, data.frame(k=k,tipo="ciclo APS",x=mb[1],fx=mb[2],
                                 err=abs(mb[1]-alpha),amp=b-a))
      break
    }
    u  <- if (abs(fa) < abs(fb)) a  else b               # fase 3
    fu <- if (abs(fa) < abs(fb)) fa else fb
    cc <- u - 2*fu*(b-a)/(fb-fa)
    if (!is.finite(cc) || abs(cc-u) > 0.5*(b-a)) cc <- 0.5*(a+b)
    e<-d; fe<-fd
    B <- bracket(a,b,cc,fa,fb); a<-B$a;b<-B$b;d<-B$d;fa<-B$fa;fb<-B$fb;fd<-B$fd
    if ((b-a) > 0.5*ab0) {                              # fase 4: contrazione
      e<-d; fe<-fd
      B <- bracket(a,b,0.5*(a+b),fa,fb)
      a<-B$a;b<-B$b;d<-B$d;fa<-B$fa;fb<-B$fb;fd<-B$fd
    }
    mb <- migliore(a,b,fa,fb)
    st <- rbind(st, data.frame(k=k,tipo="ciclo APS",x=mb[1],fx=mb[2],
                               err=abs(mb[1]-alpha),amp=b-a))
    if (b-a < tol) break
  }
  st
}

tabella(toms748(f, 2, 3), "TOMS 748 su $x^3-2x-5$, $[2,3]$ (una riga per ciclo)")
```

Table 22.4: TOMS 748 su $x^3 - 2x - 5$, $[2, 3]$ (una riga per ciclo)

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	secante	2.058823529412	-3.91e-01	3.57e-02	9.41e-01

k	passo	x_k	$f(x_k)$	$\lvert x_k-\alpha\rvert$	$b-a$
2	ciclo APS	2.103044127440	9.52e-02	8.49e-03	1.91e-02
3	ciclo APS	2.094601424679	5.57e-04	4.99e-05	9.99e-05
4	ciclo APS	2.094551481542	-2.31e-12	2.07e-13	4.14e-13

Quattro cicli, dodici valutazioni di f in tutto. La colonna $b-a$ mostra l'effetto della contrazione garantita: l'ampiezza del bracket crolla di diversi ordini di grandezza per ciclo, cosa che nessuno dei metodi precedenti riesce a fare.

22.6 Metodo di Chandrupatla

Chandrupatla [12] riprende l'interpolazione quadratica inversa di Brent ma la riscrive in **coordinate normalizzate**, ottenendo un test di ammissibilità sorprendentemente semplice.

22.6.1 Come funziona

Si parametrizza il bracket con

$$x = a + t\,(b-a), \qquad 0 < t < 1,$$

cosicché il candidato è individuato dal solo numero t: verificare che cada dentro l'intervallo significa verificare $0 < t < 1$, senza confronti fra numeri di scala diversa. Con tre punti a, b, c l'interpolazione quadratica inversa fornisce

$$t_{\mathrm{IQI}} = \frac{f(b)}{f(a)-f(b)}\cdot\frac{f(c)}{f(a)-f(c)} + \frac{c-a}{b-a}\cdot\frac{f(a)}{f(b)-f(a)}\cdot\frac{f(c)}{f(b)-f(c)}.$$

Il contributo originale è il criterio con cui si decide se fidarsi. Definite le due quantità adimensionali

$$\xi = \frac{a-b}{c-b}, \qquad \phi = \frac{f(a)-f(b)}{f(c)-f(b)},$$

che misurano quanto la configurazione dei tre punti sia «geometricamente sana», l'IQI viene accettata **solo se**

$$1-\sqrt{1-\xi} \;<\; \phi \;<\; \sqrt{\xi},$$

altrimenti si pone $t = 1/2$, cioè si bisseca. La condizione ha un'interpretazione diretta: ϕ confronta le variazioni di f con le distanze ξ, e le due disuguaglianze delimitano la regione in cui la parabola inversa non è troppo distorta. Infine t viene comunque confinato in $[t_\ell, 1-t_\ell]$, con t_ℓ proporzionale alla tolleranza, in modo che il nuovo punto non si incolli mai a un estremo.

Il metodo è senza derivate, usa **una sola valutazione per iterazione** e conserva il cambio di segno. Nei confronti sperimentali si comporta come Brent, e spesso un poco meglio, con un'implementazione molto più breve.

22.6.2 Esempio numerico

```
chandrupatla <- function(f, a, b, tol = 1e-12, nmax = 60) {
  fa <- f(a); fb <- f(b)
  if (fa*fb > 0) stop("gli estremi non racchiudono una radice")
  c <- a; fc <- fa; t <- 0.5; st <- data.frame()
  for (k in 1:nmax) {
    xt <- a + t*(b-a); ft <- f(xt)
    if (sign(ft) == sign(fa)) { c<-a; fc<-fa; a<-xt; fa<-ft }
    else { c<-b; fc<-fb; b<-a; fb<-fa; a<-xt; fa<-ft }
```

Metodo di Chandrupatla

Dati: a, b con $f(a)f(b) < 0$; tolleranza τ
Restituisce: approssimazione della radice

$t \leftarrow 1/2$
mentre non convergenza **fai**
  $x_t \leftarrow a + t(b-a)$, **valuta** $f(x_t)$
  aggiorna (a, b, c) conservando il cambio di segno,
    con a = punto appena calcolato
  $x_m \leftarrow$ fra a e b quello con $|f|$ minore
  $t_\ell \leftarrow \big(2\varepsilon_{\text{mach}}\,|x_m| + \tau\big) \big/ |b-a|$
  se $t_\ell > 1/2$ oppure $f(x_m) = 0$ **allora**
    restituisci x_m
  $\xi \leftarrow \dfrac{a-b}{c-b}$, $\quad \phi \leftarrow \dfrac{f(a)-f(b)}{f(c)-f(b)}$
  se $1 - \sqrt{1-\xi} < \phi < \sqrt{\xi}$ **allora**
    $t \leftarrow t_{\text{IQI}}$ ▷ *test di ammissibilità*
  altrimenti
    $t \leftarrow 1/2$ ▷ *bisezione*
  $t \leftarrow \min\big(\max(t, t_\ell),\, 1 - t_\ell\big)$ ▷ *resta lontano dagli estremi*

Figure 22.6: Il metodo di Chandrupatla: il test di ammissibilità è una condizione esplicita sulle due quantità adimensionali ξ e ϕ, e il passo è espresso come frazione t dell'ampiezza del bracket.

```
    if (abs(fa) < abs(fb)) { xm<-a; fm<-fa } else { xm<-b; fm<-fb }
    st <- rbind(st, data.frame(k=k, tipo=if (t==0.5) "bisezione" else "IQI",
                               x=xm, fx=fm, err=abs(xm-alpha), amp=abs(b-a)))
    tola <- 2*EPS*abs(xm) + tol
    tl   <- tola/abs(b-a)
    if (tl > 0.5 || fm == 0) break
    xi <- (a-b)/(c-b); phi <- (fa-fb)/(fc-fb)
    t  <- if (1 - sqrt(1-xi) < phi && phi < sqrt(xi))      # test di ammissibilita'
            fa/(fb-fa)*fc/(fb-fc) + (c-a)/(b-a)*fa/(fc-fa)*fb/(fc-fb)
          else 0.5
    t  <- min(max(t, tl), 1-tl)
  }
  st
}

tabella(chandrupatla(f, 2, 3), "Metodo di Chandrupatla su $x^3-2x-5$, $[2,3]$")
```

Table 22.5: Metodo di Chandrupatla su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	bisezione	2.000000000000	-1.00e+00	9.46e-02	5.00e-01
2	IQI	2.084497813674	-1.12e-01	1.01e-02	4.16e-01
3	IQI	2.094728156313	1.97e-03	1.77e-04	1.02e-02
4	IQI	2.094551168821	-3.49e-06	3.13e-07	1.77e-04
5	IQI	2.094551481543	3.39e-12	3.04e-13	3.13e-07
6	IQI	2.094551481543	3.39e-12	3.04e-13	1.00e-12

Il primo passo è per forza una bisezione (t è inizializzato a $1/2$ e non ci sono ancora tre punti); da lì in poi il test di ammissibilità è sempre superato e l'IQI porta alla precisione di macchina in cinque passi, con una sola valutazione ciascuno.

22.7 Metodo ITP (Interpolate–Truncate–Project)

ITP [29] è il più recente ed è l'unico che riesce a essere contemporaneamente **superlineare sui problemi regolari** e **minmax-ottimale nel caso peggiore**: non spende mai, nemmeno sulla funzione più ostile, più iterazioni della bisezione (a meno di un margine fissato dall'utente).

22.7.1 Come funziona

Il nome descrive le tre fasi di cui si compone ogni passo.

Interpolazione. Si parte dal candidato della falsa posizione e dal punto medio:

$$x_f = \frac{a f(b) - b f(a)}{f(b) - f(a)}, \qquad m = \frac{a+b}{2}.$$

Troncamento. La falsa posizione ha il difetto noto di incollarsi a un estremo (Sezione 20.8). Lo si corregge spostando x_f *verso il centro* di una quantità δ:

$$x_t = x_f + \sigma \min\{\delta, |m - x_f|\}, \qquad \sigma = \operatorname{sign}(m - x_f), \qquad \delta = \kappa_1 (b-a)^{\kappa_2}.$$

Il minimo impedisce di oltrepassare il punto medio. L'esponente κ_2 controlla quanto rapidamente il correttivo svanisce al restringersi dell'intervallo: con $\kappa_2 > 1$ la correzione diventa trascurabile vicino alla radice, e l'ordine superlineare della falsa posizione non viene distrutto.

Proiezione. È la fase che fornisce la garanzia. Si fissa

$$n_{1/2} = \left\lceil \log_2 \frac{b_0 - a_0}{2\varepsilon} \right\rceil,$$

il numero di passi che impiegherebbe la bisezione, e si concede uno *slack* di $n_0 \geq 0$ iterazioni in più. Detta j l'iterazione corrente, il raggio ammesso attorno al punto medio è

$$\rho_j = \max\left\{ \varepsilon\, 2^{\,n_{1/2}+n_0-j} - \frac{b-a}{2},\ 0 \right\},$$

e il candidato finale è il troncato **proiettato** su $[m - \rho_j,\ m + \rho_j]$:

$$x_{\mathrm{ITP}} = \begin{cases} x_t, & |x_t - m| \leq \rho_j, \\ m - \sigma \rho_j, & \text{altrimenti.} \end{cases}$$

Il significato di ρ_j è un *budget*: misura di quanto l'algoritmo è in anticipo sulla tabella di marcia della bisezione. Finché l'interpolazione lavora bene il bracket si restringe più in fretta di 2^{-j}, il budget resta ampio e il candidato interpolatorio passa indisturbato; se invece l'interpolazione è inefficace il budget si azzera, $\rho_j = 0$, e il candidato viene forzato nel punto medio. Da qui la garanzia: **al più $n_{1/2} + n_0$ iterazioni, sempre.**

I parametri raccomandati sono $\kappa_1 = 0.2/(b_0 - a_0)$, $\kappa_2 = 2$ e $n_0 = 1$. Una salvaguardia di aritmetica finita è necessaria in più: se il candidato arrotonda esattamente su un estremo, il bracket smetterebbe di restringersi, e in quel caso si usa il punto medio.

22.7.2 Esempio numerico

```
itp <- function(f, a, b, eps = 1e-12, k1 = NULL, k2 = 2, n0 = 1, nmax = 80) {
  ya <- f(a); yb <- f(b)
  if (ya*yb > 0) stop("gli estremi non racchiudono una radice")
  if (is.null(k1)) k1 <- 0.2/(b-a)
  n12    <- ceiling(log2((b-a)/(2*eps)))
  nmaxit <- n12 + n0
  j <- 0; st <- data.frame()
  while (b - a > 2*eps && j < nmax) {
    xf <- (a*yb - b*ya)/(yb - ya)                    # interpolazione
```

Metodo ITP

Dati: a, b con $f(a)f(b) < 0$; tolleranza $\varepsilon > 0$;
parametri $\kappa_1 > 0$, $\kappa_2 \in [1, 1+\varphi)$, $n_0 \geq 0$
Restituisce: approssimazione della radice con errore $\leq \varepsilon$

$y_a \leftarrow f(a), \quad y_b \leftarrow f(b)$
$n_{1/2} \leftarrow \lceil \log_2\big((b-a)/(2\varepsilon)\big) \rceil, \quad n_{\max} \leftarrow n_{1/2} + n_0, \quad j \leftarrow 0$
mentre $b - a > 2\varepsilon$ **fai**
  $x_f \leftarrow \dfrac{a\,y_b - b\,y_a}{y_b - y_a}, \quad m \leftarrow \dfrac{a+b}{2}$ ▷ interpolazione
  $\sigma \leftarrow \operatorname{sign}(m - x_f), \quad \delta \leftarrow \kappa_1 (b-a)^{\kappa_2}$ ▷ troncamento
  se $\delta \leq |m - x_f|$ **allora**
    $x_t \leftarrow x_f + \sigma\delta$
  altrimenti
    $x_t \leftarrow m$
  $\rho \leftarrow \max\!\left(\varepsilon\, 2^{\,n_{\max} - j} - \dfrac{b-a}{2},\, 0\right)$ ▷ proiezione
  se $|x_t - m| \leq \rho$ **allora**
    $x_{\mathrm{ITP}} \leftarrow x_t$
  altrimenti
    $x_{\mathrm{ITP}} \leftarrow m - \sigma\rho$
  se $x_{\mathrm{ITP}} \leq a$ oppure $x_{\mathrm{ITP}} \geq b$ **allora**
    $x_{\mathrm{ITP}} \leftarrow m$ ▷ *salvaguardia in virgola mobile*
  $y \leftarrow f(x_{\mathrm{ITP}})$ ▷ aggiornamento
  se $\operatorname{sign} y = \operatorname{sign} y_a$ **allora**
    $a \leftarrow x_{\mathrm{ITP}}, \; y_a \leftarrow y$
  altrimenti se $\operatorname{sign} y = \operatorname{sign} y_b$ **allora**
    $b \leftarrow x_{\mathrm{ITP}}, \; y_b \leftarrow y$
  altrimenti
    $a \leftarrow b \leftarrow x_{\mathrm{ITP}}$ ▷ *radice esatta*
  $j \leftarrow j + 1$
restituisci $(a+b)/2$

Figure 22.7: Il metodo ITP: interpolazione, troncamento e proiezione. La proiezione su un intorno del punto medio di raggio ρ garantisce che non si facciano mai più iterazioni della bisezione, più le n_0 concesse in partenza.

```
    m  <- 0.5*(a+b)
    sg <- sign(m - xf)
    dl <- k1*(b-a)^k2                                   # troncamento
    xt <- if (dl <= abs(m-xf)) xf + sg*dl else m
    r  <- max(eps*2^(nmaxit - j) - 0.5*(b-a), 0)        # proiezione
    xi <- if (abs(xt-m) <= r) xt else m - sg*r
    tipo <- if (abs(xt-m) <= r) {
              if (dl <= abs(m-xf)) "falsa pos. troncata" else "punto medio"
            } else "proiezione"
    if (xi <= a || xi >= b) { xi <- m; tipo <- "punto medio" }   # salvaguardia fp
    y <- f(xi)
    if (y*ya > 0) { a <- xi; ya <- y } else if (y*yb > 0) { b <- xi; yb <- y }
    else { a <- xi; b <- xi }
    j <- j + 1
    st <- rbind(st, data.frame(k=j, tipo=tipo, x=xi, fx=y,
                               err=abs(xi-alpha), amp=b-a))
  }
  st
}

tabella(itp(f, 2, 3), "Metodo ITP su $x^3-2x-5$, $[2,3]$")
```

Table 22.6: Metodo ITP su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	falsa pos. troncata	2.258823529412	2.01e+00	1.64e-01	2.59e-01
2	falsa pos. troncata	2.099456956203	5.49e-02	4.91e-03	9.95e-02
3	falsa pos. troncata	2.092302285104	-2.51e-02	2.25e-03	7.15e-03
4	falsa pos. troncata	2.094555514511	4.50e-05	4.03e-06	2.25e-03
5	falsa pos. troncata	2.094550461022	-1.14e-05	1.02e-06	5.05e-06
6	falsa pos. troncata	2.094551481545	3.11e-11	2.79e-12	1.02e-06
7	falsa pos. troncata	2.094551481542	-2.33e-12	2.08e-13	3.00e-12
8	falsa pos. troncata	2.094551481542	-8.88e-16	0.00e+00	2.79e-12
9	punto medio	2.094551481544	1.56e-11	1.39e-12	1.39e-12

Su questo problema la falsa posizione troncata è sempre accettata e il budget non diventa mai vincolante: nove valutazioni portano alla precisione di macchina. Vedremo in Sezione 22.13 un caso opposto, in cui il budget si esaurisce e ITP si riduce di fatto alla bisezione — comportamento non brillante ma, per costruzione, mai peggiore di essa.

22.8 Gli ibridi con derivate: uno schema comune

I quattro metodi che seguono differiscono soltanto per la **formula del candidato**: condividono bracket, test di ammissibilità e criterio d'arresto. Conviene quindi scrivere una volta sola il driver e passargli il passo come argomento.

22.8.1 Il test di ammissibilità

Rispetto ai metodi senza derivate c'è una differenza importante: un passo di tipo Newton **non restringe il bracket**. Se tutte le iterate arrivano alla radice dallo stesso lato, uno dei due estremi non si muove mai, esattamente come nella *regula falsi*. Perciò qui non si può usare $b - a \le \tau$ come criterio d'arresto: si usa l'incremento $|x_{k+1} - x_k|$, come nella classica routine `rtsafe` [32].

Il candidato c viene rifiutato in due casi:

- **esce dal bracket**, $c < a$ oppure $c > b$: il modello locale sta mentendo, e accettare il passo significherebbe perdere la radice;
- **progredisce troppo lentamente**, $|c - x_k| > \frac{1}{2}|x_k - x_{k-1}|$: il passo non dimezza nemmeno l'incremento precedente, quindi non sta facendo meglio di una bisezione e tanto vale eseguire quest'ultima, che almeno riduce l'intervallo.

In entrambi i casi si ripiega sul punto medio. Si noti che il confronto $c < a$, $c > b$ è **non stretto**: dopo l'aggiornamento del bracket l'iterata corrente *è* uno degli estremi, e un test stretto rifiuterebbe sistematicamente il candidato una volta raggiunta la convergenza, degradando il metodo a bisezione proprio quando sta per finire.

```
ibrido <- function(f, passo, a, b, df = NULL, ddf = NULL, dddf = NULL,
                   tol = 1e-12, nmax = 60) {
  fa <- f(a); fb <- f(b)
  if (fa*fb > 0) stop("gli estremi non racchiudono una radice")
  amp0 <- b - a
  x <- 0.5*(a+b); fx <- f(x)
  if (sign(fx) == sign(fa)) { a<-x; fa<-fx } else { b<-x; fb<-fx }
  dxold <- amp0; dx <- dxold; st <- data.frame()
  for (k in 1:nmax) {
    cand  <- tryCatch(passo(x, f, df, ddf, dddf), error = function(e) NA)
    fuori <- !is.finite(cand) || cand < a || cand > b
    lento <- is.finite(cand) && abs(cand - x) > 0.5*abs(dxold)
    if (fuori || lento) { dxold<-dx; dx <- 0.5*(a+b) - x; tipo <- "bisezione" }
    else                { dxold<-dx; dx <- cand - x;      tipo <- "alto ordine" }
    x <- x + dx; fx <- f(x)
    if (sign(fx) == sign(fa)) { a<-x; fa<-fx } else { b<-x; fb<-fx }
    st <- rbind(st, data.frame(k=k, tipo=tipo, x=x, fx=fx,
                               err=abs(x-alpha), amp=b-a))
    if (abs(dx) <= tol*max(1,abs(x)) || fx == 0) break
  }
  st
}
```

Ibrido con salvaguardia e passo di ordine alto

Dati: a, b con $f(a)f(b) \le 0$; funzione `passo`; tolleranza τ
Restituisce: approssimazione della radice

$x \leftarrow (a+b)/2$, **valuta** $f(x)$, aggiorna il bracket
$\Delta_{\text{prec}} \leftarrow b_0 - a_0$
mentre non convergenza **fai**
  $c \leftarrow \texttt{passo}(x)$
  se *c non è finito, oppure $c < a$ o $c > b$,*
  oppure $|c - x| > \frac{1}{2}|\Delta_{\text{prec}}|$ **allora**
    $\Delta \leftarrow \dfrac{a+b}{2} - x$ ▷ *bisezione*
  altrimenti
    $\Delta \leftarrow c - x$ ▷ *passo di ordine alto*
  $x \leftarrow x + \Delta$, **valuta** $f(x)$,
    aggiorna il bracket conservando il segno
  se $|\Delta| \le \tau \max(1, |x|)$ oppure $f(x) = 0$ **allora**
    restituisci x
  $\Delta_{\text{prec}} \leftarrow \Delta$

Figure 22.8: Lo schema ibrido generico: `passo` è un qualunque metodo di ordine alto (Newton, Halley, Chebyshev, Householder) e viene accettato solo se resta nel bracket e dimezza almeno il passo precedente.

```
passo_newton      <- function(x,f,df,ddf,dddf) x - f(x)/df(x)
passo_halley      <- function(x,f,df,ddf,dddf)
  x - 2*f(x)*df(x)/(2*df(x)^2 - f(x)*ddf(x))
passo_chebyshev   <- function(x,f,df,ddf,dddf) {
  u <- f(x)/df(x); x - u*(1 + 0.5*f(x)*ddf(x)/df(x)^2)
}
passo_householder <- function(x,f,df,ddf,dddf)
  x - (6*f(x)*df(x)^2 - 3*f(x)^2*ddf(x)) /
      (6*df(x)^3 - 6*f(x)*df(x)*ddf(x) + f(x)^2*dddf(x))
```

22.9 Metodo Newton–Bisection

Il candidato è il passo di Newton (Sezione 20.6)

$$n = x - \frac{f(x)}{f'(x)}.$$

È l'ibrido più semplice e quello che chiarisce meglio il meccanismo: vicino a una radice semplice i passi di Newton vengono accettati e si recupera la convergenza quadratica locale; lontano, dove Newton ciclerebbe o divergerebbe, il test li rifiuta e la bisezione riporta l'iterata nella regione buona. Il costo è di **due valutazioni per iterazione** (f e f').

```
tabella(ibrido(f, passo_newton, 2, 3, df, ddf, dddf),
        "Newton--Bisection su $x^3-2x-5$, $[2,3]$")
```

Table 22.7: Newton–Bisection su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	alto ordine	2.164179104478	8.08e-01	6.96e-02	1.64e-01
2	alto ordine	2.097135355811	2.89e-02	2.58e-03	9.71e-02
3	alto ordine	2.094555232390	4.19e-05	3.75e-06	9.46e-02
4	alto ordine	2.094551481550	8.84e-11	7.92e-12	9.46e-02
5	alto ordine	2.094551481542	-8.88e-16	0.00e+00	7.92e-12
6	alto ordine	2.094551481542	-8.88e-16	0.00e+00	7.92e-12

La prova decisiva è il caso in cui il metodo di Newton puro fallisce in modo spettacolare, $f(x) = \arctan x$ a partire da $x_0 = 20$ (Sezione 20.6.4):

```
alpha_sv <- alpha; alpha <- 0
cat("Newton puro da x0 = 20:\n")
```

```
Newton puro da x0 = 20:
```

```
x <- 20; for (k in 1:5) x <- c(x, x[k] - atan(x[k])*(1 + x[k]^2)); print(x)
```

```
[1]  2.000000e+01 -5.898560e+02  5.453492e+05 -4.671628e+11  3.428122e+23
[6] -1.846004e+47
```

```
tab_at <- ibrido(atan, passo_newton, -5, 20, function(x) 1/(1+x^2))
alpha <- alpha_sv
tabella(tab_at, "Newton--Bisection su $\\arctan x$, $[-5,20]$")
```

Table 22.8: Newton–Bisection su arctan x, $[-5, 20]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	bisezione	1.250000000000	8.96e-01	1.25e+00	6.25e+00
2	alto ordine	-1.046141922964	-8.08e-01	1.05e+00	2.30e+00
3	alto ordine	0.646028604063	5.74e-01	6.46e-01	1.69e+00
4	alto ordine	-0.166934309609	-1.65e-01	1.67e-01	8.13e-01
5	alto ordine	0.003084231094	3.08e-03	3.08e-03	1.70e-01
6	alto ordine	-0.000000019559	-1.96e-08	1.96e-08	3.08e-03
7	alto ordine	0.000000000000	6.62e-24	6.62e-24	1.96e-08
8	alto ordine	0.000000000000	0.00e+00	0.00e+00	1.96e-08

Newton puro diverge a 10^{47} in cinque passi; l'ibrido parte con una bisezione, poi accetta i passi di Newton e converge esattamente a zero. È il capitolo intero in una tabella.

22.10 Metodo Halley–Bisection

Il candidato è il passo di Halley (Sezione 20.12),

$$h = x - \frac{2f(x)f'(x)}{2f'(x)^2 - f(x)f''(x)},$$

che su radici semplici ha ordine locale tre. Oltre ai test comuni va verificato che il denominatore non sia troppo piccolo: si annulla quando $2f'^2 = ff''$, cioè quando la parabola tangente degenera. Il costo è di **tre valutazioni per iterazione**.

```
tabella(ibrido(f, passo_halley, 2, 3, df, ddf, dddf),
        "Halley--Bisection su $x^3-2x-5$, $[2,3]$")
```

Table 22.9: Halley–Bisection su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	alto ordine	2.104745673833	1.14e-01	1.02e-02	1.05e-01
2	alto ordine	2.094551719900	2.66e-06	2.38e-07	9.46e-02
3	alto ordine	2.094551481542	-8.88e-16	0.00e+00	2.38e-07
4	alto ordine	2.094551481542	-8.88e-16	0.00e+00	2.38e-07

22.11 Metodo Chebyshev–Bisection

Posti $u = f(x)/f'(x)$ e $L = f(x)f''(x)/f'(x)^2$ (il *grado di convessità logaritmica*), lo sviluppo di Taylor dell'inversa locale fino al secondo ordine in $f(x)$ dà il passo di Chebyshev [41]

$$c = x - u\left(1 + \frac{L}{2}\right) = x - \frac{f(x)}{f'(x)} - \frac{f''(x)f(x)^2}{2f'(x)^3}.$$

Ha lo stesso ordine tre di Halley e lo stesso costo, ma una proprietà numerica diversa: il termine correttivo è **additivo** anziché al denominatore, quindi quando f'' è grande il passo di Chebyshev si allunga senza limite mentre quello di Halley resta controllato. Halley è infatti l'approssimante razionale di Chebyshev, e in presenza di forte convessità è il più robusto dei due; il test di ammissibilità del driver rende comunque sicuri entrambi.

```
tabella(ibrido(f, passo_chebyshev, 2, 3, df, ddf, dddf),
        "Chebyshev--Bisection su $x^3-2x-5$, $[2,3]$")
```

Table 22.10: Chebyshev–Bisection su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	alto ordine	2.113682534088	2.16e-01	1.91e-02	1.14e-01
2	alto ordine	2.094555161991	4.11e-05	3.68e-06	9.46e-02
3	alto ordine	2.094551481542	-8.88e-16	0.00e+00	3.68e-06
4	alto ordine	2.094551481542	-8.88e-16	0.00e+00	3.68e-06

22.12 Metodo Householder–Bisection

La famiglia di Householder [24] generalizza Newton e Halley. Posto $g = 1/f$, il metodo di ordine locale $d + 1$ è

$$H_d(x) = x + d\,\frac{g^{(d-1)}(x)}{g^{(d)}(x)},$$

dove le derivate siano definite e il denominatore non si annulli. Svolgendo le derivate si trovano i primi tre membri della famiglia:

$$\begin{aligned}
d = 1 &: \quad H_1 = x - \frac{f}{f'} && \text{(Newton, ordine 2)},\\
d = 2 &: \quad H_2 = x - \frac{2ff'}{2f'^2 - ff''} && \text{(Halley, ordine 3)},\\
d = 3 &: \quad H_3 = x - \frac{6ff'^2 - 3f^2 f''}{6f'^3 - 6ff'f'' + f^2 f'''} && \text{(ordine 4)}.
\end{aligned}$$

Householder–Bisection designa dunque una *famiglia* di ibridi, non un singolo algoritmo: occorre specificare d. In pratica si usa raramente $d > 2$: l'indice di efficienza $(d+1)^{1/(d+1)}$ vale 1.414 per $d = 1$, 1.442 per $d = 2$, 1.414 per $d = 3$ e poi decresce, mentre il costo e la fragilità numerica delle derivate alte crescono. Il massimo è in $d = 2$, cioè Halley: è la ragione teorica per cui la famiglia di Householder si ferma, in pratica, a Halley.

Nella tabella seguente si usa $d = 3$, con $f''' = 6$.

