Radici di polinomi, tridiagonalizzazione di matrici e metodo di Arnoldi

Luigi Brugnano

http://www.math.unifi.it/~brugnano

Università degli Studi di Firenze

Problema

Determinare le radici di

$$p(z) = 0, \qquad (1)$$

dove: $p(z) = \sum_{i=0}^{n} c_i z^i \in \Pi_n$.

- Il problema è noto sin dai Sumeri (3000 a.C.)
- È stato risolto dal punto di vista teorico solo nel XVIII secolo.
- Teorema fondamentale dell'Algebra. L'equazione (1) ammette esattamente n radici in $\mathbb{C}$.

Metodi numerici

Per l'approssimzaione delle radici di $p(z)$ sono stati proposti numerosi metodi di approssimazione numerica:

- Metodi di omotopia.
- Metodi che trasformano il problema in un sistema equivalente di equazioni algebriche.
- Metodi che trasformano il problema in un problema equivalente di autovalori.

Metodi di omotopia

Es.:

$$f(z,\theta) \;=\; \theta p(z) + (1-\theta)(z^n - 1), \qquad \theta \in [0,1].$$

- Per $\theta \;=\; 0$ la soluzione è nota;
- al variare di θ le radici $z_i(\theta)$ descrivono delle traiettorie tali che $z_i(1) \;\equiv\; z_i$, radice di $p(z), \quad i \;=\; 1,\ldots,n$;
- cruciale dipendenza dai punti iniziali (ovvero dalla scelta del polinomio di cui sono note le radici).
- problemi in caso di radici multiple.

Metodi algebrici

Questi metodi utilizzano essenzialmente il metodo di Newton per determinare la soluzione di sistemi algebrici equivalenti al problema.

Es.:

$$p(z) \;=\; (z - z_1)(z - z_2) \;\equiv\; z^2 + c_1 z + c_0 \;=\; 0$$

- $$\begin{cases} x_1 + x_2 + c_1 &= 0, \\ x_1 x_2 - c_0 &= 0, \end{cases}$$

  $$\begin{cases} p[x_1] &= 0, \\ p[x_1, x_2] &= 0. \end{cases}$$

- problema della scelta del punto iniziale;
- problemi in caso di radici multiple.

Metodi agli autovalori

Es.:

$$p(z) \equiv z^n + c_{n-1}z^{n-1} + \cdots + c_1 z + c_0 = 0.$$

Le radici di $p(z)$ sono gli autovalori della matrice di compagnia

$$C = \begin{pmatrix} -c_{n-1} & 1 & & \\ -c_{n-2} & & 1 & \\ \vdots & & & \ddots \\ -c_1 & & & 1 \\ -c_0 & & & \end{pmatrix}$$

Metodi agli autovalori

- semplice da implementare;
- non necessita di punti iniziali (è semplicemente l'applicazione del metodo QR per gli autovalori);
- l'algoritmo risultante è robusto (ad es., `roots` di Matlab);
- problemi, tuttavia, nel caso di radici multiple.

Es: $p(z) = (z-1)^{10}$

```
roots(p)
ans =
   1.0476e+000
   1.0381e+000 +2.8333e-002i
   1.0381e+000 -2.8333e-002i
   1.0137e+000 +4.5003e-002i
   1.0137e+000 -4.5003e-002i
   9.8477e-001 +4.3882e-002i
   9.8477e-001 -4.3882e-002i
   9.6250e-001 +2.6523e-002i
   9.6250e-001 -2.6523e-002i

   9.5427e-001
```

Perché?