```
tabella(ibrido(f, passo_householder, 2, 3, df, ddf, dddf),
        "Householder--Bisection ($d=3$) su $x^3-2x-5$, $[2,3]$")
```

Table 22.11: Householder–Bisection ($d = 3$) su $x^3 - 2x - 5$, $[2, 3]$

k	passo	x_k	$f(x_k)$	$\lvert x_k - \alpha\rvert$	$b - a$
1	alto ordine	2.095856605977	1.46e-02	1.31e-03	9.59e-02
2	alto ordine	2.094551481543	2.51e-12	2.25e-13	9.46e-02
3	alto ordine	2.094551481542	-8.88e-16	0.00e+00	2.25e-13

Tre passi portano alla precisione di macchina, contro i sei di Newton–Bisection — ma ciascuno costa **quattro** valutazioni (f, f', f'', f''') contro le due di Newton, e nel conteggio complessivo del prossimo paragrafo il vantaggio sparisce del tutto.

22.13 Confronto sperimentale

Le tabelle precedenti confrontano i metodi *per iterazione*, che come sappiamo (Sezione 21.1) non è il criterio giusto. Contiamo allora le **valutazioni di funzione e di derivate** necessarie a ciascun metodo per raggiungere $|x_k - \alpha| \le 10^{-10}$, su cinque problemi di natura diversa:

Problema	Caratteristica
$x^3 - 2x - 5$ su $[2, 3]$	radice semplice, funzione ben condizionata
$x^{10} - 1$ su $[0.5, 1.3]$	quasi piatta a sinistra, ripidissima a destra: il caso patologico della falsa posizione
$e^{-x} - x$ su $[0, 1]$	funzione regolare, derivate di taglia moderata
$\arctan x$ su $[-5, 20]$	Newton puro diverge, bracket molto ampio
$(x - 1)^3$ su $[0, 3]$	radice **tripla**: l'interpolazione è inefficace

Table 22.12: Valutazioni di f e delle derivate per ottenere $|x_k - \alpha| \le 10^{-10}$

	$x^3 - 2x - 5$	$x^{10} - 1$	$e^{-x} - x$	$\arctan x$	$(x - 1)^3$
Dekker	7	11	7	10	> 82
Brent	7	11	6	10	> 82
Ridders	10	10	8	18	58
TOMS 748	12	19	9	12	96
Chandrupatla	7	5	6	11	36
ITP	8	31	9	11	36
Newton–Bis.	15	21	12	24	171
Halley–Bis.	21	21	15	27	201
Chebyshev–Bis.	21	45	15	39	231
Householder–Bis.	25	25	25	36	278

La tabella si legge così.

- **Sui problemi regolari** (prima, terza e quarta colonna) Brent, Dekker e Chandrupatla sono i più economici, intorno a 6–11 valutazioni. TOMS 748 non recupera il suo ordine superiore perché su problemi così facili la convergenza avviene prima che i cicli completi entrino a regime.
- **Su** $x^{10} - 1$ emergono le differenze di progetto. Chandrupatla è nettamente il migliore; ITP invece degenera: la falsa posizione è pessima su questa funzione, il budget ρ_j si azzera e l'algoritmo si riduce alla bisezione. È il prezzo della garanzia minmax, e si vede che il conto torna: le valutazioni spese sono quelle della bisezione, mai di più.
- **Sulla radice tripla** $(x - 1)^3$ il quadro si capovolge. Tutti i metodi interpolatori soffrono, perché nessun polinomio di grado basso approssima bene una funzione con derivata prima e seconda nulle nella radice; Dekker e Brent non raggiungono affatto la tolleranza nel budget di iterazioni concesso, perché continuano a fidarsi di interpolazioni che non avanzano; TOMS 748 e Ridders ci arrivano ma spendendo rispettivamente 96 e 58 valutazioni; Chandrupatla e ITP, che riconoscono presto l'inutilità dell'interpolazione e cadono sulla bisezione, ci arrivano in 36 valutazioni, cioè esattamente $\lceil \log_2(3/10^{-10}) \rceil = 35$ passi di bisezione più uno. Gli ibridi con derivate sono i peggiori, perché Newton su una radice di molteplicità m converge soltanto linearmente con fattore $1 - 1/m$ (Sezione 21.3) e ogni iterazione costa da due a quattro valutazioni.
- **Gli ibridi con derivate**, in generale, costano da due a quattro volte un metodo senza derivate per ogni iterazione, e questo annulla quasi sempre il vantaggio dell'ordine più alto.

Hanno senso quando f' (e f'') si ottengono quasi gratis insieme a f: polinomi valutati con lo schema di Horner esteso, oppure differenziazione automatica.

In pratica: quale metodo usare

- **Senza derivate, caso generale**: Brent–Dekker. È la scelta di riferimento e quella implementata nella maggior parte delle librerie; è difficile da battere in modo significativo.
- **Se serve un algoritmo breve da scrivere a mano**: Ridders. Il candidato cade sempre nel bracket senza bisogno di test di ammissibilità, quindi l'implementazione sta in poche righe e non ha casi particolari; si paga con due valutazioni per iterazione.
- **Se si vuole la robustezza di Brent in trenta righe**: Chandrupatla, che nei test sopra è il più efficiente dei metodi senza derivate.
- **Tolleranze molto strette su funzioni regolari**: TOMS 748, che sfrutta l'interpolazione di grado più alto e ha l'R-ordine maggiore.
- **Valutazioni di f costose e necessità di una garanzia**: ITP, perché il numero di iterazioni non supera mai quello della bisezione più lo slack n_0.
- **Derivata disponibile e poco costosa** (polinomi, differenziazione automatica): Newton–Bisection è semplice e sufficiente; Halley–Bisection solo se anche f'' è gratuita.
- **Radici multiple**: nessuno di questi metodi aiuta davvero. Su una radice di molteplicità **pari** non esiste nemmeno cambio di segno, quindi il bracket non si può costruire; su una radice di molteplicità dispari il bracket esiste ma, come mostra l'ultima colonna della tabella, l'interpolazione è inefficace e si ricade sulla bisezione. Occorre ricorrere alle tecniche di Sezione 21.3.

22.14 Esercizi

1. **Perché serve il troncamento.** Applicare la pura falsa posizione e poi ITP alla funzione $f(x) = x^{10} - 1$ su $[0, 1.3]$, contando le valutazioni necessarie per scendere sotto 10^{-10}. Spiegare il ruolo di $\delta = \kappa_1(b-a)^{\kappa_2}$ e che cosa cambia prendendo $\kappa_2 = 1$ invece di 2.
2. **Ridders resta nel bracket.** Verificare analiticamente che, se $f(a)f(b) < 0$, allora $\Delta = \sqrt{f(m)^2 - f(a)f(b)} > |f(m)|$ e dedurne che r cade sempre in (a, b). Che cosa succede se $f(a)f(b) = 0$?
3. **Il budget di ITP.** Modificare l'implementazione di ITP perché registri a ogni passo il valore di ρ_j, e applicarla a $x^{10} - 1$ su $[0.5, 1.3]$: a quale iterazione il budget si azzera? Confrontare con $n_0 = 0$ e $n_0 = 4$.
4. **Dekker contro Brent.** Costruire una funzione continua su $[0, 1]$ con $f(0)f(1) < 0$ per cui il metodo di Dekker impieghi molte più valutazioni della bisezione, e verificare che Brent (o `uniroot`) non degrada. *Suggerimento:* servono passi di secante che si accorcino lentamente.
5. **Newton–Bisection su una funzione non derivabile.** Applicare `ibrido` con `passo_newton` a $f(x) = \operatorname{sign}(x)\,|x|^{1/3}$ su $[-1, 2]$, dove Newton puro diverge per qualunque punto iniziale (Sezione 20.6.4). Quanti passi di Newton vengono accettati, e perché?
6. **IQI contro parabola diretta.** Per tre punti vicini a $\sqrt{2}$ su $f(x) = x^2 - 2$, confrontare lo zero ottenuto interpolando $y = f(x)$ con una parabola in x e quello ottenuto con l'interpolazione quadratica inversa. Quale dei due è sempre definito, e perché?
7. **Il test di Chandrupatla.** Disegnare nel piano (ξ, ϕ) la regione $1 - \sqrt{1-\xi} < \phi < \sqrt{\xi}$ per $\xi \in (0, 1)$ e verificare che contiene il punto corrispondente a tre punti equispaziati su una funzione lineare. Che cosa accade quando f ha un flesso fra i tre punti?
8. **Molteplicità pari.** Spiegare perché nessun metodo di questo capitolo può essere applicato a $f(x) = (x-1)^2$, e proporre una strategia alternativa basata su Sezione 21.3.
9. **Costo della derivata.** Rifare la tabella di Sezione 22.13 contando una valutazione di f e delle sue derivate come **una sola** valutazione (è il caso dei polinomi con lo schema di

Horner esteso). Come cambia la classifica?

10. **Confronto con la libreria.** Confrontare le proprie implementazioni con `uniroot` di R (che è `zeroin`, cioè Brent) su tutti e cinque i problemi di Sezione 22.13, usando `tol = 1e-12`, e commentare le differenze.



23. Iterazioni di punto fisso

Vogliamo ora dare una formulazione più generale alle iterazioni del tipo Newton-Raphson[1].

Definition 23.1 — Punto fisso. Data una funzione $g : \mathbb{R} \mapsto \mathbb{R}$, diremo che α è un punto fisso se:

$$\alpha = g(\alpha).$$

Cercare gli zeri di una funzione $f(x)$ è equivalente a cercare i punti fissi della funzione

$$g(x) = x - f(x).$$

Un modo molto naturale per cercare i punti fissi è quello di usare delle iterazioni. Ad esempio dato un punto iniziale x_0 costruendo la successione

$$\boxed{x_{k+1} = g(x_k), \qquad k = 1, 2, \ldots} \tag{23.1}$$

possiamo chiederci quando la successione $\{x_i\}_{i=0}^{\infty}$ è convergente e se converge a α. Ad esempio lo schema di Newton~20.3 può essere messo nella forma 23.1 ponendo

$$g(x) = x - \frac{f(x)}{f'(x)},$$

fondamentale per lo studio di schemi iterativi della forma 23.1 è il seguente teorema:

Theorem 23.1 — Convergenza punto fisso. Sia $g(x)$ una funzione continua e sia α un suo punto fisso, cioè $\alpha = g(\alpha)$. Sia $g(x)$ soddisfacente la seguente condizione

$$\left|g(x) - g(x')\right| \leq L \left|x - x'\right|, \tag{23.2}$$

per ogni x, x' in un intervallo chiuso $I \equiv [\alpha - \rho, \alpha + \rho]$, dove la costante L soddisfa:

$$0 \leq L < 1,$$

allora

- (i) Posto $x_0 \in I$ allora tutte le iterate x_k definite da 23.1 stanno in I, cioè

$$\alpha - \rho \leq x_k \leq \alpha + \rho, \qquad k = 1, 2, \ldots$$

[1] Isaac Newton (1642-1727) e Joseph Raphson (1668 ca.-1715 ca.)

- (ii) (esistenza) le iterate convergono ad α,

$$\lim_{k\mapsto\infty} x_k = \alpha.$$

- (iii) (unicità) α è l'unico punto fisso di $g(x)$ in $[\alpha-\rho, \alpha+\rho]$.

Proof.

- (i) basta osservare che

$$\begin{aligned} |\alpha - x_k| &= |g(\alpha) - g(x_{k-1})|, \\ &\leq L\,|\alpha - x_{k-1}| = L\,|g(\alpha) - g(x_{k-2})|, \\ &\leq L^2\,|\alpha - x_{k-2}| = L^2\,|g(\alpha) - g(x_{k-3})|, \\ &\vdots \\ &\leq L^k\,|\alpha - x_0| \leq \rho, \end{aligned} \tag{23.3}$$

  e quindi x_k è nell'intervallo $[\alpha-\rho, \alpha+\rho]$.
- (ii) sempre dalla 23.3 e dal fatto che $0 \leq L < 1$ segue che

$$\lim_{k\mapsto\infty} |\alpha - x_k| = 0.$$

- (iii) siano α e β due punti fissi allora dalla 23.2 abbiamo

$$|\alpha - \beta| = |g(\alpha) - g(\beta)| \leq L\,|\alpha - \beta| < |\alpha - \beta|.$$

  Questa contraddizione implica che $\alpha = \beta$.

■

Osservazione

Se la funzione $g(x)$ è differenziabile, si può scrivere

$$\begin{aligned} g(x) - g(x') &= g'(\eta)(x - x'), \qquad \eta \in (x, x') \\ |g(x) - g(x')| &\leq |g'(\eta)|\,|x - x'|, \end{aligned}$$

Di conseguenza, se vale

$$|g'(\zeta)| \leq L < 1, \qquad \zeta \in [\alpha-\rho, \alpha+\rho],$$

dove α è un punto fisso di $g(x)$ allora per il teorema precedente le iterazioni 23.1 convergono ad α.

▪ **Example 23.1 — Esempio.** Considerando il metodo di Newton-Raphson come uno schema a punto fisso abbiamo

$$g(x) = x - \frac{f(x)}{f'(x)}, \qquad g'(x) = \frac{f(x)f''(x)}{(f'(x))^2},$$

Se α è una radice semplice di $f(x)$ cioè $f'(\alpha) \neq 0$ allora

$$g'(\alpha) = \frac{f(\alpha)f''(\alpha)}{(f'(\alpha))^2} = 0,$$

se $f(x)$ è una funzione C^2 allora $g'(x)$ è continua nell'intorno di α e scegliendo un intorno sufficientemente piccolo di α avremo

$$|g'(x)| \leq L < 1, \qquad x \in [\alpha - \rho, \alpha + \rho]$$

Per il teorema precedente, lo schema di Newton-Raphson è quindi convergente, purché si parta da un'approssimazione sufficientemente vicina alla radice e la radice sia semplice.

▪

Osservazione

Se α è un punto fisso di $g(x)$ e inoltre $g'(\alpha) = 0$ è chiaro che nella iterazione 23.1 più ci avviciniamo alla radice α più piccola è la costante L e quindi l'avvicinamento al punto fisso è più rapido. Supponiamo ora che $g''(x)$ esista e sia continua, allora sviluppando con Taylor[2] attorno al punto fisso α la funzione $g(x)$ otteniamo:

$$g(x) = g(\alpha) + 0 + \frac{(x-\alpha)^2}{2} g''(\eta),$$

e da questa

$$|x_{k+1} - \alpha| = |g(x_k) - g(\alpha)| = \left| \frac{(x_k - \alpha)^2}{2} g''(\eta_k) \right| \leq \left| \frac{g''(\eta_k)}{2} \right| |x_k - \alpha|^2 . \tag{23.4}$$

Da questa equazione segue che l'errore alla iterata successiva è proporzionale al quadrato dell'errore precedente. In tal caso diremo che il metodo è del *secondo ordine*. Quindi in particolare il metodo di Newton-Raphson è del secondo ordine nel caso di radici semplici. Sia ora $I \equiv [\alpha - \rho, \alpha + \rho]$ un intervallo in cui lo schema è convergente e

$$\left| \frac{g''(x)}{2} \right| \leq M, \qquad x \in I$$

allora dalla 23.4

$$\begin{aligned}
|x_k - \alpha| &\leq M |x_{k-1} - \alpha|^2 , \\
&\leq M \left(M |x_{k-2} - \alpha|^2 \right)^2 = M \cdot M^2 |x_{k-2} - \alpha|^4 , \\
&\leq M \cdot M^2 \left(M |x_{k-3} - \alpha|^2 \right)^4 = M \cdot M^2 \cdot M^4 |x_{k-3} - \alpha|^8 , \\
&\vdots \\
&\leq M^{1+2+4+8+\cdots+2^{k-1}} |x_0 - \alpha|^{2^k} ,
\end{aligned}$$

osserviamo che $1 + 2 + 4 + \cdots + 2^{k-1} = 2^k - 1$ e quindi

$$|x_k - \alpha| \leq (M |x_0 - \alpha|)^{2^{k-1}} |x_0 - \alpha| .$$

Scegliamo x_0 sufficientemente vicino ad α in modo che

$$M |x_0 - \alpha| < 1,$$

allora possiamo calcolare il numero di iterazioni necessarie per ridurre l'errore iniziale $|x_0 - \alpha|$ di ad esempio 10^{-m} ponendo

$$(M |x_0 - \alpha|)^{2^{k-1}} \approx 10^{-m}, \tag{23.5}$$

e prendendo il logaritmo da entrambe le parti

$$k \approx \frac{1}{\log_{10} 2} \log \left(\frac{m}{-\log_{10}(M |x_0 - \alpha|)} \right),$$

osserviamo che sempre dalla 23.5 otteniamo

$$m \approx 2^k \log_{10}(1/(M|x_0 - \alpha|)), \tag{23.6}$$

m è proporzionale al numero di cifre esatte nella approssimazione di α. quindi la 23.6 significa che ad ogni iterazione un metodo al secondo ordine circa raddoppia le cifre esatte.

Osservazione

Supponiamo che sia α un punto fisso di $g(x)$ e inoltre $g \in C^p$ con

$$g'(\alpha) = g''(\alpha) = \cdots = g^{(p-1)}(\alpha) = 0,$$

per il teorema di Taylor

$$g(x) = g(\alpha) + \frac{(x-\alpha)^p}{p!} g^{(p)}(\eta),$$

anche qui avremo

$$|x_{k+1} - \alpha| = |g(x_k) - g(\alpha)| \leq \frac{|g^{(p)}(\eta)|}{p!} |x - \alpha|^p .$$

Da questa equazione segue che l'errore alla iterata successiva è proporzionale alla $p-$esima potenza dell'errore precedente. In tal caso diremo che il metodo è *di ordine p*.

▪ **Example 23.2 — Esempio.** Nel metodo di Newton-Raphson se α è una radice multipla di $f(x)$ cioè

$$f(x) = (x-\alpha)^n h(x), \qquad n > 1$$

dove $h(\alpha) \neq 0$ segue che

$$\begin{aligned} f'(x) &= n(x-\alpha)^{n-1} h(x) + (x-\alpha)^n h'(x) \\ f''(x) &= n(n-1)(x-\alpha)^{n-2} h(x) + 2n(x-\alpha)^{n-1} h'(x) + (x-\alpha)^n h''(x), \end{aligned}$$

e di conseguenza

$$\begin{aligned} g'(x) &= \frac{f(x)f''(x)}{(f'(x))^2} = \frac{h(x)\left(n(n-1)h(x) + 2n(x-\alpha)h'(x) + (x-\alpha)^2 h''(x)\right)}{(nh(x) + (x-\alpha)h'(x))^2}, \\ g'(\alpha) &= \frac{n(n-1)h(\alpha)^2}{n^2 h(\alpha)^2} = 1 - \frac{1}{n}, \end{aligned}$$

quindi

$$|g'(\alpha)| = 1 - \frac{1}{n} < 1$$

e lo schema iterativo converge localmente anche se ora la convergenza è solo al primo ordine.

▪

[2] Brook Taylor (1685-1731)

23.1 Elenco metodi numerici per zeri di funzione

23.1.1 Newton-Raphson

Ordine 2

$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad k = 1, 2, \ldots}$$

Newton per radici multiple. Basta applicare Newton alla funzione $F(x) = f(x)/f'(x)$ ottenendo:

$$\boxed{x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)} = x_k - \frac{f(x_k)f'(x_k)}{f'(x_k)^2 - f(x_k)f''(x_k)}, \qquad k = 1, 2, \ldots}$$

- **R. L.Burden & J. D. Faires**
  Numerical Analysis (9th ed.)
  Brooks Cole (2010)

23.1.2 Secanti

Ordine $\dfrac{1+\sqrt{5}}{2} = 1.618\ldots$

$$\boxed{x_{k+1} = \frac{f(x_k)x_{k-1} - f(x_{k-1})x_k}{f(x_k) - f(x_{k-1})}, \qquad k = 1, 2, \ldots}$$

23.1.3 Secanti (variante Brent)

$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{D_k}, \quad D_k = \begin{cases} \dfrac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} & |x_k - x_{k-1}| > h \\ \dfrac{f(x_k) - f(x_k - h)}{h} & |x_k - x_{k-1}| \leq h \end{cases} \qquad k = 1, 2, \ldots}$$

23.1.4 Parabole tangenti

Ordine 3 (si veda Sezione 20.11)

$$\boxed{x_{k+1} = x_k - 2\frac{f(x_k)}{f'(x_k)} \Bigg/ \left(1 + \sqrt{1 - 2\frac{f(x_k)f''(x_k)}{f'(x_k)^2}}\right) \qquad k = 1, 2, \ldots}$$

Da non confondere con il metodo di Müller[3] (Sezione 20.10), che interpola una parabola per tre punti, non usa derivate e ha ordine ≈ 1.84:

- **D. E. Müller**
  A method for solving algebraic equations using an automatic computer
  Mathematical Tables and Other Aids to Computation, Vol. 10, No. 56, 208-215, (1956)

23.1.5 Metodo di Halley

Ordine 3

$$\boxed{x_{k+1} = x_k - \frac{f(x_k)f'(x_k)}{f'(x_k)^2 - \frac{1}{2}f(x_k)f''(x_k)} \qquad k = 1, 2, \ldots}$$

[3] David Eugene Muller (1924-2008)

- **J. F. Traub**[4]

  – Iterative Methods for the Solution of Equations*
  Prentice-Hall, 1964.

23.1.6 Metodo di Steffensen

Ordine 2

$$x_{k+1} = x_k - \frac{f(x_k)^2}{f(x_k + f(x_k)) - f(x_k)} \qquad k = 1, 2, \dots$$

- **J. F. Steffensen**[5]
  Remarks on Iteration
  Skandinavisk Aktuarietidskrift, 1933, 16(1), 64-72.

23.1.7 Metodo delle corde tangenti

$$z = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad x_{k+1} = z - \frac{f(z)}{f'(x_k)}, \qquad k = 1, 2, \dots$$

23.1.8 Metodo di Ostrowski

Ordine 4

$$z = x_k - \frac{f(x_k)}{f'(x_k)}, \qquad x_{k+1} = z - \frac{f(z)(z - x_k)}{2f(z) - f(x_k)}, \qquad k = 1, 2, \dots$$

- **A. M. Ostrowski**[6]
  Solution of equations and systems of equations
  Academic Press, New York, 1960

In sintesi

- Ogni metodo iterativo per $f(t) = 0$ si riscrive come $t_{k+1} = g(t_k)$: la radice diventa un **punto fisso** di g.
- Se g è una contrazione ($|g'| \leq \lambda < 1$) in un intorno invariante, l'iterazione converge all'unico punto fisso, con errore che decade come λ^k.
- L'ordine di convergenza è determinato dalla prima derivata non nulla di g nel punto fisso: $g'(\xi) = 0$ dà ordine almeno due, ed è esattamente il caso di Newton.
- $|g'(\xi)| > 1$ rende il punto fisso **repulsivo**: la stessa equazione, riscritta male, produce un metodo divergente.

[4] Joseph Frederick Traub (1932-2015)
[5] Johan Frederik Steffensen (1873-1961)
[6] Alexander Markowich Ostrowski (1893-1986)



24. Zeri di polinomi

24.1 Il metodo di Sturm

24.1.1 Eliminazione delle radici multiple

Il metodo di Newton-Raphson[1] è in genere del secondo ordine; tuttavia, quando si approssima una radice multipla di un polinomio, la convergenza degrada al primo ordine. Per evitare questo rallentamento si può sostituire il polinomio con radici multiple con un polinomio che abbia le stesse radici, tutte semplici. Questo problema, apparentemente complesso, ammette una soluzione semplice. Sia $p(x)$ un polinomio monico fattorizzato come segue

$$p(x) = \prod_{i=1}^{k}(x - x_i)^{m_i},$$

allora il polinomio derivata prima si scriverà come

$$p'(x) = q(x)\prod_{i=1}^{k}(x - x_i)^{m_i - 1},$$

$$q(x) = \sum_{i=1}^{k} m_i \prod_{\substack{j=1 \\ j\neq i}}^{k}(x - x_i),$$

Osserviamo che il polinomio $q(x)$ non ha radici in comune con il polinomio $p(x)$ infatti

$$q(x_\ell) = \sum_{i=1}^{k} m_i \prod_{\substack{j=1 \\ j\neq i}}^{k}(x_\ell - x_i) = m_\ell \prod_{\substack{j=1 \\ j\neq \ell}}^{k}(x_\ell - x_i) \neq 0, \qquad \ell = 1, 2, \ldots, k$$

Quindi, i polinomi $p(x)$ e $p'(x)$ hanno in comune soltanto le radici di $p(x)$ con molteplicità maggiore di 1. Se x_i è una radice comune con molteplicità m_i allora la sua molteplicità in $p'(x)$ sarà $m_i - 1$.

Ricordiamo che un polinomio $M(x)$ è il massimo comune divisore dei polinomi $P(x)$ e $Q(x)$ se soddisfa le condizioni seguenti:

1. $M(x)$ divide $P(x)$ e $Q(x)$, cioè

$$P(x) = M(x)S(x), \qquad Q(x) = M(x)T(x), \tag{24.1}$$

dove $S(x)$ e $T(x)$ sono anch'essi polinomi (eventualmente di grado 0).

[1] Isaac Newton (1642-1727) e Joseph Raphson (1668 ca.-1715 ca.)

2. Se $N(x)$ è un altro polinomio che divide $P(x)$ e $Q(x)$, allora $N(x)$ divide $M(x)$, cioè

$$M(x) = W(x)N(x),$$

per un opportuno polinomio $W(x)$.

Theorem 24.1 — MCD e radici. Se $M(x)$ è il massimo comune divisore tra due polinomi $P(x)$ e $Q(x)$ e se α è una radice comune a $P(x)$ e $Q(x)$ cioè:

$$P(\alpha) = 0, \qquad Q(\alpha) = 0,$$

allora necessariamente α è anche una radice di $M(x)$.

Proof. Se così non fosse dalla 24.1 avremo che il polinomio $x - \alpha$ divide sia $S(x)$ che $T(x)$ cioè

$$P(x) = M(x)(x-\alpha)S^1(x), \qquad Q(x) = M(x)(x-\alpha)T^1(x),$$

e $M(x)(x-\alpha)$ sarebbe un divisore comune a $S(x)$ e $T(x)$ che non divide $M(x)$. ∎

Theorem 24.2 — Radici comuni. Se α è una radice del massimo comune divisore $M(x)$ tra i polinomi $P(x)$ e $Q(x)$ allora α è una radice comune a $P(x)$ e $Q(x)$ cioè:

$$P(\alpha) = 0, \qquad Q(\alpha) = 0,$$

Proof. dalla 24.1 segue immediatamente

$$P(\alpha) = M(\alpha)S(\alpha) = 0,$$
$$Q(\alpha) = M(\alpha)T(\alpha) = 0,$$

∎

Da questi due teoremi segue che il massimo comune divisore tra due polinomi è costituito dal prodotto delle radici in comune con molteplicità il minimo tra le due.

▪ **Example 24.1 — Esempio.** Siano

$$P(x) = 3(x-1)^3(x+1)^2(x-3),$$
$$Q(x) = 2(x-1)^2(x+1)(x-3)^3,$$

il loro massimo comune divisore è

$$M(x) = (x-1)^2(x+1)(x-3).$$

▪

Sia $m(x)$ il massimo comune divisore tra i polinomi $p(x)$ e $p'(x)$ allora per i discorsi precedenti vale

$$m(x) = \prod_{i=1}^{k}(x-x_i)^{m_i-1}.$$

Allora il polinomio

$$\frac{p(x)}{m(x)} = \prod_{i=1}^{k}(x-x_i),$$

contiene tutte le radici di $p(x)$, tutte semplici. Resta da calcolare questo massimo comune divisore; a tal fine si può ricorrere a un algoritmo classico, quello di Euclide (Euclide di Alessandria, circa 365–300 a.C.).

24.1.2 Algoritmo di Euclide

Algoritmo di Euclide per il M.C.D. di $p(x)$ e $q(x)$

Dati: polinomi $p(x)$ e $q(x)$
Restituisce: il massimo comune divisore di p e q

se $\deg(p) > \deg(q)$ **allora**
  $p_0 \leftarrow p, \quad p_1 \leftarrow q$
altrimenti
  $p_0 \leftarrow q, \quad p_1 \leftarrow p$
$i \leftarrow 1$
ripeti
  $p_{i+1} \leftarrow$ resto della divisione di p_{i-1} per p_i
    cioè $p_{i-1}(x) = s_i(x)\,p_i(x) + p_{i+1}(x)$
  $i \leftarrow i + 1$
  se $p_i \equiv 0$ **allora**
    esci dal ciclo
restituisci p_{i-1}

Figure 24.1: L'algoritmo di Euclide per polinomi: la successione dei resti termina perché il grado cala a ogni divisione, e l'ultimo resto non nullo è il M.C.D.

▪ **Example 24.2 — Esempio.** Sia dato

$$\begin{aligned} p(x) &= (x-1)^3(x-2)(x+1)^2, \\ &= x^6 - 3x^5 + 6x^3 - 3x^2 - 3x + 2, \\ p'(x) &= 6x^5 - 15x^4 + 18x^2 - 6x - 3, \end{aligned}$$

calcoliamo il massimo comune divisore $m(x)$ con l'algoritmo di Euclide:

- Inizializzazione:
$$\begin{aligned} p_0(x) &= x^6 - 3x^5 + 6x^3 - 3x^2 - 3x + 2, \\ p_1(x) &= 6x^5 - 15x^4 + 18x^2 - 6x - 3, \end{aligned}$$
- Calcolo p_2:
$$\begin{aligned} p_0(x) &= p_1(x)s_1(x) + p_2(x), \\ x^6 - 3x^5 + 6x^3 - 3x^2 - 3x + 2 &= \left(6x^5 - 15x^4 + 18x^2 - 6x - 3\right)\left(\frac{x}{6} - \frac{1}{12}\right) \\ &\quad + \left(\frac{7}{4} - 3x - \frac{x^2}{2} + 3x^3 - \frac{5}{4}x^4\right). \end{aligned}$$
- Calcolo p_3:
$$\begin{aligned} p_1(x) &= p_2(x)s_1(x) + p_3(x), \\ 6x^5 - 15x^4 + 18x^2 - 6x - 3 &= \left(-\frac{24x}{5} + \frac{12}{25}\right)\left(\frac{7}{4} - 3x - \frac{x^2}{2} + 3x^3 - \frac{5}{4}x^4\right) \\ &\quad + \frac{96}{25}\left(-x^3 + x^2 + x - 1\right). \end{aligned}$$

- Calcolo p_4:

$$p_2(x) = p_3(x)s_1(x) + p_4(x),$$
$$\frac{7}{4} - 3x - \frac{x^2}{2} + 3x^3 - \frac{5}{4}x^4 = \left(\frac{125x}{384} - \frac{175}{384}\right)\frac{96}{25}\left(-x^3 + x^2 + x - 1\right) + 0.$$

- Poiché $p_4 \equiv 0$, il massimo comune divisore è p_3

$$\frac{96}{25}\left(-x^3 + x^2 + x - 1\right),$$

e poiché è unico a meno di una costante moltiplicativa scegliamo

$$m(x) = -x^3 + x^2 + x - 1.$$

dividendo $p(x)$ per $m(x)$ otteniamo

$$\frac{p(x)}{m(x)} = -x^3 + 2x^2 + x - 2 = -(x-1)(x-2)(x+1).$$

▪

Per questioni computazionali (ad esempio nella costruzione di successioni di Sturm[2]) a volte è più conveniente usare una versione modificata dell'algoritmo, tenendo conto del fatto che se $m(x)$ è il massimo comune divisore tra due polinomi, anche $cm(x)$ lo è, dove c è uno scalare non nullo qualsiasi. La versione modificata cambia la riga

- $p_{i-1}(x) = s_i(x)p_i(x) - p_{i+1}(x)$

con

- $p_{i-1}(x) = s_i(x)p_i(x) - c_i p_{i+1}(x)$

Osserviamo che l'algoritmo Sezione 24.1.2 produce effettivamente il massimo comune divisore. Vale infatti il seguente teorema:

Theorem 24.3 — Terminazione dell'algoritmo di Euclide. L'algoritmo Sezione 24.1.2 termina in un numero finito m di passi e l'ultimo resto non nullo è il massimo comune divisore dei polinomi.

Proof. Innanzitutto osserviamo che l'algoritmo termina in un numero finito di passi. Infatti il resto della divisione ha grado strettamente minore del divisore e quindi ad ogni divisione si riduce il grado dei polinomi coinvolti. Quando il grado è 0 il polinomio è uno scalare. La divisione di un polinomio per uno scalare ha resto nullo e quindi anche in questo caso estremo l'algoritmo termina. L'algoritmo può essere quindi scritto come segue

$$\begin{aligned}
p_0(x) &= s_1(x)p_1(x) - c_1 p_2(x), && (a)\\
p_1(x) &= s_2(x)p_2(x) - c_2 p_3(x), && (b)\\
&\vdots\\
p_{k-1}(x) &= s_k(x)p_k(x) - c_k p_{k+1}(x), && (k)\\
&\vdots\\
p_{m-2}(x) &= s_{m-1}(x)p_{m-1}(x) - c_{m-1} p_m(x), && (m-1)\\
p_{m-1}(x) &= s_m(x)p_m(x), && (m)
\end{aligned} \tag{24.2}$$

e segue subito che $p_m(x)$ l'ultimo resto non nullo divide $p_{m-1}(x)$, $p_{m-2}(x)$, ..., $p_0(x)$ e quindi è un divisore comune.

Viceversa sia $q(x)$ un divisore di $p_0(x)$ e $p_1(x)$, allora dalla 24.2 (a) segue se $q(x)$ divide $p_2(x)$ e dalla 24.2 (b) segue che $q(x)$ divide $p_3(x)$ e così via fino ad arrivare alla 24.2 (m) da cui segue che $q(x)$ divide $p_m(x)$ e quindi $p_m(x)$ è il massimo comune divisore. ■

[2] Jacques Charles François Sturm (1803-1855)

24.1.3 Separazione delle radici

È utile conoscere il numero di *radici reali* contenute in un dato intervallo. Se ciò è possibile, si possono isolare tutte le radici reali mediante un metodo di bisezione e approssimarle con la precisione desiderata. Il problema si risolve grazie alle successioni che prendono il nome dal loro scopritore, Jacques Charles François Sturm (1803–1855):

Definition 24.1 — Successione di Sturm. La successione di funzioni continue definite su $[a, b]$:

$$\mathcal{F}(x) = \{f_0(x), f_1(x), \dots, f_m(x)\},$$

forma una *successione di Sturm su* $[a, b]$ se vale:

1. $f_0(x)$ ha soltanto zeri semplici in $[a, b]$;
2. $f_m(x)$ non ha zeri su $[a, b]$;
3. per ogni $\alpha \in [a, b]$ tale che $f_k(\alpha) = 0$, allora $f_{k-1}(\alpha) f_{k+1}(\alpha) < 0$;
4. per ogni $\alpha \in [a, b]$ tale che $f_0(\alpha) = 0$, allora $f_0'(\alpha) f_1(\alpha) < 0$;

Definition 24.2 — Variazione di segno. Data una successione di Sturm

$$\mathcal{F}(x) = \{f_0(x), f_1(x), \dots, f_m(x)\}$$

per ogni punti x associamo un vettore di $m + 1$ numeri reali. A questo vettore possiamo associare un numero intero detto *variazione di segno*. Questo numero rappresenta il numero di volte che scorrendo la successione di numeri c'è un cambio di segno. Ad esempio la successione

$$\{\underbrace{1, 0, -2}_{*}, \underbrace{-3, 4}_{*}, 3, 0, 1\},$$

ha 2 variazioni di segno (marcate con $*$). Osserviamo che $3, 0, 1$ ha variazione di segno nulla. Infatti lo 0 va considerato come elemento neutro e va rimosso dal computo.

Per ogni successione di Sturm vale il seguente teorema:

Theorem 24.4 — Teorema di Sturm. Data una successione di Sturm

$$\mathcal{F}(x) = \{f_0(x), f_1(x), \dots, f_m(x)\}$$

su $[a, b]$ il numero di zeri di $f_0(x)$ in (a, b) è dato dalla differenza tra il numero di variazioni di segno in $\mathcal{F}(b)$ e $\mathcal{F}(a)$.

Proof. Il numero di variazioni di segno cambia man mano che x passa da a a b solo a causa del cambio di segno di qualche funzione della successione di Sturm. Assumiamo che per un qualche $\hat{x} \in (a, b)$ valga $f_k(\hat{x}) = 0$ per $0 < k < m$. In un intorno di $\hat{x}$ dalla condizione 3 sarà

Table 24.1: variazione di segno

x	$f_{k-1}(x)$	$f_k(x)$	$f_{k+1}(x)$
$\hat{x} + \epsilon$	$+$	$\pm$	$-$
$\hat{x}$	$+$	0	$-$
$\hat{x} - \epsilon$	$+$	$\pm$	$-$

oppure

Table 24.3: variazione di segno ($f_0'(\hat{x}) < 0$, $f_1(\hat{x}) > 0$)

x	$f_0(x)$	$f_1(x)$
$\hat{x} + \epsilon$	$+$	$-$
$\hat{x}$	0	$-$
$\hat{x} - \epsilon$	$-$	$-$

x	$f_0(x)$	$f_1(x)$
$\hat{x} + \epsilon$	$-$	$+$
$\hat{x}$	0	$+$
$\hat{x} - \epsilon$	$+$	$+$

Table 24.2: variazione di segno

x	$f_{k-1}(x)$	$f_k(x)$	$f_{k+1}(x)$
$\hat{x} + \epsilon$	$-$	$\pm$	$+$
$\hat{x}$	$-$	0	$+$
$\hat{x} - \epsilon$	$-$	$\pm$	$+$

in ogni caso il passaggio da $\hat{x}$ non cambia il numero di variazioni di segno. Sia ora $\hat{x}$ uno zero di $f_0(x)$ e vediamo la variazione di segno nell'intorno di $\hat{x}$, osserviamo che dalla condizione 4 della definizione Definizione 24.1 avremo

in ogni caso in numero delle variazioni di segno cresce al passare di x per uno zero di $f_0(x)$. Combinando queste osservazioni otteniamo il teorema. ∎

24.1.4 Costruzione della successione di Sturm per un polinomio

È relativamente facile costruire una successione di Sturm per un polinomio. Sia $f_0 \equiv p_n$ un polinomio di grado n, definiamo $f_1 \equiv -p_n'$. Dividiamo $f_0(x)$ per $f_1(x)$ e chiamiamo il resto $-f_2(x)$. Poi dividiamo $f_1(x)$ per $f_2(x)$ e chiamiamo il resto $-f_3(x)$. Continuiamo così finché il procedimento termina. Abbiamo così la successione:

$$\begin{aligned}
f_0(x) &= q_1(x)f_1(x) - f_2(x),\\
f_1(x) &= q_2(x)f_2(x) - f_3(x),\\
&\vdots\\
f_{k-1}(x) &= q_k(x)f_k(x) - f_{k+1}(x),\\
&\vdots\\
f_{m-2}(x) &= q_{m-1}(x)f_{m-1}(x) - f_m(x),\\
f_{m-1}(x) &= q_m(x)f_m(x).
\end{aligned} \tag{24.3}$$

Questa successione a parte il segno di f_i è esattamente la successione per il calcolo del massimo comune divisore di un polinomio, e f_m è il massimo comune divisore tra f_0 e f_1. La successione di polinomi

$$p_i(x) = \frac{f_i(x)}{f_m(x)}, \qquad i = 0, 1, \dots, m$$

è una successione di Sturm, infatti

- $p_0(x)$ ha soltanto zeri semplici in $[a, b]$; essendo il rapporto tra il polinomio originario ed il massimo comune divisore tra il polinomio originario e la sua derivata prima.
- $p_m(x)$ non ha zeri su $[a, b]$; infatti è una costante.
- per ogni $\alpha \in [a, b]$ tale che $p_k(\alpha) = 0$, allora $p_{k-1}(\alpha)p_{k+1}(\alpha) < 0$; infatti dalla 24.3 abbiamo

$$p_{k-1}(\alpha) = -p_{k+1}(\alpha).$$

Osserviamo che se $p_{k-1}(\alpha) = p_k(\alpha) = p_{k+1}(\alpha) = 0$ allora dalla 24.3 seguirebbe il prossimo punto:

- per ogni $\alpha \in [a, b]$ tale che $p_0(\alpha) = 0$, allora $p_0'(\alpha)p_1(\alpha) = -p_0'(\alpha)^2 < 0$;

24.2 Limitazione delle radici

Per poter usare il metodo di bisezione e separare le radici con Sturm è necessario conoscere una prima stima anche se grossolana dell'intervallo in cui stanno tutte le radici di un polinomio. Per fare questo occorre osservare che la seguente matrice in forma di Frobenius[3]

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & & \vdots \\ \vdots & & & \ddots & \ddots & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ -\dfrac{a_0}{a_n} & -\dfrac{a_1}{a_n} & -\dfrac{a_2}{a_n} & \cdots & -\dfrac{a_{n-2}}{a_n} & -\dfrac{a_{n-1}}{a_n} \end{pmatrix},$$

ha come polinomio caratteristico

$$\begin{aligned} p(\lambda) &= \left|\boldsymbol{A} - \lambda \boldsymbol{I}\right| \\ &= \frac{(-1)^n}{a_n}\left(a_0 + a_1\lambda + a_2\lambda^2 + \cdots + a_{n-2}\lambda^{n-2} + a_{n-1}\lambda^{n-1} + a_n\lambda^n\right), \end{aligned} \tag{24.4}$$

(basta sviluppare lungo l'ultima riga). Applicando il teorema di Gershgorin alla matrice otteniamo che le radici del polinomio 24.4 soddisfano

$$\left|\lambda + \frac{a_{n-1}}{a_n}\right| \leq \left|\frac{a_0}{a_n}\right| + \left|\frac{a_1}{a_n}\right| + \cdots + \left|\frac{a_{n-2}}{a_n}\right|, \qquad |\lambda| \leq 1.$$

Applicando il teorema di Gershgorin alla matrice trasposta otteniamo che le radici del polinomio 24.4 soddisfano

$$|\lambda| \leq \left|\frac{a_0}{a_n}\right|, \qquad |\lambda| \leq \left|\frac{a_i}{a_n}\right| + 1, \qquad \left|\lambda + \frac{a_{n-1}}{a_n}\right| \leq 1.$$

Possiamo applicare le disuguaglianze appena viste (indebolendole un poco) a un polinomio qualunque,

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1}x^{n-1} + a_n x^n,$$

ottenendo le seguenti limitazioni

$$|x| \leq \max\left\{1, \sum_{i=0}^{n-1}\left|\frac{a_i}{a_n}\right|\right\},$$

oppure

$$|x| \leq \max\left\{\left|\frac{a_0}{a_n}\right|, 1 + \max_{i=1}^{n-1}\left|\frac{a_i}{a_n}\right|\right\}.$$

[3] Ferdinand Georg Frobenius (1849-1917)

In sintesi

- Per i polinomi si può fare di più che cercare uno zero alla volta: si **separano** le radici prima di iterare.
- L'algoritmo di Euclide applicato a p e p' dà il M.C.D. e permette di eliminare le radici multiple, restituendo un polinomio a radici semplici.
- La **successione di Sturm** conta esattamente le radici reali in un intervallo tramite le variazioni di segno: è un risultato esatto, non una stima.
- Le limitazioni delle radici (Cauchy, Lagrange) forniscono l'intervallo iniziale da cui partire con bisezione o Newton, ora a convergenza garantita.

VII Sistemi lineari: metodi iterativi

25 Preliminari: raggio spettrale, serie di Neumann, M-matrici 321

25.1 Raggio spettrale 321
25.2 Serie di Neumann 321
25.3 M-matrici 321

26 Metodi Iterativi per Sistemi Lineari . . . 323

26.1 Costruzione degli Schemi di Iterazione mediante Splitting 324
26.2 Metodo di Jacobi 324
26.3 Metodo di Gauss-Seidel 325
26.4 Metodo SOR 325
26.5 Derivazione per componenti dei metodi di Jacobi, Gauss–Seidel e SOR 326
26.6 Generalizzazione 328
26.7 Convergenza degli schemi iterativi 329
26.8 Come si trasforma l'errore secondo lo schema iterativo generale? 329
26.9 Controllo della Convergenza 333
26.10 Quale metodo converge più in fretta 334

Il capitolo si apre con i preliminari di teoria delle matrici non negative (Capitolo 25): raggio spettrale, serie di Neumann e M-matrici. Sono gli strumenti con cui si dimostrano i criteri di convergenza di Jacobi, Gauss-Seidel e SOR. Gli approfondimenti — teorema di Perron-Frobenius e teoria degli splitting — stanno nell'appendice Appendice B.



25. Preliminari: raggio spettrale, serie di Neumann, M-matrici

Questa sezione raccoglie i risultati di teoria delle matrici non negative che servono per studiare la convergenza dei metodi iterativi, e che vengono ripresi nell'appendice Appendice B. Non vengono dimostrati: sono risultati classici, e per ciascuno è indicato dove trovarne la dimostrazione.

25.1 Raggio spettrale

Corollary 25.1 — Formula di Gelfand. Sia $\boldsymbol{A}$ una matrice quadrata e $\|\cdot\|$ una qualunque norma matriciale indotta. Allora

$$\varrho(\boldsymbol{A}) = \lim_{k \mapsto \infty} \|\boldsymbol{A}^k\|^{1/k}.$$

In particolare $\varrho(\boldsymbol{A}) \leq \|\boldsymbol{A}\|$ per ogni norma indotta.

Dimostrazione in [23], capitolo sulle norme matriciali e il raggio spettrale.

25.2 Serie di Neumann

Lemma 25.1 — Convergenza della serie di Neumann. Sia $\boldsymbol{B}$ una matrice quadrata. Le seguenti affermazioni sono equivalenti:

1. la serie $\sum_{k=0}^{\infty} \boldsymbol{B}^k$ converge;
2. $\lim_{k \mapsto \infty} \boldsymbol{B}^k = \boldsymbol{0}$;
3. $\varrho(\boldsymbol{B}) < 1$.

Se valgono, allora $\boldsymbol{I} - \boldsymbol{B}$ è non singolare e

$$(\boldsymbol{I} - \boldsymbol{B})^{-1} = \sum_{k=0}^{\infty} \boldsymbol{B}^k.$$

Se inoltre $\boldsymbol{B} \geq \boldsymbol{0}$, tutti i termini della serie sono non negativi e quindi $(\boldsymbol{I} - \boldsymbol{B})^{-1} \geq \boldsymbol{I} \geq \boldsymbol{0}$.

L'equivalenza fra 2. e 3. è ripresa più avanti in questo capitolo come Teorema 26.1; per il resto si veda [43], oppure [23].

25.3 M-matrici

Definition 25.1 — M-matrice. Una matrice $A \in \mathbb{R}^{n\times n}$ è una **M-matrice** se

1. ha elementi fuori diagonale non positivi, $a_{ij} \le 0$ per $i \neq j$;
2. è non singolare e la sua inversa è non negativa, $A^{-1} \ge 0$.

Le M-matrici sono la classe naturale in cui i metodi iterativi per sistemi lineari si comportano bene: la non negatività dell'inversa è ciò che rende monotoni i confronti fra raggi spettrali usati nel seguito. Per una trattazione sistematica si veda [5].

Lemma 25.2 — Diagonale di una M-matrice. Se A è una M-matrice allora i suoi elementi diagonali sono strettamente positivi, $a_{ii} > 0$ per ogni $i = 1, \dots, n$.

Dimostrazione in [5], capitolo sulle M-matrici.

Lemma 25.3 — Caratterizzazione per traslazione. Sia A con elementi fuori diagonale non positivi. Allora A è una M-matrice se e solo se esistono $\sigma > 0$ e $B \ge 0$ tali che

$$A = \sigma I - B, \qquad \rho(B) < \sigma.$$

Si può sempre prendere $\sigma \ge \max_i a_{ii}$.

Dimostrazione in [43], capitolo sulle matrici non negative, oppure in [5].

Theorem 25.1 — Criterio di splitting per le M-matrici. Sia $A = D - B$ con D diagonale a elementi strettamente positivi e $B \ge 0$ con diagonale nulla. Posto $H = D^{-1}B$, se

$$\rho(H) < 1$$

allora A è una M-matrice e

$$A^{-1} = \left(\sum_{k=0}^{\infty} H^k \right) D^{-1} \ge 0.$$

La formula per A^{-1} segue da Lemma 25.1 applicato a H; per il resto si veda [43] o [5].

In sintesi

- Il **raggio spettrale** $\rho(B)$ governa il comportamento asintotico di B^k; la formula di Gelfand lo lega alle norme: $\rho(B) = \lim_k \|B^k\|^{1/k}$.
- La **serie di Neumann** $\sum_k B^k$ converge a $(I - B)^{-1}$ se e solo se $\rho(B) < 1$: è il criterio di convergenza di tutti i metodi iterativi del capitolo.
- Una **M-matrice** ha elementi extradiagonali ≤ 0 e inversa non negativa; ha diagonale positiva ed è caratterizzabile per traslazione, $A = sI - B$ con $B \ge 0$ e $s > \rho(B)$.
- Il criterio di splitting per le M-matrici garantisce la convergenza senza calcolare autovalori: è lo strumento operativo del capitolo successivo.



26. Metodi Iterativi per Sistemi Lineari

I metodi iterativi permettono di risolvere sistemi lineari della forma $\boldsymbol{Ax} = \boldsymbol{b}$, in cui $\boldsymbol{A}$ è una matrice e $\boldsymbol{b}$ un vettore noto. Sono particolarmente utili nel caso di matrici grandi e sparse, per le quali i metodi diretti, come la fattorizzazione di Gauss, possono richiedere troppo tempo di calcolo e memoria.

Un metodo iterativo aggiorna progressivamente un'approssimazione della soluzione $\boldsymbol{x}$ fino a raggiungere una convergenza soddisfacente. Tra i metodi più comuni figurano il metodo di Jacobi, il metodo di Gauss-Seidel e il metodo del gradiente coniugato. Questi procedimenti sfruttano le proprietà della matrice $\boldsymbol{A}$ e, in genere, convergono più rapidamente dei metodi diretti quando la matrice è ben condizionata.

Per valutare la qualità di una soluzione approssimata nei metodi iterativi, è fondamentale considerare il residuo, definito come la differenza tra il termine noto e il prodotto tra la matrice e l'approssimazione:

$$\boldsymbol{r} = \boldsymbol{b} - \boldsymbol{Ax},$$

L'obiettivo dei metodi iterativi è ridurre il residuo, avvicinandolo il più possibile a zero. Di solito, la norma del residuo viene utilizzata come criterio di convergenza: il processo iterativo continua finché tale norma non scende sotto una soglia prefissata. Un residuo piccolo indica in genere che l'approssimazione della soluzione è vicina a quella esatta.

Schema iterativo:

Si sceglie un'approssimazione iniziale $\boldsymbol{x}^0$ e si costruisce una successione di approssimazioni $\boldsymbol{x}^k$ mediante uno schema iterativo del tipo:

$$\boldsymbol{x}^{k+1} = \boldsymbol{G}(\boldsymbol{x}^k).$$

Sotto opportune condizioni, ci aspettiamo che questa successione converga, ovvero:

$$\boldsymbol{x}^* = \lim_{k\to\infty} \boldsymbol{x}^k, \qquad \boldsymbol{r}^* = \boldsymbol{b} - \boldsymbol{Ax}^* = \boldsymbol{0}.$$

In questo contesto, $\boldsymbol{x}^*$ rappresenta la soluzione esatta del sistema, e il residuo $\boldsymbol{r}^*$ deve annullarsi, indicando che la soluzione trovata soddisfa l'equazione originale.

Questioni da Definire

- Quali metodi utilizziamo per costruire gli schemi di iterazione della forma $\boldsymbol{x}^{k+1} = \boldsymbol{G}(\boldsymbol{x}^k)$?
- Quali criteri seguiamo per selezionare la soluzione iniziale $\boldsymbol{x}^0$?
- Come determinare l'accuratezza della soluzione approssimata?

- Quali strategie adottiamo per arrestare le iterazioni, considerando che non possiamo eseguire un numero infinito di passi?
- Come misuriamo l'efficienza di uno schema iterativo?

26.1 Costruzione degli Schemi di Iterazione mediante Splitting

Consideriamo la matrice $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ espressa nella forma:

$$\boldsymbol{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \cdots & A_{1,n} \\ A_{2,1} & A_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & A_{n-1,n} \\ A_{n,1} & \cdots & A_{n,n-1} & A_{n,n} \end{pmatrix},$$

e introduciamo le seguenti matrici:

$$\boldsymbol{D} = \begin{pmatrix} A_{1,1} & 0 & \cdots & 0 \\ 0 & A_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & A_{n,n} \end{pmatrix},$$

$$\boldsymbol{L} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ A_{2,1} & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ A_{n,1} & \cdots & A_{n,n-1} & 0 \end{pmatrix},$$

$$\boldsymbol{U} = \begin{pmatrix} 0 & A_{1,2} & \cdots & A_{1,n} \\ 0 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & A_{n-1,n} \\ 0 & \cdots & 0 & 0 \end{pmatrix}.$$

Utilizzando le matrici $\boldsymbol{D}$, $\boldsymbol{L}$ e $\boldsymbol{U}$, possiamo decomporre la matrice $\boldsymbol{A}$ come segue:

$$\boldsymbol{A} = \boldsymbol{D} + \boldsymbol{L} + \boldsymbol{U}.$$

Grazie a questa *decomposizione* o *splitting* di $\boldsymbol{A}$, il sistema $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ può essere riscritto in vari modi:

$$(i) \qquad \boldsymbol{D}\boldsymbol{x} = \boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x},$$

$$(ii) \qquad (\boldsymbol{D} - \boldsymbol{L})\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{U}\boldsymbol{x},$$

$$(iii) \qquad (\boldsymbol{D} - \boldsymbol{U})\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{L}\boldsymbol{x}.$$

Osserviamo che nella forma (i) la matrice $\boldsymbol{D}$ è diagonale, mentre in (ii) la matrice $(\boldsymbol{D} - \boldsymbol{L})$ è triangolare inferiore e in (iii) la matrice $(\boldsymbol{D} - \boldsymbol{U})$ è triangolare superiore.

26.2 Metodo di Jacobi

La formulazione del sistema $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ in (i)

$$\boldsymbol{D}\boldsymbol{x} = \boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}$$

suggerisce uno schema di iterazione della forma

$$\boxed{\boldsymbol{D}\boldsymbol{x}^{k+1} = \boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}^{k}}$$

noto come *metodo iterativo di Jacobi*[1].

26.3 Metodo di Gauss-Seidel

Analogamente la riformulazione in (*ii*)

$$(\boldsymbol{D}+\boldsymbol{L})\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{U}\boldsymbol{x}$$

suggerisce un secondo schema di iterazione della forma

$$\boxed{(\boldsymbol{D}+\boldsymbol{L})\boldsymbol{x}^{k+1} = \boldsymbol{b} - \boldsymbol{U}\boldsymbol{x}^{k}}$$

noto come *metodo iterativo di Gauss*[2]*-Seidel*[3].

> Osservazione
>
> A questo punto, con un po' di creatività, possiamo considerare la riformulazione (*iii*):
>
> $$(\boldsymbol{D}+\boldsymbol{U})\boldsymbol{x} = \boldsymbol{b} - \boldsymbol{L}\boldsymbol{x}$$
>
> per definire un terzo schema iterativo nel seguente modo:
>
> $$\boxed{(\boldsymbol{D}+\boldsymbol{U})\boldsymbol{x}^{k+1} = \boldsymbol{b} - \boldsymbol{L}\boldsymbol{x}^{k}}.$$
>
> È importante notare che questo schema può essere interpretato come una variante del metodo di Gauss-Seidel. In effetti, esso deriva dalla forma (*ii*), ma applicata al sistema lineare con le equazioni e le incognite invertite. Si tratta, quindi, di uno schema di Gauss-Seidel *all'indietro*.

26.4 Metodo SOR

Separiamo la parte diagonale di $\boldsymbol{A}$, cioè $\boldsymbol{D}$, in due termini *pesati* con un coefficiente ω reale e non negativo, da determinare,

$$\boldsymbol{D} = \frac{1}{\omega}\boldsymbol{D} + \left(1 - \frac{1}{\omega}\right)\boldsymbol{D}.$$

Introduciamo questi due termini al posto della matrice $\boldsymbol{D}$ nella decomposizione utilizzata per il metodo di Gauss-Seidel

$$\left(\frac{1}{\omega}\boldsymbol{D} + \boldsymbol{L}\right)\boldsymbol{x} = \boldsymbol{b} - \left[\left(1 - \frac{1}{\omega}\right)\boldsymbol{D} + \boldsymbol{U}\right]\boldsymbol{x}.$$

Si ottiene quindi un nuovo schema iterativo

$$\boxed{\left(\frac{1}{\omega}\boldsymbol{D} + \boldsymbol{L}\right)\boldsymbol{x}^{k+1} = \boldsymbol{b} - \left(\frac{\omega-1}{\omega}\boldsymbol{D} + \boldsymbol{U}\right)\boldsymbol{x}^{k}}$$

noto come **S**uccessive **O**ver **R**elaxation *method*, da cui l'acronimo **SOR**. Si noti che il metodo **SOR** dipende da un parametro ω che deve essere individuato correttamente per garantire convergenza ed efficienza.

[1] Carl Gustav Jacob Jacobi (1804-1851)
[2] Johann Carl Friedrich Gauss (1777-1855)
[3] Philipp Ludwig von Seidel (1821-1896)

26.5 Derivazione per componenti dei metodi di Jacobi, Gauss–Seidel e SOR

Partiamo dallo splitting $\boldsymbol{A} = \boldsymbol{D} + \boldsymbol{L} + \boldsymbol{U}$ e dal sistema componente per componente

$$\sum_{j=1}^{n} A_{ij} x_j = b_i, \qquad i = 1, \dots, n.$$

Separiamo il termine diagonale $A_{ii}x_i$ dalla somma:

$$A_{ii} x_i + \sum_{j \neq i} A_{ij} x_j = b_i.$$

Risolviamo per x_i ottenendo la forma fondamentale che porta ai metodi iterativi.

26.5.1 Metodo di Jacobi

Nel metodo di Jacobi tutte le componenti del nuovo vettore $\boldsymbol{x}^{k+1}$ sono calcolate usando solo i valori vecchi $\boldsymbol{x}^{k}$. Da

$$A_{ii} x_i^{k+1} + \sum_{j \neq i} A_{ij} x_j^{k} = b_i,$$

otteniamo per ogni i

$$x_i^{k+1} = \frac{1}{A_{ii}} \Big(b_i - \sum_{j \neq i} A_{ij} x_j^{k} \Big) = \frac{1}{A_{ii}} \Big(b_i - \sum_{j=1}^{i-1} A_{ij} x_j^{k} - \sum_{j=i+1}^{n} A_{ij} x_j^{k} \Big).$$

Tutti gli x_i^{k+1} possono essere calcolati *in parallelo*.

Jacobi

```
Input: A[n,n], b[n], x^(0), tol, max_iter
Output: x ≈ soluzione di A x = b

k = 0
x_new = x^(0)

while k < max_iter:
    for i = 1 to n:
        sum = 0
        for j = 1 to n:
            if j != i:
                sum += A[i,j] * x[j]
        end
        x_new[i] = (b[i] - sum) / A[i,i]
    end
    if ||x_new - x|| < tol:
        break
    x = x_new
    k = k + 1
end
```

26.5.2 Metodo di Gauss–Seidel

Nel metodo di Gauss–Seidel, quando si calcola x_i^{k+1} si usano le componenti già aggiornate per gli indici $j < i$ e le componenti vecchie per $j > i$. Partendo da

$$A_{ii}x_i^{k+1} + \sum_{j<i} A_{ij}x_j^{k+1} + \sum_{j>i} A_{ij}x_j^k = b_i,$$

si ottiene

$$x_i^{k+1} = \frac{1}{A_{ii}}\Big(b_i - \sum_{j=1}^{i-1} A_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} A_{ij}x_j^k\Big).$$

Per $i = 1$ la somma $\sum_{j<1}$ è nulla, e per $i = n$ la somma $\sum_{j>n}$ è nulla.

Gauss–Seidel

```
Input: A[n,n], b[n], x^(0), tol, max_iter
Output: x ≈ soluzione di A x = b

k = 0
x = x^(0)

while k < max_iter:
    for i = 1 to n:
        sum1 = sum_{j=1}^{i-1} A[i,j] * x[j]   # valori aggiornati
        sum2 = sum_{j=i+1}^{n} A[i,j] * x[j]   # valori vecchi
        x[i] = (b[i] - sum1 - sum2) / A[i,i]
    end
    if ||A*x - b|| < tol:
        break
    k = k + 1
end
```

26.5.3 Metodo SOR (Successive Over-Relaxation)

Il metodo SOR introduce un parametro di rilassamento ω. La derivazione componente-wise combina il vecchio valore x_i^k con l'aggiornamento di Gauss–Seidel $\tilde{x}_i^{k+1}$:

$$\tilde{x}_i^{k+1} = \frac{1}{A_{ii}}\Big(b_i - \sum_{j=1}^{i-1} A_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} A_{ij}x_j^k\Big),$$

poi applichiamo il rilassamento:

$$x_i^{k+1} = (1-\omega)x_i^k + \omega\tilde{x}_i^{k+1},$$

ossia, sostituendo $\tilde{x}_i^{k+1}$:

$$x_i^{k+1} = (1-\omega)x_i^k + \frac{\omega}{A_{ii}}\Big(b_i - \sum_{j=1}^{i-1} A_{ij}x_j^{k+1} - \sum_{j=i+1}^{n} A_{ij}x_j^k\Big).$$

- $\omega = 1$ riduce al metodo sdi Gauss–Seidel.
- $0 < \omega < 1 \rightarrow$ *under-relaxation.*
- $1 < \omega < 2 \rightarrow$ *over-relaxation* (talvolta accelera la convergenza).

SOR (Successive Over-Relaxation)

```
Input: A[n,n], b[n], x^(0), omega, tol, max_iter
Output: x ≈ soluzione di A x = b

k = 0
x = x^(0)

while k < max_iter:
    for i = 1 to n:
        sum1 = sum_{j=1}^{i-1} A[i,j] * x[j]   # valori aggiornati
        sum2 = sum_{j=i+1}^{n} A[i,j] * x[j]   # valori vecchi
        x_i_GS = (b[i] - sum1 - sum2) / A[i,i]
        x[i] = (1 - omega) * x[i] + omega * x_i_GS
    end
    if ||A*x - b|| < tol:
        break
    k = k + 1
end
```

26.6 Generalizzazione

Scriviamo $\boldsymbol{A}$, matrice quadrata non singolare di ordine n, come differenza di due matrici $\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q}$ in cui $\boldsymbol{P}$ è non singolare e *facile da invertire*.

$$(\boldsymbol{P} - \boldsymbol{Q})\boldsymbol{x} = \boldsymbol{b}, \quad \Rightarrow \quad \boldsymbol{P}\boldsymbol{x} = \boldsymbol{Q}\boldsymbol{x} + \boldsymbol{b},$$

Da questa decomposizione si ricava in maniera naturale lo schema di iterazione

$$\boxed{\boldsymbol{P}\boldsymbol{x}^{k+1} = \boldsymbol{Q}\boldsymbol{x}^{k} + \boldsymbol{b}, \quad \Rightarrow \quad \boldsymbol{x}^{k+1} = \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{x}^{k} + \boldsymbol{P}^{-1}\boldsymbol{b},}$$

Definition 26.1 — Matrice di iterazione. La matrice $\boldsymbol{P}^{-1}\boldsymbol{Q}$, che possiamo anche scrivere come segue

$$\boldsymbol{P}^{-1}\boldsymbol{Q} = \boldsymbol{P}^{-1}\Big(\boldsymbol{P} - \boldsymbol{A}\Big) = \boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{A}$$

prende il nome di *matrice di iterazione*.

Ovviamente, è indifferente scrivere lo schema di iterazione come segue

$$\boldsymbol{x}^{k+1} = (\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{A})\boldsymbol{x}^{k} + \boldsymbol{P}^{-1}\boldsymbol{b},$$

Osservazione

Tutti i metodi precedenti introdotti finora, *Jacobi*, *Gauss-Seidel*, *SOR*, sono casi particolari del metodo generale.

Table 26.1: Quadro riassuntivo

	Jacobi	**Gauss-Seidel**	**SOR**
$\boldsymbol{P}$	$\boldsymbol{D}$	$\boldsymbol{D} + \boldsymbol{L}$	$\frac{1}{\omega}\boldsymbol{D} + \boldsymbol{L}$
$\boldsymbol{Q}$	$-(\boldsymbol{L} + \boldsymbol{U})$	$-\boldsymbol{U}$	$\frac{1-\omega}{\omega}\boldsymbol{D} - \boldsymbol{U}$

	Jacobi	Gauss-Seidel	SOR
$\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{A}$	$-\boldsymbol{D}^{-1}(\boldsymbol{L}+\boldsymbol{U})$	$-(\boldsymbol{D}+\boldsymbol{L})^{-1}\boldsymbol{U}$	$\left(\frac{\boldsymbol{D}}{\omega}+\boldsymbol{L}\right)^{-1}\left(\frac{1-\omega}{\omega}\boldsymbol{D}-\boldsymbol{U}\right)$

26.7 Convergenza degli schemi iterativi

Definition 26.2 — Convergenza. Uno schema iterativo della forma

$$\boldsymbol{x}^{k+1} = \boldsymbol{G}(\boldsymbol{x}^k)$$

si dice convergente se *esiste* un vettore $\boldsymbol{x}^*$ ed un norma vettoriale $\|\cdot\|$ tali che la successione delle iterate costruita a partire da un *qualsiasi* vettore iniziale $\boldsymbol{x}^0$ converge a $\boldsymbol{x}^*$ nella norma considerata

$$\lim_{k\to\infty} \left\|\boldsymbol{x}^k - \boldsymbol{x}^*\right\| = \boldsymbol{0}.$$

Osservazione

Si noti che $\boldsymbol{x}^*$ deve soddisfare la condizione

$$\boldsymbol{x}^* = \boldsymbol{G}(\boldsymbol{x}^*)$$

Uno schema convergente nel senso della definizione precedente ha convergenza *globale*. Per problemi non-lineari uno schema iterativo può essere convergente solo per un sottoinsieme di vettori di $\mathbb{R}^n$. In tal caso si parla di convergenza *locale*. Per problemi lineari, si assume sempre che la convergenza sia globale.

Questione:
sotto quali condizioni $\boldsymbol{x}^k$ converge alla soluzione $\boldsymbol{x}^*$?

Definition 26.3 — Errore. Definiamo errore $\boldsymbol{\varepsilon}^k$ la differenza tra la soluzione *esatta* $\boldsymbol{x}^*$ e la soluzione *approssimata* $\boldsymbol{x}^k$; cioè

$$\boldsymbol{\varepsilon}^k = \boldsymbol{x}^* - \boldsymbol{x}^k,$$

Quindi la condizione che lo schema iterativo sia convergente si può esprimere anche chiedendo che l'errore sia infinitesimo in norma

$$\lim_{k\mapsto\infty} \left\|\boldsymbol{\varepsilon}^k\right\| = \boldsymbol{0}.$$

26.8 Come si trasforma l'errore secondo lo schema iterativo generale?

Osserviamo che dalle due espressioni che definiscono $\boldsymbol{x}^*$ e $\boldsymbol{x}^{k+1}$

$$\begin{aligned}\boldsymbol{P}\boldsymbol{x}^* &= \boldsymbol{Q}\boldsymbol{x}^* + \boldsymbol{b},\\ \boldsymbol{P}\boldsymbol{x}^{k+1} &= \boldsymbol{Q}\boldsymbol{x}^k + \boldsymbol{b},\end{aligned}$$

si ottiene per differenza

$$\boldsymbol{P}\left(\boldsymbol{x}^* - \boldsymbol{x}^{k+1}\right) = \boldsymbol{Q}\left(\boldsymbol{x}^* - \boldsymbol{x}^k\right),$$

da cui segue iterando

$$\begin{aligned}\varepsilon^{k+1} &= \left(\boldsymbol{P}^{-1}\boldsymbol{Q}\right)\varepsilon^{k}\\ &= \left(\boldsymbol{P}^{-1}\boldsymbol{Q}\right)^{2}\varepsilon^{k-1}\\ &= \left(\boldsymbol{P}^{-1}\boldsymbol{Q}\right)^{3}\varepsilon^{k-2}\\ &= \dots\\ &= \left(\boldsymbol{P}^{-1}\boldsymbol{Q}\right)^{k+1}\varepsilon^{0}.\end{aligned}$$

Osservazione

La convergenza di un metodo iterativo può essere definita richiedendo che l'errore sia infinitesimale in una certa norma vettoriale.
Tuttavia, tale definizione dell'errore presuppone la conoscenza della soluzione esatta $\boldsymbol{x}^*$. Si osserva che:

$$\boldsymbol{A}\varepsilon^{k} = \boldsymbol{A}\boldsymbol{x}^{*} - \boldsymbol{A}\boldsymbol{x}^{k} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{k} = \boldsymbol{r}^{k},$$

da cui si deduce immediatamente che:

$$\boldsymbol{x}^{k} \mapsto \boldsymbol{x}^{*} \qquad \Leftrightarrow \qquad \boldsymbol{r}^{k} \mapsto 0.$$

Pertanto, studiare la convergenza di un metodo iterativo è equivalente a analizzare quando il residuo tende a zero.

26.8.1 Trasformazione del residuo secondo lo schema iterativo generale

Possiamo scrivere la matrice $\boldsymbol{A}$ come:

$$\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q} = \left(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{P}^{-1}\right)\boldsymbol{P}.$$

Da questa espressione, con alcuni passaggi ovvi, otteniamo:

$$\begin{aligned}\boldsymbol{A}\boldsymbol{P}^{-1}\boldsymbol{Q} &= \left(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{P}^{-1}\right)\boldsymbol{P}\boldsymbol{P}^{-1}\boldsymbol{Q},\\ &= \boldsymbol{Q} - \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{Q},\\ &= \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{P} - \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{Q},\\ &= \boldsymbol{Q}\boldsymbol{P}^{-1}\left(\boldsymbol{P} - \boldsymbol{Q}\right),\\ &= \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{A}.\end{aligned}$$

Infine, dalla relazione $\boldsymbol{A}\boldsymbol{P}^{-1}\boldsymbol{Q} = \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{A}$, possiamo dedurre:

$$\begin{aligned}\boldsymbol{r}^{k+1} &= \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{k+1},\\ &= \boldsymbol{b} - \boldsymbol{A}\boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{x}^{k} - \boldsymbol{A}\boldsymbol{P}^{-1}\boldsymbol{b},\\ &= \boldsymbol{b} - \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{x}^{k} - \left(\boldsymbol{P} - \boldsymbol{Q}\right)\boldsymbol{P}^{-1}\boldsymbol{b},\\ &= \boldsymbol{b} - \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{x}^{k} - \boldsymbol{b} + \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{b},\\ &= \boldsymbol{Q}\boldsymbol{P}^{-1}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{k}),\\ &= \boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{r}^{k}.\end{aligned}$$

Theorem 26.1 — Convergenza. Sia $\boldsymbol{A}$ una qualsiasi matrice quadrata. Allora le seguenti condizioni sono equivalenti:

1. $\lim_{k\mapsto\infty} \boldsymbol{A}^k = \boldsymbol{0}$.
2. $\lim_{k\mapsto\infty} \boldsymbol{A}^k\boldsymbol{v} = \boldsymbol{0}$ per ogni vettore $\boldsymbol{v}$.
3. $\varrho(\boldsymbol{A}) < 1$.
4. $\|\boldsymbol{A}\| < 1$ per almeno una norma matriciale.

Theorem 26.2 — Convergenza. Dato un sistema lineare $\boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{b}$ ed una decomposizione $\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q}$ con $\boldsymbol{P}$ non singolare, allora il metodo iterativo

$$\boldsymbol{P}\boldsymbol{x}^{k+1} = \boldsymbol{Q}\boldsymbol{x}^k + \boldsymbol{b},$$

converge alla soluzione $\boldsymbol{x}^*$ se e solo se vale una delle due:

$$(i) \qquad \boxed{\varrho(\boldsymbol{Q}\boldsymbol{P}^{-1}) < 1}$$

$$(ii) \qquad \boxed{\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) < 1}$$

Proof. Dimostriamo separatamente le due tesi del teorema.

- (i) Il residuo $k-$esimo può essere scritto come

  $$\boldsymbol{r}^k = (\boldsymbol{Q}\boldsymbol{P}^{-1})^k\boldsymbol{r}^0,$$

  per cui dal teorema~1 segue che

  $$\varrho(\boldsymbol{Q}\boldsymbol{P}^{-1}) < 1 \quad \Longrightarrow \quad \lim_{k\mapsto\infty} \boldsymbol{r}^k = \boldsymbol{0}.$$

  Viceversa, se il metodo converge, si ha sempre dal teorema Teorema 26.1 che

  $$\text{per ogni } \boldsymbol{r}^0, \quad \lim_{k\mapsto\infty} \boldsymbol{r}^k = \lim_{k\mapsto\infty} (\boldsymbol{Q}\boldsymbol{P}^{-1})^k\boldsymbol{r}^0 = \boldsymbol{0} \quad \Longrightarrow$$
  $$\Longrightarrow \quad \varrho(\boldsymbol{Q}\boldsymbol{P}^{-1}) < 1.$$

- (ii)
  Si dimostra analogamente sfruttando il modo in cui si trasforma l'errore durante il procedimento iterativo. Poiché

  $$\boldsymbol{\varepsilon}^k = (\boldsymbol{P}^{-1}\boldsymbol{Q})^k\boldsymbol{\varepsilon}^0,$$

  dal teorema Teorema 26.1 segue che

  $$\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) < 1 \quad \Longrightarrow \quad \lim_{k\mapsto\infty} \boldsymbol{\varepsilon}^k = \boldsymbol{0}.$$

  Viceversa, se il metodo converge, si ha sempre dal teorema~1 che

  $$\text{per ogni } \boldsymbol{\varepsilon}^0, \quad \lim_{k\mapsto\infty} \boldsymbol{\varepsilon}^k = \lim_{k\mapsto\infty} (\boldsymbol{P}^{-1}\boldsymbol{Q})^k\boldsymbol{\varepsilon}^0 = \boldsymbol{0} \quad \Longrightarrow$$
  $$\Longrightarrow \quad \varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) < 1.$$

■

Corollary 26.1 — Corollario. Il metodo di Jacobi converge alla soluzione se e solo se

$$\varrho\left(\boldsymbol{D}^{-1}(\boldsymbol{L}+\boldsymbol{U})\right) < 1.$$

Corollary 26.2 — Corollario. Il metodo di Gauss-Seidel converge se e solo se

$$\varrho\left((\boldsymbol{D}+\boldsymbol{L})^{-1}\boldsymbol{U}\right) < 1.$$

Corollary 26.3 — Corollario. Il metodo *SOR* converge se e solo se

$$\varrho\left(\left(\frac{\boldsymbol{D}}{\omega}+\boldsymbol{L}\right)^{-1}\left(\frac{1-\omega}{\omega}\boldsymbol{D}-\boldsymbol{U}\right)\right) < 1.$$

Definition 26.4 — Raggio spettrale. Il raggio spettrale della matrice di iterazione del metodo *SOR* soddisfa

$$\varrho\left(\left(\frac{1-\omega}{\omega}\boldsymbol{D}-\boldsymbol{U}\right)\left(\frac{\boldsymbol{D}}{\omega}+\boldsymbol{L}\right)^{-1}\right) \geq |1-\omega|.$$

Theorem 26.3 — SOR convergence. Scegliere $\omega \in (0,2)$ è condizione *necessaria* per la convergenza del metodo *SOR*.

Proof. Calcoliamo il determinante della matrice di iterazione del metodo *SOR*.

$$\begin{aligned}\left|\left[\frac{\boldsymbol{D}}{\omega}+\boldsymbol{L}\right]^{-1}\left[\frac{1-\omega}{\omega}\boldsymbol{D}-\boldsymbol{U}\right]\right| &= \left|\frac{\boldsymbol{D}}{\omega}+\boldsymbol{L}\right|^{-1}\left|\frac{1-\omega}{\omega}\boldsymbol{D}-\boldsymbol{U}\right| \\ &= \frac{\prod_{i=1}^{n}\frac{1-\omega}{\omega}\boldsymbol{D}_{i,i}}{\prod_{i=1}^{n}\frac{1}{\omega}\boldsymbol{D}_{i,i}} \\ &= (1-\omega)^n.\end{aligned}$$

Osserviamo ora che se $\boldsymbol{M}$ è una qualsiasi matrice con autovalori λ_i e raggio spettrale $\varrho(\boldsymbol{M})$, allora

$$\varrho(\boldsymbol{M})^n \geq \prod_{i=1}^{n}|\lambda_i| \geq |\det \boldsymbol{M}|.$$

Prendendo $\boldsymbol{M}$ uguale alla matrice di iterazione del metodo *SOR* si ha

$$\varrho\left(\left(\frac{\boldsymbol{D}}{\omega}+\boldsymbol{L}\right)^{-1}\left(\frac{1-\omega}{\omega}\boldsymbol{D}-\boldsymbol{U}\right)\right) \geq$$
$$\left|\det\left(\frac{\boldsymbol{D}}{\omega}+\boldsymbol{L}\right)^{-1}\left(\frac{1-\omega}{\omega}\boldsymbol{D}-\boldsymbol{U}\right)\right|^{\frac{1}{n}} = |1-\omega|.$$

Indipendentemente dalla matrice $\boldsymbol{A} = \boldsymbol{D}-\boldsymbol{L}-\boldsymbol{U}$, dato che

$$\omega \notin (0,2) \quad \Longrightarrow \quad |1-\omega| \geq 1.$$

abbiamo che il raggio spettrale è > 1 se $\omega \notin (0,2)$. ∎

26.9 Controllo della Convergenza

Se $\varepsilon^{k-1} \neq \mathbf{0}$, la quantità $\|\varepsilon^k\| / \|\varepsilon^{k-1}\|$ esprime la riduzione dell'errore al k-esimo passo. La media geometrica delle riduzioni dell'errore sui primi k passi

$$\sigma_k = \left(\frac{\|\varepsilon^1\|}{\|\varepsilon^0\|} \frac{\|\varepsilon^2\|}{\|\varepsilon^1\|} \cdots \frac{\|\varepsilon^k\|}{\|\varepsilon^{k-1}\|} \right)^{1/k} = \left(\frac{\|\varepsilon^k\|}{\|\varepsilon^0\|} \right)^{1/k}$$

esprime la *riduzione media per passo* dell'errore.
Dalle proprietà delle norme naturali segue che

$$\varepsilon^k = (\boldsymbol{P}^{-1}\boldsymbol{Q})^k \varepsilon^0 \quad \Longrightarrow \quad \|\varepsilon^k\| \leq \|(\boldsymbol{P}^{-1}\boldsymbol{Q})^k\| \, \|\varepsilon^0\|$$

e quindi si ottiene la relazione

$$\sigma_k \leq \left(\|(\boldsymbol{P}^{-1}\boldsymbol{Q})^k\| \right)^{\frac{1}{k}}.$$

Quindi σ_k stima la velocità di convergenza sulle prime k iterazioni ed è controllato da una quantità che dipende dalla matrice di iterazione dello schema. Come possiamo stimare questa quantità?

26.9.1 Stima della velocità di convergenza

Theorem 26.4 — Limite potenza e norma. Sia $\boldsymbol{M} \in \mathbb{K}^{m\times n}$ e $\|\cdot\|$ una *qualunque* norma indotta. Allora

$$\lim_{k\mapsto\infty} \|(\boldsymbol{M})^k\|^{\frac{1}{k}} = \varrho(\boldsymbol{M}).$$

Osservazione

La quantità $\varrho(\boldsymbol{M})$ non dipende dalla norma matriciale utilizzata e nemmeno dall'indice di iterazione k. Possiamo assumerla come una *stima della velocità di convergenza* sui nostri metodi iterativi ponendo $\boldsymbol{M} = \boldsymbol{P}^{-1}\boldsymbol{Q}$, cioè uguale alla matrice di iterazione.

Definition 26.5 — Tasso asintotico di convergenza. Si definisce *tasso asintotico di convergenza* di uno schema iterativo la costante

$$C = -\log_{10} \varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}).$$

La costante C è all'incirca il numero di iterazioni richieste per ridurre l'errore di un fattore $\frac{1}{10}$, cioè grossomodo per avere ottenere una cifra decimale in più. Infatti

$$\left(\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})\right)^k \approx \frac{1}{10} \quad \Longrightarrow \quad k \approx -\frac{1}{\log_{10} \varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}.$$

26.9.2 Criteri di Arresto

Sia fissata una tolleranza η sull'errore. I criteri più comunemente usati sfruttano un controllo sulla variazione assoluta e relativa delle iterate al generico passo k

$$\textbf{IF} \quad \left\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\right\| \leq \eta \qquad \textbf{THEN STOP}$$
$$\textbf{IF} \quad \left\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\right\| \leq \eta \left\|\boldsymbol{x}^k\right\| \qquad \textbf{THEN STOP}$$

Tuttavia è importante rendersi conto che queste condizioni non garantiscono che la soluzione sia stata approssimata con accuratezza η

Infatti, si ha che

$$\begin{aligned}\boldsymbol{x}^k - \boldsymbol{x}^{k-1} &= \boldsymbol{x}^k - \boldsymbol{x}^* + \boldsymbol{x}^* - \boldsymbol{x}^{k-1} \\ &= -\boldsymbol{\varepsilon}^k + \boldsymbol{\varepsilon}^{k-1} \\ &= -\boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{\varepsilon}^{k-1} + \boldsymbol{\varepsilon}^{k-1} \\ &= \left(\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q}\right)\boldsymbol{\varepsilon}^{k-1}\end{aligned}$$

per cui possiamo esprimere l'errore al passo $k-1$ in funzione della variazione assoluta delle iterate al passo k

$$\boldsymbol{\varepsilon}^{k-1} = \left(\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q}\right)^{-1}\left(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\right).$$

Dalla relazione

$$\boldsymbol{\varepsilon}^{k-1} = \left(\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q}\right)^{-1}\left(\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\right), \qquad \text{(ripetiamo)}$$

passando alle norme se $\left\|\boldsymbol{P}^{-1}\boldsymbol{Q}\right\| \leq 1$ si ha

$$\begin{aligned}\left\|\boldsymbol{\varepsilon}^{k-1}\right\| &\leq \left\|(\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q})^{-1}\right\| \left\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\right\| \\ &\leq \frac{\left\|\boldsymbol{x}^k - \boldsymbol{x}^{k-1}\right\|}{1 - \left\|\boldsymbol{P}^{-1}\boldsymbol{Q}\right\|}\end{aligned}$$

per cui se $\left\|\boldsymbol{P}^{-1}\boldsymbol{Q}\right\| \approx 1$ l'errore in norma $\left\|\boldsymbol{\varepsilon}^{k-1}\right\|$ può essere ancora grande.

26.10 Quale metodo converge più in fretta

I criteri visti finora dicono *se* un metodo converge, non quale dei due converge più in fretta. Il confronto si fa sul raggio spettrale della matrice di iterazione, e il risultato chiave è il seguente: fra due splitting della stessa matrice, converge più rapidamente quello che lascia **meno** matrice in $\boldsymbol{Q}$.

> **Confronto fra due splitting**
>
> Siano $\boldsymbol{A} = \boldsymbol{P}_1 - \boldsymbol{Q}_1 = \boldsymbol{P}_2 - \boldsymbol{Q}_2$ due splitting debolmente non negativi della stessa matrice. Se $\boldsymbol{A}^{-1} \geq \boldsymbol{0}$ e $\boldsymbol{Q}_2 \geq \boldsymbol{Q}_1$ allora
>
> $$\varrho(\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1) \leq \varrho(\boldsymbol{P}_2^{-1}\boldsymbol{Q}_2) < 1.$$
>
> La dimostrazione è in appendice, Teorema C.4.

Applichiamolo a Jacobi e Gauss-Seidel. Con $\boldsymbol{A} = \boldsymbol{D} - \boldsymbol{L} - \boldsymbol{U}$ si ha

$$\boldsymbol{Q}_{\text{J}} = \boldsymbol{L} + \boldsymbol{U}, \qquad \boldsymbol{Q}_{\text{GS}} = \boldsymbol{U}.$$

Se $\boldsymbol{A}$ è una M-matrice (Definizione 25.1) i suoi elementi fuori diagonale sono non positivi, quindi $\boldsymbol{L} \geq \boldsymbol{0}$ e $\boldsymbol{U} \geq \boldsymbol{0}$, e inoltre $\boldsymbol{A}^{-1} \geq \boldsymbol{0}$ per definizione. Allora

$$\boldsymbol{Q}_{\text{GS}} = \boldsymbol{U} \;\leq\; \boldsymbol{L} + \boldsymbol{U} = \boldsymbol{Q}_{\text{J}} \qquad \Longrightarrow \qquad \varrho\left(\boldsymbol{P}_{\text{GS}}^{-1}\boldsymbol{Q}_{\text{GS}}\right) \leq \varrho\left(\boldsymbol{P}_{\text{J}}^{-1}\boldsymbol{Q}_{\text{J}}\right) < 1,$$

cioè **su una M-matrice Gauss-Seidel non è mai più lento di Jacobi**, e i due convergono o divergono insieme. Si noti che l'ipotesi è necessaria: senza di essa esistono matrici per cui vale il contrario.

Lo stesso criterio ordina famiglie più ampie di metodi. L'appendice Appendice C inquadra Jacobi, Gauss-Seidel e EWA come casi particolari di un'unica costruzione per splitting, riassunta in Tabella C.1, e ne confronta le velocità di convergenza.

In sintesi

- Ogni metodo iterativo nasce da uno **splitting** $\boldsymbol{A} = \boldsymbol{M} - \boldsymbol{N}$ con $\boldsymbol{M}$ facile da invertire; la matrice di iterazione è $\boldsymbol{B} = \boldsymbol{M}^{-1}\boldsymbol{N}$.
- Jacobi, Gauss-Seidel e SOR differiscono solo per la scelta di $\boldsymbol{M}$: diagonale, triangolare inferiore, combinazione pesata da ω.
- La condizione **necessaria e sufficiente** di convergenza è $\rho(\boldsymbol{B}) < 1$, per ogni dato iniziale; $\|\boldsymbol{B}\| < 1$ in una qualsiasi norma è solo sufficiente.
- L'errore e il residuo si trasformano con la stessa matrice $\boldsymbol{B}$: il residuo è osservabile, l'errore no, ed è questo che si usa come criterio d'arresto.
- Il parametro ω di SOR va scelto, non indovinato: per matrici consistentemente ordinate esiste un ω_{opt} esplicito in funzione di $\rho(\boldsymbol{B}_{\text{Jacobi}})$.

VIII Equazioni differenziali ordinarie

27 Equazioni Differenziali Ordinarie 339
27.1 Introduzione ai Problemi a Valore Iniziale 339
27.2 Buona posizione dei problemi a valore iniziale 341
27.3 Teoremi Fondamentali 342

28 Soluzione numerica 349
28.1 Passo Temporale 349
28.2 Principali metodi numerici per IVP 349
28.3 Il Metodo di Eulero 350
28.4 Convergenza Metodo di Eulero 350
28.5 Verifica Numerica dell'Ordine 360

29 Metodi di Taylor per ODE 365
29.1 Metodo di Eulero e Serie di Taylor 365
29.2 Metodo di Taylor al Secondo Ordine 366
29.3 Metodo di Taylor al Terzo Ordine 366
29.4 Metodo basato su Taylor con $f(t, y)$ nota 368
29.5 Metodo di Taylor migliorato 370
29.6 Sistemi di ODE 372

30 Metodi di Runge-Kutta 383
30.1 Formulazione Generale 383
30.2 Condizioni di Ordine 386
30.3 Metodi di Runge-Kutta Espliciti 387
30.4 Controllo adattivo del passo 391
30.5 Metodi di Runge-Kutta Impliciti 393

31 Stabilità assoluta 401
31.1 L'equazione test di Dahlquist 401
31.2 Funzione e regione di stabilità 401
31.3 I metodi fondamentali 402
31.4 A-stabilità e L-stabilità 404
31.5 Dal caso scalare ai sistemi 404
31.6 Diagrammi di stabilità 405
31.7 Stabilità e accuratezza 406
31.8 Quadro riassuntivo 407

References 409



27. Equazioni Differenziali Ordinarie

Un'**equazione differenziale ordinaria** (ODE) è un'equazione che coinvolge una o più funzioni di una sola variabile indipendente e le loro derivate. Il termine **ordinaria** distingue queste equazioni dalle **equazioni differenziali alle derivate parziali**, che dipendono da più variabili indipendenti.

Le ODE sono strumenti matematici ampiamente utilizzati per modellare e analizzare sistemi dinamici in diversi ambiti, tra cui la fisica, l'economia, la biologia e le scienze sociali. Studiosi come Newton[1], Euler, Leibniz, i Bernoulli, Riccati, Clairaut e d'Alembert hanno contribuito allo sviluppo e all'applicazione di queste equazioni.

Un esempio noto di ODE è la seconda legge del moto di Newton:

$$\mathcal{F}(y(t)) = my''(t)$$

dove $\mathcal{F}$ rappresenta una forza applicata come funzione della posizione $y(t)$ e m è la massa. Questa equazione descrive come l'accelerazione di un oggetto sia legata alla forza applicata, un principio fondamentale della meccanica classica.

> Convenzioni di notazione
>
> In tutto il capitolo t indica la variabile indipendente e y l'incognita. La teoria è enunciata per $y(t)$ a valori in $\mathbb{R}^d$, ma nei capitoli numerici che seguono la trattazione è scritta nel **caso scalare** per non appesantire le formule: tutti gli schemi e tutte le stime si estendono ai sistemi sostituendo y con $\boldsymbol{y}$, f con $\boldsymbol{f}$ e il valore assoluto con una norma. Restano in grassetto le quantità che sono vettori o matrici anche nel caso scalare, come i coefficienti $\boldsymbol{A}$, $\boldsymbol{b}$, $\boldsymbol{c}$ del tableau di Butcher.

27.1 Introduzione ai Problemi a Valore Iniziale

Prima di esaminare i metodi numerici di soluzione, è utile richiamare alcune definizioni e proprietà delle equazioni differenziali ordinarie (ODE) e dei problemi a valore iniziale (IVP).

27.1.1 Equazione Differenziale Ordinaria

Un'**equazione differenziale ordinaria** (ODE) è un'equazione che coinvolge una funzione $y(t)$ definita su un intervallo $I \subset \mathbb{R}$ e a valori reali, complessi oppure nello spazio vettoriale $\mathbb{R}^d$ o $\mathbb{C}^d$. La forma generale di un'ODE è:

[1] Isaac Newton (1642-1727)

$$F\left(t, y(t), y'(t), y''(t), \dots, y^{(n)}(t)\right) = 0$$

L'**ordine** dell'equazione differenziale è determinato dalla derivata di ordine più alto $y^{(n)}(t)$ presente.

Se $y(t)$ assume valori in uno spazio di dimensione $d > 1$, si parla di **sistema di equazioni differenziali**.

27.1.2 ODE esplicita

Un'**equazione differenziale esplicita** del primo ordine ha la forma:

$$y'(t) = f\left(t, y(t)\right)$$

Un'equazione di ordine superiore (ordine n) si dice esplicita se può essere scritta nella forma:

$$y^{(n)}(t) = F\left(t, y(t), y'(t), y''(t), \dots, y^{(n-1)}(t)\right)$$

27.1.3 ODE autonoma

Un'equazione differenziale della forma precedente è chiamata **autonoma** se il termine a destra f non dipende esplicitamente da t, il che significa:

$$y'(t) = f\left(y(t)\right)$$

27.1.4 Riduzione a sistemi del primo ordine

Ogni equazione differenziale di ordine superiore può essere riscritta come un sistema di equazioni del primo ordine. Se l'equazione originale è esplicita, anche il sistema risultante sarà esplicito.

Per trasformare un'equazione differenziale di ordine superiore in un sistema di equazioni del primo ordine, introduciamo variabili ausiliarie per ciascuna derivata di $y(t)$:

$$y_0(t) = y(t), \quad y_1(t) = y'(t), \quad \dots, \quad y_{n-1}(t) = y^{(n-1)}(t)$$

In questo modo, l'equazione differenziale di ordine n si riscrive come un sistema di n equazioni del primo ordine:

$$\begin{Bmatrix} y_0'(t) - y_1(t) \\ y_1'(t) - y_2(t) \\ \vdots \\ y_{n-2}'(t) - y_{n-1}(t) \\ F\left(t, y_0(t), y_1(t), \dots, y_{n-1}(t), y_{n-1}'(t)\right) \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{Bmatrix}$$

Nel caso di un'equazione esplicita, questo sistema diventa:

$$\begin{Bmatrix} y_0'(t) \\ y_1'(t) \\ \vdots \\ y_{n-2}'(t) \\ y_{n-1}'(t) \end{Bmatrix} = \begin{Bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_{n-1}(t) \\ F\left(t, y_0(t), y_1(t), \dots, y_{n-1}(t)\right) \end{Bmatrix}$$

27.1.5 Problema a Valore Iniziale (IVP)

Il *problema a valore iniziale* (IVP), detto anche *problema di Cauchy*[2], associa all'ODE una *condizione iniziale*. Essa specifica il valore della funzione y in un punto t_0 del dominio e, sotto opportune ipotesi, permette di individuare una soluzione unica.

Data una coppia iniziale $(t_0, y_0) \in \mathbb{R} \times \mathbb{R}^d$, consideriamo $f(t, y(t))$ una funzione a valori in $\mathbb{R}^d$, definita in un intorno $I \times U \subset \mathbb{R} \times \mathbb{R}^d$ del punto iniziale. Il problema a valore iniziale consiste nel trovare una funzione $y(t)$ tale che:

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases}$$

27.1.5.1 Soluzione Locale

Una *soluzione locale* del problema a valore iniziale (IVP) è una funzione $y(t)$ continuamente derivabile che soddisfa la condizione $y(t_0) = y_0$. È definita su un intervallo J contenente t_0, nel quale sono ben definite sia $y(t)$ sia $f(t, y(t))$.

27.1.5.2 Equazione Differenziale Lineare

Un'equazione differenziale è considerata *lineare* se può essere espressa come una combinazione lineare delle derivate di $y(t)$:

$$y^{(n)}(t) = \sum_{i=1}^{n-1} a_i(t) y^{(i)}(t) + r(t)$$

dove $a_i(t)$ e $r(t)$ sono funzioni continue di t. Se $r(t) = 0$, l'equazione si dice *omogenea*; altrimenti, si dice *non omogenea*.

27.2 Buona posizione dei problemi a valore iniziale

Un problema matematico si dice *ben posto* se soddisfa le seguenti condizioni di Hadamard[3]:

- Esiste una soluzione.
- La soluzione è unica.
- La soluzione dipende continuamente dai dati iniziali.

Per i problemi a valore iniziale (IVP), la terza condizione è spesso sostituita da un criterio quantitativo noto come *continuità di Lipschitz*[4].

27.2.1 Condizione di Lipschitz

La funzione $f(t, y)$ soddisfa una *condizione di Lipschitz uniforme* nel suo dominio $D = I \times \Omega \subset \mathbb{R} \times \mathbb{R}^d$ rispetto a y se esiste una costante positiva L tale che:

$$\forall t \in I, \quad \forall x, y \in \Omega : |f(t, x) - f(t, y)| \leq L|x - y|$$

Se questa condizione vale per sottoinsiemi compatti di D, si parla di *condizione di Lipschitz locale*.

27.2.2 Funzione Localmente Lipschitziana

Una funzione $f(x)$ è localmente Lipschitziana se esistono un $\Delta > 0$ e una costante L tali che:

$$|f(x) - f(x_0)| \leq L|x - x_0|, \quad |x - x_0| < \Delta$$

[2] Augustin-Louis Cauchy (1789-1857)
[3] Jacques Salomon Hadamard (1865-1963)
[4] Rudolph Otto Sigismund Lipschitz (1832-1903)

▪ **Example 27.1 — Esempio di funzione non Lipschitziana.** La funzione

$$f(x) = \text{sign}(x)\sqrt{|x|}$$

non è Lipschitziana, infatti se lo fosse allora esiste L tale che:

$$\begin{aligned} |\sqrt{x} - \sqrt{0}| \leq L|x-0|, &\qquad \text{for } x > 0 \\ |\sqrt{x}| \leq L|x| = L|\sqrt{x}||\sqrt{x}| & \\ 1 \leq L|\sqrt{x}|, &\qquad \text{for } x > 0 \end{aligned}$$

ma per ogni L basta scegliere x abbastanza piccolo per violare $1 \leq L|\sqrt{x}|$.

▪

Osservazione

Se $f(x)$ è (localmente) Lipschitziana allora $f(x)$ è continua. Infatti $f(x)$ è continua in x_0 se per ogni $\epsilon > 0$ esiste $\delta > 0$ tale che

$$|f(x) - f(x_0)| \leq \epsilon, \qquad \forall |x - x_0| \leq \delta$$

ma $f(x)$ è Lipschitziana quindi

$$|f(x) - f(x_0)| \leq L|x - x_0|, \qquad |x - x_0| \leq \Delta \tag{27.1}$$

dato $\epsilon > 0$ scegliendo

$$\delta \leq \min\left(\frac{\epsilon}{L}, \Delta\right)$$

($\delta \leq \Delta$ serve per avere la diseguaglianza 27.1) allora vale

$$|f(x) - f(x_0)| \leq L|x - x_0| \leq L\delta = \min(\epsilon, L\Delta) \leq \epsilon$$

Osservazione

Se $f(x) \in C^1$ allora è Lipschitziana (in un intervallo chiuso). Usando la espansione di Taylor[5]

$$f(x) = f(x_0) + f'(y)(x - x_0), \qquad x \in I(x_0, x)$$

da cui

$$|f(x) - f(x_0)| \leq |f'(y)|\,|x - x_0| \leq \sup_{w \in [a,b]} |f'(w)|\,|x - x_0| \leq L|x - x_0|$$

dove $L = \sup_{w\in[a,b]} |f'(w)|$ e la diseguaglianza vale per ogni $x, x_0 \in [a, b]$.

27.3 Teoremi Fondamentali

27.3.1 Teorema di Esistenza di Peano

Il teorema di Peano[6] garantisce l'esistenza di almeno una soluzione di un IVP se la funzione $f(t, y)$ è continua in un intorno del punto iniziale (t_0, y_0).

[5] Brook Taylor (1685-1731)
[6] Giuseppe Peano (1858-1932)

27.3.1.1 Enunciato del Teorema

Sia $f(t, y)$ una funzione continua definita sull'insieme chiuso:

$$\bar{D} = \left\{(t, y) \in \mathbb{R} \times \mathbb{R}^d \mid |t - t_0| \leq \alpha,\ |y - y_0| \leq \beta\right\}$$

con $\alpha, \beta > 0$. Allora esiste almeno una soluzione $y(t)$ continua nell'intervallo $I = [t_0 - T, t_0 + T]$, dove $T = \min(\alpha, \frac{\beta}{M})$ e $M = \max_{(t,y)\in D} |f(t, y)|$.

▪ **Example 27.2 — Esempio di non unicità di Peano.** la seguente ODE

$$\begin{cases} y' = f(x, y) = \sqrt{|y|}, \\ y(0) = 0, \end{cases}$$

ha due soluzioni

- (a) $y(t) = 0$
- (b) $y(t) = \frac{t^2}{4}\text{sign}(t)$

la prima soluzione si verifica immediatamente. Per la seconda basta osservare che

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t}\frac{t^2}{4}\text{sign}(t) &= \frac{\mathrm{d}}{\mathrm{d}t}\begin{cases} \frac{t^2}{4} & t > 0 \\ -\frac{t^2}{4} & t \leq 0 \end{cases} \\ &= \begin{cases} \frac{t}{2} & t > 0 \\ -\frac{t}{2} & t \leq 0 \end{cases} \\ &= \frac{|t|}{2} \end{aligned}$$

e

$$\sqrt{|y(t)|} = \sqrt{\left|\frac{t^2}{4}\text{sign}(t)\right|} = \sqrt{\frac{t^2}{4}} = \frac{|t|}{2}$$

e quindi è una seconda soluzione dell'ODE.

▪

27.3.2 Teorema di Picard-Lindelöf

Il teorema afferma che se $f(t, y)$ soddisfa la condizione Lipschitz rispetto a y, allora esiste una soluzione unica locale per l'IVP.

Theorem 27.1 — Esistenza e Unicità: Picard-Lindelöf. Sia $f(t, y)$ continua nel cilindro:

$$D = \{(t, y) \in \mathbb{R} \times \mathbb{R}^d \mid |t - t_0| \leq a, |y - y_0| \leq b\}$$

Inoltre, supponiamo che f sia limitata, con una costante

$$M = \max_{(t,y)\in D} |f(t, y)|$$

e soddisfi la *condizione di Lipschitz* con costante L su D. Allora, il problema a valore iniziale:

$$\begin{cases} y'(t) = f(t, y(t)) \\ y(t_0) = y_0 \end{cases}$$

ha una soluzione unica nell'intervallo $I = [t_0 - T, t_0 + T]$, dove:

$$T = \min\left(a, \frac{b}{M}\right)$$

Per stabilire l'*unicità* della soluzione è necessario utilizzare la *condizione di Lipschitz* insieme al *Lemma di Grönwall*[7], che è lo strumento chiave per controllare la crescita di una funzione che soddisfa una disuguaglianza integrale.

27.3.3 Lemma di Grönwall

Lemma 27.1 — Disuguaglianza di Grönwall. Sia $u : [t_0, T] \to [0, +\infty)$ continua e sia $b : [t_0, T] \to [0, +\infty)$ integrabile. Se

$$u(t) \le A + \int_{t_0}^{t} b(s)\, u(s)\, \mathrm{d}s, \qquad A \ge 0, \quad t \in [t_0, T] \tag{27.2}$$

allora

$$u(t) \le A \exp\left(\int_{t_0}^{t} b(s)\, \mathrm{d}s\right), \qquad t \in [t_0, T] \tag{27.3}$$

Proof. Si definisca il membro destro della 27.2

$$v(t) = A + \int_{t_0}^{t} b(s)\, u(s)\, \mathrm{d}s$$

cosicché $u(t) \le v(t)$ e v è assolutamente continua con $v(t_0) = A$. Per quasi ogni t vale

$$v'(t) = b(t)\, u(t) \le b(t)\, v(t)$$

dove si è usato $u \le v$ e $b \ge 0$. Posto

$$B(t) = \int_{t_0}^{t} b(s)\, \mathrm{d}s$$

si moltiplica la disequazione differenziale per il fattore integrante $e^{-B(t)} > 0$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\Big(e^{-B(t)} v(t)\Big) = e^{-B(t)}\Big(v'(t) - b(t)\, v(t)\Big) \le 0$$

quindi $t \mapsto e^{-B(t)} v(t)$ è non crescente e dunque

$$e^{-B(t)} v(t) \le e^{-B(t_0)} v(t_0) = A \qquad \Longrightarrow \qquad v(t) \le A\, e^{B(t)}$$

Essendo $u(t) \le v(t)$ si ottiene la 27.3. □ ∎

Due casi particolari sono di uso continuo:

- se $b(s) \equiv L$ costante, la stima diventa

$$u(t) \le A\, e^{L(t - t_0)}$$

[7] Thomas Hakon Grönwall (1877-1932)

- se $A = 0$, allora necessariamente $u(t) \equiv 0$: è questo il meccanismo che produce l'unicità.

Esiste anche una forma più generale, utile per le stime di perturbazione: se $a(t)$ è non negativa e **non decrescente** e

$$u(t) \le a(t) + \int_{t_0}^{t} b(s)\, u(s)\, \mathrm{d}s$$

allora, con lo stesso argomento,

$$u(t) \le a(t) \exp\left(\int_{t_0}^{t} b(s)\, \mathrm{d}s \right)$$

27.3.4 Dimostrazione di Picard-Lindelöf

La dimostrazione è *costruttiva*: produce una successione di funzioni che converge alla soluzione. È sufficiente dimostrare il risultato sull'intervallo in avanti

$$I_+ = [t_0,\ t_0 + T]$$

perché la dimostrazione su $[t_0 - T, t_0]$ è identica a meno di invertire il verso del tempo, e l'unicità permette di raccordare le due soluzioni.

Proof. iterate di Picard

Il punto di partenza è l'osservazione che l'IVP è equivalente all'*equazione integrale*

$$y(t) = y_0 + \int_{t_0}^{t} f\big(s, y(s)\big)\, \mathrm{d}s \tag{27.4}$$

Si parte dalla funzione costante $y^{(0)}(t) = y_0$ e si definiscono ricorsivamente le *iterate di Picard*[8]

$$y^{(n+1)}(t) = y_0 + \int_{t_0}^{t} f\big(s, y^{(n)}(s)\big)\, \mathrm{d}s \tag{27.5}$$

1. Ogni iterata resta nel cilindro. Se $\|y^{(n)}(t) - y_0\| \le b$ per $t \in I_+$, allora

$$\|y^{(n+1)}(t) - y_0\| \le \int_{t_0}^{t} \|f(s, y^{(n)}(s))\|\, \mathrm{d}s \le M\,|t - t_0| \le M\,T \le b$$

per la definizione $T = \min(a, b/M)$. Poiché $y^{(0)}$ soddisfa la condizione, per induzione tutte le iterate sono ben definite su I_+ e restano in D.

2. La successione è uniformemente di Cauchy. La prima differenza soddisfa

$$\|y^{(1)}(t) - y^{(0)}(t)\| \le M\,|t - t_0|$$

e, usando la condizione di Lipschitz, per induzione si ottiene

$$\|y^{(n+1)}(t) - y^{(n)}(t)\| \le M\,L^{n}\,\frac{|t - t_0|^{n+1}}{(n+1)!} \tag{27.6}$$

Infatti, supposta vera la stima al livello $n-1$,

$$\begin{aligned} \|y^{(n+1)}(t) - y^{(n)}(t)\| &\le L \int_{t_0}^{t} \|y^{(n)}(s) - y^{(n-1)}(s)\|\, \mathrm{d}s \\ &\le L \int_{t_0}^{t} M\,L^{n-1}\,\frac{(s - t_0)^{n}}{n!}\, \mathrm{d}s = M\,L^{n}\,\frac{|t - t_0|^{n+1}}{(n+1)!} \end{aligned}$$

[8] Charles Émile Picard (1856-1941)

Uniformemente su I_+ vale quindi

$$\left\|y^{(n+1)}(t) - y^{(n)}(t)\right\| \le M\,L^n\,\frac{T^{n+1}}{(n+1)!}$$

e la serie numerica

$$\sum_{n=0}^{\infty} M\,L^n\,\frac{T^{n+1}}{(n+1)!} = \frac{M}{L}\left(e^{LT} - 1\right)$$

converge. Per il criterio di Weierstrass[9] la successione $y^{(n)}$ converge *uniformemente* su I_+ ad una funzione continua y. Si noti che la presenza del fattoriale nella 27.6 garantisce la convergenza su tutto l'intervallo consentito dal vincolo $MT \le b$, **senza** richiedere $LT < 1$.

3. Il limite soddisfa l'equazione integrale. Dalla condizione di Lipschitz

$$\left\|f(t, y^{(n)}(t)) - f(t, y(t))\right\| \le L\left\|y^{(n)}(t) - y(t)\right\|$$

per cui anche gli integrandi convergono uniformemente ed è lecito passare al limite nella 27.5, ottenendo la 27.4. Poiché l'integrando è continuo, il teorema fondamentale del calcolo integrale dà

$$y'(t) = f(t, y(t)), \qquad y(t_0) = y_0$$

4. La soluzione è unica. Siano w e y due soluzioni con lo stesso dato iniziale. Sottraendo le rispettive equazioni integrali e usando la condizione di Lipschitz

$$\|w(t) - y(t)\| \le L \int_{t_0}^{t} \|w(s) - y(s)\|\,\mathrm{d}s$$

per $t \ge t_0$ (per $t \le t_0$ l'argomento è identico invertendo gli estremi di integrazione). Il Lemma 27.1 con $A = 0$ e $b \equiv L$ forza $\|w(t) - y(t)\| = 0$, cioè $w \equiv y$ su I. □ ■

> Osservazione
>
> Una dimostrazione più breve si ottiene applicando il *teorema delle contrazioni di Banach*[10] all'operatore definito dalla 27.4, ma solo su un intervallo abbastanza corto da garantire $LT < 1$. La stima con il fattoriale usata sopra è didatticamente più forte perché mostra direttamente perché la successione di Picard converge su **tutto** l'intervallo permesso dal vincolo geometrico $MT \le b$.

27.3.5 Dipendenza continua dai dati iniziali

La stessa disuguaglianza di Grönwall fornisce molto più dell'unicità: dà una stima *quantitativa* di quanto due soluzioni possano allontanarsi se partono da dati iniziali diversi.

Siano w e y due soluzioni della stessa ODE con $w(t_0) = w_0$ e $y(t_0) = y_0$. Dalle equazioni integrali

$$\|w(t) - y(t)\| \le \|w_0 - y_0\| + L \int_{t_0}^{t} \|w(s) - y(s)\|\,\mathrm{d}s$$

e il Lemma 27.1 con $A = \|w_0 - y_0\|$ dà

$$\boxed{\|w(t) - y(t)\| \le e^{L|t - t_0|}\,\|w_0 - y_0\|} \tag{27.7}$$

[9] Karl Theodor Wilhelm Weierstrass (1815-1897)
[10] Stefan Banach (1892-1945)

Quindi la stessa ipotesi che garantisce l'unicità produce anche la *dipendenza continua dai dati iniziali* e dunque la **buona posizione** locale del problema di Cauchy, nel senso di Hadamard: esistenza, unicità e dipendenza continua.

Attenzione

La 27.7 è una stima *pessimistica*: il fattore $e^{L|t-t_0|}$ cresce esponenzialmente anche quando le soluzioni reali restano vicine (ad esempio per $y' = -y$, dove $L = 1$ ma le soluzioni si avvicinano). Essa garantisce la buona posizione su intervalli finiti, non la stabilità per tempi lunghi. Questa distinzione è esattamente quella che ritroveremo fra *convergenza* (comportamento per $h \to 0$ su intervallo finito) e *stabilità assoluta* (comportamento per $n \to \infty$ a passo h fissato) dei metodi numerici.

27.3.6 Prolungamento delle soluzioni

Il teorema di Picard-Lindelöf è un risultato **locale**: garantisce esistenza e unicità solo sull'intervallo $I = [t_0 - T, t_0 + T]$. Nulla impedisce che la soluzione cessi di esistere in tempo finito.

Theorem 27.2 — Teorema di prolungamento. Sia $D \subset \mathbb{R} \times \mathbb{R}^d$ aperto e sia $f : D \to \mathbb{R}^d$ continua e localmente lipschitziana rispetto a y. Sia $y : (\alpha, \beta) \to \mathbb{R}^d$ la soluzione *massimale* dell'IVP. Se $\beta < +\infty$ allora il grafico $(t, y(t))$ esce definitivamente da ogni compatto contenuto in D per $t \uparrow \beta$.

Equivalentemente: se il grafico resta in un compatto $K \subset D$, la soluzione può essere prolungata oltre β.

L'estinzione in tempo finito può quindi avvenire per due motivi: la soluzione diventa illimitata, oppure il suo grafico si avvicina al bordo del dominio. L'esempio classico del primo caso è

$$y' = y^2, \qquad y(0) = 1 \qquad \Longrightarrow \qquad y(t) = \frac{1}{1 - t}$$

che è di classe C^∞ e localmente lipschitziana ovunque, ma esplode per $t \to 1^-$: nessuna ipotesi di regolarità *locale* può garantire esistenza globale.

Esistenza globale

Se f è globalmente lipschitziana rispetto a y su tutta la striscia $[a, b] \times \mathbb{R}^d$, allora la soluzione esiste su tutto $[a, b]$. Infatti dalla 27.7 applicata con $y \equiv$ soluzione e dal confronto con la soluzione nulla si ricava una limitazione a priori di $\|y(t)\|$ su ogni intervallo limitato, che per il Teorema 27.2 impedisce la terminazione in tempo finito. È questo il caso, in particolare, dei **sistemi lineari** $y' = A(t)y + g(t)$ con A e g continue.

In sintesi

- Ogni ODE di ordine n si riduce a un **sistema del primo ordine**: basta quindi studiare $y' = f(t, y)$.
- **Peano** dà esistenza con la sola continuità di f, ma non unicità: l'esempio $y' = \sqrt{|y|}$ lo mostra.
- **Picard-Lindelöf** aggiunge la condizione di Lipschitz e ottiene esistenza e unicità locali; la dimostrazione è costruttiva (iterate di Picard).
- Il **lemma di Grönwall** è lo strumento che trasforma una disuguaglianza integrale in una stima esponenziale: da esso discendono sia l'unicità sia la dipendenza continua $\|y - w\| \le e^{L|t-t_0|}\|y_0 - w_0\|$.

- Il risultato è **locale**: il Teorema 27.2 dice che una soluzione cessa di esistere solo uscendo da ogni compatto, come in $y' = y^2$.



28. Soluzione numerica

In molti casi, trovare una soluzione analitica di un problema a valore iniziale (IVP) è impraticabile, a causa della complessità degli integrali o delle funzioni coinvolte. Si ricorre quindi a metodi numerici per approssimare la soluzione. Questi metodi si applicano sia ai singoli IVP sia ai sistemi di equazioni differenziali ordinarie (ODE). Poiché le equazioni differenziali di ordine superiore possono essere riformulate come sistemi del primo ordine, i metodi numerici per questi sistemi sono centrali nella risoluzione delle ODE.

28.1 Passo Temporale

Sia $I = [t_0, t_0 + T]$ un intervallo temporale, che suddividiamo in n sottointervalli, detti anche passi temporali. Ogni sottointervallo $I_k = [t_{k-1}, t_k]$ ha ampiezza $h_k = t_k - t_{k-1}$, con $t_n = t_0 + T$. Se i passi sono uniformi, $h_k = h$ per ogni k. La scelta di h è cruciale, poiché influisce sulla precisione e sulla stabilità del metodo numerico.

28.2 Principali metodi numerici per IVP

28.2.1 Metodi a Un Passo

Nei metodi a un passo, la soluzione al tempo t_{k+1} dipende soltanto da quella calcolata al tempo t_k. In altre parole, l'approssimazione y_{k+1} si basa unicamente sull'informazione relativa a y_k. Il più semplice di questi metodi è quello di Eulero[1].

28.2.2 Metodi a Più Passi

I metodi a più passi utilizzano le informazioni di diversi passi precedenti per calcolare la nuova approssimazione y_{k+1}. Sono più complessi e spesso più accurati dei metodi a un passo, poiché tengono conto di una parte più ampia della storia della soluzione. Un esempio noto è il metodo di Adams-Bashforth.

28.2.3 Metodi Espliciti

Un metodo esplicito (o di integrazione in avanti) calcola la soluzione al passo successivo usando soltanto valori già noti. A partire dalla condizione iniziale y_0 al tempo t_0, le approssimazioni successive sono determinate da una relazione del tipo:

$$y_{k+1} = y_k + h_k F(t_k, y_k),$$

[1] Leonhard Euler (1707-1783)

dove F rappresenta l'incremento della soluzione al passo k. I metodi espliciti sono meno costosi dal punto di vista computazionale e semplici da implementare, ma possono richiedere passi temporali piccoli per garantire la stabilità della soluzione.

28.2.4 Metodi Impliciti

Nei metodi impliciti (o di integrazione all'indietro), invece, la funzione F dipende dal valore della soluzione al tempo t_{k+1}. A ogni passo occorre quindi risolvere un'equazione per determinare y_{k+1}:

$$y_{k+1} = y_k + h_k F(t_{k+1}, y_{k+1}).$$

Questi metodi hanno un costo computazionale maggiore, poiché richiedono la risoluzione di sistemi di equazioni, spesso in modo iterativo; possono però offrire una maggiore stabilità, soprattutto per i problemi rigidi. Un esempio è il metodo di Eulero implicito.

28.3 Il Metodo di Eulero

Il metodo di Eulero è uno dei metodi numerici più semplici per la risoluzione degli IVP. Si tratta di un metodo a un passo che approssima la soluzione tramite una forma semplificata dell'espansione in serie di Taylor[2]. Consideriamo un problema di valore iniziale:

$$\begin{cases} y'(t) = f(t, y), \\ y(t_0) = y_0. \end{cases}$$

L'espansione di Taylor di $y(t)$ attorno al punto iniziale t_0 è:

$$y(t_0 + h) = y_0 + hy'(t_0) + O(h^2),$$

dove

$$y'(t_0) = f(t_0, y_0).$$

28.3.1 Metodo di Eulero Esplicito

Trascurando i termini di ordine superiore, otteniamo la formula del metodo di Eulero esplicito:

$$y_{k+1} = y_k + hf(t_k, y_k).$$

28.3.2 Metodo di Eulero Implicito

Nel metodo di Eulero implicito, si usa un'analisi simile, ma si considera l'approssimazione alla derivata in t_{k+1} anziché in t_k. L'equazione risultante è:

$$y_{k+1} = y_k + hf(t_{k+1}, y_{k+1}),$$

che richiede la risoluzione implicita di y_{k+1} a ogni passo, rendendo il metodo più stabile ma computazionalmente più costoso.

28.4 Convergenza Metodo di Eulero

28.4.1 Convergenza Metodo di Eulero Esplicito

Consideriamo la soluzione esatta $y(t_k)$ al punto t_k e introduciamo l'errore

$$\epsilon_k = y(t_k) - y_k,$$

[2] Brook Taylor (1685-1731)

che rappresenta la differenza tra la soluzione esatta e quella approssimata al passo k. Per analizzare l'errore del metodo di Eulero esplicito, confrontiamo l'espansione di Taylor della soluzione esatta con il passo di avanzamento del metodo.

28.4.1.1 Taylor e metodo di Eulero

L'espansione di Taylor della soluzione esatta $y(t)$ attorno a t_k è:

$$y(t_{k+1}) = y(t_k + h) = y(t_k) + y'(t_k)h + \frac{y''(\zeta_k)}{2}h^2,$$

dove $\zeta_k \in [t_k, t_{k+1}]$ è un punto intermedio.
Il metodo di Eulero esplicito approssima la soluzione al passo successivo come:

$$y_{k+1} = y_k + hf(t_k, y_k).$$

Sottraendo l'espressione di y_{k+1} dall'espansione di Taylor di $y(t_{k+1})$, otteniamo:

$$\begin{aligned} y(t_{k+1}) - y_{k+1} = y(t_k) - y_k \\ + h\big(f(t_k, y(t_k)) - f(t_k, y_k)\big) \\ + \frac{y''(\zeta_k)}{2}h^2. \end{aligned}$$

Questo può essere riscritto come:

$$\epsilon_{k+1} = \epsilon_k + h\big(f(t_k, y(t_k)) - f(t_k, y_k)\big) + \frac{y''(\zeta_k)}{2}h^2.$$

28.4.1.2 Stima dell'Errore

Assumendo che $f(t, y)$ sia lipschitziana rispetto a y, ovvero:

$$|f(t_k, y(t_k)) - f(t_k, y_k)| \leq L|y(t_k) - y_k| = L|\epsilon_k|,$$

e che esista una costante M_2 tale che

$$|y''(t)| \leq M_2 \quad \text{per ogni } t \in [a, b],$$

possiamo stimare l'errore al passo successivo come:

$$|\epsilon_{k+1}| \leq (1 + hL)|\epsilon_k| + \frac{M_2}{2}h^2. \tag{28.1}$$

usando questa diseguaglianza possiamo dimostrare la convergenza del metodo di Eulero.

Lemma 28.1 Sia

$$E_h = \max_{k=0}^{n} |\epsilon_k|$$

l'errore massimo e

$$h = \frac{b - a}{n}$$

il passo temporale. Per il metodo di Eulero esplicito si ha la seguente stima:

$$E_h \leq Ch, \qquad C = \frac{2Le^{(b-a)}}{LM_2}$$

dove L è la costante di Lipschitz[a] di $f(t, y)$ rispetto a y, e M_2 è un limite superiore per il modulo della derivata seconda della soluzione.

[a] Rudolph Otto Sigismund Lipschitz (1832-1903)

Proof. Ponendo $A = 1 + hL$ e $B = M_2 \frac{h^2}{2}$ nella stima dell'errore data in 28.1, otteniamo che l'errore si propaga con la seguente limitazione:

$$|\varepsilon_{k+1}| \leq A|\varepsilon_k| + B,$$

dove $\varepsilon_0 = y(t_0) - y_0 = 0$. Utilizzando la ricorrenza, possiamo scrivere:

$$\begin{aligned}
|\varepsilon_1| &\leq A|\varepsilon_0| + B = B, \\
|\varepsilon_2| &\leq A|\varepsilon_1| + B = AB + B = (A+1)B, \\
|\varepsilon_3| &\leq A|\varepsilon_2| + B = A(A+1)B + B = (A^2 + A + 1)B, \\
|\varepsilon_4| &\leq A|\varepsilon_3| + B = A(A^2 + A + 1)B + B = (A^3 + A^2 + A + 1)B, \\
&\vdots \\
|\varepsilon_{k+1}| &\leq (A^k + A^{k-1} + \cdots + A^2 + A + 1)B.
\end{aligned}$$

La sommatoria $A^k + A^{k-1} + \cdots + A^2 + A + 1$ è una serie geometrica, che può essere espressa come:

$$A^k + A^{k-1} + \cdots + A^2 + A + 1 = \frac{A^{k+1} - 1}{A - 1}.$$

Di conseguenza, abbiamo:

$$E_h = \max |\varepsilon_k| \leq \frac{A^n - 1}{A - 1} B \leq \frac{A^n}{A - 1} B.$$

Sostituendo $A = 1 + hL$ e $B = M_2 \frac{h^2}{2}$, otteniamo:

$$E_h \leq \frac{A^n}{A - 1} B = \frac{(1 + hL)^n}{hL} M_2 \frac{h^2}{2}.$$

Utilizzando la disuguaglianza valida per $z \geq 0$:

$$1 + z \leq e^z, \tag{28.2}$$

risultante dalla serie di Taylor, otteniamo:

$$E_h \leq \frac{(e^{hL})^n}{2L} M_2 h = \frac{e^{nhL}}{2L} M_2 h = \left(\frac{e^{(b-a)L}}{2L} M_2 \right) h.$$

■

28.4.2 Convergenza Metodo di Eulero Implicito

Analogamente al metodo esplicito, consideriamo la soluzione esatta $y(t_k)$ al punto t_k e l'errore $\epsilon_k = y(t_k) - y_k$.

28.4.2.1 Taylor e metodo di Eulero Implicito

L'espansione di Taylor della soluzione esatta $y(t)$ attorno a t_{k+1} è:

$$y(t_{k+1}) = y(t_k) + y'(t_{k+1})h - \frac{y''(\zeta_k)}{2}h^2,$$

dove $\zeta_k \in [t_k, t_{k+1}]$.
Nel metodo di Eulero implicito, l'approssimazione al passo successivo si basa su:

$$y_{k+1} = y_k + hf(t_{k+1}, y_{k+1}),$$

dove la funzione $f(t, y)$ è valutata al passo successivo (t_{k+1}, y_{k+1}).
Sottraendo l'espressione di y_{k+1} dalla serie di Taylor, otteniamo:

$$\begin{aligned} y(t_{k+1}) - y_{k+1} = y(t_k) - y_k \\ + h\big(f(t_{k+1}, y(t_{k+1})) - f(t_{k+1}, y_{k+1})\big) \\ - \frac{y''(\zeta_k)}{2}h^2, \end{aligned}$$

che può essere riscritta come:

$$\epsilon_{k+1} = \epsilon_k + h\big(f(t_{k+1}, y(t_{k+1})) - f(t_{k+1}, y_{k+1})\big) - \frac{y''(\zeta_k)}{2}h^2.$$

28.4.2.2 Stima dell'Errore

Assumiamo che $f(t, y)$ sia lipschitziana rispetto a y, cioè:

$$|f(t_{k+1}, y(t_{k+1})) - f(t_{k+1}, y_{k+1})| \leq L|y(t_{k+1}) - y_{k+1}| = L|\epsilon_{k+1}|,$$

e che esista una costante M_2 tale che $|y''(t)| \leq M_2$ per ogni $t \in [a, b]$.
Otteniamo quindi la seguente stima per l'errore al passo successivo:

$$|\epsilon_{k+1}| \leq |\epsilon_k| + hL|\epsilon_{k+1}| + \frac{M_2}{2}h^2.$$

Riorganizzando i termini, otteniamo:

$$|\epsilon_{k+1}| \leq \frac{|\epsilon_k| + \frac{M_2}{2}h^2}{1 - hL}.$$

Da questa disuguaglianza possiamo dimostrare la convergenza del metodo di Eulero implicito.

Lemma 28.2 — Convergenza di Eulero implicito. Sia

$$E_h = \max_{k=0}^{n} |\epsilon_k|$$

l'errore massimo e sia

$$h = \frac{b - a}{n}$$

il passo temporale. Per il metodo di Eulero implicito si ha la seguente stima:

$$E_h \leq Ch, \qquad C = \frac{e^{2(b-a)L}}{L}M_2,$$

dove L è la costante di Lipschitz di $f(t, y)$ rispetto a y e M_2 è una stima per il massimo modulo della derivata seconda della soluzione.

Proof. Riprendiamo la stima dell'errore per $|\epsilon_{k+1}|$:

$$|\epsilon_{k+1}| \leq \frac{|\epsilon_k| + \frac{M_2}{2}h^2}{1 - hL}.$$

Poniamo

$$A = \frac{1}{1 - hL} \quad \text{e} \quad B = A\frac{M_2}{2}h^2$$

in modo da riscrivere la relazione come:

$$|\epsilon_{k+1}| \leq A|\epsilon_k| + B.$$

Facendo lo stesso schema del Lemma sulla convergenza del metodo di Eulero esplicito, otteniamo:

$$E_h \leq \frac{A^{n+1} - 1}{A - 1}B.$$

Assumendo h sufficientemente piccolo affinché $Lh \leq 1/2$, possiamo scrivere:

$$\begin{aligned} A = \frac{1}{1 - hL} &= \sum_{k=0}^{\infty}(hL)^k \\ &= 1 + hL\sum_{k=1}^{\infty}(hL)^k \leq 1 + 2hL, \\ A - 1 = \frac{1}{1 - hL} - 1 &\geq hL, \end{aligned}$$

e quindi, usando anche 28.2:

$$\frac{A^n - 1}{A - 1} \leq \frac{(1 + 2hL)^n - 1}{hL} \leq \frac{e^{2nhL} - 1}{hL} \leq \frac{e^{2(b-a)L}}{hL}.$$

Poiché $A \leq 2$ per $hL \leq 1/2$, si ha

$$B = A\frac{M_2}{2}h^2 \leq M_2h^2,$$

e unendo le disuguaglianze, otteniamo la stima cercata. □ ■

28.4.3 Generalizzazione

Il metodo per dimostrare la convergenza di schemi numerici, come quelli di Eulero esplicito e implicito, può essere esteso a metodi generici sotto certe ipotesi. In questa sezione, vediamo come formalizzare un metodo numerico generico (sia esplicito che implicito) e dimostrare la sua convergenza.

Consideriamo l'equazione differenziale ordinaria (ODE):

$$\begin{cases} y'(t) = f(t, y), \\ y(a) = a. \end{cases}$$

Un **metodo esplicito a un passo** può essere scritto genericamente come:

$$y_0 = a, \qquad y_{k+1} = \Phi(t_k, y_k; h)$$

dove $\Phi(t, y; h)$ è la funzione che rappresenta il passo di avanzamento. Ad esempio, nel caso di Eulero esplicito:

$$\Phi(t, y; h) = y + hf(t, y)$$

28.4.4 Funzione di avanzamento per altri schemi

Vediamo ora come scrivere la funzione di avanzamento $\Phi(t, y; h)$ per altri schemi numerici.

28.4.4.1 Metodo di Collatz

Il metodo di Collatz può essere descritto come segue:

$$\begin{aligned} y_{k+1/2} &= y_k + \frac{h}{2} f(t_k, y_k) \\ y_{k+1} &= y_k + hf\left(t_k + \frac{h}{2},\ y_{k+1/2}\right) \end{aligned}$$

Questo schema può essere riarrangiato in una forma compatibile con $\Phi(t, y; h)$:

$$y_{k+1} = y_k + hf\left(t_k + \frac{h}{2},\ y_k + \frac{h}{2} f(t_k, y_k)\right)$$

per cui la funzione di avanzamento per Collatz diventa

$$\Phi(t, y; h) = y + hf\left(t + \frac{h}{2},\ y + \frac{h}{2} f(t, y)\right)$$

28.4.4.2 Metodo di Heun

Il metodo di Heun è definito da:

$$\begin{aligned} \widetilde{y}_{k+1} &= y_k + hf(t_k, y_k) \\ y_{k+1} &= y_k + \frac{h}{2}\left(f(t_k, y_k) + f(t_{k+1}, \widetilde{y}_{k+1})\right) \end{aligned}$$

Riarrangiando anche questo schema, otteniamo:

$$y_{k+1} = y_k + \frac{h}{2}\Big(f(t_k, y_k) + f\big(t_k + h, y_k + hf(t_k, y_k)\big)\Big)$$

per cui la funzione di avanzamento per Heun diventa

$$\Phi(t, y; h) = y + \frac{h}{2}\Big(f(t, y) + f\big(t + h, y + hf(t, y)\big)\Big)$$

28.4.5 Propagazione dell'errore

Sostituendo la soluzione esatta nella funzione di avanzamento, otteniamo:

$$y(t + h) = \Phi(t, y(t)) + \tau(t, h)$$

dove $\tau(t, h)$ è l'errore locale di troncamento. Ora calcoliamo l'errore di troncamento per alcuni metodi.

28.4.5.1 Errore di Troncamento per Eulero

Per Eulero esplicito, abbiamo:

$$\begin{aligned}\tau(t,h) &= y(t+h) - \Phi(t, y(t), h)\\ &= y(t+h) - y(t) - hf(t, y(t))\\ &= \cancel{y(t)} + \cancel{y'(t)h} + y''(\zeta)h^2/2 - \cancel{y(t)} - \cancel{hy'(t)}\\ &= \frac{h^2}{2}y''(\zeta)\end{aligned}$$

dove $\zeta \in [t, t+h]$.

28.4.5.2 Errore di Troncamento per Collatz

Per il metodo di Collatz:

$$\begin{aligned}\tau(t,h) &= y(t+h) - \Phi(t, y(t), h)\\ &= y(t+h) - y(t) - hf\left(t + \frac{h}{2},\ y(t) + \frac{h}{2}f(t, y(t))\right)\\ &= \cancel{y(t)} + y'(t)h + y''(t)h^2/2 + y'''(\zeta)h^3/6\\ &\quad - \cancel{y(t)} - h\left(f(t, y(t)) + \frac{h}{2}\frac{\partial f(t, y(t))}{\partial t} + \frac{h}{2}\frac{\partial f(t, y(t))}{\partial t}f(t, y(t)) + \mathcal{O}(h^2)\right)\\ &= \cancel{f(t, y(t))h} + \frac{h^2}{2}\frac{\mathrm{d}}{\mathrm{d}t}f(t, y(t)) + y'''(\zeta)h^3/6\\ &\quad \cancel{-hf(t, y(t))} - \frac{h^2}{2}\frac{\partial f(t, y(t))}{\partial t} - \frac{h^2}{2}\frac{\partial f(t, y(t))}{\partial t}f(t, y(t)) + \mathcal{O}(h^3)\\ &= \frac{h^2}{2}\cancel{\left(\frac{\partial f(t, y(t))}{\partial t} + \frac{\partial f(t, y(t))}{\partial t}f(t, y(t))\right)} + y'''(\zeta)h^3/6\\ &\quad - \frac{h^2}{2}\cancel{\left(\frac{\partial f(t, y(t))}{\partial t} + \frac{\partial f(t, y(t))}{\partial t}f(t, y(t))\right)} + \mathcal{O}(h^3)\\ &= \mathcal{O}(h^3)\end{aligned}$$

Analogamente, si può calcolare l'errore per altri metodi, ad esempio Heun, che risulta essere $\mathcal{O}(h^2)$. In generale, se la soluzione esatta non esplode e la funzione $f(t, y)$ è sufficientemente regolare, l'errore locale di troncamento soddisferà:

$$|\tau(t,h)| \ \leq\ C\,h^{p+1} \tag{28.3}$$

ove $p \geq 1$ dipende dal metodo utilizzato.

28.4.6 Convergenza dei metodi a un passo

I conti precedenti si ripetono quasi identici per ogni metodo a un passo. Conviene allora enunciarli una volta sola, in forma generale, e ricavarne una stima **esplicita** dell'errore globale.

Scriviamo un generico metodo a un passo nella forma *incrementale*

$$y_{k+1} = y_k + h\,\Psi(t_k, y_k; h) \tag{28.4}$$

dove Ψ è detta *funzione di incremento*. Ad esempio $\Psi = f(t_k, y_k)$ per Eulero esplicito e $\Psi = f\left(t_k + \frac{h}{2},\ y_k + \frac{h}{2}f(t_k, y_k)\right)$ per Collatz.

Definition 28.1 — Difetto locale e ordine. Detta $y(t)$ la soluzione esatta, il *difetto locale* è

$$d_{k+1} = y(t_{k+1}) - y(t_k) - h\,\Psi\big(t_k, y(t_k); h\big)$$

Il metodo ha **ordine** p se, su ogni intervallo limitato,

$$|d_{k+1}| \le C\,h^{p+1}$$

ed è **consistente** se $|d_{k+1}|/h \to 0$ per $h \to 0$.

Si noti la relazione con l'errore locale di troncamento usato sopra: $d_{k+1} = \tau(t_k, h)$. L'*errore globale* è invece

$$\epsilon_k = y(t_k) - y_k$$

La distinzione è essenziale: un difetto $\mathcal{O}(h^{p+1})$ viene prodotto ad ognuno dei circa T/h passi, quindi ci si attende un errore globale $\mathcal{O}(h^p)$ **purché** le perturbazioni non vengano amplificate troppo. Il teorema seguente rende preciso questo "purché".

Theorem 28.1 — Convergenza dei metodi a un passo. Si supponga che, per $0 < h \le h_0$, la funzione di incremento sia uniformemente lipschitziana rispetto alla seconda variabile,

$$\big|\Psi(t,u;h) - \Psi(t,v;h)\big| \le L_\Psi\,|u - v| \tag{28.5}$$

e che il difetto locale soddisfi $|d_{k+1}| \le Ch^{p+1}$. Allora, su un intervallo fissato $[t_0, t_0 + T]$, l'errore globale soddisfa

$$\boxed{|\epsilon_k| \le \frac{C}{L_\Psi}\Big(e^{L_\Psi (t_k - t_0)} - 1\Big)h^p} \tag{28.6}$$

quando $L_\Psi > 0$, e $|\epsilon_k| \le C\,T\,h^p$ nel caso limite $L_\Psi = 0$. In particolare il metodo converge con ordine p.

Proof. Sottraendo il passo numerico 28.4 dal passo esatto si ottiene

$$\epsilon_{k+1} = \epsilon_k + h\Big[\Psi\big(t_k, y(t_k); h\big) - \Psi\big(t_k, y_k; h\big)\Big] + d_{k+1}$$

e quindi, per la 28.5,

$$|\epsilon_{k+1}| \le (1 + hL_\Psi)\,|\epsilon_k| + Ch^{p+1} \tag{28.7}$$

Poiché $\epsilon_0 = 0$, iterando la 28.7 si ricava

$$|\epsilon_k| \le Ch^{p+1} \sum_{j=0}^{k-1} (1 + hL_\Psi)^j$$

Per $L_\Psi > 0$ la somma geometrica vale $\big[(1 + hL_\Psi)^k - 1\big]/(hL_\Psi)$, da cui

$$\begin{aligned} |\epsilon_k| &\le \frac{C}{L_\Psi}\Big[(1 + hL_\Psi)^k - 1\Big]h^p \\ &\le \frac{C}{L_\Psi}\Big(e^{L_\Psi k h} - 1\Big)h^p = \frac{C}{L_\Psi}\Big(e^{L_\Psi (t_k - t_0)} - 1\Big)h^p \end{aligned}$$

dove si è usato $1 + \xi \le e^\xi$ per $\xi \ge 0$. Se $L_\Psi = 0$ la somma vale k e si ottiene direttamente $|\epsilon_k| \le Ckh^{p+1} \le CTh^p$. □ ∎

Il teorema è il meccanismo elementare dietro la regola

$$\text{errore locale } \mathcal{O}(h^{p+1}) \quad \Longrightarrow \quad \text{errore globale } \mathcal{O}(h^p)$$

e mostra anche **a quale prezzo**: la costante cresce come $e^{L_\Psi T}$, quindi la stima è utile su intervalli finiti e si degrada esponenzialmente al crescere di T. Si confronti con la 27.7: è la stessa struttura, ed è la stessa ragione.

Da dove viene L_Ψ

Per Eulero esplicito $\Psi = f$, quindi $L_\Psi = L$, la costante di Lipschitz di f. Per un metodo a più stadi Ψ è una composizione di valutazioni di f e si ottiene $L_\Psi = L(1 + \mathcal{O}(hL))$: la lipschitzianità di f è dunque l'**unica** ipotesi sostanziale, la stessa che garantisce l'esistenza e l'unicità della soluzione (Teorema 27.1). Senza di essa non c'è né soluzione unica da approssimare né convergenza da dimostrare.

Quello che il teorema non dice

La 28.6 vale per $h \to 0$ su un intervallo **fissato**. Non dice nulla sul comportamento per $k \to \infty$ a passo h fisso: quello è l'oggetto della stabilità assoluta (Capitolo 31), ed è una proprietà del tutto indipendente dall'ordine p.

28.4.7 Generalizzazione (metodi impliciti)

Un **metodo implicito a un passo** può essere scritto genericamente come:

$$y_0 = a, \qquad y_{k+1} = \Phi(t_k, y_k, y_{k+1}; h)$$

dove $\Phi(t, y, z; h)$ è la funzione di avanzamento che coinvolge sia il punto noto y_k che il punto incognito y_{k+1}. Ad esempio, per il metodo di **Eulero implicito**, la funzione di avanzamento assume la forma:

$$\Phi(t, y, z; h) = z + hf(t, z)$$

dove $z = y_{k+1}$, ovvero il nuovo valore che stiamo cercando.

28.4.7.1 Errore Locale di Troncamento

Per definire la precisione del metodo, consideriamo l'errore locale di troncamento. Questo rappresenta la differenza tra la soluzione esatta al passo successivo $y(t + h)$ e la previsione del metodo numerico. L'errore di troncamento per un metodo implicito è dato da:

$$\tau(t, h) = y(t + h) - \Phi(t, y(t), y(t + h); h)$$

28.4.7.2 Errore di troncamento di Eulero implicito

Per il metodo di **Eulero implicito**, l'errore locale di troncamento si calcola come segue:

$$\begin{aligned}
\tau(t, h) &= y(t + h) - \Phi(t, y(t), y(t + h), h) \\
&= y(t + h) - y(t) - hf(t + h, y(t + h)) \\
&= y(t + h) - y(t) - hy'(t + h) \\
&= \cancel{y(t + h)} - (\cancel{y(t + h)} - \cancel{hy'(t + h)} + h^2/2y''(\zeta)) - \cancel{hy'(t + h)} \\
&= -\frac{h^2}{2} y''(\zeta)
\end{aligned}$$

dove $\zeta \in [t, t+h]$. Notiamo che l'errore locale di troncamento per Eulero implicito è:

$$\tau(t,h) = -\frac{h^2}{2} y''(\zeta)$$

che è dello stesso ordine dell'errore per il metodo di Eulero esplicito, ovvero $\mathcal{O}(h^2)$.

28.4.7.3 Metodo di Crank-Nicolson

Il metodo di **Crank-Nicolson**[3], basato sulla regola dei trapezi, è un metodo implicito che può essere scritto come:

$$y_{k+1} = y_k + \frac{h}{2}\Big(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\Big)$$

per cui la funzione di avanzamento prende la forma

$$\Phi(t, y, z; h) = y + \frac{h}{2}\Big(f(t, y) + f(t+h, z)\Big)$$

L'errore locale di troncamento per questo metodo si calcola come:

$$\begin{aligned}\tau(t,h) &= y(t+h) - \Phi(t, y(t), y(t+h), h) \\ &= y(t+h) - y(t) - \frac{h}{2}\Big(f(t, y(t)) + f(t+h, y(t+h))\Big) \\ &= \mathcal{O}(h^3)\end{aligned}$$

che si ottiene dopo una lunga espansione di Taylor in più variabili e che qui è omessa. L'errore di troncamento è di ordine $\mathcal{O}(h^3)$, più preciso rispetto a Eulero implicito.

28.4.8 Dimostrazione della convergenza

Analogamente al caso esplicito, possiamo dimostrare la convergenza dei metodi impliciti utilizzando l'errore locale di troncamento e le proprietà di Lipschitz delle funzioni di avanzamento.

28.4.8.1 Errore locale di troncamento

Per i metodi impliciti, come nel caso di Eulero esplicito, l'errore locale di troncamento $\tau(h)$ è della forma:

$$|\tau(h)| \leq Ch^{p+1},$$

dove p dipende dal metodo utilizzato. Per Eulero implicito abbiamo $p = 1$, mentre per Crank-Nicolson $p = 2$.

28.4.8.2 Soluzione esatta e soluzione numerica

Consideriamo l'equazione che lega la soluzione esatta $y(t_{k+1})$ alla soluzione numerica y_{k+1}, includendo l'errore locale di troncamento:

$$y(t_{k+1}) = \Phi(t_k, y(t_k), y(t_{k+1}); h) + \tau_k(h).$$

Sottraendo l'equazione che descrive il metodo numerico:

$$y_{k+1} = \Phi(t_k, y_k, y_{k+1}; h),$$

otteniamo:

$$y(t_{k+1}) - y_{k+1} = \Phi(t_k, y(t_k), y(t_{k+1}); h) - \Phi(t_k, y_k, y_{k+1}; h) + \tau_k(h).$$

[3] John Crank (1916-2006) e Phyllis Nicolson (1917-1968)

28.4.8.3 Uso della condizione di Lipschitz

Supponiamo che la funzione $\Phi(t, y, z; h)$ sia Lipschitziana rispetto a y e z con costanti L_y e L_z. Applicando la condizione di Lipschitz:

$$|\Phi(t, y, z; h) - \Phi(t, y', z'; h)| \leq L_y|y - y'| + L_z|z - z'|,$$

otteniamo una relazione dell'errore ricorsivo simile a quella del metodo esplicito:

$$|\epsilon_{k+1}| \leq (1 + hL_z)|\epsilon_k| + Ch^{p+1}.$$

28.4.8.4 Conclusione

Ripetendo il ragionamento induttivo utilizzato per Eulero esplicito, possiamo dedurre che, sotto opportune ipotesi di regolarità su $f(t, y)$ e con una condizione di Lipschitz, l'errore globale soddisfa:

$$E_h \leq C'h^p,$$

dove p dipende dal metodo scelto. Ad esempio, per Eulero implicito abbiamo $p = 1$, mentre per Crank-Nicolson $p = 2$.

Abbiamo mostrato che la dimostrazione di convergenza per i metodi impliciti può seguire un approccio simile a quello utilizzato per i metodi espliciti. La chiave sta nell'analisi dell'errore locale di troncamento e nell'uso della condizione di Lipschitz sulla funzione di avanzamento. I metodi impliciti, come Eulero implicito e Crank-Nicolson, presentano vantaggi in termini di stabilità e precisione, soprattutto per sistemi **stiff**, sebbene richiedano la risoluzione di equazioni non lineari a ogni passo.

28.5 Verifica Numerica dell'Ordine

Quando si sviluppa o implementa un metodo numerico, è fondamentale verificarne il corretto comportamento rispetto all'ordine teorico di convergenza. Questo controllo garantisce che il metodo funzioni correttamente e rispetti le aspettative teoriche.

28.5.1 Analisi Teorica

Un metodo numerico si dice di ordine p se l'errore massimo

$$E_h = \max_{k=0}^{n} |y_k - y(t_k)|$$

al variare del passo h segue la relazione:

$$E_h \approx Ch^p. \tag{28.8}$$

dove:

- y_k è la soluzione numerica al passo t_k
- $y(t_k)$ è la soluzione esatta nello stesso punto,
- C è una costante che dipende dal problema ma non da h
- h è il passo del di avanzamento metodo.

28.5.1.1 Stima dell'Ordine Numerico

Utilizzando soluzioni calcolate con passi h_1 e h_2, e indicando i rispettivi errori come E_{h_1} e E_{h_2}, possiamo calcolare il rapporto:

$$\frac{E_{h_1}}{E_{h_2}} \approx \frac{Ch_1^p}{Ch_2^p} = \left(\frac{h_1}{h_2}\right)^p$$

Prendendo il logaritmo si ottiene una relazione lineare tra il rapporto degli errori e quello dei passi:

$$\log \frac{E_{h_1}}{E_{h_2}} \approx p \log \frac{h_1}{h_2},$$

Questa formula permette di stimare l'ordine p:

$$\boxed{p \approx \frac{\log \frac{E_{h_1}}{E_{h_2}}}{\log \frac{h_1}{h_2}},} \tag{28.9}$$

28.5.1.2 Verifica Grafica

Riscrivendo E_h della 28.8 in forma logaritmica:

$$\log E_h \approx \log \left(Ch^p\right) = \log C + p \log h$$

si nota che rappresentando graficamente $\log h$ (asse t) e $\log E_h$ (asse y) si ottiene una retta con:

- intercetta pari a $\log C$
- La pendenza stimata p rappresenta l'ordine numerico del metodo. Se il valore coincide con l'ordine teorico, il metodo è implementato correttamente.

Una buona linearità tra $\log h$ e $\log E_h$ conferma che il comportamento del metodo è coerente con la stima della convergenza.

28.5.1.3 Esempio numerico

Per verificare numericamente l'ordine di un metodo, è utile calcolare gli errori per una serie di valori di h e rappresentarli graficamente.

ODE con soluzione esatta

```r
# Funzione rhs della ODE (y' = f(t, y))
f <- function(t, y) { -2 * y + t^2 }

# Soluzione esatta del problema per confronto
exact_solution <- function(t,y0) {
  t*(t-1)/2 + 1/4 + exp(-2*t)*(y0 - 1/4)
}
```

metodo di Eulero

```r
# Implementazione del metodo di Eulero
euler_method <- function(f, t0, y0, h, t_end) {
  t <- seq(t0, t_end, by = h)
  n <- length(t)
  y <- numeric(n)
  y[1] <- y0
  for (i in 1:(n - 1)) {
    y[i + 1] <- y[i] + h * f(t[i], y[i])
  }
  data.frame(t = t, y = y)
}
```

calcolo la soluzione per vari valori di h

```r
# Parametri iniziali
t0 <- 0
```

```
y0 <- 0.25 # Soluzione iniziale esatta: y(0) = 1/4
t_end <- 1
h_values <- c(0.1, 0.05, 0.025, 0.01, 0.005, 0.0025 )

# Calcolo dell'errore massimo per ogni passo
errors <- numeric(length(h_values))

for (i in seq_along(h_values)) {
  h <- h_values[i]
  solution  <- euler_method(f, t0, y0, h, t_end)
  exact     <- exact_solution(solution$t,y0)
  errors[i] <- max(abs(solution$y - exact)) # Errore massimo
}
```

stampo lista degli errori in funzione di h

```
# Tabella degli errori
error_table <- data.frame(h = h_values, Error = errors)
print(error_table)
```

```
       h        Error
1 0.1000 0.0223156454
2 0.0500 0.0109802918
3 0.0250 0.0054467990
4 0.0100 0.0021684511
5 0.0050 0.0010825254
6 0.0025 0.0005408387
```

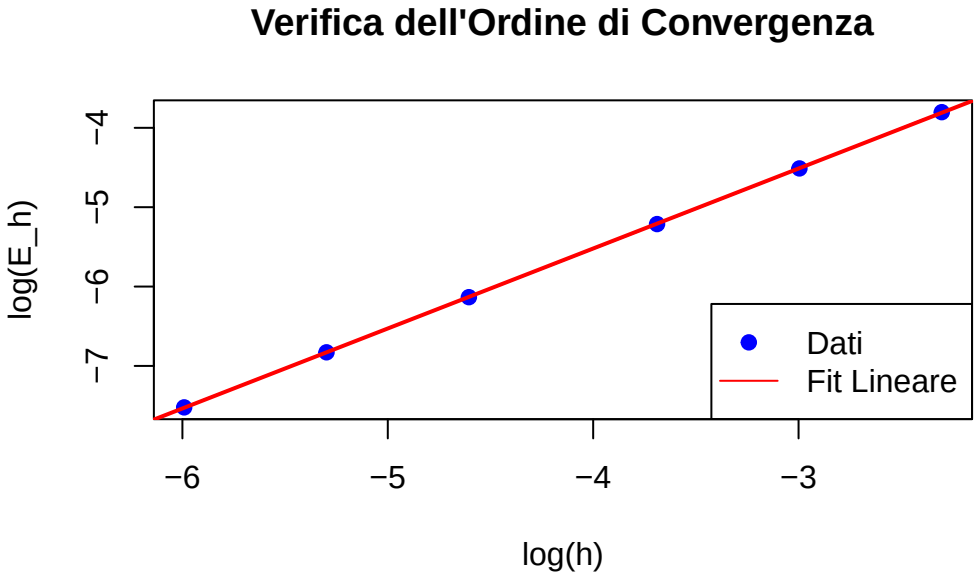
Stimo errore con 28.9

```
# Calcolo dell'ordine di convergenza
log_h <- log(h_values)
log_E <- log(errors)
fit <- lm(log_E ~ log_h) # Regressione lineare
p_estimate <- coef(fit)[2] # Stima dell'ordine
cat("Stima dell'ordine di convergenza p:", p_estimate, "\n")
```

```
Stima dell'ordine di convergenza p: 1.007675
```

errore in log-log plot

```
# Plot dell'errore
plot(
  log_h, log_E,
  type = "b", col = "blue", pch = 19,
  xlab = "log(h)", ylab = "log(E_h)",
  main = "Verifica dell'Ordine di Convergenza"
)
abline(fit, col = "red", lwd = 2)
legend(
  "bottomright",
  legend = c("Dati", "Fit Lineare"),
  col = c("blue", "red"),
  pch = c(19, NA), lty = c(NA, 1)
)
```



In sintesi

- Un metodo a un passo si scrive $y_{k+1} = y_k + h\Psi(t_k, y_k; h)$; tutto il capitolo studia il legame fra il **difetto locale** $\mathcal{O}(h^{p+1})$ e l'**errore globale** $\mathcal{O}(h^p)$.
- Il Teorema 28.1 dà la stima esplicita $\dfrac{C}{L_\Psi}(e^{L_\Psi(t_k - t_0)} - 1)h^p$: la lipschitzianità di f è l'unica ipotesi sostanziale.
- Eulero esplicito ha ordine 1 ed è il caso elementare; Collatz (punto medio) e Heun salgono a ordine 2 con due valutazioni di f.
- Eulero implicito e Crank-Nicolson richiedono di risolvere un'equazione a ogni passo, ma hanno proprietà di stabilità molto migliori (Capitolo 31).
- L'ordine si **verifica** numericamente dal rapporto degli errori a passi dimezzati: $p \approx \log(E_{h_1}/E_{h_2}) / \log(h_1/h_2)$.



29. Metodi di Taylor per ODE

Consideriamo la seguente equazione differenziale ordinaria (ODE):

$$\begin{cases} y' = f(t, y) \\ y(a) = y_a \end{cases} \tag{29.1}$$

Sotto le usuali condizioni di regolarità per $f(t, y)$, possiamo garantire l'esistenza e l'unicità della soluzione.

Per approssimare numericamente la soluzione nell'intervallo $[a, b]$, dividiamo l'intervallo in n parti uguali con ampiezza $h = (b - a)/n$. Cerchiamo di calcolare la soluzione in corrispondenza dei punti:

$$t_k = a + kh, \qquad k = 0, 1, \dots, n.$$

La soluzione esatta $y(t)$ soddisfa:

$$y'(t) = f(t, y(t)), \qquad y(a) = y_a.$$

Possiamo sfruttare lo sviluppo in serie di Taylor[1] per costruire metodi numerici che approssimano la soluzione.

29.1 Metodo di Eulero e Serie di Taylor

Utilizziamo la serie di Taylor per approssimare la soluzione $y(t)$ al primo ordine:

$$y(t_{k+1}) = y(t_k) + y'(t_k)h + \frac{y''(\zeta)}{2}h^2, \qquad \zeta \in (t_k, t_{k+1}),$$

dove $y'(t_k)$ si ottiene direttamente dalla ODE come $y'(t_k) = f(t_k, y(t_k))$. Otteniamo così:

$$y(t_{k+1}) = y(t_k) + f(t_k, y(t_k))h + \frac{y''(\zeta)}{2}h^2.$$

Trascurando il termine d'errore $\mathcal{O}(h^2)$, otteniamo il metodo di **Eulero**[2] **esplicito**:

$$y_{k+1} = y_k + f(t_k, y_k)h,$$

che è un metodo del primo ordine.

[1] Brook Taylor (1685-1731)
[2] Leonhard Euler (1707-1783)

29.2 Metodo di Taylor al Secondo Ordine

Per ottenere una migliore approssimazione, possiamo considerare un'espansione della serie di Taylor fino al secondo ordine:

$$y(t_{k+1}) = y(t_k) + y'(t_k)h + \frac{y''(t_k)}{2}h^2 + \frac{y'''(\zeta)}{6}h^3, \quad \zeta \in (t_k, t_{k+1})$$

Il problema che emerge è il calcolo di $y''(t_k)$, che non è immediatamente disponibile. Tuttavia, possiamo calcolarlo derivando la ODE rispetto a t:

$$y''(t) = \frac{\mathrm{d}}{\mathrm{d}t}y'(t) = \frac{\mathrm{d}}{\mathrm{d}t}f(t, y(t)).$$

Applicando la **chain rule**, otteniamo:

$$y''(t) = \frac{\partial f(t, y(t))}{\partial t} + \frac{\partial f(t, y(t))}{\partial y} \cdot y'(t).$$

Sostituendo $y'(t)$ con $f(t, y(t))$, possiamo esprimere $y''(t)$ come:

$$y''(t) = \frac{\partial f(t, y(t))}{\partial t} + \frac{\partial f(t, y(t))}{\partial y} \cdot f(t, y(t)).$$

Sostituendo questa espressione nello sviluppo di Taylor, otteniamo:

$$\begin{aligned} y(t_{k+1}) = y(t_k) &+ f(t_k, y(t_k))h \\ &+ \frac{h^2}{2}\left(\frac{\partial f(t_k, y(t_k))}{\partial t} + \frac{\partial f(t_k, y(t_k))}{\partial y} \cdot f(t_k, y(t_k))\right) \\ &+ \mathcal{O}(h^3). \end{aligned}$$

Trascurando il termine $\mathcal{O}(h^3)$, otteniamo un **metodo di Taylor al secondo ordine**:

$$y_{k+1} = y_k + f(t_k, y_k)h + \frac{h^2}{2}\left(\frac{\partial f(t_k, y_k)}{\partial t} + \frac{\partial f(t_k, y_k)}{\partial y} \cdot f(t_k, y_k)\right).$$

Remark 29.1 Il calcolo delle derivate parziali di $f(t, y)$ è essenziale per costruire metodi di Taylor di ordine superiore. Tuttavia, per funzioni complesse, il calcolo manuale delle derivate può risultare molto laborioso e soggetto a errori. In questi casi, l'uso del **calcolo simbolico** diventa fondamentale.

Software come **Mathematica**, **Maple**, o librerie di Python come **SymPy** permettono di automatizzare il processo di differenziazione simbolica.

29.3 Metodo di Taylor al Terzo Ordine

Per ottenere una **migliore** approssimazione della soluzione, possiamo estendere la serie di Taylor fino al **terzo** ordine:

$$\begin{aligned} y(t_{k+1}) = y(t_k) + y'(t_k)h + \frac{y''(t_k)}{2}h^2 \\ + \frac{y'''(t_k)}{6}h^3 + \frac{y^{(4)}(\zeta)}{24}h^4, \quad \zeta \in (t_k, t_{k+1}). \end{aligned}$$

Il calcolo di $y''(t_k)$ è già stato fatto nel metodo di Taylor al secondo ordine. Per $y'''(t_k)$, possiamo procedere come segue:

$$
\begin{aligned}
y'''(t) &= \frac{\mathrm{d}}{\mathrm{d}t} y''(t) \\
&= \frac{\mathrm{d}}{\mathrm{d}t} \left(\frac{\partial f(t, y(t))}{\partial t} + \frac{\partial f(t, y(t))}{\partial y} \cdot f(t, y(t)) \right) \\
&= \frac{\partial^2 f(t, y(t))}{\partial t^2} + \frac{\partial^2 f(t, y(t))}{\partial y^2} f(t, y(t))^2 \\
&\quad + \left(\frac{\partial f(t, y(t))}{\partial y} \right)^2 f(t, y(t)) + 2 \frac{\partial^2 f(t, y(t))}{\partial t \partial y} f(t, y(t)) \\
&\quad + \frac{\partial f(t, y(t))}{\partial t} \cdot \frac{\partial f(t, y(t))}{\partial y}.
\end{aligned}
$$

29.3.1 Notazione Compatta

Per semplificare la scrittura e il calcolo, possiamo introdurre alcune scorciatoie:

$$
\begin{aligned}
f_k &= f(t_k, y(t_k)), \\
f_{t,k} &= \frac{\partial f(t_k, y(t_k))}{\partial t}, \\
f_{y,k} &= \frac{\partial f(t_k, y(t_k))}{\partial y}, \\
f_{tt,k} &= \frac{\partial^2 f(t_k, y(t_k))}{\partial t^2}, \\
f_{yy,k} &= \frac{\partial^2 f(t_k, y(t_k))}{\partial y^2}, \\
f_{ty,k} &= \frac{\partial^2 f(t_k, y(t_k))}{\partial t \partial y}.
\end{aligned}
$$

Sostituendo queste espressioni nello sviluppo di Taylor, otteniamo:

$$
\begin{aligned}
y(t_{k+1}) &= y(t_k) + h f_k \\
&\quad + \frac{h^2}{2} \left(f_{t,k} + f_{y,k} f_k \right) \\
&\quad + \frac{h^3}{6} \left(f_{tt,k} + f_{yy,k} f_k^2 + (f_{y,k}^2 + 2 f_{ty,k}) f_k + f_{t,k} f_{y,k} \right) \\
&\quad + \mathcal{O}(h^4).
\end{aligned}
$$

29.3.2 Metodo di Taylor al Terzo Ordine

Trascurando il termine d'errore $\mathcal{O}(h^4)$, otteniamo il **metodo di Taylor al terzo ordine**:

$$
\begin{aligned}
y_{k+1} &= y_k + f(t_k, y_k) h \\
&\quad + \frac{h^2}{2} \left(\frac{\partial f(t_k, y_k)}{\partial t} + \frac{\partial f(t_k, y_k)}{\partial y} \cdot f(t_k, y_k) \right) \\
&\quad + \frac{h^3}{6} \left(f_{tt,k} + f_{yy,k} f_k^2 + (f_{y,k}^2 + 2 f_{ty,k}) f_k + f_{t,k} f_{y,k} \right).
\end{aligned}
$$

29.3.3 Considerazioni sul metodo

> **Nota**
>
> I metodi basati su Taylor richiedono il calcolo di derivate parziali multiple della funzione $f(t, y)$. Questo pone alcune sfide significative:
> - **Derivate simboliche**: Per usare un metodo di ordine superiore, è necessario calcolare simbolicamente (o con un software di algebra computazionale) un numero crescente di derivate parziali.
> - **Crescita della complessità**: La complessità del metodo cresce rapidamente con l'ordine, rendendo l'implementazione e il calcolo più difficili per ordini superiori al terzo.
>
> Tuttavia, se la forma analitica di $f(t, y)$ è nota in anticipo, è possibile costruire un metodo numerico basato sulla serie di Taylor specifico per quella funzione, ottenendo così un'accurata approssimazione della soluzione.

29.4 Metodo basato su Taylor con $f(t, y)$ nota

Sia $y(t)$ la soluzione analitica della seguente equazione differenziale ordinaria (ODE):

$$y'(t) = f(t, y(t)).$$

Consideriamo la serie di Taylor della soluzione:

$$y(t + h) = y(t) + hy'(t) + \frac{h^2}{2}y''(t) + \cdots + \frac{h^p}{p!}y^{(p)}(t) + \mathcal{O}(h^{p+1}), \tag{29.2}$$

dove ogni derivata k-esima può essere espressa in termini della funzione $f(t, y(t))$. A partire da questa relazione, possiamo costruire delle funzioni $D_k(t, y)$ che hanno la proprietà:

$$y^{(k)}(t) = D_k(t, y(t)),$$

ovvero, se valutate sulla soluzione esatta $y(t)$, restituiscono la derivata k-esima della soluzione stessa.

29.4.1 Definizione ricorsiva di $D_k(t, y)$

La costruzione delle funzioni $D_k(t, y)$ è ricorsiva e parte dalla definizione:

$$D_1(t, y) = f(t, y),$$

poiché $y'(t) = f(t, y(t)) = D_1(t, y(t))$.
Se supponiamo che $y^{(k)}(t) = D_k(t, y(t))$, possiamo derivare $y^{(k+1)}(t)$ come segue:

$$\begin{aligned} y^{(k+1)}(t) &= \frac{\mathrm{d}}{\mathrm{d}t}y^{(k)}(t) \\ &= \frac{\mathrm{d}}{\mathrm{d}t}D_k(t, y(t)) \\ &= \frac{\partial D_k(t, y(t))}{\partial t} + \frac{\partial D_k(t, y(t))}{\partial y} \cdot y'(t) \\ &= \frac{\partial D_k(t, y(t))}{\partial t} + \frac{\partial D_k(t, y(t))}{\partial y} \cdot f(t, y(t)). \end{aligned}$$

Da questa relazione, otteniamo la seguente formula ricorsiva per $D_{k+1}(t, y)$:

$$\begin{cases} D_1(t, y) = f(t, y), \\ D_{k+1}(t, y) = \dfrac{\partial D_k(t, y)}{\partial t} + \dfrac{\partial D_k(t, y)}{\partial y} \cdot f(t, y), \quad k = 1, 2, \dots \end{cases}$$

29.4.2 Espansione della soluzione con $D_k(t, y)$

Sostituendo queste funzioni ricorsive nella serie di Taylor, otteniamo l'espansione completa:

$$\begin{aligned} y(t + h) = y(t) + hD_1(t, y(t)) + \frac{h^2}{2} D_2(t, y(t)) \\ + \frac{h^3}{6} D_3(t, y(t)) + \cdots + \frac{h^p}{p!} D_p(t, y(t)) + \mathcal{O}(h^{p+1}). \end{aligned}$$

Da qui possiamo derivare un metodo numerico che approssima la soluzione della ODE fino all'ordine p:

$$\begin{cases} y_{k+1} = y_k + hy'_k + \dfrac{h^2}{2} y''_k + \dfrac{h^3}{6} y'''_k + \cdots + \dfrac{h^p}{p!} y_k^{(p)}, \\ \\ y'_k = D_1(t_k, y_k), \\ y''_k = D_2(t_k, y_k), \\ y'''_k = D_3(t_k, y_k), \\ \quad \vdots \\ y_k^{(p)} = D_p(t_k, y_k). \end{cases}$$

29.4.3 Esempio: $f(t, y) = t + y + ty$

Consideriamo la seguente equazione differenziale ordinaria:

$$\begin{cases} y' = t + y + ty, \\ y(a) = y_a. \end{cases}$$

Per questa ODE, le funzioni $D_k(t, y)$ si calcolano come segue:

$$D_1(t,y) = t + y + ty,$$

$$\begin{aligned} D_2(t,y) &= \frac{\partial D_1(t,y)}{\partial t} + \frac{\partial D_1(t,y)}{\partial y} \cdot D_1(t,y) \\ &= y + 1 + (t+1)(ty + t + y), \end{aligned}$$

$$\begin{aligned} D_3(t,y) &= \frac{\partial D_2(t,y)}{\partial t} + \frac{\partial D_2(t,y)}{\partial y} \cdot D_1(t,y) \\ &= (y+1)t^3 + (3y+2)t^2 + (6y+4)t + 4y + 1, \end{aligned}$$

$$\begin{aligned} D_4(t,y) &= \frac{\partial D_3(t,y)}{\partial t} + \frac{\partial D_3(t,y)}{\partial y} \cdot D_1(t,y) \\ &= (y+1)t^4 + (4y+3)t^3 + (12y+9)t^2 + (16y+8)t + 10y + 4, \end{aligned}$$

$$\begin{aligned} D_5(t,y) &= \frac{\partial D_4(t,y)}{\partial t} + \frac{\partial D_4(t,y)}{\partial y} \cdot D_1(t,y) \\ &= (y+1)t^5 + (5y+4)t^4 + (20y+16)t^3 + (40y+25)t^2 + (50y+28)t + 26y + 8, \end{aligned}$$

$$\begin{aligned} D_6(t,y) &= \frac{\partial D_5(t,y)}{\partial t} + \frac{\partial D_5(t,y)}{\partial y} \cdot D_1(t,y) \\ &= (y+1)t^6 + (6y+5)t^5 + (30y+25)t^4 + (80y+56)t^3 + (150y+98)t^2 + (156y+76)t + 76y + 28. \end{aligned}$$

29.4.3.1 Metodo Numerico di Ordine 6

Utilizzando fino alla derivata sesta, possiamo costruire un metodo numerico di ordine 6 per approssimare la soluzione:

$$\begin{cases} y_{k+1} = y_k + hy'_k + \frac{h^2}{2!}y''_k + \frac{h^3}{3!}y'''_k + \frac{h^4}{4!}y''''_k + \frac{h^5}{5!}y'''''_k + \frac{h^6}{6!}y''''''_k, \\ \\ y'_k = D_1(t_k, y_k), \\ y''_k = D_2(t_k, y_k), \\ y'''_k = D_3(t_k, y_k), \\ y''''_k = D_4(t_k, y_k), \\ y'''''_k = D_5(t_k, y_k), \\ y''''''_k = D_6(t_k, y_k). \end{cases}$$

29.5 Metodo di Taylor migliorato

La costruzione delle funzioni $D_k(t,y)$ richiede la sostituzione di $y'(t) = D_1(t, y(t))$ che poi viene derivato nel calcolo delle funzioni successive. Se si evita questa sostituzione si ottengono delle funzioni ugualmente calcolabili ma in generale più semplici.

Consideriamo la sequenza

$$
\begin{aligned}
y'(t) &= f(t, y(t)) = D_1(t, y(t)) \\
y''(t) &= \frac{\mathrm{d}}{\mathrm{d}t} D_1(t, y(t)) \\
&= \frac{\partial D_1(t, y(t))}{\partial t} + \frac{\partial D_1(t, y(t))}{\partial y} y'(t) \\
&= D_2(t, y(t), y'(t)) \\
y'''(t) &= \frac{\mathrm{d}}{\mathrm{d}t} D_2(t, y(t), y'(t)) \\
&= \frac{\partial D_2(t, y(t), y'(t))}{\partial t} + \frac{\partial D_2(t, y(t), y'(t))}{\partial y} y'(t) + \frac{\partial D_2(t, y(t), y'(t))}{\partial y'} y''(t) \\
&= D_3(t, y(t), y'(t), y''(t)) \\
&\vdots
\end{aligned}
$$

questa suggerisce la definizione delle funzioni

$$D_k(t, y, y', \dots, y^{(k-1)})$$

tali che data la soluzione esatta $y(t)$ vale

$$y^{(k)}(t) = D_k(t, y(t), y'(t), \dots, y^{(k-1)}(t))$$

e che soddisfano la ricorrenza:

$$
\begin{cases}
D_1(t, y) = f(t, y), \\
D_{k+1}(t, y, y', \dots, y^{(k)}) = \dfrac{\partial D_k(t, y, y', \dots, y^{(k-1)})}{\partial t} \\
\qquad\qquad + \displaystyle\sum_{j=0}^{k-1} \frac{\partial D_k(t, y, y', \dots, y^{(k-1)})}{\partial y^{(j)}} y^{(j+1)}, \quad k = 1, 2, \dots, \quad y^{(0)} := y
\end{cases}
$$

29.5.1 Esempio: $f(t, y) = t + y + ty$

Consideriamo la seguente equazione differenziale ordinaria:

$$
\begin{cases}
y' = t + y + ty, \\
y(a) = y_a.
\end{cases}
$$

Per questa ODE, le funzioni $D_k(t, \dots)$ si calcolano come segue (omettiamo gli argomenti per semplicità)

$$
\begin{aligned}
D_1 &= t + y + ty, \\
D_2 &= \frac{\partial D_1}{\partial t} + \frac{\partial D_1}{\partial y} y' \\
&= (t+1)y' + y + 1, \\
D_3 &= \frac{\partial D_2}{\partial t} + \frac{\partial D_2}{\partial y} y' + \frac{\partial D_2}{\partial y'} y'' \\
&= (t+1)y'' + 2y', \\
D_4 &= \frac{\partial D_3}{\partial t} + \frac{\partial D_3}{\partial y} y' + \frac{\partial D_3}{\partial y'} y'' + \frac{\partial D_3}{\partial y''} y''' \\
&= (t+1)y''' + 3y'' \\
D_5 &= \frac{\partial D_4}{\partial t} + \frac{\partial D_4}{\partial y} y' + \frac{\partial D_4}{\partial y'} y'' + \frac{\partial D_4}{\partial y''} y''' + \frac{\partial D_4}{\partial y'''} y'''' \\
&= (t+1)y'''' + 4y''', \\
D_6 &= \frac{\partial D_5}{\partial t} + \frac{\partial D_5}{\partial y} y' + \frac{\partial D_5}{\partial y'} y'' + \frac{\partial D_5}{\partial y''} y''' + \frac{\partial D_5}{\partial y'''} y'''' + \frac{\partial D_5}{\partial y''''} y''''' \\
&= (t+1)y''''' + 5y''''.
\end{aligned}
$$

29.5.1.1 Metodo Numerico di Ordine 6

Ora possiamo costruire un metodo numerico di ordine 6, utilizzando le derivate fino alla sesta:

$$
\begin{cases}
y_{k+1} = y_k + hy'_k + \frac{h^2}{2!} y''_k + \frac{h^3}{3!} y'''_k + \frac{h^4}{4!} y''''_k + \frac{h^5}{5!} y'''''_k + \frac{h^6}{6!} y''''''_k, \\
\\
y'_k = (t_k + 1)y_k + t_k, \\
y''_k = (t_k + 1)y'_k + y_k + 1, \\
y'''_k = (t_k + 1)y''_k + 2y'_k, \\
y''''_k = (t_k + 1)y'''_k + 3y''_k, \\
y'''''_k = (t_k + 1)y''''_k + 4y'''_k, \\
y''''''_k = (t_k + 1)y'''''_k + 5y''''_k.
\end{cases}
$$

Questo metodo, basato sull'espansione in serie di Taylor, permette di ottenere un'approssimazione di ordine elevato con una complessità computazionale ridotta rispetto alla versione semplice del metodo.

29.6 Sistemi di ODE

Tutto quello che abbiamo visto vale, parola per parola, anche quando l'incognita non è un numero ma un vettore. È il caso interessante nelle applicazioni: un modello di popolazioni, un circuito, un sistema meccanico, un modello epidemiologico sono tutti sistemi di equazioni accoppiate.

29.6.1 Il caso vettoriale: la ricorrenza non cambia

Consideriamo il problema di Cauchy vettoriale

$$\begin{cases} \mathbf{y}'(t) = \mathbf{f}(t, \mathbf{y}(t)), \\ \mathbf{y}(a) = \mathbf{y}_a, \end{cases} \qquad \mathbf{y}(t) \in \mathfrak{R}^m, \quad \mathbf{f} : \mathfrak{R} \times \mathfrak{R}^m \to \mathfrak{R}^m. \tag{29.3}$$

In componenti, $\mathbf{y} = (y_1, \dots, y_m)^T$ e $\mathbf{f} = (f_1, \dots, f_m)^T$: le m equazioni sono accoppiate perché ogni f_i può dipendere da tutte le componenti.

Lo sviluppo di Taylor si applica a ciascuna componente, quindi

$$\mathbf{y}(t+h) = \mathbf{y}(t) + h\,\mathbf{y}'(t) + \frac{h^2}{2}\mathbf{y}''(t) + \cdots + \frac{h^p}{p!}\mathbf{y}^{(p)}(t) + \mathcal{O}(h^{p+1}), \tag{29.4}$$

ed esattamente come nel caso scalare il problema è esprimere $\mathbf{y}''$, $\mathbf{y}'''$, … in funzione di $\mathbf{f}$. Deriviamo la prima equazione rispetto a t usando la regola della catena, che per una funzione a valori vettoriali diventa

$$\mathbf{y}''(t) = \frac{\mathrm{d}}{\mathrm{d}t}\mathbf{f}(t, \mathbf{y}(t)) = \frac{\partial \mathbf{f}}{\partial t} + \underbrace{\frac{\partial \mathbf{f}}{\partial \mathbf{y}}}_{\text{Jacobiana}} \mathbf{y}'(t) = \mathbf{f}_t + J_{\mathbf{f}}\,\mathbf{f}, \tag{29.5}$$

dove

$$J_{\mathbf{f}}(t, \mathbf{y}) = \frac{\partial \mathbf{f}}{\partial \mathbf{y}} = \begin{pmatrix} \dfrac{\partial f_1}{\partial y_1} & \cdots & \dfrac{\partial f_1}{\partial y_m} \\ \vdots & & \vdots \\ \dfrac{\partial f_m}{\partial y_1} & \cdots & \dfrac{\partial f_m}{\partial y_m} \end{pmatrix} \in \mathfrak{R}^{m \times m}$$

è la **matrice Jacobiana** di $\mathbf{f}$, e $J_{\mathbf{f}}\,\mathbf{f}$ è un ordinario prodotto matrice–vettore. Confrontando 29.5 con la formula scalare $y'' = f_t + f_y f$ si vede che **non è cambiato nulla**: la derivata parziale $\partial f/\partial y$ è diventata una matrice e il prodotto è diventato un prodotto matrice–vettore.

Lo stesso vale per tutta la ricorrenza. Definendo le funzioni $\mathbf{D}_k : \mathfrak{R} \times \mathfrak{R}^m \to \mathfrak{R}^m$ come nel caso scalare,

$$\begin{cases} \mathbf{D}_1(t, \mathbf{y}) = \mathbf{f}(t, \mathbf{y}), \\ \mathbf{D}_{k+1}(t, \mathbf{y}) = \dfrac{\partial \mathbf{D}_k}{\partial t} + J_{\mathbf{D}_k}(t, \mathbf{y})\,\mathbf{f}(t, \mathbf{y}), \qquad k = 1, 2, \dots \end{cases} \tag{29.6}$$

si ottiene $\mathbf{y}^{(k)}(t) = \mathbf{D}_k(t, \mathbf{y}(t))$ e quindi il **metodo di Taylor di ordine p per sistemi**

$$\mathbf{y}_{k+1} = \mathbf{y}_k + \sum_{j=1}^{p} \frac{h^j}{j!}\,\mathbf{D}_j(t_k, \mathbf{y}_k). \tag{29.7}$$

> **Nota**
>
> Il prezzo del caso vettoriale non è concettuale ma di **taglia**. Nel caso scalare al passo k servono le derivate parziali di una funzione; qui serve la Jacobiana $m \times m$ di $\mathbf{D}_k$, cioè m^2 derivate parziali, e le espressioni crescono molto in fretta. Per questo, in pratica, non si costruiscono mai esplicitamente le $\mathbf{D}_k$: si propagano invece i **coefficienti** di Taylor, come mostriamo ora.

29.6.2 Dalle derivate ai coefficienti di Taylor

Conviene cambiare incognita. Invece delle derivate $\mathbf{y}^{(k)}$ usiamo i **coefficienti di Taylor normalizzati**

$$\mathbf{y}_{[k]} = \frac{\mathbf{y}^{(k)}(t)}{k!}, \qquad \text{cosicché} \qquad \mathbf{y}(t+h) = \sum_{k \ge 0} \mathbf{y}_{[k]}\, h^k. \tag{29.8}$$

Il vantaggio è che la relazione fra la funzione e la sua derivata diventa banale: se $\mathbf{g}(t) = \mathbf{y}'(t)$ allora

$$\mathbf{y}_{[k+1]} = \frac{\mathbf{g}_{[k]}}{k+1}, \tag{29.9}$$

perché derivare abbassa di uno l'indice e moltiplica per $k+1$. Dunque, **se so calcolare i coefficienti del secondo membro $\mathbf{f}$, i coefficienti della soluzione si ottengono dividendo per $k+1$**. Il problema si sposta tutto sul calcolo dei coefficienti di $\mathbf{f}$ a partire da quelli di $\mathbf{y}$.

29.6.3 La regola di Leibniz e i sistemi polinomiali

Per i secondi membri che si incontrano più spesso — polinomi nelle componenti di $\mathbf{y}$ — quel calcolo è elementare, perché somme e prodotti si trasformano in operazioni sui coefficienti:

$$\begin{aligned} (a+b)_{[k]} &= a_{[k]} + b_{[k]}, \\ (\lambda\, a)_{[k]} &= \lambda\, a_{[k]}, \\ (a\, b)_{[k]} &= \sum_{j=0}^{k} a_{[j]}\, b_{[k-j]}. \end{aligned} \tag{29.10}$$

L'ultima è la regola di Leibniz scritta nei coefficienti normalizzati — un **prodotto di convoluzione**, lo stesso che si usa per moltiplicare due serie di potenze. Si verifica subito a partire da $(ab)^{(k)} = \sum_j \binom{k}{j} a^{(j)} b^{(k-j)}$ dividendo per $k!$.

L'algoritmo che ne segue è semplice: si parte da $\mathbf{y}_{[0]}$, il valore corrente; si calcolano i coefficienti di $\mathbf{f}$ di ordine k con 29.10; si ricava $\mathbf{y}_{[k+1]}$ con 29.9; si ripete fino a $k = p-1$; infine si somma la serie troncata 29.8. Nessuna derivata simbolica, nessuna Jacobiana, costo $\mathcal{O}(p^2 m)$ per passo.

29.6.4 Esempio: il modello preda–predatore

Il modello di Lotka–Volterra[3][4] descrive due popolazioni: le prede $u(t)$ e i predatori $v(t)$.

$$\begin{cases} u' = \alpha\, u - \beta\, uv, \\ v' = \delta\, uv - \gamma\, v, \end{cases} \qquad u(0) = u_0, \quad v(0) = v_0. \tag{29.11}$$

Le prede crescono con tasso α in assenza di predatori e vengono predate con frequenza proporzionale agli incontri uv; i predatori si riproducono in proporzione agli stessi incontri e muoiono con tasso γ. Il sistema è **autonomo**, quindi $\mathbf{f}_t = \mathbf{0}$ e 29.5 si riduce a $\mathbf{y}'' = J_{\mathbf{f}}\mathbf{f}$ con

$$J_{\mathbf{f}}(u,v) = \begin{pmatrix} \alpha - \beta v & -\beta u \\ \delta v & \delta u - \gamma \end{pmatrix}, \qquad \mathbf{y}'' = \begin{pmatrix} \alpha u' - \beta\,(u'v + uv') \\ \delta\,(u'v + uv') - \gamma v' \end{pmatrix}.$$

La seconda scrittura è già la regola di Leibniz applicata al prodotto uv, e suggerisce la ricorrenza generale. Posto $(uv)_{[k]} = \sum_{j=0}^{k} u_{[j]} v_{[k-j]}$,

[3] Alfred James Lotka (1880-1949)
[4] Vito Volterra (1860-1940)

$$\begin{cases} u_{[k+1]} = \dfrac{\alpha\, u_{[k]} - \beta\,(uv)_{[k]}}{k+1}, \\ v_{[k+1]} = \dfrac{\delta\,(uv)_{[k]} - \gamma\, v_{[k]}}{k+1}, \end{cases} \qquad k = 0, 1, \ldots, p-1, \tag{29.12}$$

con $u_{[0]} = u_k$ e $v_{[0]} = v_k$ i valori correnti. Sono poche righe di codice e funzionano per **qualunque** ordine p.

```
# prodotto di Cauchy: (a*b)_[k] = sum_j a_[j] b_[k-j]
cauchy <- function(a, b, k) sum(a[1:(k + 1)] * b[(k + 1):1])

# coefficienti di Taylor di Lotka-Volterra fino all'ordine p
lv_coef <- function(u0, v0, p, par) {
  u <- numeric(p + 1); v <- numeric(p + 1)
  u[1] <- u0; v[1] <- v0                          # u[j+1] contiene u_[j]
  for (k in 0:(p - 1)) {
    uv <- cauchy(u, v, k)
    u[k + 2] <- (par$alpha * u[k + 1] - par$beta  * uv       ) / (k + 1)
    v[k + 2] <- (par$delta * uv       - par$gamma * v[k + 1] ) / (k + 1)
  }
  list(u = u, v = v)
}

# un passo = somma della serie troncata; poi si itera
lv_taylor <- function(u0, v0, h, n, p, par) {
  U <- numeric(n + 1); V <- numeric(n + 1)
  U[1] <- u0; V[1] <- v0; pw <- h^(0:p)
  for (i in 1:n) {
    C <- lv_coef(U[i], V[i], p, par)
    U[i + 1] <- sum(C$u * pw)
    V[i + 1] <- sum(C$v * pw)
  }
  cbind(t = (0:n) * h, u = U, v = V)
}
```

Integriamo con $\alpha = 2/3$, $\beta = 4/3$, $\gamma = \delta = 1$ e $u_0 = v_0 = 0{,}9$. Per questi valori l'equilibrio non banale è $(\gamma/\delta,\ \alpha/\beta) = (1,\ 1/2)$ e le orbite sono curve **chiuse** attorno ad esso: le due popolazioni oscillano indefinitamente, con i predatori in ritardo di fase sulle prede.

```
par_lv <- list(alpha = 2/3, beta = 4/3, gamma = 1, delta = 1)
u0 <- 0.9; v0 <- 0.9; h <- 0.05; Tend <- 30
S8 <- lv_taylor(u0, v0, h, Tend/h, 8, par_lv)   # ordine 8
S1 <- lv_taylor(u0, v0, h, Tend/h, 1, par_lv)   # ordine 1 = Eulero

op <- par(mfrow = c(1, 2), mar = c(4.2, 4.2, 2.6, 0.8))
matplot(S8[, "t"], S8[, c("u", "v")], type = "l", lty = 1, lwd = 2,
        col = c(4, 2), xlab = "t", ylab = "popolazione",
        main = "Preda-predatore nel tempo")
legend("topright", c("prede u(t)", "predatori v(t)"), col = c(4, 2),
       lwd = 2, bty = "n", cex = 0.85)

plot(S1[, "u"], S1[, "v"], type = "l", lwd = 1.6, col = "grey55",
     xlab = "prede u", ylab = "predatori v", main = "Ritratto di fase",
     xlim = range(S1[, "u"], S8[, "u"]), ylim = range(S1[, "v"], S8[, "v"]))
lines(S8[, "u"], S8[, "v"], lwd = 2.2, col = 4)
points(par_lv$gamma/par_lv$delta, par_lv$alpha/par_lv$beta,
       pch = 4, lwd = 2, col = 2, cex = 1.3)
points(u0, v0, pch = 19, cex = 1)
legend("topright", c("Eulero (p=1)", "Taylor p=8", "equilibrio"),
```

```
       col = c("grey55", 4, 2), lwd = c(1.6, 2.2, NA), pch = c(NA, NA, 4),
       bty = "n", cex = 0.8)
par(op)
```

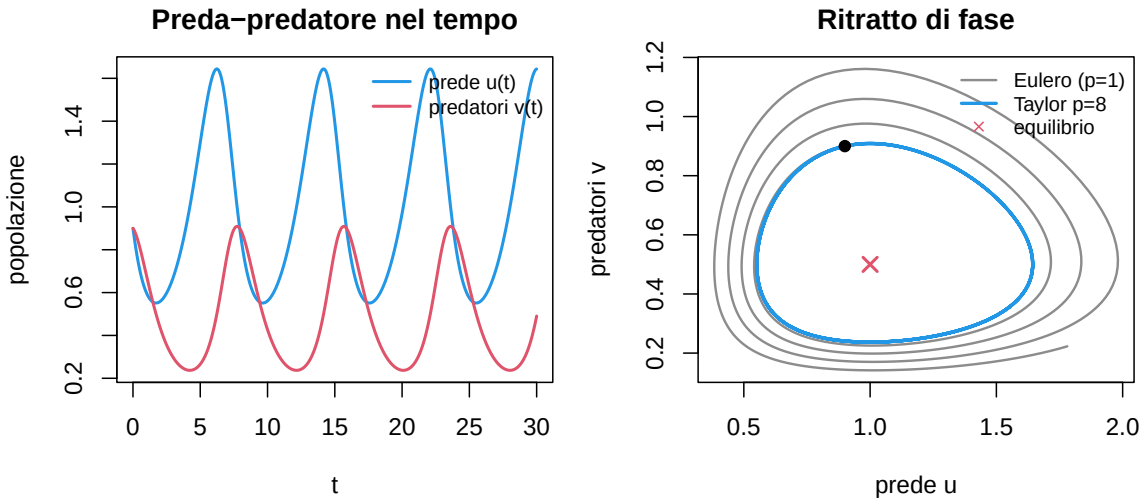


Figure 29.1: Preda-predatore integrato con il metodo di Taylor. A sinistra le due popolazioni nel tempo: i predatori seguono le prede con un ritardo di fase. A destra il ritratto di fase: l'orbita esatta `echiusa e il metodo di ordine 8 la riproduce, mentre Eulero (ordine 1) a parita` di passo spiraleggia verso l'esterno, cioe` crea energia dal nulla.

Il pannello di destra mostra un fenomeno che il solo errore puntuale non racconta. Il sistema 29.11 possiede una quantità conservata,

$$V(u, v) = \delta\, u - \gamma \ln u + \beta\, v - \alpha \ln v, \tag{29.13}$$

costante lungo ogni soluzione: lo si verifica derivando e raccogliendo, $\dot V = (\delta u - \gamma)\big[(\alpha - \beta v) + (\beta v - \alpha)\big] = 0$. Le orbite sono dunque le curve di livello di V, cioè curve chiuse. Eulero non conserva V e l'orbita numerica si allarga ad ogni giro; il metodo di ordine 8, a parità di passo, la mantiene chiusa.

```
V <- function(u, v, p) p$delta*u - p$gamma*log(u) + p$beta*v - p$alpha*log(v)
V0 <- V(u0, v0, par_lv)
deriva <- sapply(c(1, 2, 4, 8), function(p) {
  S <- lv_taylor(u0, v0, h, Tend/h, p, par_lv)
  max(abs(V(S[, "u"], S[, "v"], par_lv) - V0))
})
knitr::kable(data.frame(ordine_p = c(1, 2, 4, 8),
                        deriva_massima_di_V = sprintf("%.2e", deriva)),
             align = c("r", "r"))
```

Table 29.1: Deriva massima della quantita`conservata V su 30 unita` di tempo, a parita` di passo h=0.05. L'ordine del metodo si legge direttamente nella colonna di destra.

ordine_p	deriva_massima_di_V
1	2.26e-01
2	1.73e-04
4	6.74e-08
8	2.35e-14

Table 29.1: Deriva massima della quantitaconservata `V` su `30` `unita` di tempo, a parita‘ di passo h=0.05. L’ordine del metodo si legge direttamente nella colonna di destra.

ordine_p	deriva_massima_di_V

29.6.5 Esempio: il modello SIR

Il modello SIR[56] divide una popolazione di N individui in tre gruppi: i **suscettibili** S, gli **infetti** I e i **rimossi** R (guariti o deceduti).

$$\begin{cases} S' = -\dfrac{\beta}{N} S\, I, \\ I' = \dfrac{\beta}{N} S\, I - \gamma\, I, \\ R' = \gamma\, I, \end{cases} \qquad S(0) + I(0) + R(0) = N. \tag{29.14}$$

Il termine $\beta SI/N$ conta i contagi per unità di tempo: è proporzionale agli incontri fra suscettibili e infetti. Il parametro γ è il tasso di rimozione, quindi $1/\gamma$ è la durata media dell’infezione, e il rapporto

$$\mathcal{R}_0 = \frac{\beta}{\gamma}$$

è il **numero di riproduzione di base**: il numero medio di contagi causati da un infetto in una popolazione interamente suscettibile. Se $\mathcal{R}_0 > 1$ l’epidemia parte, se $\mathcal{R}_0 < 1$ si spegne.

Il secondo membro è di nuovo polinomiale, con un solo prodotto SI: la ricorrenza sui coefficienti è immediata. Posto $(SI)_{[k]} = \sum_{j=0}^{k} S_{[j]} I_{[k-j]}$,

$$\begin{cases} S_{[k+1]} = \dfrac{1}{k+1}\left(-\dfrac{\beta}{N}(SI)_{[k]}\right), \\ I_{[k+1]} = \dfrac{1}{k+1}\left(\dfrac{\beta}{N}(SI)_{[k]} - \gamma I_{[k]}\right), \\ R_{[k+1]} = \dfrac{1}{k+1}\,\gamma\, I_{[k]}. \end{cases} \tag{29.15}$$

Il metodo conserva la popolazione, a ogni ordine

Sommando le tre righe di 29.15 si ottiene, per ogni $k \geq 0$,

$$S_{[k+1]} + I_{[k+1]} + R_{[k+1]} = \frac{1}{k+1}\left(-\frac{\beta}{N}(SI)_{[k]} + \frac{\beta}{N}(SI)_{[k]} - \gamma I_{[k]} + \gamma I_{[k]}\right) = 0.$$

Tutti i coefficienti di ordine ≥ 1 della somma $S + I + R$ sono nulli, quindi la somma calcolata non cambia da un passo all’altro: il vincolo $S + I + R = N$ è soddisfatto **esattamente**, a meno degli errori di arrotondamento, per qualunque ordine p e qualunque passo h. Non è una proprietà di tutti i metodi: va verificata caso per caso, ed è una delle prime cose da controllare in un codice.

```
sir_coef <- function(S0, I0, R0, p, par) {
  S <- numeric(p + 1); I <- numeric(p + 1); R <- numeric(p + 1)
```

[5] William Ogilvy Kermack (1898-1970)
[6] Anderson Gray McKendrick (1876-1943)

```
  S[1] <- S0; I[1] <- I0; R[1] <- R0
  for (k in 0:(p - 1)) {
    si <- cauchy(S, I, k) * par$beta / par$N
    S[k + 2] <- (-si                          ) / (k + 1)
    I[k + 2] <- ( si - par$gamma * I[k + 1] ) / (k + 1)
    R[k + 2] <- (      par$gamma * I[k + 1] ) / (k + 1)
  }
  list(S = S, I = I, R = R)
}

sir_taylor <- function(y0, h, n, p, par) {
  out <- matrix(0, n + 1, 4); out[1, ] <- c(0, y0); pw <- h^(0:p)
  for (i in 1:n) {
    C <- sir_coef(out[i, 2], out[i, 3], out[i, 4], p, par)
    out[i + 1, ] <- c(i*h, sum(C$S*pw), sum(C$I*pw), sum(C$R*pw))
  }
  colnames(out) <- c("t", "S", "I", "R"); out
}
```

Integriamo una popolazione di $N = 1000$ individui con un solo infetto iniziale, $\beta = 0{,}3$ e $\gamma = 0{,}1$ al giorno, quindi $\mathcal{R}_0 = 3$.

```
par_sir <- list(beta = 0.30, gamma = 0.10, N = 1000)
sir <- sir_taylor(c(999, 1, 0), h = 0.5, n = 240, p = 6, par_sir)

par(mar = c(4.2, 4.2, 2.8, 0.8))
matplot(sir[, "t"], sir[, c("S", "I", "R")], type = "l", lty = 1, lwd = 2.2,
        col = c(4, 2, 3), xlab = "giorni", ylab = "individui",
        main = "Modello SIR,  R0 = 3")
ip <- which.max(sir[, "I"])
abline(v = sir[ip, "t"], lty = 3, col = "grey50")
abline(h = par_sir$N * par_sir$gamma / par_sir$beta, lty = 2, col = "grey50")
points(sir[ip, "t"], sir[ip, "I"], pch = 19, col = 2)
text(sir[ip, "t"], sir[ip, "I"],
     sprintf(" picco: %.0f infetti al giorno %.0f", sir[ip, "I"], sir[ip, "t"]),
     pos = 4, cex = 0.8)
text(72, par_sir$N*par_sir$gamma/par_sir$beta, "soglia  N/R0", pos = 3,
     cex = 0.78, col = "grey30")
legend("right", c("suscettibili S", "infetti I", "rimossi R"),
       col = c(4, 2, 3), lwd = 2.2, bty = "n")
```

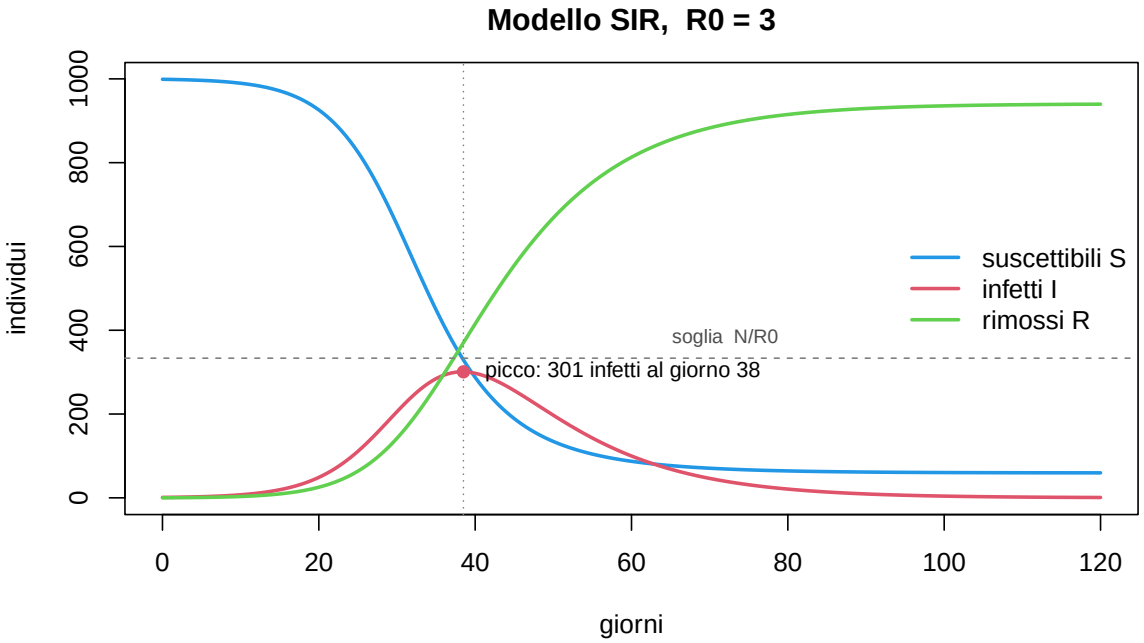


Figure 29.2: Modello SIR con R0 = 3, integrato con Taylor di ordine 6 e passo di mezza giornata. La curva degli infetti cresce, raggiunge un picco quando i suscettibili scendono sotto la soglia N/R0 e poi decade; una parte della popolazione non viene mai contagiata.

Tre letture del grafico, tutte verificabili sui numeri.

- Il picco degli infetti cade dove la curva S attraversa la soglia $N/\mathcal{R}_0$ (linea tratteggiata): è lì che $I' = 0$, perché $I' = I\,(\beta S/N - \gamma)$ cambia segno quando $S = \gamma N/\beta$.
- L'epidemia si spegne **prima** di contagiare tutti: la frazione finale $R(\infty)/N$ risolve $x = 1 - e^{-\mathcal{R}_0 x}$, che per $\mathcal{R}_0 = 3$ dà $x \simeq 0{,}94$.
- La somma $S + I + R$ resta pari a N a meno dell'arrotondamento, come garantito sopra.

```
x_inf <- uniroot(function(x) x - (1 - exp(-3*x)), c(0.1, 0.999))$root
knitr::kable(data.frame(
  grandezza = c("scarto massimo su S+I+R-N", "picco di I",
                "giorno del picco", "R(120)/N calcolato",
                "R(infinito)/N teorico"),
  valore    = sprintf("%.6g",
                c(max(abs(rowSums(sir[, 2:4]) - par_sir$N)),
                  max(sir[, "I"]), sir[which.max(sir[, "I"]), "t"],
                  sir[nrow(sir), "R"]/par_sir$N, x_inf))),
             align = c("l", "r"))
```

Table 29.2: Verifiche sulla soluzione numerica del SIR.

grandezza	valore
scarto massimo su S+I+R-N	2.72848e-12
picco di I	300.768
giorno del picco	38.5
R(120)/N calcolato	0.939588
R(infinito)/N teorico	0.940481

29.6.6 Ordine e costo

Resta da controllare che il metodo abbia davvero l'ordine annunciato anche nel caso vettoriale. Confrontiamo la soluzione a T fissato con una soluzione di riferimento, al variare del passo.

```
rif_lv  <- deSolve::ode(c(u0, v0), c(0, 20),
             function(t, y, p) list(c(p$alpha*y[1] - p$beta*y[1]*y[2],
                                      p$delta*y[1]*y[2] - p$gamma*y[2])),
             par_lv, atol = 1e-13, rtol = 1e-13)[2, 2:3]
rif_sir <- deSolve::ode(c(999, 1, 0), c(0, 120),
             function(t, y, p) list(c(-p$beta*y[1]*y[2]/p$N,
                                       p$beta*y[1]*y[2]/p$N - p$gamma*y[2],
                                       p$gamma*y[2])),
             par_sir, atol = 1e-13, rtol = 1e-13)[2, 2:4]

ordini <- c(1, 2, 4, 8); cols <- c(1, 2, 4, 3)
h_lv  <- c(0.4, 0.2, 0.1, 0.05, 0.025)
h_sir <- c(4, 2, 1, 0.5, 0.25)
E_lv <- sapply(ordini, function(p) sapply(h_lv, function(hh) {
  S <- lv_taylor(u0, v0, hh, round(20/hh), p, par_lv)
  max(abs(S[nrow(S), 2:3] - rif_lv)) }))
E_sir <- sapply(ordini, function(p) sapply(h_sir, function(hh) {
  S <- sir_taylor(c(999, 1, 0), hh, round(120/hh), p, par_sir)
  max(abs(S[nrow(S), 2:4] - rif_sir)) }))

op <- par(mfrow = c(1, 2), mar = c(4.2, 4.4, 2.8, 0.8))
for (k in 1:2) {
  H <- if (k == 1) h_lv else h_sir
  E <- if (k == 1) E_lv else E_sir
  E[E < 1e-14] <- NA
  matplot(H, E, type = "b", pch = 16, lty = 1, lwd = 2, col = cols, log = "xy",
          xlab = "passo h", ylab = "errore finale",
          main = if (k == 1) "Preda-predatore (T=20)" else "SIR (T=120)")
  for (j in seq_along(ordini))
    lines(H, E[1, j] * (H/H[1])^ordini[j], lty = 3, col = cols[j])
  legend("bottomright", paste0("p = ", ordini), col = cols, lwd = 2,
         pch = 16, bty = "n", cex = 0.8)
}
par(op)
```

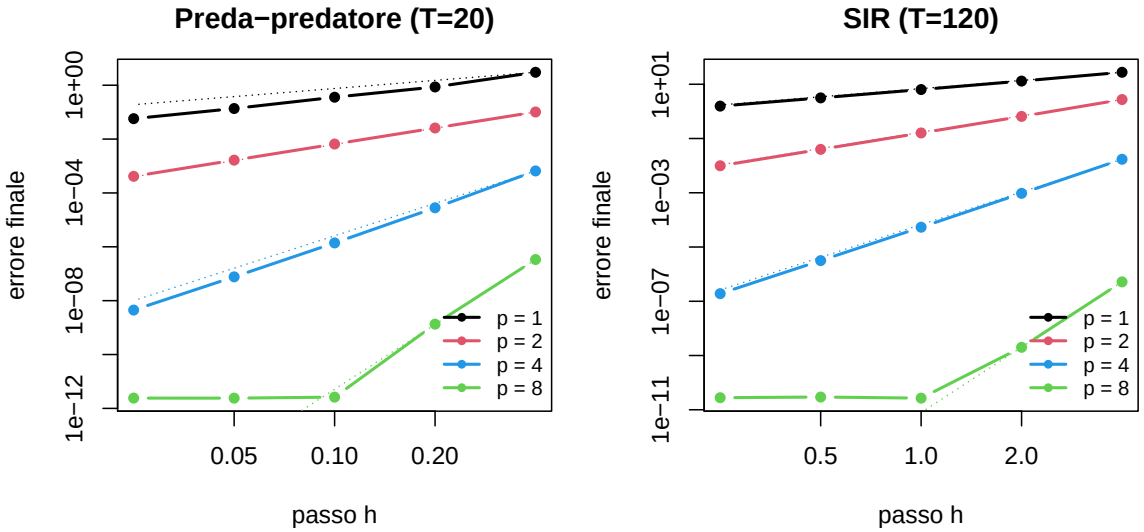


Figure 29.3: Errore al tempo finale in funzione del passo, in scala doppiamente logaritmica, per i due sistemi. Le rette tratteggiate hanno pendenza p: i punti vi si allineano, confermando l'ordine. Per p=8 la curva si appiattisce attorno a 1e-12 perche' l'errore di arrotondamento prende il sopravvento su quello di troncamento.

Il costo di un passo è $\mathcal{O}(p^2 m)$ operazioni: la somma di convoluzione di 29.10 costa $\mathcal{O}(k)$ al passo k, e si somma su $k = 0, \dots, p-1$. Confrontato con un Runge–Kutta di pari ordine, che richiede un numero di valutazioni di $\mathbf{f}$ che cresce più che linearmente in p, il metodo di Taylor è competitivo proprio quando p è alto e il secondo membro è polinomiale — e allora permette ordini, 20 o 30, che con Runge–Kutta non sono praticabili. Il limite resta quello di sempre: serve conoscere la forma analitica di $\mathbf{f}$ e saperne propagare i coefficienti, cosa che per un secondo membro generico richiede differenziazione automatica.

In sintesi

- Derivando ripetutamente la ODE si esprimono y'', y''', … in funzione di f e delle sue derivate parziali: nasce così il metodo di Taylor di ordine p.
- Il metodo raggiunge qualunque ordine con **un solo punto**, ma richiede le derivate di f: è pratico solo con derivazione simbolica o automatica.
- La notazione compatta $D_k(t, y)$ organizza la ricorsione e rende il metodo implementabile per p arbitrario.
- È il confronto che motiva i metodi di Runge-Kutta: stesso ordine, **nessuna derivata**, al prezzo di più valutazioni di f per passo.
- Per i **sistemi** la costruzione è identica: $\partial f/\partial y$ diventa la matrice Jacobiana e il prodotto diventa matrice–vettore (29.6).
- In pratica non si costruiscono le D_k: si propagano i **coefficienti di Taylor** con 29.9 e la regola di Leibniz 29.10. Per secondi membri polinomiali — come preda–predatore e SIR — bastano poche righe di codice e si raggiunge qualunque ordine, con costo $\mathcal{O}(p^2 m)$ per passo.
- Conviene controllare se il metodo preserva le quantità conservate del modello: il SIR mantiene $S + I + R = N$ esattamente a ogni ordine, mentre l'invariante di Lotka–Volterra viene conservato solo tanto meglio quanto più alto è l'ordine.



30. Metodi di Runge-Kutta

Il metodo di Eulero[1], sebbene utile in molti casi, ha limitazioni di accuratezza ed efficienza, soprattutto per problemi di grandi dimensioni, ad alta dimensionalità o con equazioni differenziali rigide. Per ottenere risultati più precisi e gestire tolleranze d'errore più stringenti, ci affidiamo ai **metodi di Runge-Kutta**[2].

30.1 Formulazione Generale

Consideriamo un problema di Cauchy[3] generico per un'equazione differenziale ordinaria (ODE):

$$\begin{cases} y' = f(t, y), \\ y(a) = y_a. \end{cases}$$

Un metodo di Runge-Kutta a un passo per approssimare la soluzione di questo problema può essere scritto come:

$$y_{k+1} = \Phi(t_k, y_k, h) = y_k + Ahf(t_k, y_k) + Bhf(t_k + Ch, y_k + Dhf(t_k, y_k)).$$

dove h è il passo di discretizzazione e i coefficienti A, B, C, D devono essere scelti per garantire l'accuratezza del metodo.

Richiamo: sviluppo di Taylor in due variabili

Tutto quello che segue poggia sullo sviluppo di f attorno a (t, y) in **entrambi** gli argomenti. Conviene enunciarlo con il resto quantitativo, perché è da lì che escono sia le condizioni d'ordine sia le costanti delle stime d'errore.

Proposition 30.1 — Sviluppo di Taylor con resto. Sia $f \in C^1$ in un intorno di (t, y). Allora

$$f(t + \alpha, y + \beta) = f(t, y) + \alpha\, f_t(t, y) + \beta\, f_y(t, y) + r_1(\alpha, \beta)$$

e, se inoltre il gradiente (f_t, f_y) è lipschitziano con costante M_1 in quell'intorno,

$$\boxed{|r_1(\alpha, \beta)| \le \frac{M_1}{2}\left(\alpha^2 + \beta^2\right)} \tag{30.1}$$

[1] Leonhard Euler (1707-1783)
[2] Carl David Tolmé Runge (1856-1927) e Martin Wilhelm Kutta (1867-1944)
[3] Augustin-Louis Cauchy (1789-1857)

Se $f \in C^2$ vale lo sviluppo al second'ordine

$$f(t+\alpha, y+\beta) = f + \alpha f_t + \beta f_y + \frac{1}{2}\Big(\alpha^2 f_{tt} + 2\alpha\beta f_{ty} + \beta^2 f_{yy}\Big) + r_2(\alpha,\beta)$$

con $|r_2| = \mathcal{O}\big((\alpha^2+\beta^2)^{3/2}\big)$, e $|r_2| \le \frac{M_2}{6}(\alpha^2+\beta^2)^{3/2}$ se l'hessiana è lipschitziana con costante M_2.

Nei metodi di Runge-Kutta si applica sempre con $\alpha = c_i h$ e $\beta = \mathcal{O}(h)$: ogni incremento è proporzionale ad h, quindi il resto 30.1 è $\mathcal{O}(h^2)$ e, al second'ordine, $\mathcal{O}(h^3)$. È questo che permette di troncare gli sviluppi all'ordine voluto e di raccogliere i termini in potenze di h.

> **Quanta regolarità serve davvero**
>
> Per ottenere un metodo di ordine p occorre $f \in C^p$: le condizioni d'ordine coinvolgono le derivate parziali fino all'ordine $p-1$ e la soluzione esatta deve essere C^{p+1}. Se f è solo lipschitziana, il metodo **converge** comunque (Teorema 28.1 è basato sulla sola lipschitzianità) ma con ordine 1: l'ordine alto è una proprietà che si paga in regolarità, non solo in stadi.

Calcolo dell'Errore Locale di Troncamento

Per valutare l'accuratezza del metodo, calcoliamo l'errore locale di troncamento $\tau(h)$, che rappresenta la differenza tra la soluzione esatta $y(t+h)$ e la stima data dal metodo $\Phi(t, y(t), h)$:

$$\begin{aligned}\tau(h) &= y(t+h) - \Phi(t, y(t), h) \\ &= y(t+h) - y(t) - Ahf(t, y(t)) - Bhf(t + Ch, y(t) + Dhf(t, y(t)))\end{aligned}$$

dove $y(t)$ è la soluzione esatta per la quale possiamo scrivere la serie di Taylor[4]

$$y(t+h) = y(t) + y'(t)h + y''(t)\frac{h^2}{2} + \cdots + y^{(p)}(t)\frac{h^p}{p!} + \mathcal{O}(h^{p+1})$$

Dalla ODE possiamo scrivere i primi due termini della serie come segue

$$y(t+h) = y(t) + \underbrace{f(t, y(t))}_{y'(t)} h + \underbrace{\left(\frac{\partial f(t, y(t))}{\partial t} + \frac{\partial f(t, y(t))}{\partial y} f(t, y(t))\right)}_{y''(t)} \frac{h^2}{2} + \mathcal{O}(h^3)$$

Sviluppo dell'Errore Locale di Troncamento

Sostituiamo questa espansione in $\tau(h)$:

$$\begin{aligned}\tau(h) = {} & \cancel{y(t)} + f(t, y(t))h + \left(\frac{\partial f(t, y(t))}{\partial t} + \frac{\partial f(t, y(t))}{\partial y} f(t, y(t))\right)\frac{h^2}{2} + \mathcal{O}(h^3) \\ & - \cancel{y(t)} - Ahf(t, y(t)) \\ & - Bh\left(f(t, y(t)) + \frac{\partial f(t, y(t))}{\partial t} Ch + \frac{\partial f(t, y(t))}{\partial y} Dhf(t, y(t)) + \mathcal{O}(h^2)\right) \\ = {} & h \cdot f(t, y(t))(1 - A - B) + \frac{h^2}{2} \cdot \frac{\partial f(t, y(t))}{\partial t}(1 - 2BC) \\ & + \frac{h^2}{2} \cdot \frac{\partial f(t, y(t))}{\partial y} f(t, y(t))(1 - 2BD) + \mathcal{O}(h^3)\end{aligned}$$

[4] Brook Taylor (1685-1731)

Per ottenere un metodo di **secondo ordine**, imponiamo le condizioni seguenti:

$$\begin{cases} 1 - A - B = 0 \\ 1 - 2BC = 0 \\ 1 - 2BD = 0 \end{cases}$$

Queste condizioni assicurano che l'errore di troncamento $\tau(h) = \mathcal{O}(h^3)$, quindi il metodo globale sarà del secondo ordine.

Famiglia RK del secondo ordine

Risolvendo il sistema di equazioni sopra, possiamo esprimere i parametri A, C, e D in funzione di B come segue:

$$A = 1 - B, \qquad C = D = \frac{1}{2B}.$$

Pertanto, i metodi di Runge-Kutta del secondo ordine assumono la forma generale:

$$y_{k+1} = y_k + (1 - B)hf(t_k, y_k) + Bhf\left(t_k + \frac{h}{2B}, y_k + \frac{h}{2B}f(t_k, y_k)\right).$$

Definiamo ora:

$$b_1 = 1 - B, \quad b_2 = B, \quad c_1 = 0, \quad c_2 = \frac{1}{2B}, \quad A_{21} = \frac{1}{2B}.$$

Possiamo riscrivere il metodo come segue:

$$y_{k+1} = y_k + b_1K_1 + b_2K_2$$

$$K_1 = hf(t_k + c_1h, y_k)$$

$$K_2 = hf\left(t_k + c_2h, y_k + A_{21}K_1\right)$$

Questa è la formulazione classica dei metodi di Runge-Kutta, che permette di approssimare soluzioni di ODE fino a un errore del secondo ordine con combinazioni lineari delle valutazioni di f in punti intermedi.

Estensione della Procedura

Il metodo di Runge-Kutta può essere esteso includendo termini aggiuntivi per ottenere una maggiore precisione. Consideriamo una formulazione estesa del metodo:

$$\begin{aligned} y_{k+1} &= \Phi(t_k, y_k, h) \\ &= y_k + Ahf(t_k + Bh, y_k + Chf(t_k, y_k)) \\ &\quad + Dhf(t_k + Eh, y_k + Fhf(t_k, y_k)). \end{aligned}$$

Anche in questo caso, utilizzando l'espansione di Taylor e minimizzando l'errore locale di troncamento, otteniamo un sistema di condizioni non lineari sui coefficienti A, B, C, D, E e F che devono essere soddisfatte. Queste condizioni conducono a intere famiglie di metodi di Runge-Kutta con proprietà diverse in termini di accuratezza e stabilità.

Un metodo di Runge-Kutta generico è costruito a partire da una formula di avanzamento nella forma generale:

$$y_{k+1} = y_k + \sum_{i=1}^{s} b_i K_i,$$

dove

$$K_1 = hf\left(t_k + hc_1, y_k + \sum_{j=1}^{s} a_{1j} K_j\right),$$

$$K_2 = hf\left(t_k + hc_2, y_k + \sum_{j=1}^{s} a_{2j} K_j\right),$$

$$\vdots$$

$$K_s = hf\left(t_k + hc_s, y_k + \sum_{j=1}^{s} a_{sj} K_j\right).$$

Questa formulazione introduce un totale di $2s + s^2$ coefficienti, dove:

- $b_1, b_2, \ldots, b_s$ rappresentano i pesi per i termini K_i,
- $c_1, c_2, \ldots, c_s$ rappresentano i coefficienti temporali,
- a_{ij} (con $i, j = 1, 2, \ldots, s$) rappresentano i coefficienti delle combinazioni lineari delle valutazioni di f nei punti intermedi.

Per determinare questi coefficienti, si applica la procedura della serie di Taylor in più variabili e si confrontano i termini della serie con l'espansione della soluzione esatta. Questo permette di ridurre l'errore locale di troncamento a un ordine $\mathcal{O}(h^p)$ desiderato.

Poiché il calcolo di queste condizioni può essere complesso, esistono già tabelle che forniscono le condizioni di ordine per i coefficienti b_i, c_j, e a_{ij} fino a ordini molto elevati.

30.2 Condizioni di Ordine

Ordine 1 (Metodo di Eulero)

Un metodo di Runge-Kutta ha ordine 1 se soddisfa la seguenti condizioni:

1. $\displaystyle\sum_{j=1}^{s} a_{ij} = c_i, \qquad i = 1, 2, \ldots, s$
2. $\displaystyle\sum_{i=1}^{s} b_i = 1$

Ordine 2

Un metodo di Runge-Kutta ha ordine 2 se soddisfa in aggiunta la seguente condizione:

3. $\displaystyle\sum_{i=1}^{s} b_i c_i = \frac{1}{2}$

Ordine 3

Un metodo di Runge-Kutta ha ordine 3 se soddisfa in aggiunta le seguenti condizioni:

4. $\displaystyle\sum_{i=1}^{s} b_i c_i^2 = \frac{1}{3}$
5. $\displaystyle\sum_{i,j=1}^{s} b_i a_{ij} c_j = \frac{1}{6}$

Ordine 4

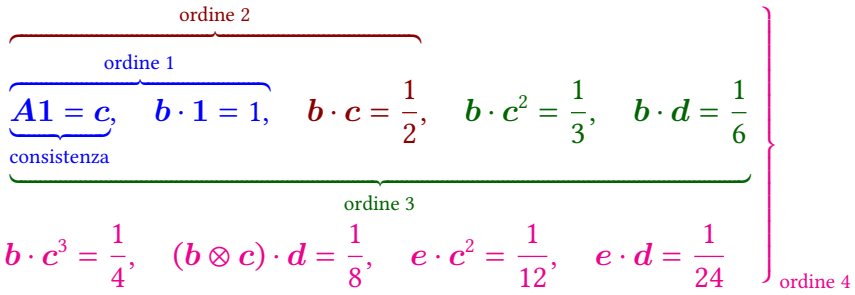
Un metodo di Runge-Kutta ha ordine 4 se soddisfa in aggiunta le seguenti condizioni:

6. $\sum_{i=1}^{s} b_i c_i^3 = \frac{1}{4}$
7. $\sum_{i,j=1}^{s} b_i c_i a_{ij} c_j = \frac{1}{8}$
8. $\sum_{i,j=1}^{s} b_i a_{ij} c_j^2 = \frac{1}{12}$
9. $\sum_{i,j,k=1}^{s} b_i a_{ij} a_{jk} c_k = \frac{1}{24}$

Ordini Superiori

Le condizioni di ordine per $p > 4$ diventano sempre più complesse e richiedono un numero maggiore di condizioni sui coefficienti.

- **Ordine 5** richiede 17 condizioni.
- **Ordine 6** richiede 37 condizioni.
- **Ordine 7** richiede 85 condizioni.
- **Ordine 8** richiede 200 condizioni.
- **Ordine 9** richiede 486 condizioni.
- **Ordine 10** richiede 1205 condizioni.



$$\boldsymbol{d} = A\boldsymbol{c}, \qquad \boldsymbol{e} = A^T\boldsymbol{b}, \qquad \boldsymbol{c}^2 = \boldsymbol{c}\otimes\boldsymbol{c}, \qquad \boldsymbol{c}^3 = \boldsymbol{c}\otimes\boldsymbol{c}\otimes\boldsymbol{c}$$

Di seguito, riportiamo una lista dei principali metodi di Runge-Kutta comunemente utilizzati.

30.3 Metodi di Runge-Kutta Espliciti

La forma generale di un metodo di Runge-Kutta esplicito è un metodo a un passo che approssima la soluzione utilizzando più fasi intermedie all'interno di ciascun passo temporale. Il metodo è rappresentato come segue:

$$
\begin{aligned}
y_{k+1} &= y_k + \sum_{i=1}^{s} b_i K_i, \\
&\begin{cases}
K_1 = hf\Big(t_k, y_k\Big), \\
K_2 = hf\Big(t_k + hc_2, y_k + a_{21}K_1\Big), \\
K_3 = hf\Big(t_k + hc_3, y_k + a_{31}K_1 + a_{32}K_2\Big), \\
\quad \vdots \\
K_s = hf\Big(t_k + hc_s, y_k + \sum_{j=1}^{s-1} a_{sj}K_j\Big),
\end{cases}
\end{aligned}
$$

dove $K_1, K_2, \dots, K_s$ sono valori intermedi chiamati *stadi*, e s è il numero di stadi. Le costanti a_{ij}, b_i e c_i definiscono il metodo di Runge-Kutta specifico e sono scelte per raggiungere un'accuratezza di ordine superiore. Il metodo calcola più somme pesate delle valutazioni della funzione (i valori K_i) per ottenere approssimazioni più accurate della soluzione.

I valori $t_k + hc_i$ rappresentano punti di quadratura nell'intervallo temporale $[t_k, t_{k+1}]$, e le quantità K_i/h approssimano i valori della funzione in questi punti.

30.3.1 Tableau di Butcher

Per semplificare la notazione dei metodi di Runge-Kutta, i coefficienti a_{ij}, b_i e c_i vengono spesso riassunti in una forma compatta chiamata *tableau di Butcher*[5]. Il tableau per il metodo di Runge-Kutta esplicito è dato da:

$$
\begin{array}{c|cccccc}
0 & 0 & 0 & 0 & 0 & 0 \\
c_2 & a_{21} & 0 & 0 & 0 & 0 \\
c_3 & a_{31} & a_{32} & 0 & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & 0 & 0 \\
c_s & a_{s1} & a_{s2} & \dots & a_{s,s-1} & 0 \\
\hline
 & b_1 & b_2 & \dots & b_{s-1} & b_s
\end{array}
$$

Euler Esplicito

Il metodo di Euler esplicito rappresentato utilizzando un tableau di Butcher:

$$
\begin{array}{c|c}
0 & 0 \\
\hline
 & 1
\end{array}
\tag{30.2}
$$

Metodo del Punto Medio

Il Metodo del Punto Medio, noto anche come Metodo di Collatz, è una variante del metodo di Euler esplicito. Questo approccio è classificato come metodo del secondo ordine e consiste in due fasi, consentendo di migliorare l'accuratezza nell'integrazione numerica.

[5] John Charles Butcher (1933)

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1/2 & 1/2 & 0 \\ \hline & 0 & 1 \end{array} \tag{30.3}$$

Metodo di Heun

Il Metodo di Heun, spesso noto come la regola del trapezio esplicita, rappresenta un miglioramento rispetto al metodo di Euler esplicito standard. Questa tecnica è classificata come un metodo del secondo ordine e utilizza due fasi per ottenere una maggiore accuratezza nell'integrazione numerica.

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & 1/2 & 1/2 \end{array} \tag{30.4}$$

Metodo di Ralston

Il metodo di Ralston è una tecnica del secondo ordine che utilizza due fasi, progettata specificamente per ottenere un limite minimo all'errore locale.

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 2/3 & 2/3 & 0 \\ \hline & 1/4 & 3/4 \end{array} \tag{30.5}$$

Metodo generico del secondo ordine

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \alpha & \alpha & 0 \\ \hline & 1-\dfrac{1}{2\alpha} & \dfrac{1}{2\alpha} \end{array} \qquad \alpha \in (0,1) \tag{30.6}$$

Runge-Kutta del terzo ordine (RK3)

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 \\ 1 & -1 & 2 & 0 \\ \hline & 1/6 & 2/3 & 1/6 \end{array} \tag{30.7}$$

Metodo del terzo ordine di Heun

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1/3 & 1/3 & 0 & 0 \\ 2/3 & 0 & 2/3 & 0 \\ \hline & 1/4 & 0 & 3/4 \end{array} \tag{30.8}$$

Metodo del terzo ordine di Ralston

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 \\ 3/4 & 0 & 3/4 & 0 \\ \hline & 2/9 & 1/3 & 4/9 \end{array} \tag{30.9}$$

Runge-Kutta SSP (SSPRK3)

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1/2 & 1/4 & 1/4 & 0 \\ \hline & 1/6 & 1/6 & 2/3 \end{array} \tag{30.10}$$

Metodo generico del terzo ordine

Con $\alpha \neq 0, 2/3, 1$:

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ \alpha & \alpha & 0 & 0 \\ 1 & 1 + \dfrac{1-\alpha}{\alpha(3\alpha-2)} & -\dfrac{1-\alpha}{\alpha(3\alpha-2)} & 0 \\ \hline & \dfrac{1}{2} - \dfrac{1}{6\alpha} & \dfrac{1}{6\alpha(1-\alpha)} & \dfrac{2-3\alpha}{6\alpha(1-\alpha)} \end{array} \tag{30.11}$$

Metodo di Runge-Kutta classico (RK4)

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 1/2 & 1/2 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ \hline & 1/6 & 2/6 & 2/6 & 1/6 \end{array} \tag{30.12}$$

Ralston metodo del quarto ordine

È un metodo del secondo ordine con due stadi, progettato per minimizzare l'errore di troncamento.

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ 2/5 & 2/5 & 0 & 0 & 0 \\ c_3 & a_{3,1} & a_{3,2} & 0 & 0 \\ 1 & a_{4,1} & a_{4,2} & a_{4,3} & 0 \\ \hline & b_1 & b_2 & b_3 & b_4 \end{array} \tag{30.13}$$

dove

$$
\begin{aligned}
b_1 &= \frac{263}{1812} + \frac{2}{151}\sqrt{5} \\
b_2 &= \frac{125}{3828} - \frac{250}{957}\sqrt{5} \\
b_3 &= \frac{3426304}{5924787} + \frac{553984}{1974929}\sqrt{5} \\
b_4 &= \frac{10}{41} - \frac{4}{123}\sqrt{5} \\
c_3 &= \frac{7}{8} - \frac{3}{16}\sqrt{5} \\
a_{3,1} &= \frac{357}{256}\sqrt{5} - \frac{2889}{1024} \\
a_{4,1} &= \frac{1047}{3020}\sqrt{5} - \frac{673}{1208} \\
a_{3,2} &= \frac{3785}{1024} - \frac{405}{256}\sqrt{5} \\
a_{4,2} &= -\frac{975}{2552} - \frac{1523}{1276}\sqrt{5} \\
a_{4,3} &= \frac{93408}{48169} + \frac{203968}{240845}\sqrt{5}
\end{aligned}
$$

Runge-Kutta del quarto ordine: regola 3/8

0	0	0	0	0
1/3	1/3	0	0	0
2/3	−1/3	1	0	0
1	1	−1	1	0
	1/8	3/8	3/8	1/8

30.4 Controllo adattivo del passo

Finora il passo h è stato tenuto costante. Nei codici reali non lo è quasi mai: dove la soluzione varia poco conviene un passo grande, dove varia rapidamente un passo piccolo. Per farlo serve una **stima dell'errore locale** che costi poco.

Coppie annidate

L'idea delle *coppie annidate* (*embedded pairs*) è di combinare gli **stessi** stadi K_i con due diversi vettori di pesi, $\boldsymbol{b}$ e $\widehat{\boldsymbol{b}}$, ottenendo due approssimazioni di ordine diverso:

$$
y_{k+1} = y_k + \sum_{i=1}^{s} b_i K_i, \qquad \widehat{y}_{k+1} = y_k + \sum_{i=1}^{s} \widehat{b}_i K_i
$$

La loro differenza

$$
\boxed{e_{k+1}^{\text{est}} = y_{k+1} - \widehat{y}_{k+1} = \sum_{i=1}^{s} \left(b_i - \widehat{b}_i\right) K_i} \tag{30.14}
$$

è una stima dell'errore locale che **non costa nessuna valutazione aggiuntiva** di f: è questo il punto. Il tableau si scrive con due righe sotto la linea:

$$\begin{array}{c|c} \boldsymbol{c} & \boldsymbol{A} \\ \hline & \boldsymbol{b}^T \\ \hline & \widehat{\boldsymbol{b}}^T \end{array}$$

Accettazione del passo e controllore

Per un sistema la stima si normalizza componente per componente con una tolleranza assoluta e una relativa:

$$\mathrm{err} = \left[\frac{1}{d} \sum_{j=1}^{d} \left(\frac{e_j^{\mathrm{est}}}{\mathrm{atol}_j + \mathrm{rtol}_j \max\left(|y_{k,j}|, |y_{k+1,j}|\right)} \right)^2 \right]^{1/2}$$

Il passo è **accettato** se $\mathrm{err} \le 1$, altrimenti è rifiutato e ripetuto. In entrambi i casi il passo successivo è

$$\boxed{h_{\mathrm{new}} = \eta\, h\, \mathrm{err}^{-1/q}} \tag{30.15}$$

dove q è l'ordine della stima 30.14 e $\eta \in (0,1)$ è un fattore di sicurezza (tipicamente 0.8–0.9). L'esponente $-1/q$ segue dall'ipotesi $e^{\mathrm{est}} \approx C h^q$: per avere $\mathrm{err} = 1$ occorre scalare h di $\mathrm{err}^{-1/q}$. In pratica il fattore si limita a un intervallo, ad esempio $[0.2,\ 5]$, per evitare oscillazioni del passo.

Esempio: la coppia di Bogacki-Shampine 3(2)

È il metodo usato da `ode23` di MATLAB e da `BS3` in molte librerie.

$$\begin{array}{c|cccc} 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ \frac{3}{4} & 0 & \frac{3}{4} & 0 & 0 \\ 1 & \frac{2}{9} & \frac{1}{3} & \frac{4}{9} & 0 \\ \hline & \frac{2}{9} & \frac{1}{3} & \frac{4}{9} & 0 \\ \hline & \frac{7}{24} & \frac{1}{4} & \frac{1}{3} & \frac{1}{8} \end{array} \tag{30.16}$$

Il primo vettore di pesi dà ordine 3, il secondo ordine 2, quindi $e^{\mathrm{est}} = \mathcal{O}(h^3)$ e nella 30.15 si usa $q = 3$.

> FSAL: *first same as last*
>
> Si osservi che l'ultima riga di $\boldsymbol{A}$ coincide con $\boldsymbol{b}^T$ e che $c_4 = 1$. Ne segue
>
> $$K_4 = h f\left(t_k + h,\ y_{k+1}\right) = h f\left(t_{k+1}, y_{k+1}\right)$$
>
> cioè l'ultimo stadio di un passo **accettato** è esattamente il primo stadio del passo successivo. Il metodo ha quattro stadi ma, dopo il primo passo, costa solo **tre** valutazioni di f per passo. La stessa proprietà la possiede la coppia di Dormand-Prince 5(4), che è il `ode45` di MATLAB.

Il controllo adattivo del passo è la ragione principale per cui i metodi espliciti moderni sono molto più efficienti di RK4 a passo fisso nei codici di uso generale: non perché siano più accurati a parità di passo, ma perché scelgono il passo da soli.

30.5 Metodi di Runge-Kutta Impliciti

Finora abbiamo discusso esclusivamente dei *metodi di Runge-Kutta espliciti*. Tuttavia, quando si affrontano equazioni *rigide*, è consigliabile utilizzare i *metodi di Runge-Kutta impliciti*.

Il *metodo di Runge-Kutta implicito* generico è un metodo a un passo caratterizzato dalla seguente rappresentazione:

$$
\begin{aligned}
y_{k+1} &= y_k + \sum_{i=1}^{s} b_i K_i \\
&\begin{cases}
K_1 = hf\left(t_k + hc_1,\, y_k + \displaystyle\sum_{j=1}^{s} a_{1j} K_j\right) \\
K_2 = hf\left(t_k + hc_2,\, y_k + \displaystyle\sum_{j=1}^{s} a_{2j} K_j\right) \\
\quad\vdots \\
K_s = hf\left(t_k + hc_s,\, y_k + \displaystyle\sum_{j=1}^{s} a_{sj} K_j\right)
\end{cases}
\end{aligned}
\tag{30.17}
$$

Il tableau per 30.17 sarà quindi della forma:

$$
\begin{array}{c|cccc}
c_1 & a_{11} & a_{12} & \dots & a_{1s} \\
c_2 & a_{21} & a_{22} & \dots & a_{2s} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
c_s & a_{s1} & a_{s2} & \dots & a_{ss} \\
\hline
 & b_1 & b_2 & \dots & b_s
\end{array}
$$

30.5.1 Soluzione degli stadi impliciti

La 30.17 non è una formula ma un **sistema non lineare**: gli s stadi $K_1, \dots, K_s$ compaiono a destra e a sinistra. Per un problema di dimensione d si tratta di un sistema di $s \cdot d$ equazioni da risolvere **a ogni passo**. È qui che i metodi dei capitoli precedenti tornano utili.

Raccogliendo gli stadi nel vettore $\boldsymbol{K} = (K_1, \dots, K_s)$, il sistema è

$$
\boldsymbol{G}(\boldsymbol{K}) = \boldsymbol{0}, \qquad G_i(\boldsymbol{K}) = K_i - hf\left(t_k + c_i h,\; y_k + \sum_{j=1}^{s} a_{ij} K_j\right) \tag{30.18}
$$

Iterazione di punto fisso

La scelta più semplice è riscrivere la 30.18 come iterazione di punto fisso (Capitolo 23):

$$
K_i^{(m+1)} = hf\left(t_k + c_i h,\; y_k + \sum_{j=1}^{s} a_{ij} K_j^{(m)}\right), \qquad m = 0, 1, 2, \dots
$$

partendo ad esempio da $K_i^{(0)} = hf(t_k, y_k)$. Se f è lipschitziana con costante L rispetto a y, la mappa è una contrazione non appena

$$\boxed{h\,L \max_i \sum_{j=1}^{s} |a_{ij}| < 1} \tag{30.19}$$

e in tal caso l'iterazione converge linearmente con fattore pari al membro sinistro della 30.19.

Perché il punto fisso non basta

La 30.19 impone $h \lesssim 1/L$. Ma i metodi impliciti servono proprio per i problemi **stiff** (Capitolo 31), dove L è enorme: la condizione di contrazione reintrodurrebbe esattamente il vincolo sul passo che si voleva evitare. Su un problema stiff il punto fisso vanifica l'implicitezza, e occorre il metodo di Newton.

Metodo di Newton

Applicando il metodo di Newton (Capitolo 20) alla 30.18 si itera

$$\boldsymbol{J_G}\big(\boldsymbol{K}^{(m)}\big)\,\Delta\boldsymbol{K}^{(m)} = -\boldsymbol{G}\big(\boldsymbol{K}^{(m)}\big), \qquad \boldsymbol{K}^{(m+1)} = \boldsymbol{K}^{(m)} + \Delta\boldsymbol{K}^{(m)}$$

dove il jacobiano ha struttura a blocchi $s \times s$, ciascun blocco di dimensione $d \times d$:

$$\boxed{\big[\boldsymbol{J_G}\big]_{ij} = \delta_{ij}\,\boldsymbol{I} - h\,a_{ij}\,\boldsymbol{J_f}\Big(t_k + c_j h,\ y_k + \sum_l a_{jl} K_l\Big)} \tag{30.20}$$

con $\boldsymbol{J_f} = \partial \boldsymbol{f}/\partial \boldsymbol{y}$ e δ_{ij} la delta di Kronecker. Il sistema lineare ad ogni iterazione si risolve con la fattorizzazione LU (Capitolo 1) oppure, se sd è grande e la matrice è sparsa, con un metodo iterativo (Capitolo 26).

A differenza del punto fisso, la convergenza di Newton è **quadratica** e non richiede alcuna condizione del tipo 30.19: il passo h resta libero, che è esattamente il motivo per cui si è scelto un metodo implicito.

Accorgimenti pratici

- **Newton semplificato**: si congela $\boldsymbol{J_f}$ in (t_k, y_k) e si riusa la stessa fattorizzazione per tutte le iterazioni e spesso per più passi consecutivi. Si perde la convergenza quadratica ma si risparmia il costo dominante, che è la fattorizzazione $\mathcal{O}((sd)^3)$.
- **Metodi DIRK**: se $\boldsymbol{A}$ è triangolare inferiore ($a_{ij} = 0$ per $j > i$), la 30.20 è triangolare a blocchi e gli stadi si risolvono **uno alla volta**, con s sistemi $d \times d$ invece di uno $sd \times sd$. È la ragione per cui i metodi DIRK sono preferiti nei codici di produzione.
- **Predittore**: un buon $\boldsymbol{K}^{(0)}$, ottenuto estrapolando dal passo precedente, riduce spesso le iterazioni a due o tre.

Metodo di Eulero Implicito

Il metodo di Eulero implicito rappresentato utilizzando un tableau:

$$\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array} \tag{30.21}$$

Metodo del Punto Medio Implicito

Il metodo del punto medio implicito è un'adattamento del metodo di Eulero esplicito, spesso chiamato metodo di Collatz. Si tratta di un metodo del secondo ordine caratterizzato da due stadi, che offre una maggiore precisione nella risoluzione delle equazioni differenziali ordinarie.

Metodo originale

$$y_{k+1} = y_k + hf\left(t_k + \frac{1}{2}h, \frac{y_k + y_{k+1}}{2}\right)$$

formulazione come Runge-Kutta

$$\begin{aligned} y_{k+1} &= y_k + K_1 \\ K_1 &= hf\left(t_k + \frac{1}{2}h, \frac{y_{k+1} + y_k}{2}\right) \\ &= hf\left(t_k + \frac{1}{2}h, y_k + \frac{y_{k+1} - y_k}{2}\right) \\ &= hf\left(t_k + \frac{1}{2}h, y_k + \frac{1}{2}K_1\right) \end{aligned}$$

per cui il tableau diventa

$$\begin{array}{c|c|c} 1/2 & 1/2 \\ \hline & 1 \end{array} \tag{30.22}$$

Metodo di Crank-Nicolson

Il metodo di Crank-Nicolson[6] corrisponde alla regola trapezoidale implicita.

Metodo originale

$$y_{k+1} = y_k + \frac{1}{2}h\left(f(t_k, y_k) + f(t_{k+1}, y_{k+1})\right)$$

formulazione come Runge-Kutta

$$\begin{aligned} y_{k+1} &= y_k + \frac{1}{2}K_1 + \frac{1}{2}K_2 \\ K_1 &= hf(t_k, y_k) \\ K_2 &= hf(t_k + h, y_{k+1}) \\ &= hf\left(t_k + h, y_k + \frac{1}{2}K_1 + \frac{1}{2}K_2\right) \end{aligned}$$

per cui il tableau diventa

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1/2 & 1/2 \\ \hline & 1/2 & 1/2 \end{array} \tag{30.23}$$

Metodi di collocazione

I tableau impliciti che seguono non sono stati tirati a sorte: nascono tutti da una sola idea, la *collocazione*, che mette insieme l'interpolazione (Capitolo 7) e la quadratura (Capitolo 10).

Si fissano s nodi distinti $0 \le c_1 < \cdots < c_s \le 1$ e si cerca un **polinomio** $u(t)$ di grado $\le s$ tale che

$$u(t_k) = y_k, \qquad u'\big(t_k + c_i h\big) = f\Big(t_k + c_i h,\ u(t_k + c_i h)\Big), \quad i = 1, \dots, s \tag{30.24}$$

cioè che soddisfi l'equazione differenziale **esattamente** negli s nodi. Si pone poi $y_{k+1} = u(t_{k+1})$.

[6] John Crank (1916-2006) e Phyllis Nicolson (1917-1968)

Theorem 30.1 — Collocazione e Runge-Kutta. Il metodo di collocazione 30.24 coincide con il metodo di Runge-Kutta implicito di tableau

$$a_{ij} = \int_0^{c_i} \ell_j(\sigma)\,\mathrm{d}\sigma, \qquad b_j = \int_0^1 \ell_j(\sigma)\,\mathrm{d}\sigma \tag{30.25}$$

dove ℓ_j sono i polinomi elementari di Lagrange sui nodi $c_1, \dots, c_s$.

Proof. Il polinomio u' ha grado $\leq s-1$ ed è noto negli s nodi, dove vale K_i/h con $K_i = hf(t_k + c_i h, u(t_k + c_i h))$. Per l'unicità del polinomio interpolante (Capitolo 6),

$$u'(t_k + \sigma h) = \frac{1}{h} \sum_{j=1}^{s} K_j\, \ell_j(\sigma)$$

Integrando da 0 a c_i e poi da 0 a 1 si ottengono rispettivamente

$$u(t_k + c_i h) = y_k + \sum_{j=1}^{s} K_j \int_0^{c_i} \ell_j(\sigma)\,\mathrm{d}\sigma, \qquad u(t_{k+1}) = y_k + \sum_{j=1}^{s} K_j \int_0^{1} \ell_j(\sigma)\,\mathrm{d}\sigma$$

che sono esattamente le equazioni degli stadi e del passo di un metodo di Runge-Kutta con i coefficienti 30.25. □ ■

La 30.25 spiega l'intera tabella dei metodi impliciti che segue, e spiega anche il loro ordine: poiché b_j sono i pesi della formula di quadratura interpolatoria sui nodi c_j, il metodo di collocazione **eredita l'ordine della quadratura**.

Table 30.1: Corrispondenza fra formule di quadratura e metodi di collocazione.

Nodi c_i	Quadratura	Metodo RK	Stadi	Ordine
zeri di Legendre su $(0,1)$	Gauss-Legendre	Gauss-Legendre	s	$2s$
estremi inclusi	Gauss-Lobatto	Lobatto IIIA	s	$2s-2$
un estremo incluso	Gauss-Radau	Radau IIA	s	$2s-1$
$c_1 = 1$	—	Eulero implicito	1	1
$c_1 = \frac{1}{2}$	punto medio	punto medio implicito	1	2

Il salto da ordine s (quanto basterebbe a un metodo generico a s stadi) a ordine $2s$ è lo stesso fenomeno della quadratura gaussiana: la scelta ottimale dei nodi raddoppia l'ordine. Per questo i metodi di Gauss-Legendre sono i metodi impliciti di ordine massimo a parità di stadi.

Metodi di Gauss-Legendre

Questi metodi si basano sui punti della quadratura di Gauss-Legendre[7].

[7] Carl Friedrich Gauss (1777-1855) e Adrien-Marie Legendre (1752-1833)

Metodo di Gauss-Legendre del quarto ordine

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \\ & \frac{1}{2}+\frac{\sqrt{3}}{2} & \frac{1}{2}-\frac{\sqrt{3}}{2} \end{array} \tag{30.26}$$

Metodo di Gauss-Legendre del sesto ordine

$$\begin{array}{c|ccc} \frac{1}{2}-\frac{\sqrt{15}}{10} & \frac{5}{36} & \frac{2}{9}-\frac{\sqrt{15}}{15} & \frac{5}{36}-\frac{\sqrt{15}}{30} \\ \frac{1}{2} & \frac{5}{36}+\frac{\sqrt{15}}{24} & \frac{2}{9} & \frac{5}{36}-\frac{\sqrt{15}}{24} \\ \frac{1}{2}+\frac{\sqrt{15}}{10} & \frac{5}{36}+\frac{\sqrt{15}}{30} & \frac{2}{9}+\frac{\sqrt{15}}{15} & \frac{5}{36} \\ \hline & \frac{5}{18} & \frac{4}{9} & \frac{5}{18} \\ & -\frac{5}{6} & \frac{8}{3} & -\frac{5}{6} \end{array} \tag{30.27}$$

Metodi di Lobatto

Metodo di Lobatto IIIA del quarto ordine

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1/2 & 5/24 & 1/3 & -1/24 \\ 1 & 1/6 & 2/3 & 1/6 \\ \hline & 1/6 & 2/3 & 1/6 \end{array} \tag{30.28}$$

Lobatto IIIB del quarto ordine

$$\begin{array}{c|ccc} 0 & 1/6 & -1/6 & 0 \\ 1/2 & 1/6 & 1/3 & 0 \\ 1 & 1/6 & 5/6 & 0 \\ \hline & 1/6 & 2/3 & 1/6 \end{array} \tag{30.29}$$

Lobatto IIIC del quarto ordine

$$\begin{array}{c|ccc} 0 & 1/6 & -1/3 & 1/6 \\ 1/2 & 1/6 & 5/12 & -1/12 \\ 1 & 1/6 & 2/3 & 1/6 \\ \hline & 1/6 & 2/3 & 1/6 \end{array} \tag{30.30}$$

Lobatto IIIC* del quarto ordine

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ 1/2 & 1/4 & 1/4 & 0 \\ 1 & 0 & 1 & 0 \\ \hline & 1/6 & 2/3 & 1/6 \end{array} \tag{30.31}$$

Metodi di Radau

Metodo di Radau IA del quinto ordine

$$\begin{array}{c|ccc} 0 & \frac{1}{9} & \frac{-1-\sqrt{6}}{18} & \frac{-1+\sqrt{6}}{18} \\ \frac{3}{5}-\frac{\sqrt{6}}{10} & \frac{1}{9} & \frac{11}{45}+\frac{7\sqrt{6}}{360} & \frac{11}{45}-\frac{43\sqrt{6}}{360} \\ \frac{3}{5}+\frac{\sqrt{6}}{10} & \frac{1}{9} & \frac{11}{45}+\frac{43\sqrt{6}}{360} & \frac{11}{45}-\frac{7\sqrt{6}}{360} \\ \hline & \frac{1}{9} & \frac{4}{9}+\frac{\sqrt{6}}{36} & \frac{4}{9}-\frac{\sqrt{6}}{36} \end{array} \tag{30.32}$$

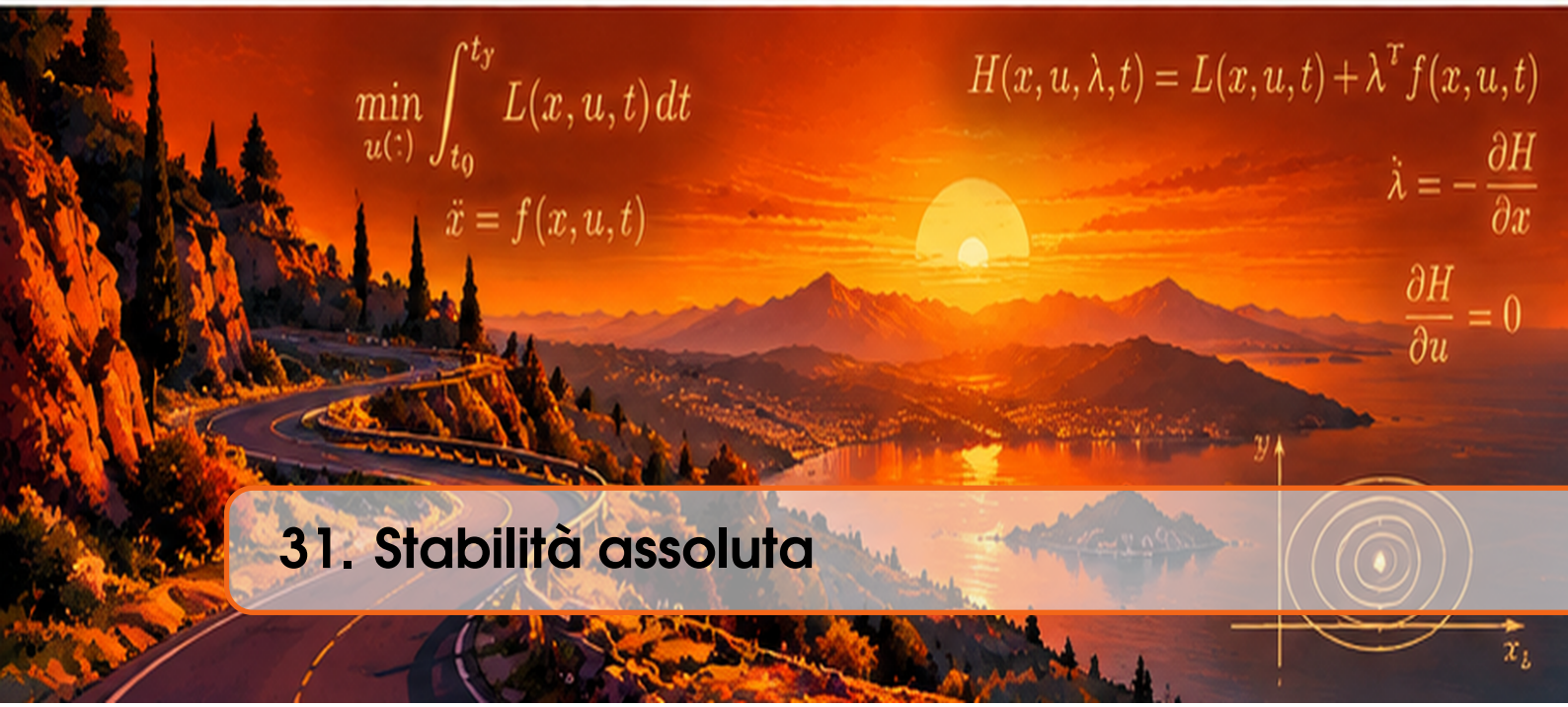
Radau II del quinto ordine

$$\begin{array}{c|ccc} \frac{2}{5}-\frac{\sqrt{6}}{10} & \frac{11}{45}-\frac{7\sqrt{6}}{360} & \frac{37}{225}-\frac{169\sqrt{6}}{1800} & -\frac{2}{225}+\frac{\sqrt{6}}{75} \\ \frac{2}{5}+\frac{\sqrt{6}}{10} & \frac{37}{225}+\frac{169\sqrt{6}}{1800} & \frac{11}{45}+\frac{7\sqrt{6}}{360} & -\frac{2}{225}-\frac{\sqrt{6}}{75} \\ 1 & \frac{4}{9}-\frac{\sqrt{6}}{36} & \frac{4}{9}+\frac{\sqrt{6}}{36} & \frac{1}{9} \\ \hline & \frac{4}{9}-\frac{\sqrt{6}}{36} & \frac{4}{9}+\frac{\sqrt{6}}{36} & \frac{1}{9} \end{array} \tag{30.33}$$

In sintesi

- I metodi di Runge-Kutta ottengono ordine alto valutando f in più **stadi** interni, senza mai derivare f; il tableau di Butcher $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ li individua completamente.

- Le **condizioni d'ordine** si ricavano confrontando lo sviluppo di Taylor del metodo con quello della soluzione esatta; il loro numero esplode con p (17 condizioni per $p = 5$, 1205 per $p = 10$).
- Il numero di stadi necessari supera l'ordine da $p \geq 5$ in poi: non esiste un RK esplicito a 5 stadi di ordine 5.
- Le **coppie annidate** (Sezione 30.4) danno una stima dell'errore locale a costo nullo e permettono il controllo automatico del passo; la proprietà FSAL riduce ulteriormente le valutazioni.
- I metodi impliciti richiedono di risolvere un sistema non lineare di dimensione sd a ogni passo (Sezione 30.5.1): punto fisso sui problemi non stiff, Newton su quelli stiff.
- Gauss-Legendre, Lobatto e Radau non sono tableau arbitrari ma metodi di **collocazione** (Teorema 30.1): ereditano nodi, pesi e ordine dalle corrispondenti formule di quadratura.



31. Stabilità assoluta

L'accuratezza da sola non basta a decidere se un metodo numerico per ODE sia utilizzabile. Per problemi che contengono modi che decadono rapidamente un metodo può essere di ordine elevato e ciò nonostante richiedere un passo h piccolissimo solo per **non esplodere**. Questo capitolo sviluppa il concetto di *stabilità assoluta* e confronta le regioni di stabilità dei metodi visti nei capitoli precedenti.

Tre nozioni da non confondere

- **Buona posizione** (27.7): proprietà del *problema continuo*.
- **Convergenza** (Lemma 28.1): comportamento dell'errore per $h \to 0$ su un intervallo **finito**.
- **Stabilità assoluta**: comportamento della soluzione numerica per $n \to \infty$ a passo h **fissato**.

Sono indipendenti: un metodo convergente può essere inutilizzabile su un problema stiff, e un metodo stabile può essere del tutto impreciso.

31.1 L'equazione test di Dahlquist

Il modello di riferimento è l'equazione test scalare

$$y'(t) = \lambda\, y(t), \qquad \lambda \in \mathbb{C} \tag{31.1}$$

la cui soluzione esatta avanza di un passo secondo

$$y(t_{n+1}) = e^{q}\, y(t_n), \qquad q = h\lambda$$

Se $\mathrm{Re}(\lambda) < 0$ la soluzione esatta **decade**. È ragionevole chiedere che un buon metodo numerico riproduca questo decadimento per il valore $q = h\lambda$ scelto.

Perché proprio questa equazione? Perché, come vedremo nella Sezione 31.5, la linearizzazione di un sistema attorno ad una traiettoria e la diagonalizzazione della matrice jacobiana riconducono il caso vettoriale ad una collezione di equazioni scalari della forma 31.1, una per ogni autovalore.

31.2 Funzione e regione di stabilità

Applicando un metodo a un passo alla 31.1 si ottiene sempre una relazione della forma

$$y_{n+1} = R(q)\, y_n \qquad \Longrightarrow \qquad y_n = R(q)^n\, y_0$$

dove $R(q)$ è detta *funzione di stabilità* del metodo. La soluzione numerica resta limitata per $n \to \infty$ se e solo se $|R(q)| \le 1$.

Definition 31.1 — Regione di stabilità assoluta. Dato un metodo a un passo con funzione di stabilità $R(q)$, la sua *regione di stabilità assoluta* è

$$\mathcal{S} = \left\{ q \in \mathbb{C} \ : \ |R(q)| \le 1 \right\} \tag{31.2}$$

La frontiera di $\mathcal{S}$ è descritta da $|R(q)| = 1$. Per ogni metodo **consistente** vale

$$R(q) = 1 + q + \mathcal{O}(q^2)$$

cioè R riproduce localmente l'esponenziale e^q vicino all'origine; più precisamente, un metodo di ordine p ha

$$R(q) = \sum_{k=0}^{p} \frac{q^k}{k!} + \mathcal{O}(q^{p+1})$$

Stabilità dal tableau di Butcher

Per un metodo di Runge-Kutta[1] con tableau $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$ a s stadi (Capitolo 30) applicato alla 31.1 si ha

$$\boldsymbol{Z} = y_n \mathbf{1} + q\boldsymbol{A}\boldsymbol{Z} \qquad \Longrightarrow \qquad \boldsymbol{Z} = y_n (\boldsymbol{I} - q\boldsymbol{A})^{-1} \mathbf{1}$$

e quindi

$$\boxed{R(q) = 1 + q\, \boldsymbol{b}^T (\boldsymbol{I} - q\boldsymbol{A})^{-1} \mathbf{1}} \tag{31.3}$$

Se il metodo è **esplicito**, $\boldsymbol{A}$ è strettamente triangolare inferiore, quindi $\boldsymbol{A}^s = \mathbf{0}$ e la serie di Neumann[2] $(\boldsymbol{I} - q\boldsymbol{A})^{-1} = \sum_{k \ge 0} q^k \boldsymbol{A}^k$ è finita: $R(q)$ è un **polinomio** di grado $\le s$. Se il metodo è implicito, $R(q)$ è una **funzione razionale**.
Questa è l'osservazione decisiva: un polinomio non costante è illimitato, quindi $|R(q)| \le 1$ non può valere su un insieme illimitato. Ne segue che

> **nessun metodo di Runge-Kutta esplicito può avere regione di stabilità illimitata.**

31.3 I metodi fondamentali

31.3.1 Eulero esplicito

$$y_{n+1} = y_n + h\lambda y_n = (1 + q) y_n \qquad \Longrightarrow \qquad \boxed{R_{\mathrm{EE}}(q) = 1 + q}$$

La condizione $|1 + q| \le 1$ descrive il **disco chiuso di raggio 1 centrato in** -1. Sull'asse reale negativo

$$-2 \le q \le 0$$

[2] Carl David Tolmé Runge (1856-1927) e Martin Wilhelm Kutta (1867-1944)
[2] Carl Gottfried Neumann (1832-1925)

quindi per λ reale negativo il metodo richiede

$$h \leq \frac{2}{|\lambda|} \tag{31.4}$$

Si tratta di un vincolo di **pura stabilità**: può essere enormemente più severo del passo necessario a risolvere accuratamente la parte lenta della soluzione.

31.3.2 Runge-Kutta espliciti del secondo ordine

Tutti i metodi RK espliciti a due stadi di ordine due (punto medio, Heun[3], Ralston[4]) hanno la **stessa** funzione di stabilità

$$\boxed{R_{\mathrm{RK2}}(q) = 1 + q + \frac{q^2}{2}}$$

I coefficienti dei tableau sono diversi, ma sull'equazione test i metodi coincidono. L'intervallo di stabilità sull'asse reale negativo è ancora $[-2, 0]$: **aumentare l'ordine da 1 a 2 non allarga il passo ammissibile** per un modo reale negativo.

31.3.3 Runge-Kutta classico del quarto ordine

$$\boxed{R_{\mathrm{RK4}}(q) = 1 + q + \frac{q^2}{2} + \frac{q^3}{6} + \frac{q^4}{24}}$$

cioè il polinomio di Taylor[5] di grado 4 di e^q. L'intervallo di stabilità reale è circa

$$-2.7853 \lesssim q \leq 0$$

un po' più largo di Eulero[6], ma pur sempre **limitato**.

31.3.4 Eulero implicito

$$y_{n+1} = y_n + q y_{n+1} \qquad \Longrightarrow \qquad \boxed{R_{\mathrm{EI}}(q) = \frac{1}{1-q}}$$

La condizione $|R| \leq 1$ diventa $|1 - q| \geq 1$, cioè l'**esterno** del disco di raggio 1 centrato in $+1$. Tale insieme contiene tutto il semipiano sinistro chiuso. Inoltre

$$R_{\mathrm{EI}}(q) \longrightarrow 0 \qquad \text{per } |q| \to \infty,\ \mathrm{Re}(q) < 0$$

quindi i modi molto rigidi vengono fortemente **smorzati**.

31.3.5 Metodo dei trapezi (Crank-Nicolson)

$$y_{n+1} = y_n + \frac{q}{2}\left(y_n + y_{n+1}\right) \qquad \Longrightarrow \qquad \boxed{R_{\mathrm{CN}}(q) = \frac{1 + q/2}{1 - q/2}}$$

Si verifica subito che $|R_{\mathrm{CN}}(q)| \leq 1$ se e solo se $\mathrm{Re}(q) \leq 0$: la regione di stabilità è **esattamente** il semipiano sinistro. Però

$$R_{\mathrm{CN}}(q) \longrightarrow -1 \qquad \text{per } q \to -\infty$$

I modi rigidi **non** vengono smorzati: sopravvivono numericamente come oscillazioni di segno alterno. È questa la differenza pratica fondamentale fra trapezi ed Eulero implicito.

[3] Karl Heun (1859-1929)
[4] Anthony Ralston (1930-2019)
[5] Brook Taylor (1685-1731)
[6] Leonhard Euler (1707-1783)

31.4 A-stabilità e L-stabilità

Definition 31.2 — Metodo A-stabile. Un metodo a un passo si dice *A-stabile* se

$$\left\{ q \in \mathbb{C} \ : \ \mathrm{Re}(q) \leq 0 \right\} \subseteq S$$

cioè se ogni modo stabile dell'equazione test resta numericamente stabile **per ogni** passo $h > 0$.

Definition 31.3 — Metodo L-stabile. Un metodo A-stabile si dice *L-stabile* se in più

$$R(q) \longrightarrow 0 \qquad \text{per } |q| \to \infty \text{ nel semipiano sinistro}$$

L'A-stabilità **impedisce la crescita** dei modi rigidi stabili; la L-stabilità li **sopprime**. Dalla discussione precedente:

- Eulero implicito è A-stabile **e** L-stabile;
- il metodo dei trapezi è A-stabile ma **non** L-stabile;
- nessun metodo di Runge-Kutta esplicito è A-stabile (31.3).

Barriera di Dahlquist

Si può dimostrare (seconda barriera di Dahlquist[7]) che un metodo lineare multipasso A-stabile ha ordine al più 2, e che fra questi il metodo dei trapezi è quello con la costante d'errore minima. Per i metodi di Runge-Kutta **impliciti** invece non c'è barriera: i metodi di Gauss-Legendre[8] (Capitolo 30) sono A-stabili a ogni ordine, i metodi di Radau[9] IIA sono L-stabili. La dimostrazione si trova in [21:V].

31.5 Dal caso scalare ai sistemi

Si consideri il sistema lineare

$$\boldsymbol{y}' = \boldsymbol{A}\boldsymbol{y}$$

Per un metodo di Runge-Kutta la funzione di stabilità si estende formalmente all'argomento matriciale:

$$\boldsymbol{y}_{n+1} = R(h\boldsymbol{A})\, \boldsymbol{y}_n$$

Se $\boldsymbol{A}$ è diagonalizzabile, $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}$, allora

$$R(h\boldsymbol{A}) = \boldsymbol{V}\, R(h\boldsymbol{\Lambda})\, \boldsymbol{V}^{-1}$$

e poiché $R(h\boldsymbol{\Lambda})$ è diagonale, il problema si disaccoppia in tante equazioni test scalari quanti sono gli autovalori. La condizione pratica è quindi

$$\boxed{h\lambda_j \in S \quad \text{per ogni autovalore } \lambda_j \text{ di } \boldsymbol{A}} \tag{31.5}$$

[9] Germund Dahlquist (1925-2005)
[9] Carl Friedrich Gauss (1777-1855) e Adrien-Marie Legendre (1752-1833)
[9] Jean Charles Rodolphe Radau (1835-1911)

Limiti del criterio spettrale

La 31.5 è esatta per matrici **normali**. Per matrici fortemente **non normali** gli autovalori da soli possono non bastare: la crescita transitoria è governata dal numero di condizionamento di $\boldsymbol{V}$ e non dallo spettro. I diagrammi di stabilità vanno quindi letti come diagnostica, non come teoria completa, soprattutto per problemi non lineari.

31.5.0.1 Problemi stiff

Si consideri

$$\boldsymbol{y}' = \boldsymbol{A}\boldsymbol{y}, \qquad \boldsymbol{A} = \begin{bmatrix} -1 & 0 \\ 0 & -1000 \end{bmatrix}$$

I due modi decadono come e^{-t} ed e^{-1000t}. Dopo un tempo brevissimo il secondo modo è numericamente nullo e la soluzione è governata interamente dal primo: per **accuratezza** basterebbe $h \approx 10^{-1}$. Ma Eulero esplicito, per la 31.4, richiede

$$h \leq \frac{2}{1000} = 2 \cdot 10^{-3}$$

per non esplodere, cioè un passo **500 volte più piccolo** di quello richiesto dall'accuratezza, e lo richiede per tutta la durata dell'integrazione. Un problema con questa caratteristica si dice *stiff*; una misura grossolana è il rapporto

$$\frac{\max_j |\mathrm{Re}(\lambda_j)|}{\min_j |\mathrm{Re}(\lambda_j)|}$$

detto *rapporto di stiffness* (qui vale 1000). Per questi problemi i metodi impliciti A-stabili non sono un lusso ma una necessità.

31.6 Diagrammi di stabilità

Le regioni che seguono sono calcolate direttamente dalla definizione 31.2, tracciando la curva di livello $|R(q)| = 1$.

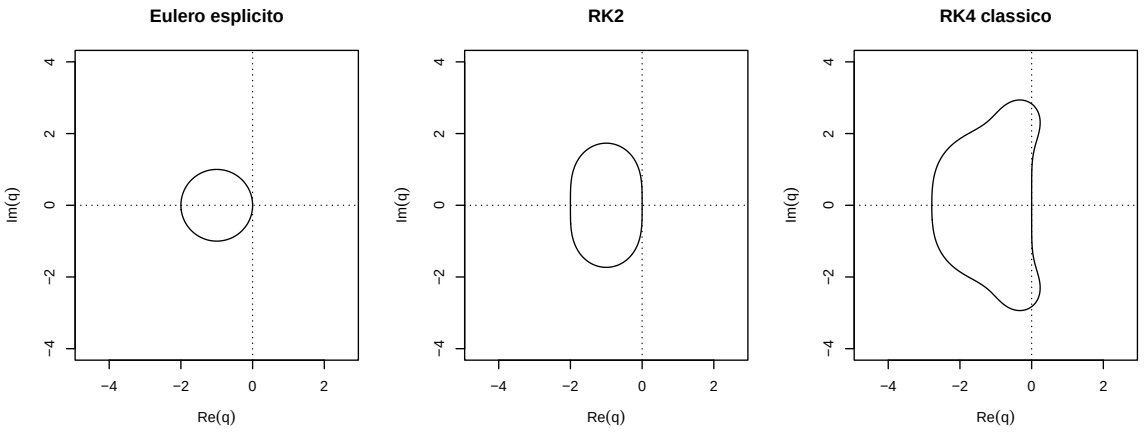


Figure 31.1: Regioni di stabilità assoluta di Eulero esplicito, RK2 e RK4 classico. La regione stabile è l'interno della curva.

Le regioni sono tutte **limitate**: aumentare l'ordine ne cambia forma ed estensione, ma non produce A-stabilità.

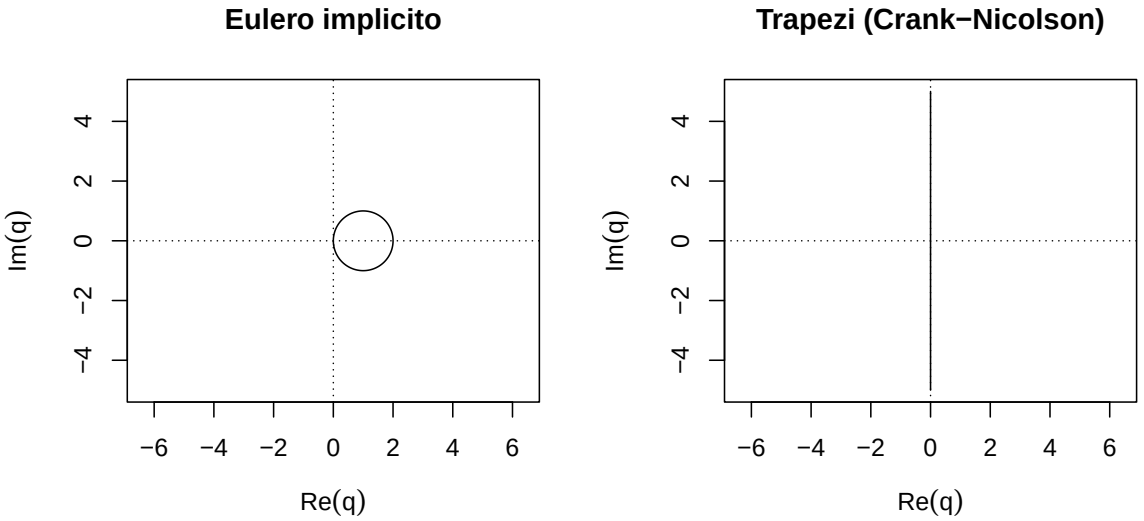


Figure 31.2: Frontiere di stabilità di Eulero implicito e del metodo dei trapezi. Entrambi contengono il semipiano sinistro; solo Eulero implicito è L-stabile.

> **Come si legge il disegno**
>
> Per Eulero implicito la frontiera è la circonferenza $|1 - q| = 1$, ma la parte **stabile è l'esterno** del cerchio, non l'interno. Per il metodo dei trapezi la frontiera è l'asse immaginario e la parte stabile è tutto il semipiano sinistro. Una curva di livello, da sola, non dice da che parte si sta: va sempre letta insieme alla definizione 31.2.

Il grafico seguente mostra il confronto fra $|R(q)|$ ed $|e^q| = e^q$ sull'asse reale negativo: si vede dove ogni metodo perde il contatto con la soluzione esatta.

31.7 Stabilità e accuratezza

Il fatto che q appartenga a $\mathcal{S}$ garantisce solo che la propagazione numerica resti limitata: **non** dice che l'approssimazione sia accurata. Eulero implicito è stabile per q negativo arbitrariamente grande, ma

$$R_{\mathrm{EI}}(q) = \frac{1}{1 - q}$$

approssima male e^q a passo grossolano. La scelta pratica di h deve quindi soddisfare **due** condizioni separate:

1. **stabilità**: gli autovalori scalati $h\lambda_j$ devono stare in $\mathcal{S}$;
2. **accuratezza**: l'errore globale deve rispettare la tolleranza richiesta.

Per problemi non stiff è di norma l'accuratezza a determinare h; per problemi stiff un metodo esplicito è invece limitato dalla stabilità.

31.7.1 Problemi oscillanti e asse immaginario

Per problemi quasi conservativi gli autovalori stanno vicino all'asse immaginario e conta il comportamento di $R(i\omega h)$. Si distinguono due errori:

- errore di **dissipazione**: $|R(i\omega h)| \neq 1$;
- errore di **dispersione** (o di fase): $\arg R(i\omega h) \neq \omega h$.

Un metodo può quindi essere assolutamente stabile e riprodurre comunque male le oscillazioni. Si noti che Eulero esplicito ha $|R(i\theta)| = \sqrt{1 + \theta^2} > 1$: **nessun** tratto dell'asse immaginario, tranne

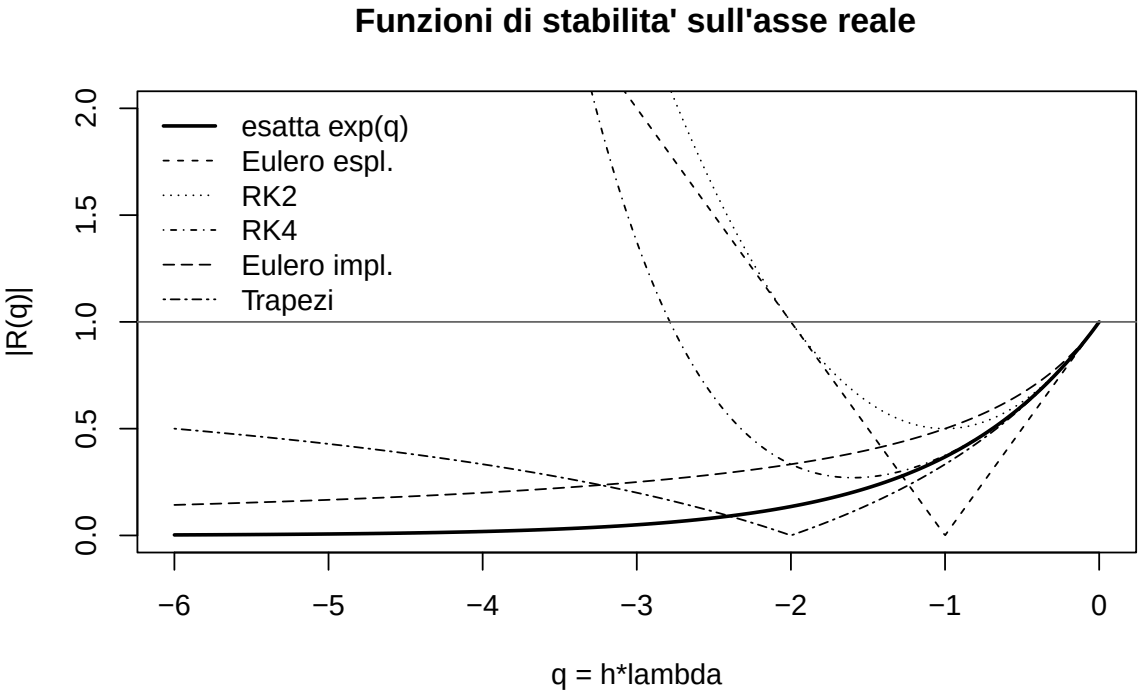


Figure 31.3: Modulo della funzione di stabilità sull'asse reale negativo confrontato con la soluzione esatta e^q. La soglia $|R| = 1$ delimita la stabilità.

l'origine, appartiene alla sua regione di stabilità, per cui esso è inutilizzabile sui problemi puramente oscillanti.

31.7.2 Procedura pratica

1. Linearizzare localmente la ODE: $\boldsymbol{y}' \approx \boldsymbol{J_f}(t, \boldsymbol{y})\,\boldsymbol{y} + \cdots$
2. Stimare gli autovalori λ_j dello jacobiano $\boldsymbol{J_f}$.
3. Scalarli con il passo candidato: $q_j = h\lambda_j$.
4. Verificare che i q_j stiano in $\mathcal{S}$.
5. Imporre **separatamente** la tolleranza di accuratezza.

31.8 Quadro riassuntivo

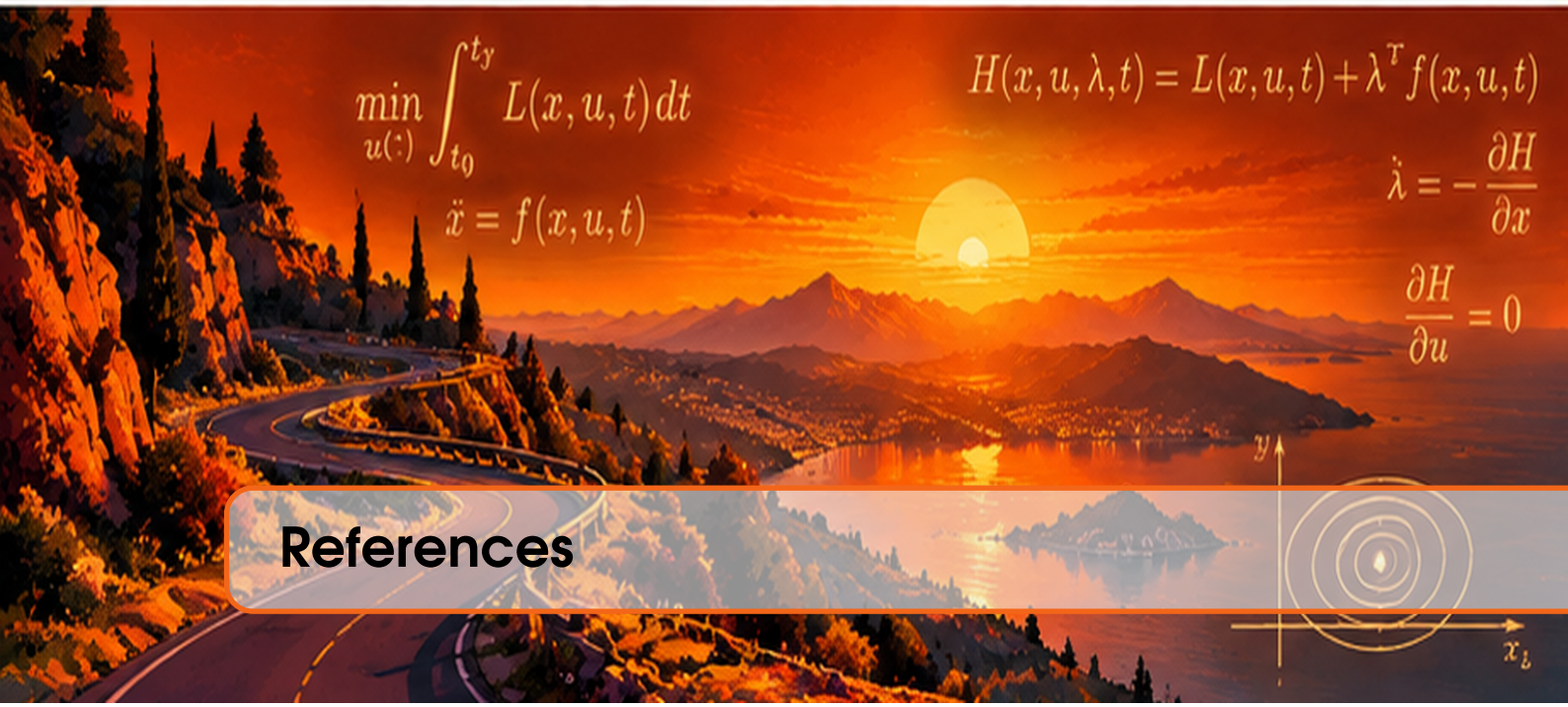
Table 31.1: Confronto fra i metodi a un passo studiati.

Metodo	Ordine	Tipo	$R(q)$	Stabilità su $\mathbb{R}^-$	A-stab.	L-stab.
Eulero esplicito	1	esplicito	$1 + q$	$[-2, 0]$	no	no
RK2 (Heun, punto medio, Ralston)	2	esplicito	$1 + q + \frac{q^2}{2}$	$[-2, 0]$	no	no
RK4 classico	4	esplicito	$\sum_{k=0}^{4} \frac{q^k}{k!}$	$\approx [-2.7853,\ 0]$	no	no

Metodo	Ordine	Tipo	$R(q)$	Stabilità su $\mathbb{R}^-$	A-stab.	L-stab.
Eulero implicito	1	implicito	$\dfrac{1}{1-q}$	$(-\infty, 0]$	**sì**	**sì**
Trapezi (Crank-Nicolson)	2	implicito	$\dfrac{1+q/2}{1-q/2}$	$(-\infty, 0]$	**sì**	no
Punto medio implicito	2	implicito	$\dfrac{1+q/2}{1-q/2}$	$(-\infty, 0]$	**sì**	no
Gauss-Legendre (s stadi)	$2s$	implicito	Padé (s, s)	$(-\infty, 0]$	**sì**	no
Radau IIA (s stadi)	$2s-1$	implicito	Padé $(s-1, s)$	$(-\infty, 0]$	**sì**	**sì**

I punti essenziali del capitolo:

- le regioni di stabilità dei metodi di Runge-Kutta **espliciti** sono sempre limitate, perché R è un polinomio;
- un metodo A-stabile contiene tutto il semipiano sinistro, e deve necessariamente essere implicito;
- la L-stabilità aggiunge lo smorzamento dei modi molto rigidi;
- il metodo dei trapezi è A-stabile ma non L-stabile;
- su un problema stiff il passo di un metodo esplicito è dettato dalla stabilità, non dall'accuratezza;
- stabilità e accuratezza sono requisiti **indipendenti**.



References

[1] 2025. NIST Digital Library of Mathematical Functions. Recuperato da https://dlmf.nist.gov/

[2] Milton Abramowitz e Irene A. Stegun (A c. di). 1964. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. National Bureau of Standards, Washington, D.C. Recuperato da https://personal.math.ubc.ca/~cbm/aands/

[3] G. E. Alefeld, F. A. Potra, e Yixun Shi. 1995. Algorithm 748: Enclosing Zeros of Continuous Functions. *ACM Transactions on Mathematical Software* 21, 3 (1995), 327–344. https://doi.org/10.1145/210089.210111

[4] Bernhard Beckermann. 2000. The condition number of real Vandermonde, Krylov and positive definite Hankel matrices. *Numerische Mathematik* 85, 4 (2000), 553–577. https://doi.org/10.1007/PL00005392

[5] A. Berman e R. J. Plemmons. 1994. *Nonnegative Matrices in the Mathematical Sciences*. SIAM.

[6] Roberto Bevilacqua, Dario Bini, Milvio Capovani, e Ornella Menchi. 1992. *Metodi Numerici*. Zanichelli.

[7] Åke Björck. 1996. *Numerical Methods for Least Squares Problems*. SIAM.

[8] Richard P. Brent. 1973. *Algorithms for Minimization without Derivatives*. Prentice-Hall, Englewood Cliffs, NJ.

[9] J. C. P. Bus e T. J. Dekker. 1975. Two Efficient Algorithms with Guaranteed Convergence for Finding a Zero of a Function. *ACM Transactions on Mathematical Software* 1, 4 (1975), 330–345. https://doi.org/10.1145/355656.355659

[10] J. C. Butcher. 1987. *The Numerical analysis of ordinary differential equations : Runge-Kutta and general linear methods* (First ed.). Wiley, Chichester.

[11] Frank Cameron. 2006. A Matlab package for automatically generating Runge-Kutta trees, order conditions, and truncation error coefficients. *ACM Trans. Math. Softw.* 32, 2 (giugno 2006), 274–298. https://doi.org/10.1145/1141885.1141892

[12] Tirupathi R. Chandrupatla. 1997. A New Hybrid Quadratic/Bisection Algorithm for Finding the Zero of a Nonlinear Function without Using Derivatives. *Advances in Engineering Software* 28, 3 (1997), 145–149. https://doi.org/10.1016/S0965-9978(96)00051-8

[13] Theodore S. Chihara. 1978. *An Introduction to Orthogonal Polynomials*. Gordon; Breach, New York.

[14] T. J. Dekker. 1969. Finding a Zero by Means of Successive Linear Interpolation. In *Constructive Aspects of the Fundamental Theorem of Algebra*, B. Dejon e P. Henrici (a c. di). Wiley-Interscience, London, 37–48.

[15] Carl Friedrich Gauss. 1814. Methodus nova integralium valores per approximationem inveniendi. *Commentationes Societatis Regiae Scientiarum Gottingensis Recentiores* 3, (1814).

[16] Gene H. Golub e Charles F. Van Loan. 1989. *Matrix Computations*. Johns Hopkins Univ. Press.

[17] Gene H. Golub e John H. Welsch. 1969. Calculation of Gauss quadrature rules. *Mathematics of Computation* 23, 106 (1969), 221–230. https://doi.org/10.2307/2004418

[18] L. A. Hageman e D. M. Young. 1981. *Applied Iterative Methods*. Academic Press.

[19] W. W. Hager. 1988. *Applied Numerical Linear Algebra*. Prentice Hall.

[20] E. Hairer, S. P. Nørsett, e G. Wanner. 1993. *Solving Ordinary Differential Equations I: Nonstiff Problems* (2nd ed.). Springer.

[21] Ernst Hairer e Gerhard Wanner. 1996. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems* (2ª ed.). Springer-Verlag, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-05221-7

[22] Nicholas J. Higham. 2002. *Accuracy and Stability of Numerical Algorithms* (2nd ed.). SIAM, Philadelphia, PA.

[23] R. A. Horn e C. R. Johnson. 2012. *Matrix Analysis* (2nd ed.). Cambridge University Press.

[24] Alston S. Householder. 1970. *The Numerical Treatment of a Single Nonlinear Equation*. McGraw-Hill, New York.

[25] Frank P. Incropera, David P. DeWitt, Theodore L. Bergman, e Adrienne S. Lavine. 2006. *Fundamentals of Heat and Mass Transfer* (6th ed.). John Wiley & Sons.

[26] Carl Gustav Jacob Jacobi. 1826. Ueber Gauß' neue Methode, die Werthe der Integrale näherungsweise zu finden. *Journal für die reine und angewandte Mathematik* 1, (1826), 301–308. Recuperato da https://eudml.org/doc/183033

[27] S. Lang e I. Mencattini. 2014. *Algebra lineare*. Bollati Boringhieri.

[28] Adrien-Marie Legendre. 1785. Recherches sur l'attraction des sphéroïdes homogènes. *Mémoires de Mathématiques et de Physique, présentés à l'Académie Royale des Sciences par divers savans* 10, (1785), 411–435.

[29] I. F. D. Oliveira e R. H. C. Takahashi. 2021. An Enhancement of the Bisection Method Average Performance Preserving Minmax Optimality. *ACM Transactions on Mathematical Software* 47, 1 (2021), 1–24. https://doi.org/10.1145/3423597

[30] J. M. Ortega. 1987. *Matrix Theory (A Second Course)*. Plenum Press.

[31] J. M. Ortega e W. C. Rheinboldt. 1970. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press.

[32] William H. Press, Saul A. Teukolsky, William T. Vetterling, e Brian P. Flannery. 2007. *Numerical Recipes: The Art of Scientific Computing* (3ª ed.). Cambridge University Press, Cambridge.

[33] A. Quarteroni, F. Saleri, e P. Gervasio. 2017. *Calcolo Scientifico: Esercizi e problemi risolti con MATLAB e Octave*. Springer Milan.

[34] C. Ridders. 1979. A New Algorithm for Computing a Single Root of a Real Continuous Function. *IEEE Transactions on Circuits and Systems* 26, 11 (1979), 979–980. https://doi.org/10.1109/TCS.1979.1084580

[35] Adel S. Sedra, Kenneth C. Smith, Tony Chan Carusone, e Vincent Gaudet. 2020. *Microelectronic Circuits* (8th ed.). Oxford University Press.

[36] J. M. Smith, H. C. Van Ness, Michael M. Abbott, e M. T. Swihart. 2005. *Introduction to Chemical Engineering Thermodynamics* (7th ed.). McGraw-Hill Education.

[37] Thomas Joannes Stieltjes. 1894. Recherches sur les fractions continues. *Annales de la Faculté des sciences de Toulouse, 1re série* 8, 4 (1894), J1–J122. Recuperato da https://www.numdam.org/item/AFST_1894_1_8_4_J1_0

[38] J. Stoer e R. Bulirsch. 1975. *Introduzione All' Analisi Numerica*. Zanichelli.

[39] Gábor Szegő. 1939. *Orthogonal Polynomials*. American Mathematical Society, Providence, RI.

[40] Pafnouti L. Tchebichef. 1858. Sur les fractions continues. *Journal de Mathématiques Pures et Appliquées, 2e série* 3, (1858), 289–323. Recuperato da https://numdam.org/item/JMPA_1858_2_3__289_0/

[41] Joseph F. Traub. 1964. *Iterative Methods for the Solution of Equations*. Prentice-Hall, Englewood Cliffs, NJ.

[42] Lloyd N. Trefethen e David Bau. 1997. *Numerical Linear Algebra*. SIAM.

[43] R. S. Varga. 2000. *Matrix Iterative Analysis* (2nd ed.). Springer. https://doi.org/10.1007/978-3-642-05156-2

[44] Frank M. White. 2011. *Fluid Mechanics* (7th ed.). McGraw-Hill Education.

[45] Z. I. Woźnicki. 1994. Nonnegative splitting theory. *Japan Journal of Industrial and Applied Mathematics* 11, 2 (1994), 289–342. https://doi.org/10.1007/BF03167226

IX Complementi sui polinomi ortogonali

A Complementi sui polinomi ortogonali 417

A.1 Simmetria: cosa succede se il peso è pari . . . 417
A.2 Derivate dei polinomi ortogonali 418
A.3 La formula di Christoffel–Darboux 419
A.4 Completezza e proiezione ortogonale 420

Il Capitolo 16 contiene quanto basta per costruire i minimi quadrati di Capitolo 17 e le quadrature gaussiane di Capitolo 18: la ricorrenza a tre termini, la matrice di Jacobi, la localizzazione degli zeri, i polinomi del nucleo e il problema di minimo vincolato.

Quello che segue è il resto della cassetta degli attrezzi: la simmetria dei polinomi quando il peso è pari, le formule per le derivate, la formula di Christoffel–Darboux e il quadro di completezza in cui si colloca la proiezione ortogonale. Sono risultati classici, utili appena si va oltre il programma del corso, ma nessuno di essi è richiesto per i due capitoli applicativi.



A. Complementi sui polinomi ortogonali

> **Materiale di complemento**
>
> Questa appendice raccoglie quattro risultati sui polinomi ortogonali che **non servono** per i minimi quadrati di Capitolo 17 né per le quadrature gaussiane di Capitolo 18, e che sono stati tolti dal Capitolo 16 per non appesantirlo. Restano qui perché sono strumenti di calcolo standard: si incontrano in ogni trattazione dell'argomento e servono appena si va un passo oltre il programma.

A.1 Simmetria: cosa succede se il peso è pari

Supponiamo che l'intervallo sia simmetrico rispetto all'origine, $(a, b) = (-c, c)$, e che il peso sia una **funzione pari**,

$$w(-x) = w(x) \qquad \text{per ogni } x.$$

È il caso dei polinomi di Legendre ($w \equiv 1$ su $[-1, 1]$), di Chebyshev ($w = (1 - x^2)^{-1/2}$) e in generale di Gegenbauer. Sotto questa ipotesi i polinomi ortogonali sono alternativamente pari e dispari:

$$p_n(-x) = (-1)^n\, p_n(x). \tag{A.1}$$

Vediamo perché, usando i polinomi monici π_n e l'unicità garantita da Teorema 14.1. Definiamo il polinomio «riflesso»

$$q_n(x) = (-1)^n\, \pi_n(-x)$$

e verifichiamo che possiede le due proprietà che caratterizzano π_n.

È monico di grado n. Sostituire $x \mapsto -x$ in π_n moltiplica il termine di testa x^n per $(-1)^n$; il fattore $(-1)^n$ davanti lo riporta a $+1$.

È ortogonale a tutti i polinomi di grado minore. Sia r con $\deg r < n$. Nel calcolo

$$\langle q_n, r\rangle_w = (-1)^n \int_{-c}^{c} \pi_n(-x)\, r(x)\, w(x)\, \mathrm{d}x$$

operiamo il cambio di variabile $t = -x$. L'intervallo si ribalta in sé (è simmetrico), $\mathrm{d}x$ diventa $-\mathrm{d}t$ con scambio degli estremi, e $w(-t) = w(t)$ perché il peso è pari:

$$\langle q_n, r\rangle_w = (-1)^n \int_{-c}^{c} \pi_n(t)\, r(-t)\, w(t)\,\mathrm{d}t = (-1)^n \big\langle \pi_n,\ r(-\cdot)\big\rangle_w = 0,$$

perché anche $r(-\cdot)$ ha grado minore di n e π_n è ortogonale a tutti questi polinomi. Dunque q_n è monico di grado n e ortogonale a $\mathcal{P}_{n-1}$: per l'unicità $q_n = \pi_n$, che è esattamente A.1.

Da A.1 si leggono tre conseguenze immediate.

- **Gli zeri sono simmetrici.** Se $p_n(x_0) = 0$ allora anche $p_n(-x_0) = \pm p_n(x_0) = 0$: gli zeri si presentano in coppie $\pm x_0$.
- **Per n dispari, lo zero è $x = 0$.** Ponendo $x = 0$ in A.1 con n dispari si ottiene $p_n(0) = -p_n(0)$, cioè $p_n(0) = 0$. L'origine è dunque sempre un nodo delle formule gaussiane di ordine dispari.
- **Nella ricorrenza $b_k = 0$.** Infatti $b_k = \langle x\phi_k, \phi_k\rangle_w$ è l'integrale di $x\,\phi_k(x)^2 w(x)$, prodotto di una funzione dispari (x) per una pari ($\phi_k^2 w$): l'integrale su un intervallo simmetrico di una funzione dispari è nullo. Si ritrova così la 16.7: la matrice di Jacobi dei polinomi di Legendre ha diagonale nulla.

Per i Jacobi con $\alpha \neq \beta$ il peso **non** è pari, e infatti la simmetria non vale nella forma A.1. La sostituzione $x \mapsto -x$ trasforma però il peso in

$$(1+x)^{\alpha}(1-x)^{\beta} = w_{\beta,\alpha}(x),$$

cioè nel peso con i due parametri scambiati. Ripetendo l'argomento di sopra si ottiene

$$P_n^{(\alpha,\beta)}(-x) = (-1)^n\, P_n^{(\beta,\alpha)}(x). \tag{A.2}$$

Per $\alpha = \beta$ i due membri riguardano la stessa famiglia e si ricade in A.1.

A.2 Derivate dei polinomi ortogonali

Serve spesso la derivata di un polinomio ortogonale: per esempio nella formula dei pesi di Gauss–Legendre $\omega_i = 2/\big[(1-x_i^2)P_n'(x_i)^2\big]$ di Capitolo 18, o per raffinare gli zeri con il metodo di Newton. Derivare la forma monomiale sarebbe disastroso; conviene una formula che riusi i valori già calcolati con la ricorrenza.

Per i polinomi di Legendre vale

$$(1-x^2)\,P_n'(x) = n\big[P_{n-1}(x) - x\,P_n(x)\big], \qquad -1 < x < 1, \tag{A.3}$$

che si ricava dalla formula di Rodrigues 14.6 con alcune integrazioni per parti (il conto è in [39]). Dal punto di vista pratico A.3 dice che, una volta ottenuti $P_{n-1}(x)$ e $P_n(x)$ con la ricorrenza a tre termini, la derivata costa **due moltiplicazioni e una divisione**, senza alcun lavoro aggiuntivo.

Agli estremi $x = \pm 1$ la formula va letta come un limite, perché il fattore $1 - x^2$ si annulla; il valore corretto è

$$P_n'(1) = \frac{n(n+1)}{2}, \qquad P_n'(-1) = (-1)^{n-1}\frac{n(n+1)}{2}.$$

Per i Jacobi vale invece una formula di natura diversa, che non lega la derivata alla stessa famiglia ma a una famiglia *vicina*:

$$\frac{\mathrm{d}}{\mathrm{d}x}P_n^{(\alpha,\beta)}(x) = \frac{n+\alpha+\beta+1}{2}\,P_{n-1}^{(\alpha+1,\beta+1)}(x). \tag{A.4}$$

Vale la pena fermarsi un attimo su cosa dice. Derivando si abbassa il grado di uno, come sempre; ma il risultato non è un polinomio qualunque: è di nuovo un polinomio di Jacobi, con **entrambi i parametri aumentati di uno**, cioè ortogonale rispetto al peso $(1-x)^{\alpha+1}(1+x)^{\beta+1}$, che si annulla agli estremi più rapidamente. In altre parole la derivazione manda famiglie ortogonali in famiglie ortogonali: una proprietà tutt'altro che scontata, che è alla base delle formule di quadratura di Gauss–Lobatto e dei metodi spettrali.

A.3 La formula di Christoffel–Darboux

Nel Sezione 16.3 si incontra ripetutamente la somma

$$\sum_{k=0}^{n-1} \frac{p_k(x)\, p_k(y)}{h_k}, \qquad h_k = \langle p_k, p_k \rangle_w, \tag{A.5}$$

dove p_k è la famiglia **monica**. Calcolarla come sta scritta richiede di valutare n polinomi. La formula di Christoffel–Darboux[1][2] riduce tutto a **due** valutazioni:

$$\sum_{k=0}^{n-1} \frac{p_k(x)p_k(y)}{h_k} = \frac{p_n(x)\, p_{n-1}(y) - p_{n-1}(x)\, p_n(y)}{h_{n-1}\,(x-y)}, \qquad x \neq y. \tag{A.6}$$

La dimostrazione è un conto diretto, che conviene seguire passo passo.

Passo 1: la ricorrenza monica e un'identità sui h_k. Nella normalizzazione monica la ricorrenza si scrive

$$p_{k+1}(t) = (t - b_k)\, p_k(t) - a_k^2\, p_{k-1}(t), \qquad p_{-1} = 0,\ p_0 = 1. \tag{A.7}$$

Ci servirà il valore di a_k^2. Moltiplichiamo scalarmente A.7 per p_{k-1}: a sinistra si ottiene 0 perché $p_{k+1} \perp p_{k-1}$, e a destra sopravvivono due termini,

$$0 = \langle t\, p_k, p_{k-1} \rangle_w - a_k^2\, h_{k-1}.$$

Ora $\langle t\, p_k, p_{k-1} \rangle_w = \langle p_k,\ t\, p_{k-1} \rangle_w$ e $t\, p_{k-1}$ è monico di grado k, dunque uguale a p_k più un polinomio di grado minore, che è ortogonale a p_k. Resta $\langle t\, p_k, p_{k-1} \rangle_w = h_k$ e quindi

$$a_k^2 = \frac{h_k}{h_{k-1}}. \tag{A.8}$$

Passo 2: scrivere la ricorrenza in due punti e sottrarre. Scriviamo A.7 una volta in $t = x$ e una volta in $t = y$:

$$p_{k+1}(x) = (x - b_k)p_k(x) - a_k^2 p_{k-1}(x),$$
$$p_{k+1}(y) = (y - b_k)p_k(y) - a_k^2 p_{k-1}(y).$$

Moltiplichiamo la prima per $p_k(y)$, la seconda per $p_k(x)$ e sottraiamo. I termini con b_k compaiono come $-b_k p_k(x)p_k(y)$ in entrambe le righe e si **cancellano**:

$$\underbrace{p_{k+1}(x)p_k(y) - p_k(x)p_{k+1}(y)}_{=:\,F_k} = (x-y)\, p_k(x)p_k(y) + a_k^2 \underbrace{\left[p_k(x)p_{k-1}(y) - p_{k-1}(x)p_k(y)\right]}_{=F_{k-1}}. \tag{A.9}$$

Conviene dare un nome alla combinazione che si ripete: posto

$$F_k = p_{k+1}(x)\, p_k(y) - p_k(x)\, p_{k+1}(y),$$

la A.9 diventa semplicemente

$$F_k = (x-y)\, p_k(x)p_k(y) + a_k^2\, F_{k-1}.$$

Passo 3: dividere per h_k e far telescopare. Dividiamo per h_k e usiamo A.8, cioè $a_k^2/h_k = 1/h_{k-1}$:

[1] Elwin Bruno Christoffel (1829-1900)
[2] Jean Gaston Darboux (1842-1917)

$$\frac{F_k}{h_k} - \frac{F_{k-1}}{h_{k-1}} = (x-y)\,\frac{p_k(x)p_k(y)}{h_k}. \tag{A.10}$$

A sinistra compare la differenza di due termini consecutivi della stessa successione $k \mapsto F_k/h_k$: sommando su $k = 0, 1, \dots, n-1$ tutti i termini intermedi si elidono e resta solo la differenza fra l'ultimo e il primo,

$$\frac{F_{n-1}}{h_{n-1}} - \frac{F_{-1}}{h_{-1}} = (x-y)\sum_{k=0}^{n-1}\frac{p_k(x)p_k(y)}{h_k}.$$

Il termine $F_{-1} = p_0(x)p_{-1}(y) - p_{-1}(x)p_0(y)$ è nullo perché $p_{-1} = 0$. Esplicitando F_{n-1} e dividendo per $x - y \neq 0$ si ottiene A.6.

Il caso $y = x$. Per $y \to x$ numeratore e denominatore di A.6 tendono entrambi a zero. Applicando la regola di de l'Hôpital rispetto a y (o, equivalentemente, riconoscendo nel rapporto un rapporto incrementale) si ottiene la **forma confluente**

$$\sum_{k=0}^{n-1}\frac{p_k(x)^2}{h_k} = \frac{p_n'(x)\,p_{n-1}(x) - p_{n-1}'(x)\,p_n(x)}{h_{n-1}}, \tag{A.11}$$

Il membro sinistro è una somma di quadrati, quindi **strettamente positivo**: è la quantità che compare al denominatore della funzione di Christoffel 16.17 e, attraverso di essa, nei pesi delle formule gaussiane.

A.4 Completezza e proiezione ortogonale

Per un peso di Jacobi i polinomi sono densi in $L^2_w(-1,1)$. La proiezione ortogonale di $f \in L^2_w$ sulla classe $\mathcal{P}_n$ è

$$\Pi_n f = \sum_{k=0}^{n} c_k \phi_k, \qquad c_k = \langle f, \phi_k \rangle_w.$$

Poiché $f - \Pi_n f$ è ortogonale a $\mathcal{P}_n$, per qualunque $q \in \mathcal{P}_n$ si ha la decomposizione di Pitagora

$$\|f - q\|_w^2 = \|f - \Pi_n f\|_w^2 + \|\Pi_n f - q\|_w^2.$$

Questa prova mostra che $\Pi_n f$ è l'unica migliore approssimazione in norma pesata. La densità implica $\|f - \Pi_n f\|_w \to 0$.

In sintesi

- Se l'intervallo è simmetrico e il peso è pari, i polinomi sono alternativamente pari e dispari: gli zeri vengono in coppie $\pm x_0$, l'origine è uno zero nei gradi dispari e la matrice di Jacobi ha diagonale nulla.
- Le derivate si ottengono senza derivare la forma monomiale: per i Legendre con A.3, per i Jacobi con A.4, che manda la famiglia (α, β) nella famiglia $(\alpha+1, \beta+1)$.
- La formula di Christoffel–Darboux riduce una somma di n termini a due sole valutazioni; la sua forma confluente dà il reciproco della funzione di Christoffel.
- La proiezione ortogonale è la migliore approssimazione in norma pesata, e la densità dei polinomi garantisce che l'errore tenda a zero.

rron-Frobenius, M-matrici e splitting

X

B Il teorema di Perron-Frobenius 425

B.1 Introduzione 425
B.2 Teorema di Perron-Frobenius 428

C Confronto fra splitting e velocità di convergenza 431

C.1 Autovalori del prodotto 431
C.2 Splitting convergente 431
C.3 Splitting vari 431
C.4 Confronto raggio spettrale 432
C.5 Splitting e inversa 433
C.6 Monotonia e splitting debole 433
C.7 Splitting debolemente non negativi 435
C.8 Raggio spettrale e splitting debolmente non negativi 436
C.9 Splitting non negativi e convergenza 436
C.10 Confronto della velocità di convergenza fra alcuni metodi iterativi 436

Appendice in costruzione

Questo materiale non fa parte del programma d'esame ed è **incompleto**. Raccoglie i risultati di teoria spettrale delle matrici non negative che giustificano i criteri di convergenza dei metodi iterativi visti nel capitolo Capitolo 26.
Stato attuale:

- alcune dimostrazioni del testo sono abbozzate e vanno riviste;
- la notazione non è ancora del tutto allineata al resto del libro.

L'appendice è organizzata in due parti: il teorema di Perron-Frobenius e la teoria degli splitting, con il confronto fra le velocità di convergenza dei metodi iterativi. I preliminari su raggio spettrale, serie di Neumann e M-matrici — che qui si danno per noti — sono all'inizio del capitolo sui metodi iterativi, in Capitolo 25.



B. Il teorema di Perron-Frobenius

B.1 Introduzione

Il teorema di Perron-Frobenius è fondamentale nello studio degli splitting non negativi e dei teoremi di confronto sulle M-matrici. In questa sezione dimostriamo i due risultati principali.

B.1.1 Autovalori e raggio spettrale

Theorem B.1 Sia $\boldsymbol{A} \geq 0$ una matrice quadrata, $\boldsymbol{x} \geq 0$ un vettore non nullo e $\alpha \geq 0$. Se vale

$$\alpha \boldsymbol{x} \leq \boldsymbol{A}\boldsymbol{x}$$

allora $\alpha \leq \varrho(\boldsymbol{A})$. Se inoltre $\alpha \boldsymbol{x} < \boldsymbol{A}\boldsymbol{x}$ allora $\alpha < \varrho(\boldsymbol{A})$.

Proof. Osserviamo che

$$\begin{aligned} \alpha \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{A}^2\boldsymbol{x} \quad &\Longrightarrow \quad \alpha^2 \boldsymbol{x} \leq \boldsymbol{A}^2\boldsymbol{x} \\ \alpha \boldsymbol{A}^{k-1}\boldsymbol{x} \leq \boldsymbol{A}^k\boldsymbol{x} \quad &\Longrightarrow \quad \alpha^k \boldsymbol{x} \leq \boldsymbol{A}^k\boldsymbol{x} \end{aligned} \tag{B.1}$$

moltiplicando le disuguaglianze B.1 per il vettore $\boldsymbol{e} = \big(1, 1, \dots, 1\big)^T$

$$\alpha^k \boldsymbol{e}^T\boldsymbol{x} \leq \boldsymbol{e}^T\boldsymbol{A}^k\boldsymbol{x} \Downarrow \alpha^k \left\|\boldsymbol{x}\right\|_1 \leq \boldsymbol{e}^T\boldsymbol{A}^k\boldsymbol{x} \leq \left\|\boldsymbol{e}\right\|_1 \left\|\boldsymbol{A}^k\right\|_1 \left\|\boldsymbol{x}\right\|_1$$

estraendo le radici k-esime e passando al limite per $k \mapsto \infty$ usando il corollario Corollario 25.1

$$\alpha \left\|\boldsymbol{x}\right\|_1^{\frac{1}{k}} \leq \left\|\boldsymbol{e}\right\|_1^{\frac{1}{k}} \left\|\boldsymbol{A}^k\right\|_1^{\frac{1}{k}} \left\|\boldsymbol{x}\right\|_1^{\frac{1}{k}} \alpha = \lim_{k\mapsto\infty} \alpha \left\|\boldsymbol{x}\right\|_1^{\frac{1}{k}} \leq \lim_{k\mapsto\infty} \left\|\boldsymbol{e}\right\|_1^{\frac{1}{k}} \left\|\boldsymbol{A}^k\right\|_1^{\frac{1}{k}} \left\|\boldsymbol{x}\right\|_1^{\frac{1}{k}} = \varrho(\boldsymbol{A})$$

se $\alpha \boldsymbol{x} < \boldsymbol{A}\boldsymbol{x}$ allora esiste $\alpha < \alpha'$ tale che $\alpha' \boldsymbol{x} \leq \boldsymbol{A}\boldsymbol{x}$ e quindi $\alpha < \alpha' \leq \varrho(\boldsymbol{A})$. ∎

B.1.2 Autovalore matrice irriducibile

Lemma B.1 Sia $\boldsymbol{A} \geq 0$ irriducibile. Allora $\varrho(\boldsymbol{A}) > 0$ è un autovalore di $\boldsymbol{A}$ ed esiste un vettore $\boldsymbol{u} > 0$ tale che $\boldsymbol{A}\boldsymbol{u} = \varrho(\boldsymbol{A})\boldsymbol{u}$.

Proof. Poiché $\boldsymbol{A}$ è irriducibile, si ha

$$\boldsymbol{A}\boldsymbol{e} > 0$$

da cui, scegliendo $\epsilon > 0$ sufficientemente piccolo, segue

$$\boldsymbol{A}\boldsymbol{e} > \epsilon\boldsymbol{e}$$

Applicando il teorema Teorema B.1 otteniamo $\varrho(\boldsymbol{A}) > \epsilon > 0$. Sia $\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{x}$, $\boldsymbol{x} \neq 0$ con $|\lambda| = \varrho(\boldsymbol{A})$, ne segue,

$$\boldsymbol{A}\,|\boldsymbol{x}| = \varrho(\boldsymbol{A})\,|\boldsymbol{x}| \qquad \text{con} \qquad |\boldsymbol{x}| \geq 0,$$

da cui

$$\varrho(\boldsymbol{A})\,|\boldsymbol{x}| = |\lambda|\,|\boldsymbol{x}| = |\lambda\boldsymbol{x}| = |\boldsymbol{A}\boldsymbol{x}| \leq |\boldsymbol{A}|\,|\boldsymbol{x}| = |\boldsymbol{A}|\,|\boldsymbol{x}|\,.$$

Abbiamo quindi

$$\boldsymbol{y} = \boldsymbol{A}\,|\boldsymbol{x}| - \varrho(\boldsymbol{A})\,|\boldsymbol{x}| \geq 0 \tag{B.2}$$

Osserviamo che dalla B.2

$$(\boldsymbol{A}\,|\boldsymbol{x}|)_i = 0 \quad \Longrightarrow \quad |x_i| = 0$$

ed ancora

$$|x_i| > 0 \quad \Longrightarrow \quad (\boldsymbol{A}\,|\boldsymbol{x}|)_i = 0$$

quindi il vettore $|\boldsymbol{x}|$ ed il vettore $\boldsymbol{A}\,|\boldsymbol{x}|$ hanno gli eventuali zeri nelle stesse posizioni. Sia

$$\boldsymbol{w} \quad \text{definito come:} \quad w_i = \begin{cases} 1 & |x_i| = 0 \\ 0 & \text{altrimenti} \end{cases}$$

senza perdere in generalità effettuando eventualmente una permutazione di indici possiamo supporre che

$$\boldsymbol{w} = \left(0, 0, \ldots, 0, 1, 1, \ldots, 1\right)^T = \begin{pmatrix} 0 \\ \boldsymbol{e} \end{pmatrix}$$

possiamo quindi scrivere

$$0 = \boldsymbol{w}^T \boldsymbol{A}\,|\boldsymbol{x}| = \begin{pmatrix} 0 & \boldsymbol{e}^T \end{pmatrix} \begin{pmatrix} \boldsymbol{B} & \boldsymbol{C} \\ \boldsymbol{D} & \boldsymbol{E} \end{pmatrix} \begin{pmatrix} |\boldsymbol{x}^1| \\ 0 \end{pmatrix}$$

dove

$$|\boldsymbol{x}| = \begin{pmatrix} |\boldsymbol{x}^1| \\ 0 \end{pmatrix}$$

e sviluppando i termini

$$0 = \boldsymbol{e}^T \boldsymbol{D}\,|\boldsymbol{x}^1| \quad \Longrightarrow \quad \boldsymbol{D} = 0$$

contraddicendo l'ipotesi di irriducibilità. Da questo segue che $|\boldsymbol{x}| > 0$ e $\boldsymbol{A}\,|\boldsymbol{x}| > 0$. Se $\boldsymbol{y} = 0$ il teorema è dimostrato, altrimenti poniamo

$$\boldsymbol{E} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 1 & \cdots & 1 & 1 \end{pmatrix}$$

e definiamo la matrice

$$\boldsymbol{A}_\epsilon = \boldsymbol{A} + \epsilon \boldsymbol{E} > 0, \quad \forall \epsilon > 0$$

ripetendo i ragionamenti fin qui fatti ora per la matrice $\boldsymbol{A}_\epsilon$ arriviamo alla conclusione che $\boldsymbol{y} = \boldsymbol{A}_\epsilon |\boldsymbol{x}_\epsilon| - \varrho(\boldsymbol{A}_\epsilon)|\boldsymbol{x}_\epsilon| \geq 0$. Supponiamo $\boldsymbol{y} \neq 0$ moltiplicando $\boldsymbol{y}$ per $\boldsymbol{A}_\epsilon$ otteniamo

$$0 < \boldsymbol{A}_\epsilon \boldsymbol{y} = \boldsymbol{A}_\epsilon^2 |\boldsymbol{x}_\epsilon| - \varrho(\boldsymbol{A}_\epsilon)\boldsymbol{A}_\epsilon |\boldsymbol{x}_\epsilon| = \boldsymbol{A}_\epsilon \boldsymbol{z} - \varrho(\boldsymbol{A}_\epsilon)\boldsymbol{z}\boldsymbol{z} = \boldsymbol{A}_\epsilon |\boldsymbol{x}_\epsilon| > 0$$

cioè $\varrho(\boldsymbol{A}_\epsilon)\boldsymbol{z} < \boldsymbol{A}_\epsilon \boldsymbol{z}$ ed applicando il teorema Teorema B.1 si ha $\varrho(\boldsymbol{A}_\epsilon) < \varrho(\boldsymbol{A}_\epsilon)$ cioè una contraddizione. Quindi

$$\boldsymbol{A}_\epsilon |\boldsymbol{x}_\epsilon| = \varrho(\boldsymbol{A}_\epsilon)|\boldsymbol{x}_\epsilon|, \qquad \epsilon > 0$$

usando la continuità e passando al limite per $\epsilon \mapsto 0$ segue $\boldsymbol{A}|\boldsymbol{x}| = \varrho(\boldsymbol{A})|\boldsymbol{x}|$. Allora $\boldsymbol{u} = |\boldsymbol{x}| > 0$ è l'autovettore con autovalore $|\lambda| = \varrho(\boldsymbol{A}) > 0$ che cercavamo. ∎

B.1.3 Autovalori e raggio spettrale

Lemma B.2 Sia $\boldsymbol{A} \geq 0$ irriducibile ed $\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$, $\boldsymbol{x} \neq 0$ con $|\lambda| = \varrho(\boldsymbol{A})$ allora per un qualche angolo $\theta \in \mathbb{R}$ vale $e^{-i\theta}\boldsymbol{x} = |\boldsymbol{x}|$.

Proof. osserviamo che dal lemma Lemma B.1 possiamo scrivere

$$\boldsymbol{A}|\boldsymbol{x}| = \varrho(\boldsymbol{A})|\boldsymbol{x}| \quad \text{con} \quad |\boldsymbol{x}| > 0$$

da questa possiamo ricavare

$$\varrho(\boldsymbol{A})|\boldsymbol{x}| = |\lambda||\boldsymbol{x}| = |\boldsymbol{A}\boldsymbol{x}| \leq \boldsymbol{A}|\boldsymbol{x}| = \varrho(\boldsymbol{A})|\boldsymbol{x}| \Downarrow |\boldsymbol{A}\boldsymbol{x}| = \boldsymbol{A}|\boldsymbol{x}|$$

e passando alle componenti

$$\sum_{j=1}^{n} A_{i,j}|x_j| = \left|\sum_{j=1}^{n} A_{i,j}x_j\right| \qquad \text{per } i = 1, 2, \dots, n$$

Per avere l'uguaglianza tutti i numeri $A_{i,j}x_j$ devono stare sullo stesso raggio nel piano complesso. Sia $\theta \in \mathbb{R}$ l' angolo di questo raggio, allora

$$A_{i,j}x_j = e^{i\theta}A_{i,j}|x_j| \qquad \text{e quindi} \qquad A_{i,j}e^{-i\theta}x_j = A_{i,j}|x_j| > 0$$

e poiché $\boldsymbol{A}$ è irriducibile, fissato j esisterà un indice i per cui $A_{i,j} > 0$ e da questo $e^{-i\theta}x_j > 0$ ∎

B.1.4 Limitazione autovalori

Theorem B.2 Sia data una matrice $\boldsymbol{A} \geq 0$ irriducibile, allora risulta $|\lambda| < \varrho(\boldsymbol{A})$ per ogni autovalore $\lambda \neq \varrho(\boldsymbol{A})$.

Proof. Per definizione di raggio spettrale $|\lambda| \leq \varrho(\boldsymbol{A})$ per ogni λ autovalore di $\boldsymbol{A}$. Supponiamo $|\lambda| = \varrho(\boldsymbol{A})$ con $\boldsymbol{A}\boldsymbol{x} = \lambda\boldsymbol{x}$, dal lemma precedente ponendo $\boldsymbol{w} = e^{-i\theta}\boldsymbol{x} > 0$ otteniamo $\boldsymbol{A}\boldsymbol{w} = \lambda\boldsymbol{w}$ ed essendo i vettori $\boldsymbol{A}\boldsymbol{w}$ e $\boldsymbol{w}$ reali segue che λ è reale e quindi $\lambda = \varrho(\boldsymbol{A})$. ∎

B.1.5 Raggio spettrale e autovalore

Theorem B.3 Sia $\boldsymbol{A} \geq 0$ irriducibile allora $\varrho(\boldsymbol{A})$ è un autovalore algebricamente semplice di $\boldsymbol{A}$.

Proof. Se così non fosse esisterebbero due autovettori $\boldsymbol{x}$ ed $\boldsymbol{y}$ tali che

$$\boldsymbol{A}\boldsymbol{x} = \varrho(\boldsymbol{A})\boldsymbol{x}, \quad \boldsymbol{A}\boldsymbol{y} = \varrho(\boldsymbol{A})\boldsymbol{y} \qquad \text{con} \qquad \|\boldsymbol{x}\|_1 = \|\boldsymbol{y}\|_1 = 1.$$

per il lemma Lemma B.2 $\boldsymbol{p} = e^{-i\theta_1}\boldsymbol{x} > 0$, $\boldsymbol{q} = e^{-i\theta_2}\boldsymbol{y} > 0$, per due opportuni angoli θ_1 e θ_2. Sia

$$\beta = \min\left\{ \frac{q_i}{p_i} \ : \ i = 1, 2, \dots, n \right\}, \qquad \boldsymbol{r} = \boldsymbol{q} - \beta\boldsymbol{p}$$

allora $\boldsymbol{r} \geq 0$ ed almeno una sua componente è nulla, se $\boldsymbol{r}$ non fosse identicamente nullo si avrebbe

$$\boldsymbol{A}\boldsymbol{r} = \boldsymbol{A}\boldsymbol{q} - \beta\boldsymbol{A}\boldsymbol{p} = \varrho(\boldsymbol{A})(\boldsymbol{q} - \beta\boldsymbol{p}) = \varrho(\boldsymbol{A})\boldsymbol{r}$$

e quindi $\boldsymbol{A}\boldsymbol{r} - \varrho(\boldsymbol{A})\boldsymbol{r} = 0$. Ripetendo i ragionamenti fatti nel lemma Lemma B.1 per l'B.2 dalla irriducibilità segue che $\boldsymbol{r} > 0$ e poiché questo non può essere vero concludiamo che $\boldsymbol{r} = 0$ da cui $\boldsymbol{q} = \beta\boldsymbol{p}$ e di conseguenza $\boldsymbol{x} = \beta e^{i(\theta_1 - \theta_2)}\boldsymbol{y}$. ∎

Questi risultati si riassumono nel ben noto teorema di Perron-Frobenius:

B.2 Teorema di Perron-Frobenius

Theorem B.4 Se $\boldsymbol{A} \geq 0$ è irriducibile allora

1. $\varrho(\boldsymbol{A}) > 0$
2. $\varrho(\boldsymbol{A})$ è un autovalore algebricamente semplice di $\boldsymbol{A}$
3. $|\lambda| < \varrho(\boldsymbol{A})$ per ogni autovalore $\lambda \neq \varrho(\boldsymbol{A})$ cioè $\varrho(\boldsymbol{A})$ è l'unico autovalore di massimo modulo.
4. all' autovalore $\varrho(\boldsymbol{A})$ corrisponde un autovettore $\boldsymbol{x} > 0$ detto vettore di Perron[a].
5. $\varrho(\boldsymbol{A})$ è una funzione crescente nelle variabili $A_{i,j}$ cioè se $\boldsymbol{B} \geq \boldsymbol{A} > 0$ con $\boldsymbol{A} \neq \boldsymbol{B}$ allora $\varrho(\boldsymbol{B}) > \varrho(\boldsymbol{A})$.

[a] Oskar Perron (1880-1975)

Proof. I punti $1 - 4$ sono stati dimostrati nei teoremi precedenti. Bisogna solo provare la proprietà di monotonia del punto 5. Sia $\boldsymbol{x} > 0$ il vettore di Perron per la matrice $\boldsymbol{A}$ allora avremo

$$\boldsymbol{B}\boldsymbol{x} \geq \boldsymbol{A}\boldsymbol{x} = \varrho(\boldsymbol{A})\boldsymbol{x}, \qquad \text{con} \qquad \boldsymbol{A}\boldsymbol{x} \neq \boldsymbol{B}\boldsymbol{x}$$

e definendo

$$\boldsymbol{y} = \boldsymbol{B}\boldsymbol{x} - \varrho(\boldsymbol{A})\boldsymbol{x} \geq 0, \qquad \text{con} \qquad \boldsymbol{y} \neq 0$$

e $\boldsymbol{z} = \boldsymbol{B}\boldsymbol{x} > 0$ possiamo calcolare

$$0 < \boldsymbol{B}\boldsymbol{y} = \boldsymbol{B}\boldsymbol{z} - \varrho(\boldsymbol{A})\boldsymbol{B}\boldsymbol{x} = \boldsymbol{B}\boldsymbol{z} - \varrho(\boldsymbol{A})\boldsymbol{z}$$

e quindi $\boldsymbol{B}\boldsymbol{z} > \varrho(\boldsymbol{A})\boldsymbol{z}$ ed applicando il teorema Teorema B.1 otteniamo $\varrho(\boldsymbol{A}) < \varrho(\boldsymbol{B})$ ∎

con procedimenti di passaggio al limite possiamo provare dei risultati analoghi per matrici non negative. Infatti se $\boldsymbol{A} \geq 0$ definendo la matrice $\boldsymbol{E} > 0$ come nel lemma Lemma B.1

$$\boldsymbol{A}_\epsilon = \boldsymbol{A} + \epsilon\boldsymbol{E} > 0, \qquad \forall \epsilon > 0$$

è irriducibile e quindi possiamo applicare il teorema di Perron. Passando al limite per $\epsilon \mapsto 0$ in generale perdiamo le disuguaglianze strette e le molteplicità; quindi:

B.2.1 Monotonia raggio spettrale

Theorem B.5 Se $\boldsymbol{A} \geq 0$ allora

1. $\varrho(\boldsymbol{A})$ è un autovalore di $\boldsymbol{A}$
2. Se $\varrho(\boldsymbol{A}) > 0$ esiste un corrispondente autovettore $\boldsymbol{x} \geq 0$.
3. $\varrho(\boldsymbol{A})$ è una funzione non decrescente nelle entrate $A_{i,j}$ cioè se $\boldsymbol{A} \geq \boldsymbol{B} \geq 0$ allora $\varrho(\boldsymbol{A}) > \varrho(\boldsymbol{B})$.

Proof. basta osservare che

$$\begin{aligned} \boldsymbol{Q}_2 \geq \boldsymbol{Q}_1 &\implies \boldsymbol{A}^{-1}\boldsymbol{Q}_2 \geq \boldsymbol{A}^{-1}\boldsymbol{Q}_1 \implies \\ \sigma(\boldsymbol{A}^{-1}\boldsymbol{Q}_2) \geq \sigma(\boldsymbol{A}^{-1}\boldsymbol{Q}_1) &\implies \sigma(\boldsymbol{P}_2^{-1}\boldsymbol{Q}_2) \geq \sigma(\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1) \end{aligned}$$

■



C. Confronto fra splitting e velocità di convergenza

Questo capitolo studia le proprietà degli splitting di una matrice e, a partire da esse, confronta la velocità di convergenza dei metodi iterativi che ne derivano. Il punto di partenza è un risultato sugli autovalori del prodotto di due matrici, valido anche quando le matrici sono singolari.

C.1 Autovalori del prodotto

Lemma C.1 Siano $\boldsymbol{A}$ e $\boldsymbol{B}$ due matrici quadrate, allora $\boldsymbol{AB}$ e $\boldsymbol{BA}$ hanno gli stessi autovalori ed in particolare

$$\varrho(\boldsymbol{AB}) = \varrho(\boldsymbol{BA})$$

Proof. sia $\boldsymbol{A}$ non singolare e λ autovalore del prodotto $\boldsymbol{AB}$ allora

$$0 = \det \lambda \boldsymbol{I} - \boldsymbol{AB} = \det \boldsymbol{A} \det \lambda \boldsymbol{A}^{-1} - \boldsymbol{B} = \det \lambda \boldsymbol{A}^{-1} - \boldsymbol{B} \det \boldsymbol{A} = \det \lambda \boldsymbol{I} - \boldsymbol{BA}$$

quindi λ è autovalore del prodotto $\boldsymbol{BA}$ e di conseguenza anche i raggi spettrali coincidono. Sia $\boldsymbol{A}$ singolare allora per $\mu \neq 0$ sufficientemente piccolo la matrice $\boldsymbol{A}_\mu = \mu \boldsymbol{I} + \boldsymbol{A}$ è non singolare quindi

$$\varrho(\boldsymbol{A}_\mu \boldsymbol{B}) = \varrho(\boldsymbol{B}\boldsymbol{A}_\mu)$$

e per continuità segue la tesi. ∎

C.2 Splitting convergente

Definition C.1 La decomposizione $\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q}$ è detta splitting convergente se $\boldsymbol{P}$ è non singolare e $\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) < 1$.

Questa definizione di splitting richiede di conoscere il raggio spettrale della matrice $\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})$. Vogliamo quindi individuare condizioni sufficienti che garantiscano la convergenza dello splitting associato a una matrice. Introduciamo a questo scopo le definizioni seguenti.

C.3 Splitting vari

Definition C.2 La decomposizione $\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q}$ è detta \

- **splitting regolare**
  se:
  1. $\boldsymbol{P}$ è monotona cioè $\boldsymbol{P}^{-1} \geq 0$
  2. $\boldsymbol{Q} \geq 0$.
- **splitting non negativo**
  se:
  1. $\boldsymbol{P}$ è monotona
  2. $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0$ e $\boldsymbol{Q}\boldsymbol{P}^{-1} \geq 0$.
- **splitting debolmente non negativo**
  se:
  1. $\boldsymbol{P}$ è monotona
  2. $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0$ oppure $\boldsymbol{Q}\boldsymbol{P}^{-1} \geq 0$.
- **splitting debole**
  se:
  2. $\boldsymbol{P}$ è non singolare
  3. $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0$ oppure $\boldsymbol{Q}\boldsymbol{P}^{-1} \geq 0$.

Osserviamo che valgono le seguenti implicazioni

splitting regolare

$\Downarrow$

splitting non negativo

$\Downarrow$

splitting debolmente non negativo

$\Downarrow$

splitting debole

Applicazione del teorema di Perron-Frobenius agli splitting

Studiamo ora le proprietà degli splitting mediante la teoria di Perron-Frobenius. Il teorema di Perron-Frobenius permette di confrontare i raggi spettrali di matrici $\boldsymbol{A}$ e $\boldsymbol{B}$ nel caso $\boldsymbol{A} \geq |\boldsymbol{B}| \geq 0$. Confronti di questo tipo sono molto utili nello studio dei metodi iterativi che derivano da splitting. Infatti in questo modo si riesce a confrontare il raggio spettrale della matrice di iterazione e quindi la convergenza dei metodi.

C.4 Confronto raggio spettrale

Lemma C.2 Siano $\boldsymbol{A}$ e $\boldsymbol{B}$ due matrici con $\boldsymbol{A} \geq |\boldsymbol{B}| \geq 0$ allora

$$\varrho(\boldsymbol{A}) \geq \varrho(\boldsymbol{B})$$

la disuguaglianza è stretta se la matrice $\boldsymbol{B}$ è irriducibile.

Proof. osserviamo che dai teoremi di Perron-Frobenius segue immediatamente che

$$\begin{aligned} \varrho(\boldsymbol{A}) &\geq \varrho(|\boldsymbol{B}|) \qquad \text{se } \boldsymbol{B} \text{ è riducibile} \\ \varrho(\boldsymbol{A}) &> \varrho(|\boldsymbol{B}|) \qquad \text{se } \boldsymbol{B} \text{ è irriducibile} \end{aligned}$$

quindi basta dimostrare che $\varrho(|\boldsymbol{B}|) \geq \varrho(\boldsymbol{B})$. Per fare questo basta osservare che

$$\left\|\boldsymbol{B}^k\right\|_2 = \left\|\left|\boldsymbol{B}^k\right|\right\|_2 \leq \left\||\boldsymbol{B}|^k\right\|_2 \quad \Longrightarrow \quad \left\|\boldsymbol{B}^k\right\|_2^{\frac{1}{k}} \leq \left\||\boldsymbol{B}|^k\right\|_2^{\frac{1}{k}} \quad \Longrightarrow \quad \varrho(\boldsymbol{B}) \leq \varrho(|\boldsymbol{B}|)$$

per la formula di Gelfand[1], Corollario 25.1. ∎

Diamo ora un lemma che sarà utile nello studio dei metodi iterativi

C.5 Splitting e inversa

Lemma C.3 Sia $\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q}$ uno splitting di $\boldsymbol{A}$. Se $\boldsymbol{A}$ ed $\boldsymbol{P}$ sono non singolari allora

$$\boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1}$$

e di conseguenza la matrice $\boldsymbol{P}^{-1}\boldsymbol{Q}$ commuta con la matrice $\boldsymbol{A}^{-1}\boldsymbol{Q}$ e la matrice $\boldsymbol{Q}\boldsymbol{P}^{-1}$ commuta con la matrice $\boldsymbol{Q}\boldsymbol{A}^{-1}$.

Proof. basta osservare che

$$\begin{aligned}\boldsymbol{P}^{-1} = (\boldsymbol{A}+\boldsymbol{Q})^{-1} &= \boldsymbol{A}^{-1}(\boldsymbol{I}+\boldsymbol{Q}\boldsymbol{A}^{-1})^{-1} \\ &= (\boldsymbol{I}+\boldsymbol{A}^{-1}\boldsymbol{Q})^{-1}\boldsymbol{A}^{-1}\end{aligned}$$

da cui si ha

$$\begin{aligned}\boldsymbol{A}^{-1} &= \boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1} \\ \boldsymbol{A}^{-1} &= \boldsymbol{P}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1}\end{aligned} \quad \Longrightarrow \quad \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1}$$

e moltiplicando a desta e a sinistra per $\boldsymbol{Q}$

$$\begin{aligned}\boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1} &= \boldsymbol{Q}\boldsymbol{A}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1} \\ \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1}\boldsymbol{Q} &= \boldsymbol{A}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{Q}\end{aligned} \quad \Longrightarrow \quad \begin{aligned}(\boldsymbol{Q}\boldsymbol{P}^{-1})(\boldsymbol{Q}\boldsymbol{A}^{-1}) &= (\boldsymbol{Q}\boldsymbol{A}^{-1})(\boldsymbol{Q}\boldsymbol{P}^{-1}) \\ (\boldsymbol{P}^{-1}\boldsymbol{Q})(\boldsymbol{A}^{-1}\boldsymbol{Q}) &= (\boldsymbol{A}^{-1}\boldsymbol{Q})(\boldsymbol{P}^{-1}\boldsymbol{Q})\end{aligned}$$

otteniamo la tesi. ∎

C.6 Monotonia e splitting debole

Theorem C.1 Sia $\boldsymbol{Q} = \boldsymbol{P} - \boldsymbol{Q}$ uno *splitting debole*. Se $\boldsymbol{A}$ è monotona allora

$$\boldsymbol{A}^{-1} \geq \boldsymbol{P}^{-1} \tag{C.1}$$

Proof. utilizzando il lemma Lemma B.2 Se $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0$ allora vale

$$\boldsymbol{A}^{-1} = \boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1} \geq \boldsymbol{P}^{-1}$$

Se $\boldsymbol{Q}\boldsymbol{P}^{-1} \geq 0$ allora vale

$$\boldsymbol{A}^{-1} = \boldsymbol{P}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1} \geq \boldsymbol{P}^{-1}$$

in ogni caso la C.1 è soddisfatta. ∎

C.6.1 Proprietà splitting debole

[1] Israel Moiseevič Gel'fand (1913-2009)

Theorem C.2 Sia $\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q}$ uno splitting *debolmente non negativo*. Se $\boldsymbol{A}$ è monotona allora

1. $\boldsymbol{A}^{-1} \geq \boldsymbol{P}^{-1}$
2. $\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) = \varrho(\boldsymbol{Q}\boldsymbol{P}^{-1}) < 1$
3. $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0 \implies \boldsymbol{A}^{-1}\boldsymbol{Q} \geq \boldsymbol{P}^{-1}\boldsymbol{Q}$ ed $\boldsymbol{Q}\boldsymbol{P}^{-1} \geq 0 \implies \boldsymbol{Q}\boldsymbol{A}^{-1} \geq \boldsymbol{Q}\boldsymbol{P}^{-1}$
4. $\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) = \dfrac{\varrho(\boldsymbol{A}^{-1}\boldsymbol{Q})}{1 + \varrho(\boldsymbol{A}^{-1}\boldsymbol{Q})}$
5. viceversa se $\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) = \varrho(\boldsymbol{Q}\boldsymbol{P}^{-1}) < 1$ allora $\boldsymbol{A}^{-1} \geq 0$.

Proof. 1 segue dal teorema Teorema C.1.

2 Assumiamo che $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0$.

Possiamo scrivere allora

$$\begin{aligned}
\boldsymbol{A}^{-1} &= \boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1} \\
&= \boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}(\boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1}) \\
&= \boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1}) \\
&= \boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1} + (\boldsymbol{P}^{-1}\boldsymbol{Q})^2(\boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{A}^{-1}) \\
&= \boldsymbol{P}^{-1} + \boldsymbol{P}^{-1}\boldsymbol{Q}\boldsymbol{P}^{-1} + (\boldsymbol{P}^{-1}\boldsymbol{Q})^2\boldsymbol{P}^{-1} + (\boldsymbol{P}^{-1}\boldsymbol{Q})^3\boldsymbol{A}^{-1} \\
&\vdots \\
&= (\boldsymbol{I} + \boldsymbol{P}^{-1}\boldsymbol{Q} + \cdots + (\boldsymbol{P}^{-1}\boldsymbol{Q})^k)\boldsymbol{P}^{-1} + (\boldsymbol{P}^{-1}\boldsymbol{Q})^{k+1}\boldsymbol{A}^{-1}
\end{aligned}$$

e di conseguenza la serie

$$(\boldsymbol{I} + \boldsymbol{P}^{-1}\boldsymbol{Q} + \cdots + (\boldsymbol{P}^{-1}\boldsymbol{Q})^k)\boldsymbol{P}^{-1} = \boldsymbol{A}^{-1} - (\boldsymbol{P}^{-1}\boldsymbol{Q})^{k+1}\boldsymbol{A}^{-1} \leq \boldsymbol{A}^{-1}$$

è monotona crescente e limitata, quindi convergente. Per il lemma Lemma 25.1 si ha $\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) < 1$.\ 3 Dal lemma Lemma 25.1 per la convergenza della serie

$$\boldsymbol{I} + \boldsymbol{P}^{-1}\boldsymbol{Q} + \cdots + (\boldsymbol{P}^{-1}\boldsymbol{Q})^k + \cdots = (\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q})^{-1} \geq \boldsymbol{I}$$

Analogamente

$$\boldsymbol{I} + \boldsymbol{Q}\boldsymbol{P}^{-1} + \cdots + (\boldsymbol{Q}\boldsymbol{P}^{-1})^k + \cdots = (\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{P}^{-1})^{-1} \geq \boldsymbol{I}$$

e quindi se $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0$

$$\boldsymbol{A}^{-1}\boldsymbol{Q} = (\boldsymbol{P} - \boldsymbol{Q}\boldsymbol{Q})^{-1}\boldsymbol{Q} = (\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q})^{-1}\boldsymbol{P}^{-1}\boldsymbol{Q} \geq \boldsymbol{P}^{-1}\boldsymbol{Q}.$$

Del resto, se $\boldsymbol{Q}\boldsymbol{P}^{-1} \geq 0$

$$\boldsymbol{Q}\boldsymbol{A}^{-1} = \boldsymbol{Q}(\boldsymbol{P} - \boldsymbol{Q})^{-1} = \boldsymbol{Q}\boldsymbol{P}^{-1}(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{P}^{-1})^{-1} \geq \boldsymbol{Q}\boldsymbol{P}^{-1}$$

4 Osserviamo che

$$\boldsymbol{A}^{-1}\boldsymbol{Q} = (\boldsymbol{P} - \boldsymbol{Q}\boldsymbol{Q})^{-1}\boldsymbol{Q} = (\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q})^{-1}\boldsymbol{P}^{-1}\boldsymbol{Q} = \sum_{i=1}^{\infty}(\boldsymbol{P}^{-1}\boldsymbol{Q})^i$$

Osserviamo inoltre che

$$\varrho\left(\sum_{i=1}^{\infty}(\boldsymbol{P}^{-1}\boldsymbol{Q})^i\right) \leq \sum_{i=1}^{\infty}(\boldsymbol{P}^{-1}\boldsymbol{Q})^i = \frac{\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}{1 - \varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}$$

Poiché $\boldsymbol{P}^{-1}\boldsymbol{Q} \geq 0$ per il teorema di Perron-Frobenius esiste un autovalore $\lambda = \varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})^i) < 1$ e quindi

$$\Gamma = \sum_{i=1}^{\infty} \varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})^i = \frac{\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}{1-\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}$$

è un autovalore di $\sum_{i=1}^{\infty}(\boldsymbol{P}^{-1}\boldsymbol{Q})^i$ per cui

$$\varrho\left(\sum_{i=1}^{\infty}(\boldsymbol{P}^{-1}\boldsymbol{Q})^i\right) \geq \frac{\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}{1-\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}$$

il che implica

$$\varrho(\boldsymbol{A}^{-1}\boldsymbol{Q}) = \frac{\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}{1-\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q})}$$

5 sia ora $\varrho(\boldsymbol{P}^{-1}\boldsymbol{Q}) = \varrho(\boldsymbol{Q}\boldsymbol{P}^{-1}) < 1$. Allora la serie positiva

$$\boldsymbol{I} + \boldsymbol{P}^{-1}\boldsymbol{Q} + \cdots + (\boldsymbol{P}^{-1}\boldsymbol{Q})^k + \cdots = (\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q})^{-1} \geq \boldsymbol{I}$$

è convergente e quindi

$$\boldsymbol{A}^{-1} = (\boldsymbol{I} - \boldsymbol{P}^{-1}\boldsymbol{Q})^{-1}\boldsymbol{P}^{-1} \geq 0$$

■

Questo teorema in sostanza stabilisce che, se $\boldsymbol{A}$ è una matrice monotona, allora uno splitting debolmente non negativo è convergente. Viceversa, se la matrice $\boldsymbol{A}$ ha uno splitting debolmente non negativo che converge, allora è monotona.

C.7 Splitting debolemente non negativi

Theorem C.3 Siano $\boldsymbol{A} = \boldsymbol{P}_1 - \boldsymbol{Q}_1 = \boldsymbol{P}_2 - \boldsymbol{Q}_2$ due splitting *debolmente non negativi* di una matrice monotona $\boldsymbol{A}$. Se vale una delle seguenti disuguaglianze

$$\boldsymbol{A}^{-1}\boldsymbol{Q}_2 \geq \boldsymbol{A}^{-1}\boldsymbol{Q}_1 \geq 0, \boldsymbol{A}^{-1}\boldsymbol{Q}_2 \geq \boldsymbol{Q}_1\boldsymbol{A}^{-1} \geq 0$$
$$\boldsymbol{Q}_2\boldsymbol{A}^{-1} \geq \boldsymbol{Q}_1\boldsymbol{A}^{-1} \geq 0, \boldsymbol{Q}_2\boldsymbol{A}^{-1} \geq \boldsymbol{A}^{-1}\boldsymbol{Q}_1 \geq 0$$

allora

$$\varrho(\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1) \leq \varrho(\boldsymbol{Q}_2\boldsymbol{P}_2^{-1}) < 1$$

Proof. dimostriamolo ad esempio per la prima disuguaglianza. Per il teorema di Perron-Frobenius

$$\boldsymbol{A}^{-1}\boldsymbol{Q}_2 \geq \boldsymbol{A}^{-1}\boldsymbol{Q}_1 \geq 0 \quad \Longrightarrow \quad \varrho(\boldsymbol{A}^{-1}\boldsymbol{Q}_2) \geq \varrho(\boldsymbol{A}^{-1}\boldsymbol{Q}_1)$$

e per il teorema Teorema C.2 punto 5 segue

$$\varrho(\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1) = \frac{\varrho(\boldsymbol{A}^{-1}\boldsymbol{Q}_1)}{1+\varrho(\boldsymbol{A}^{-1}\boldsymbol{Q}_1)} \leq \frac{\varrho(\boldsymbol{A}^{-1}\boldsymbol{Q}_2)}{1+\varrho(\boldsymbol{A}^{-1}\boldsymbol{Q}_2)} = \varrho(\boldsymbol{P}_2^{-1}\boldsymbol{Q}_2)$$

in modo analogo si dimostrano le altre disuguaglianze. ■

Nel teorema Teorema C.3 il confronto dei raggi spettrali delle matrici di iterazione non è molto agevole perché ci sono di mezzo dei prodotti di matrici. Saranno quindi utili dei teoremi che confrontano direttamente le matrici $\boldsymbol{P}$ e $\boldsymbol{Q}$.

C.8 Raggio spettrale e splitting debolmente non negativi

Theorem C.4 Siano $\boldsymbol{A} = \boldsymbol{P}_1 - \boldsymbol{Q}_1 = \boldsymbol{P}_2 - \boldsymbol{Q}_2$ due *splitting debolmente non negativi*. Allora se $\boldsymbol{A}^{-1} \geq 0$, $\boldsymbol{Q}_2 \geq \boldsymbol{Q}_1$ segue

$$\varrho(\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1) \leq \varrho(\boldsymbol{P}_2^{-1}\boldsymbol{Q}_2) < 1$$

C.9 Splitting non negativi e convergenza

Theorem C.5 Siano $\boldsymbol{A} = \boldsymbol{P}_1 - \boldsymbol{Q}_1 = \boldsymbol{P}_2 - \boldsymbol{Q}_2$ due *splitting non negativi* del tipo $\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1, \boldsymbol{P}_2^{-1}\boldsymbol{Q}_2 \geq 0$ con $\boldsymbol{P}_1^{-1} \geq \boldsymbol{P}_2^{-1}$. Se $\boldsymbol{A}^{-1} > 0$ oppure $\boldsymbol{A}^{-1} \geq 0$ con $\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1$ e $\boldsymbol{P}_2^{-1}\boldsymbol{Q}_2$ irriducibili allora

$$\sigma(\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1) \leq \sigma(\boldsymbol{P}_2^{-1}\boldsymbol{Q}_2) < 1$$

se inoltre $\boldsymbol{P}_1^{-1} > \boldsymbol{P}_2^{-1}$

$$\sigma(\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1) < \sigma(\boldsymbol{P}_2^{-1}\boldsymbol{Q}_2) < 1$$

Proof. Basta osservare che

$$\boldsymbol{A}^{-1} = \boldsymbol{P}_1^{-1} + \boldsymbol{P}_1^{-1}\boldsymbol{Q}_1\boldsymbol{A}^{-1} = \boldsymbol{P}_2^{-1} + \boldsymbol{A}^{-1}\boldsymbol{Q}_2\boldsymbol{P}_2^{-1}$$

e quindi

$$\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1\boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}\boldsymbol{Q}_2\boldsymbol{P}_2^{-1} = \boldsymbol{P}_2^{-1} - \boldsymbol{P}_1^{-1} \leq 0$$

Siano $\boldsymbol{x}_1, \boldsymbol{x}_2 \geq 0$ i vettori di Frobenius[2] delle matrici $\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1$, $\boldsymbol{Q}_2\boldsymbol{P}_2^{-1}$ rispettivamente sinistro e destro allora

$$\boldsymbol{x}_1^T\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1\boldsymbol{A}^{-1}\boldsymbol{x}_2 \leq \boldsymbol{x}_1^T\boldsymbol{A}^{-1}\boldsymbol{Q}_2\boldsymbol{P}![2,-1]\boldsymbol{x}_2 \quad \Longrightarrow \sigma(\boldsymbol{P}![1,-1]\boldsymbol{Q}_1)\boldsymbol{x}_1^T\boldsymbol{A}^{-1}\boldsymbol{x}_2 \leq \sigma(\boldsymbol{Q}_2\boldsymbol{P}_2^{-1})\boldsymbol{x}_1^T\boldsymbol{A}^{-1}\boldsymbol{x}_2$$

Pertanto se $\boldsymbol{A}^{-1} > 0$ segue che $\boldsymbol{x}_1^T\boldsymbol{A}^{-1}\boldsymbol{x}_2 > 0$. Se $\boldsymbol{A}^{-1} \geq 0$ con $\boldsymbol{P}_1^{-1}\boldsymbol{Q}_1$ e $\boldsymbol{Q}_2\boldsymbol{P}_2^{-1}$ irriducibili si ha $\boldsymbol{x}_1, \boldsymbol{x}_2 > 0$ e quindi $\boldsymbol{x}_1^T\boldsymbol{A}^{-1}\boldsymbol{x}_2 > 0$. Osserviamo che se la disuguaglianza è soddisfatta in senso stretto allora varrà anche sui raggi spettrali. ∎

C.10 Confronto della velocità di convergenza fra alcuni metodi iterativi

Applichiamo ora i teoremi fin qui dimostrati allo studio della convergenza dei metodi iterativi. Dato il seguente sistema lineare

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

decomponiamo $\boldsymbol{A}$ come $\boldsymbol{A} = \boldsymbol{D} - \boldsymbol{L} - \boldsymbol{U}$, dove $\boldsymbol{D}$ è una matrice diagonale non singolare, $\boldsymbol{L}$ è triangolare inferiormente ed $\boldsymbol{U}$ è triangolare superiormente. Introduciamo altre due matrici: $\mathcal{L}$ triangolare inferiormente e $\mathcal{U}$ triangolare superiormente. Allora possiamo definire la fattorizzazione $\boldsymbol{A} = \boldsymbol{P} - \boldsymbol{Q}$ dove:

$$\boldsymbol{P} = \left[\boldsymbol{I} - (\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}\right] \boldsymbol{K} \left[\boldsymbol{I} - \boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U})\right]$$

[2] Ferdinand Georg Frobenius (1849-1917)

con $\boldsymbol{K}$ una matrice diagonale non singolare definita in maniera implicita dalla seguente relazione

$$\boldsymbol{K} = \boldsymbol{D} - \text{diag}\left\{(\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U})\right\} \tag{C.2}$$

dove con $diag(\boldsymbol{A})$ si indica la matrice diagonale ottenuta ponendo a zero gli elementi non diagonali di $\boldsymbol{A}$. La C.2 può essere riscritta come

$$\begin{aligned}
\boldsymbol{P} &= [\boldsymbol{I} - (\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{1}]\boldsymbol{K}[\boldsymbol{I} - \boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U})] \\
&= \boldsymbol{K} - (\boldsymbol{L} + \mathcal{L}) - (\boldsymbol{U} + \mathcal{U}) + (\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U}) \\
&= \boldsymbol{D} - \boldsymbol{L} - \boldsymbol{U} + \boldsymbol{K} - \boldsymbol{D} - \mathcal{L} - \mathcal{U} + (\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U}) \\
&= \boldsymbol{A} - \mathcal{L} - \mathcal{U} - \text{diag}\left\{(\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}\boldsymbol{U} + \mathcal{U})\right\} + (\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U}) \\
&= \boldsymbol{A} - \mathcal{L} - \mathcal{U} + \text{offdiag}\left\{(\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U})\right\}
\end{aligned} \tag{C.3}$$

dove $\text{offdiag}\{\boldsymbol{A}\} = \boldsymbol{A} - \text{diag}\{\boldsymbol{A}\}$. Dalla C.3 segue

$$\boldsymbol{Q} = \text{offdiag}\left\{(\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U})\right\} - \mathcal{L} - \mathcal{U} \tag{C.4}$$

Con queste matrici il metodo iterativo diventa

$$\boldsymbol{x}_{n+1} = [\boldsymbol{I} - \boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U})]^{-1}\boldsymbol{K}^{-1}[\boldsymbol{I} - (\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}](\boldsymbol{Q}\boldsymbol{x}![n] + \boldsymbol{b}) \tag{C.5}$$

che può essere decomposta nei due passi

$$\begin{aligned}
\boldsymbol{x}![n + 1/2] &= \boldsymbol{K}^{-1}\left[\boldsymbol{I} - (\boldsymbol{L} + \mathcal{L})\boldsymbol{K}^{-1}\right](\boldsymbol{Q}\boldsymbol{x}![n] + \boldsymbol{b}) \\
\boldsymbol{x}![n + 1] &= \left[\boldsymbol{I} - \boldsymbol{K}^{-1}(\boldsymbol{U} + \mathcal{U})\right]^{-1}\boldsymbol{x}![n + 1/2]
\end{aligned} \tag{C.6}$$

che sono facili da realizzare, poiché le matrici da invertire sono triangolari. Questo metodo, detto AGA, conduce, in base alla scelta delle matrici, alla costruzione di metodi iterativi classici, come i metodi di Jacobi[3], Gauss[4]-Seidel[5] ed EWA (si veda [45]).

Table C.1: Confronto metodi iterativi

	Jacobi	Gauss-Seidel	EWA
$\boldsymbol{K}$	$\boldsymbol{D}$	$\boldsymbol{D}$	$\boldsymbol{D}_{\boldsymbol{B}} - \text{diag}\left\{\boldsymbol{L}\boldsymbol{D}_{\boldsymbol{B}}^{-1}\boldsymbol{U}\right\}$
$\boldsymbol{L}$	$-\boldsymbol{L}$	$-\boldsymbol{L}$	$\boldsymbol{0}$
$\boldsymbol{U}$	$-\boldsymbol{U}$	$\boldsymbol{0}$	$\boldsymbol{0}$
$\boldsymbol{P}$	$\boldsymbol{D}$	$\boldsymbol{D} - \boldsymbol{L}$	$(\boldsymbol{I} - \boldsymbol{L}\boldsymbol{K}_{\boldsymbol{B}}^{-1})\boldsymbol{K}_{\boldsymbol{B}}(\boldsymbol{I} - \boldsymbol{K}_{\boldsymbol{B}}^{-1}\boldsymbol{U})$
$\boldsymbol{Q}$	$\boldsymbol{L} + \boldsymbol{U}$	$\boldsymbol{L}$	$\text{offdiag}\left\{\boldsymbol{L}\boldsymbol{D}_{\boldsymbol{B}}^{-1}\boldsymbol{U}\right\}$
$\boldsymbol{P}^{-1}$	$\boldsymbol{D}^{-1}$	$(\boldsymbol{D} - \boldsymbol{L})^{-1}$	$(\boldsymbol{I} - \boldsymbol{K}_{\boldsymbol{B}}^{-1}\boldsymbol{U})^{-1}\boldsymbol{K}_{\boldsymbol{B}}^{-1}(\boldsymbol{I} - \boldsymbol{L}\boldsymbol{K}_{\boldsymbol{B}}^{-1})^{-1}$
$\boldsymbol{P}^{-1}\boldsymbol{Q}$	$\boldsymbol{D}^{-1}(\boldsymbol{L} + \boldsymbol{U})$	$(\boldsymbol{D} - \boldsymbol{L})^{-1}\boldsymbol{U}$	$(\boldsymbol{I} - \boldsymbol{K}_{\boldsymbol{B}}^{-1}\boldsymbol{U})^{-1}\boldsymbol{K}_{\boldsymbol{B}}^{-1}(\boldsymbol{I} - \boldsymbol{L}\boldsymbol{K}_{\boldsymbol{B}}^{-1})^{-1}\text{offdiag}\left\{\boldsymbol{L}\boldsymbol{D}_{\boldsymbol{B}}^{-1}\boldsymbol{U}\right\}$

[3] Carl Gustav Jacob Jacobi (1804-1851)
[4] Carl Friedrich Gauss (1777-1855)
[5] Philipp Ludwig von Seidel (1821-1896)

C.10.1 Specializzazione nel caso a due colori

Sia $\boldsymbol{A}$ una matrice a due colori, cioè della forma

$$\boldsymbol{A} = \left(\begin{array}{c|c} \boldsymbol{D}_1 & -\boldsymbol{U} \\ \hline -\boldsymbol{L} & \boldsymbol{D}_2 \end{array}\right) \tag{C.7}$$

dove $\boldsymbol{D}_1$ e $\boldsymbol{D}_2$ sono matrici diagonali. La relazione C.3 diventa

$$\begin{aligned} \left(\begin{array}{c|c} \boldsymbol{K}_1 & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{K}_2 \end{array}\right) &= \left(\begin{array}{c|c} \boldsymbol{D}_1 & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{D}_2 \end{array}\right) - \operatorname{diag}\left\{\left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{0} \\ \hline \boldsymbol{L}+\mathcal{L} & \boldsymbol{0} \end{array}\right) \times \left(\begin{array}{c|c} \boldsymbol{K}_1^{-1} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{K}_2^{-1} \end{array}\right) \times \left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{U}+\mathcal{U} \\ \hline \boldsymbol{0} & \boldsymbol{0} \end{array}\right)\right\} \\ &= \left(\begin{array}{c|c} \boldsymbol{D}_1 & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{D}_2 \end{array}\right) - \operatorname{diag}\left\{\left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{0} \\ \hline \boldsymbol{0} & (\boldsymbol{L}+\mathcal{L})\boldsymbol{K}_1^{-1}(\boldsymbol{U}+\mathcal{U}) \end{array}\right)\right\} \\ &= \left(\begin{array}{c|c} \boldsymbol{D}_1 & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{D}_2 - \operatorname{diag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{K}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\} \end{array}\right) \end{aligned} \tag{C.8}$$

quindi la matrice diagonale $\boldsymbol{K}$ è facilmente calcolabile, inoltre la C.2 diventa

$$\begin{aligned} \boldsymbol{P} &= \left(\begin{array}{c|c} \boldsymbol{I}_1 & \boldsymbol{0} \\ \hline -(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{I}_2 \end{array}\right) \times \left(\begin{array}{c|c} \boldsymbol{D}_1 & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{K}_2 \end{array}\right) \times \left(\begin{array}{c|c} \boldsymbol{I}_1 & -\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U}) \\ \hline \boldsymbol{0} & \boldsymbol{I}_2 \end{array}\right) \\ &= \left(\begin{array}{c|c} \boldsymbol{D}_1 & -(\boldsymbol{U}+\mathcal{U}) \\ \hline -(\boldsymbol{L}+\mathcal{L}) & \boldsymbol{D}_2 - \operatorname{diag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\} + (\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U}) \end{array}\right) \\ &= \left(\begin{array}{c|c} \boldsymbol{D}_1 & -(\boldsymbol{U}+\mathcal{U}) \\ \hline -(\boldsymbol{L}+\mathcal{L}) & \boldsymbol{D}_2 + \operatorname{offdiag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\} \end{array}\right) \end{aligned} \tag{C.9}$$

per cui

$$\boldsymbol{Q} = \left(\begin{array}{c|c} \boldsymbol{0} & -\boldsymbol{U} \\ \hline -\boldsymbol{L} & \operatorname{offdiag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\} \end{array}\right) \tag{C.10}$$

se $\boldsymbol{K}$ è non singolare allora possiamo calcolare $\boldsymbol{P}^{-1}$ come segue

$$\boldsymbol{P}^{-1} = \left(\begin{array}{c|c} \boldsymbol{I}_1 & \boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U}) \\ \hline \boldsymbol{0} & \boldsymbol{I}[2] \end{array}\right) \times \left(\begin{array}{c|c} \boldsymbol{D}_1^{-1} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \left(\boldsymbol{D}_2 - \operatorname{diag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\}\right)^{-1} \end{array}\right) \times \left(\begin{array}{c|c} \boldsymbol{I}_1 & \boldsymbol{0} \\ \hline (\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{I}_2 \end{array}\right)$$

osserviamo che dalla condizione $\boldsymbol{Q} \geq \boldsymbol{0}$ segue che devono essere $\boldsymbol{L}, \boldsymbol{U} \leq \boldsymbol{0}$. Ci chiediamo ora sotto quali condizioni la matrice diagonale $\boldsymbol{K}$ è non singolare.

C.10.2 M-matrice e elementi diagonali

Theorem C.6 Sia

$$\boldsymbol{A} = \left(\begin{array}{c|c} \boldsymbol{D}_1 & -\boldsymbol{U} \\ \hline -\boldsymbol{L} & \boldsymbol{D}_2 \end{array}\right) \tag{C.11}$$

una M-matrice a due colori, allora se $-\boldsymbol{L} \leq \mathcal{L} \leq \boldsymbol{0}$, $-\boldsymbol{U} \leq \mathcal{U} \leq \boldsymbol{0}$, segue che la matrice

$$\boldsymbol{K}_2 = \boldsymbol{D}_2 - \operatorname{diag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\} \tag{C.12}$$

è non singolare con elementi diagonali strettamente positivi.

Proof. Osserviamo che la matrice

$$\begin{aligned}\boldsymbol{B} &= \left(\begin{array}{c|c} \boldsymbol{D}_1 & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{D}_2 - (\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U}) \end{array}\right) \\ &= \left(\begin{array}{c|c} \boldsymbol{I}_1 & \boldsymbol{0} \\ \hline (\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{I}_2 \end{array}\right)\left(\begin{array}{c|c} \boldsymbol{D}_1 & -(\boldsymbol{U}+\mathcal{U}) \\ \hline -(\boldsymbol{L}+\mathcal{L}) & \boldsymbol{D}_2 \end{array}\right)\left(\begin{array}{c|c} \boldsymbol{I}_1 & \boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U}) \\ \hline \boldsymbol{0} & \boldsymbol{I}_2 \end{array}\right)\end{aligned} \tag{C.13}$$

contiene la matrice $\boldsymbol{K}_2 = \boldsymbol{D}_2 - \operatorname{diag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\}$. Se proviamo che la parte diagonale della matrice $\boldsymbol{B}$ strettamente positiva abbiamo dimostrato il teorema. Poiché $\boldsymbol{A}$ è una M-matrice, dal lemma Lemma 25.3 segue che

$$\boldsymbol{A} = \rho\boldsymbol{I} - \boldsymbol{B} \tag{C.14}$$

dove

$$\rho = \|\operatorname{diag}\{\boldsymbol{A}\}\|_\infty, \qquad \boldsymbol{B} = \rho\boldsymbol{I} - \left(\begin{array}{c|c} \boldsymbol{D}_1 & -\boldsymbol{U} \\ \hline -\boldsymbol{L} & \boldsymbol{D}_2 \end{array}\right) = \left(\begin{array}{c|c} \rho\boldsymbol{I}_1 - \boldsymbol{D}_1 & \boldsymbol{U} \\ \hline \boldsymbol{L} & \rho\boldsymbol{I}_2 - \boldsymbol{D}_2 \end{array}\right) \geq \boldsymbol{0} \tag{C.15}$$

ed ancora $\rho > \varrho(\boldsymbol{B})$. Definendo

$$\boldsymbol{C} = \left(\begin{array}{c|c} \boldsymbol{D}_1 & -\boldsymbol{U}-\mathcal{U} \\ \hline -\boldsymbol{L}-\mathcal{L} & \boldsymbol{D}_2 \end{array}\right) \tag{C.16}$$

osserviamo che la matrice $\boldsymbol{C}$ si può scrivere come

$$\begin{aligned}\boldsymbol{C} &= \left(\begin{array}{c|c} \boldsymbol{D}_1 & -\boldsymbol{U}-\mathcal{U} \\ \hline -\boldsymbol{L}-\mathcal{L} & \boldsymbol{D}_2 \end{array}\right) \\ &= \rho\boldsymbol{I} - \left\{\rho\boldsymbol{I} - \left(\begin{array}{c|c} \boldsymbol{D}_1 & -\boldsymbol{U}-\mathcal{U} \\ \hline -\boldsymbol{L}-\mathcal{L} & \boldsymbol{D}_2 \end{array}\right)\right\} \\ &= \rho\boldsymbol{I} - \bar{\boldsymbol{B}}\end{aligned} \tag{C.17}$$

se vale la condizione $-\boldsymbol{L} \leq \mathcal{L} \leq \boldsymbol{0}$, $-\boldsymbol{U} \leq \mathcal{U} \leq \boldsymbol{0}$, allora abbiamo

$$\boldsymbol{0} \leq \bar{\boldsymbol{B}} = \left(\begin{array}{c|c} \rho\boldsymbol{I} - \boldsymbol{D}_1 & \boldsymbol{U}+\mathcal{U} \\ \hline \boldsymbol{L}+\mathcal{L} & \rho\boldsymbol{I} - \boldsymbol{D}_2 \end{array}\right) \leq \left(\begin{array}{c|c} \rho\boldsymbol{I} - \boldsymbol{D}_1 & \boldsymbol{U} \\ \hline \boldsymbol{L} & \rho\boldsymbol{I} - \boldsymbol{D}_2 \end{array}\right) = \boldsymbol{B} \tag{C.18}$$

e applicando il lemma Lemma C.2 otteniamo

$$\varsigma(\bar{\boldsymbol{B}}) \leq \varsigma(\boldsymbol{B}) < \rho \tag{C.19}$$

quindi, per il lemma Lemma 25.3, $\boldsymbol{C}$ è una M-matrice. Osserviamo che la matrice $\boldsymbol{D}_2 - (\boldsymbol{L} + \mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})$ ha elementi non diagonali non positivi. Sia

$$\boldsymbol{C}^{-1} = \left(\begin{array}{c|c} \boldsymbol{C}_{1,1} & \boldsymbol{C}_{1,2} \\ \hline \boldsymbol{C}_{2,1} & \boldsymbol{C}_{2,2} \end{array}\right) \geq \boldsymbol{0} \tag{C.20}$$

allora dalla C.13 segue

$$\begin{aligned}\boldsymbol{B}^{-1} &= \left(\begin{array}{c|c} \boldsymbol{D}_1^{-1} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \left(\boldsymbol{D}_2 - (\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right)^{-1} \end{array}\right) \\ &= \left(\begin{array}{c|c} \boldsymbol{I}_1 & -\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U}) \\ \hline \boldsymbol{0} & \boldsymbol{I}_2 \end{array}\right)\left(\begin{array}{c|c} \boldsymbol{C}_{1,1} & \boldsymbol{C}_{1,2} \\ \hline \boldsymbol{C}_{2,1} & \boldsymbol{C}_{2,2} \end{array}\right)\left(\begin{array}{c|c} \boldsymbol{I}_1 & \boldsymbol{0} \\ \hline -(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{I}_2 \end{array}\right) \\ &= \left(\begin{array}{c|c} \boldsymbol{C}_{1,1} - \boldsymbol{C}_{1,2}(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} - \boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\boldsymbol{C}_{1,2} + \boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\boldsymbol{C}_{2,2}(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{C}_{2,2} - \boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\boldsymbol{C}_{1,2} \\ \hline \boldsymbol{C}_{2,1} - \boldsymbol{C}_{2,2}(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{C}_{2,2} \end{array}\right.\end{aligned} \tag{C.21}$$

e di conseguenza $\boldsymbol{D}_2 - (\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U}) = \boldsymbol{C}_{2,2} \geq \boldsymbol{0}$, quindi $\boldsymbol{D}_2 - (\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})$ è una M-matrice e per il lemma Lemma 25.2 ha elementi diagonali strettamente positivi. ∎

Osserviamo che le matrici $\boldsymbol{Q}$ per i metodi di Jacobi, Gauss-Seidel e EWA diventano

$$\begin{aligned}
\boldsymbol{Q}_J &= \left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{U} \\ \hline \boldsymbol{U} & \boldsymbol{0} \end{array}\right), \\
\boldsymbol{Q}_{GS} &= \left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{0} \\ \hline \boldsymbol{L} & \boldsymbol{0} \end{array}\right), \\
\boldsymbol{Q}_{EWA} &= \left(\begin{array}{c|c} \boldsymbol{0} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \operatorname{offdiag}\left\{\boldsymbol{L}\boldsymbol{D}_1^{-1}\boldsymbol{U}\right\} \end{array}\right)
\end{aligned} \tag{C.22}$$

è immediato verificare che $\boldsymbol{Q}_J \geq \boldsymbol{Q}_{GS}$ e quindi applicando il teorema Teorema C.3 nel caso $\boldsymbol{A}$ sia una M-matrice si verifica il fatto noto che

$$\varrho(\boldsymbol{P}_{GS}^{-1}\boldsymbol{Q}_{GS}) \leq \varrho(\boldsymbol{P}_J^{-1}\boldsymbol{Q}_J) \tag{C.23}$$

purtroppo la matrice $\boldsymbol{Q}_{EWA}$ non è confrontabile con le matrici $\boldsymbol{Q}_J$ e $\boldsymbol{Q}_{GS}$. Possiamo però applicare il teorema Teorema C.4 sulle matrici

$$\begin{aligned}
\boldsymbol{P}_J^{-1} &= \left(\begin{array}{c|c} \boldsymbol{D}_1^{-1} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{D}_2^{-1} \end{array}\right), \\
\boldsymbol{P}_{GS}^{-1} &= \left(\begin{array}{c|c} \boldsymbol{D}_1^{-1} & \boldsymbol{D}_1^{-1}\boldsymbol{U}\boldsymbol{D}_2^{-1} \\ \hline \boldsymbol{0} & \boldsymbol{D}_2^{-1} \end{array}\right), \\
\boldsymbol{P}_{EWA}^{-1} &= \left(\begin{array}{c|c} \boldsymbol{D}_1^{-1} + \boldsymbol{D}_1^{-1}\boldsymbol{U}\boldsymbol{K}_{2,EWA}^{-1}\boldsymbol{L}\boldsymbol{D}_1^{-1} & \boldsymbol{D}_1^{-1}\boldsymbol{U}\boldsymbol{K}_{2,EWA}^{-1} \\ \hline \boldsymbol{K}_{2,EWA}^{-1}\boldsymbol{L}\boldsymbol{D}_1^{-1} & \boldsymbol{K}_{2,EWA}^{-1} \end{array}\right).
\end{aligned} \tag{C.24}$$

infatti

$$\boldsymbol{K}_{2,EWA}^{-1} = \boldsymbol{D}_2 - \operatorname{diag}\left\{\boldsymbol{L}\boldsymbol{D}_1^{-1}\boldsymbol{U}\right\} \leq \boldsymbol{D}_2 \tag{C.25}$$

e di conseguenza $\boldsymbol{K}_{2,EWA}^{-1} \geq \boldsymbol{D}_2^{-1}$ e ancora

$$\boldsymbol{P}_{EWA}^{-1} \geq \boldsymbol{P}_{GS}^{-1} \geq \boldsymbol{P}_J^{-1}, \quad \Longrightarrow \quad \varrho(\boldsymbol{P}_{EWA}^{-1}\boldsymbol{Q}_{EWA}) \leq \varrho(\boldsymbol{P}_{GS}^{-1}\boldsymbol{Q}_{GS}) \leq \varrho(\boldsymbol{P}_J^{-1}\boldsymbol{Q}_J) \tag{C.26}$$

C.10.3 AGA

Theorem C.7 Nel caso $\boldsymbol{L} \leq \mathcal{L} \leq \boldsymbol{0}, \boldsymbol{U} \leq \mathcal{U} \leq \boldsymbol{0}$, per un qualunque metodo AGA abbiamo:

$$\varrho(\boldsymbol{P}_{EWA}^{-1}\boldsymbol{Q}_{EWA}) \leq \varrho(\boldsymbol{P}_{AGA}^{-1}\boldsymbol{Q}_{AGA})$$

Proof. Osserviamo che

$$\boldsymbol{P}_{AGA}^{-1} = \left(\begin{array}{c|c} \boldsymbol{D}_1^{-1} + \boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\boldsymbol{K}_{2,AGA}^{-1}(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\boldsymbol{K}_{2,AGA}^{-1} \\ \hline \boldsymbol{K}_{2,AGA}^{-1}(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1} & \boldsymbol{K}_{2,AGA}^{-1} \end{array}\right) \tag{C.27}$$

e

$$\begin{aligned}
\boldsymbol{K}_{2,AGA} &= \boldsymbol{D}_2 - \operatorname{diag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\} \\
&= \boldsymbol{D}_2 - \operatorname{diag}\left\{\boldsymbol{L}\boldsymbol{D}_1^{-1}\boldsymbol{U}\right\} - \operatorname{diag}\left\{\mathcal{L}\boldsymbol{D}_1^{-1}(\boldsymbol{U}+\mathcal{U})\right\} - \operatorname{diag}\left\{(\boldsymbol{L}+\mathcal{L})\boldsymbol{D}_1^{-1}\mathcal{U}\right\} \\
&\geq \boldsymbol{D}_2 - \operatorname{diag}\left\{\boldsymbol{L}\boldsymbol{D}_1^{-1}\boldsymbol{U}\right\} = \boldsymbol{K}_{2,EWA}
\end{aligned}$$

di conseguenza $\boldsymbol{K}_{2,EWA}^{-1} \geq \boldsymbol{K}_{2,AGA}^{-1}$ in questo particolare caso. Vale inoltre

$$\boldsymbol{P}_{EWA}^{-1} \geq \boldsymbol{P}_{AGA}^{-1}$$

e per il teorema Teorema C.3

$$\varrho(\boldsymbol{P}_{EWA}^{-1}\boldsymbol{Q}_{EWA}) \leq \varrho(\boldsymbol{P}_{AGA}^{-1}\boldsymbol{Q}_{AGA})$$

■

Osservazione

Togliendo la condizione sufficiente $-\boldsymbol{L} \leq \mathcal{L} \leq \boldsymbol{0}$, $-\boldsymbol{U} \leq \mathcal{U} \leq \boldsymbol{0}$ otteniamo che in generale i metodi AGA (di cui EWA caso particolare) possono avere convergenza più elevata del metodo EWA. Per informazioni più dettagliate è utile leggere [45] e le referenze ivi contenute.

C.10.4 Confronto raggi spettrali

Theorem C.8 Sia $\boldsymbol{B} \in L(\mathbb{R}^n)$ e $\boldsymbol{C} \in L(\mathbb{C}^n)$. Allora se $|\boldsymbol{C}| \leq \boldsymbol{B}$ allora $\varrho(\boldsymbol{C}) \leq \varrho(\boldsymbol{B})$.

Proof. Sia $\sigma = \varrho(\boldsymbol{B})$ ed $\epsilon > 0$ arbitrario, se poniamo

$$\boldsymbol{B}_1 = (\sigma + \epsilon)^{-1}\boldsymbol{B}, \qquad \boldsymbol{C}_1 = (\sigma + \epsilon)^{-1}\boldsymbol{C},$$

ovviamente $\varrho(\boldsymbol{B}_1) < 1$ ed,

$$|\boldsymbol{C}_1|^k \leq |\boldsymbol{B}_1|^k \qquad k = 1, 2, \dots,$$

poiché $\lim_{k\mapsto\infty} \boldsymbol{B}_1^k = 0$ allora $\lim_{k\mapsto\infty} \left|\boldsymbol{C}_1^k\right| = 0$ e di conseguenza \ $\varrho(\boldsymbol{C}_1) < 1$ e $\varrho(\boldsymbol{C}) < \sigma + \epsilon$ ed essendo ϵ arbitrario $\varrho(\boldsymbol{C}) \leq \sigma$. ■

C.10.5 M-matrice caratterizzazione

Theorem C.9 Sia $\boldsymbol{A}_1 \in \mathcal{L}(\mathbb{R}^n, \mathbb{R}^n)$ una M-matrice. Posto $\boldsymbol{A}_1 = \boldsymbol{D}_1 - \boldsymbol{B}_1$ con $\boldsymbol{D}_1$ parte diagonale di $\boldsymbol{A}_1$, se $\boldsymbol{D}_2 \in \mathcal{L}(\mathbb{R}^n, \mathbb{R}^n)$ è diagonale e $\boldsymbol{D}_2 \geq 0$, $\boldsymbol{B}_2 \geq \boldsymbol{0}$, $\boldsymbol{B}_2 \leq \boldsymbol{B}_1$ allora,

$$\boldsymbol{A} = \boldsymbol{D}_1 + \boldsymbol{D}_2 - (\boldsymbol{B}_1 - \boldsymbol{B}_2),$$

è una M-matrice ed $\boldsymbol{A}^{-1} \leq \boldsymbol{A}_1^{-1}$.

Proof. Sia

$$\begin{aligned} \boldsymbol{D} &= \boldsymbol{D}_1 + \boldsymbol{D}_2, & \boldsymbol{B} &= \boldsymbol{B}_1 - \boldsymbol{B}_2, \\ \boldsymbol{H} &= \boldsymbol{D}^{-1}\boldsymbol{B}, & \boldsymbol{H}_1 &= \boldsymbol{D}_1^{-1}\boldsymbol{B}_1, \end{aligned}$$

allora

$$\boldsymbol{D} \geq \boldsymbol{D}_1 \quad \Longrightarrow \quad \boldsymbol{D}^{-1} \leq \boldsymbol{D}_1^{-1}$$

e inoltre,

$$0 \leq \boldsymbol{B} \leq \boldsymbol{B}_1 \ , \ 0 \leq \boldsymbol{H} \leq \boldsymbol{H}_1,$$

dal teorema Teorema C.8 si ha $\varrho(\boldsymbol{H}) \leq \varrho(\boldsymbol{H}_1) < 1$ ed allora il teorema Teorema 25.1 assicura che $\boldsymbol{A}$ è una M-matrice. Poiché $\boldsymbol{A} \geq \boldsymbol{A}_1$ allora moltiplicando a destra e a sinistra per $\boldsymbol{A}^{-1}$ ed $\boldsymbol{A}_1^{-1}$ (si può perché sono entrambe positive e invertibili).

$$\boldsymbol{A}^{-1}\boldsymbol{A}\boldsymbol{A}_1^{-1} \geq \boldsymbol{A}^{-1}\boldsymbol{A}_1^{-1}\boldsymbol{A}_1^{-1},$$
$$\boldsymbol{A}_1^{-1} \geq \boldsymbol{A}^{-1},$$

■

Corollary C.1 Sia $\boldsymbol{A} \in \mathcal{L}(\mathbb{R}^n, \mathbb{R}^n)$ una matrice di Stieltjes. Allora $\boldsymbol{A}$ è definita positiva.

Proof. Sia λ autovalore di $\boldsymbol{A}$ con $\lambda \leq 0$ allora se poniamo $\boldsymbol{B}_2 = 0$ e $\boldsymbol{D}_2 = -\lambda\boldsymbol{I}$ avremo che $\boldsymbol{A} - \lambda\boldsymbol{I}$ è una M-matrice perché contraddice la singolarità di $\boldsymbol{A} - \lambda\boldsymbol{I}$. ■

C.10.6 Mappa isotona

Definition C.3 Una mappa $\boldsymbol{F} : D \subset \mathbb{R}^n \mapsto \mathbb{R}^n$ è detta *isotona* (antitona) su $D_o \subset D$ se $\boldsymbol{F}(\boldsymbol{x}) \leq \boldsymbol{F}(\boldsymbol{y})(\boldsymbol{F}(\boldsymbol{x}) \geq \boldsymbol{F}(\boldsymbol{y}))$ quando $\boldsymbol{x} \leq \boldsymbol{y}\ \forall\, \boldsymbol{x}, \boldsymbol{y} \in D_o$.



Indice analitico

algoritmo di Aitken-Neville, 104, 105
algoritmo di Clenshaw, 187
algoritmo di Golub-Welsch, 225
armoniche sferiche, 235

barriera di Dahlquist, 404

cancellazione numerica, 65, 247, 254
chaos polinomiale, 235
collocazione spettrale, 234
complemento di Schur, 34, 40, 42, 44
condizione di Haar, 84
condizione di Lipschitz, 341, 343, 345, 360
convergenza
    lineare, 274
    locale, 248
    quadratica, 274, 276
costante di Lebesgue, 93, 109

delta di Kronecker, 58, 89
difetto locale, 357
differenze divise, 97, 98, 100, 102, 103
dipendenza continua dai dati, 346, 347
disuguaglianza
    di Cauchy-Schwarz, 186

eliminazione di Gauss, 19, 20, 22, 23, 31
equazione
    test di Dahlquist, 401, 403, 404
equazioni normali, 191
equioscillazione, 172
errore
    di dispersione, 406
    di dissipazione, 406
    di integrazione, 120, 122, 125
    di interpolazione, 85, 108, 109

fattorizzazione
    di Cholesky, 37, 44
    LQ, 72
    LU, 30, 31, 33
    QR, 53, 57, 61, 63, 65, 71, 72
fenomeno di Gibbs, 232
fenomeno di Runge, 93, 209
fill-in, 35, 37
formula di Christoffel-Darboux, 419
formula di Gelfand, 321, 433
formula di Rodrigues, 161
formule di quadratura, 117, 118, 120–122, 129–131, 137, 145, 146
    aperte, 118
    chiuse, 118
    dei trapezi, 123, 131, 142, 359, 403, 404, 406, 408
    di Gauss-Jacobi, 227
    di Gauss-Legendre, 119, 140, 223, 396, 404
    di Gauss-Lobatto, 119, 141, 229
    di Gauss-Radau, 119, 142, 229
    di Newton-Cotes, 118, 119, 125, 140, 223
    di Simpson, 123, 125, 131, 132, 142
    gaussiane, 118, 221
funzione
    di incremento, 356
    di stabilità, 401–404, 406
    Gamma, 169
funzione di Christoffel, 187
funzione generatrice, 165
funzione peso, 157

identità di Pitagora, 201
integrazione
    adattativa, 146
    composita, 129

interlacciamento degli zeri, 183
interpolazione
  di Lagrange, 89, 91, 92, 221
  di Newton, 95
iterate di Picard, 345
iterazione di punto fisso, 239, 303–306

lemma
  di Grönwall, 344

M-matrice, 322, 334, 438, 441
mappa isotona, 442
matrice
  di Frobenius, 24–27, 30, 32, 43
  di Gram, 158, 165
  di Hankel, 159
  di Hilbert, 160
  di iterazione, 328, 332, 333, 432
  di Jacobi, 181
  di scambio, 24, 28, 32
  di Vandermonde, 82, 190
  irriducibile, 425, 427, 428, 432
  Jacobiana, 373
  monotona, 432–435
  simmetrica definita positiva, 39, 42, 442
  sparsa, 44
  tridiagonale, 181
metodi di collocazione, 395
metodi di Runge-Kutta, 383, 385–388, 390, 393, 395, 402–404, 408
metodi spettrali, 231
metodo
  della parabola tangente, 255
  delle secanti, 239, 247, 249
  di bisezione, 239–241, 251, 313, 315
  di Crank-Nicolson, 359, 360, 395, 403
  di Eulero esplicito, 350, 351, 354, 356, 359, 360, 365, 394, 402, 405, 406
  di Eulero implicito, 350, 352, 353, 358–360, 394, 403, 404, 406
  di Gauss-Seidel, 323, 325, 328, 332, 334
  di Halley, 257, 307
  di Jacobi, 323, 324, 326, 328, 331, 334, 437, 440
  di Newton, 96, 102, 103, 239, 244, 274, 304–306, 309
  SOR, 325–328, 332
minimi quadrati, 49, 53, 54, 57, 190
minori
  principali di testa, 40
modello
  preda-predatore, 374
  SIR, 377
momenti del peso, 157

nodi
  di Chebyshev, 108, 109, 171
  di Chebyshev-Lobatto, 108, 234
  di interpolazione, 83, 84, 89, 93, 103, 104, 119
  equispaziati, 86, 92, 108
norma
  uniforme, 171

ordine di accuratezza, 121, 125
ordine di convergenza, 362
ortogonalizzazione di Gram-Schmidt, 158

pivoting, 23, 31–33, 35, 37, 44
  parziale, 31, 33, 34
  pivot numerico, 34, 44
  totale, 33, 34
polinomi del nucleo, 185
polinomi ortogonali, 118, 158
  di Chebyshev, 171
  di Gegenbauer, 174
  di Hermite, 174, 235
  di Jacobi, 169
  di Laguerre, 174
  di Legendre, 161
polinomio
  di Bernstein, 79
  di miglior approssimazione, 93
  monico, 171
problema
  stiff, 360, 401, 405, 406, 408
problema a valore iniziale, 341, 343
problema di Cauchy, 341, 347, 383
prodotto scalare pesato, 157
proiezione ortogonale, 215, 420
pseudo-inversa di Moore-Penrose, 69–72

radici multiple, 274, 276, 307, 309
raggio spettrale, 321, 332, 334, 425, 427, 429, 431, 432, 436
regola di Leibniz, 374
regula falsi, 239, 247, 251, 252
retta di regressione, 50, 52
ricorrenza
  all'indietro di Miller, 187
ricorrenza a tre termini, 162, 177, 179
riflessione di Householder, 59, 65
rotazione di Givens, 58, 59, 61, 63

Runge-Kutta
coppie annidate, 391

serie di Neumann, 321, 402
serie di Taylor, 245, 257, 350, 352, 353, 365, 366, 368, 369, 372, 384, 386
sostituzione
all'indietro, 19, 21–23
in avanti, 102
sovradattamento, 209
splitting, 322, 324, 326, 334, 425, 431–436
debole, 432, 433
debolmente non negativo, 432, 435
non negativo, 432
regolare, 432
stabilità
A-stabilità, 404, 408
assoluta, 347, 401, 402, 405
L-stabilità, 404, 406, 408
regione di, 401–403
successione
monotona, 253, 266
successione di Sturm, 313, 314

tableau di Butcher, 388, 402
tasso asintotico di convergenza, 333
teorema
delle contrazioni, 346
di Faber, 93
di Favard, 180
di interlacciamento di Cauchy, 183
di Perron-Frobenius, 425, 428, 432, 434, 435
di prolungamento, 347
di Rolle, 85, 123
di Weierstrass, 107, 346