Se al posto di

$$z^n = 0$$

avessi

$$z^n = \varepsilon$$

gli zeri diverrebbero le radici dell'unità moltiplicate per

$$\sqrt[n]{\varepsilon}.$$

Ad esempio, per

$$\varepsilon \approx 2.2 \cdot 10^{-16}$$

la perturbazione vale pressappoco

$$2.7 \cdot 10^{-2}.$$

L'algoritmo euclideo

Assegnato

$$p_0(z) \equiv p(z) = \sum_{i=0}^{n} c_i z^i, \qquad c_n = 1,$$

ed un ulteriore polinomio $p_1(z) \in \Pi_{n-1}$, possiamo definire la successione

$$\begin{aligned} p_0(z) &= q_1(z)p_1(z) - p_2(z) \\ p_1(z) &= q_2(z)p_2(z) - p_3(z) \\ &\vdots \\ p_{m-2}(z) &= q_{m-1}(z)p_{m-1}(z) - p_m(z) \\ p_{m-1}(z) &= q_m(z)p_m(z) \end{aligned} \qquad (2)$$

dove $m \leq n$ e $0 \leq \deg(p_{j+1}) < \deg(p_j)$.

L'algoritmo euclideo

$p_m(z)$ è un fattore comune di $p_0(z), \ldots, p_{m-1}(z)$:

- è il massimo comun divisore di $p_0(z)$ e $p_1(z)$;
- le funzioni $f_i(z) = \frac{p_i(z)}{p_m(z)}$, $i = 0, \ldots, m$, sono polinomi;
- Teorema. Se

$$p_1(z) = \alpha \cdot p_0'(z)$$

allora z^* è una radice di molteplicità k per $p_0(z)$ se e solo se z^* è una radice di molteplicità $k-1$ per $p_m(z)$.

Caso particolare

Se $p_0(z)$ è un polinomio monico, posso modificare la procedura (2) in modo da avere tutti i polinomi $p_i(z)$ monici. In particolare, se:

$$\begin{aligned} p_0(z) &= (z-\alpha_1)p_1(z) - \beta_1 p_2(z) \\ p_1(z) &= (z-\alpha_2)p_2(z) - \beta_2 p_3(z) \\ &\vdots \\ p_{m-2}(z) &= (z-\alpha_{m-1})p_{m-1}(z) - \beta_{m-1}p_m(z) \\ p_{m-1}(z) &= (z-\alpha_m)p_m(z), \end{aligned} \qquad (2.1)$$

con $\deg(p_i) = n-i,\ i = 0,\ldots,m \leq n$, diremo che la procedura termina regolarmente.

Terminazione regolare

Scegliendo

$$p_1(z) = \frac{p_0'(z)}{n},$$

allora:

- $m = n$: in tal caso $p_n(z) \equiv 1$. Pertanto le radici di $p(z)$ sono tutte semplici.
  In tal caso il metodo QR applicato alla matrice di compagnia funziona egregiamente.
- $m < n$: in tal caso, le radici del polinomio

$$\frac{p_0(z)}{p_m(z)}$$

sono semplici: sono tutte e sole le radici distinte di $p(z)$.

Forma matriciale

Se

$$T_m = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{m-1} \\ & & 1 & \alpha_m \end{pmatrix},$$

si ottiene

$$x \begin{pmatrix} p_1(x) \\ p_2(x) \\ \vdots \\ p_m(x) \end{pmatrix} = T_m \begin{pmatrix} p_1(x) \\ p_2(x) \\ \vdots \\ p_m(x) \end{pmatrix} + \begin{pmatrix} p_0(x) \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Forma matriciale

Teorema 1. x è autovalore di T_m se e solo se x è radice di

$$\frac{p_0(z)}{p_m(z)}.$$

Teorema 2. Se T_m è reale, allora $\beta_i > 0,\ i = 1, \ldots, m-1$ se e solo se tutti gli autovalori di T_m sono reali.

Osservazione. Gli autovalori di T_m sono tutti distinti.

Conseguenza. Il metodo QR per autovalori risulta efficace.

Conclusioni

In caso di terminazione regolare, applicando il metodo QR alla matrice T_m, si determinano le radici (semplici) di

$$\frac{p_0(z)}{p_m(z)}.$$

Se $m < n$, l'intera procedura si riapplica a $p_m(z)$.

Teorema. z è radice di molteplicità ν per $p(z)$ se e solo se z è radice di molteplicità $\nu - 1$ per $p_m(z)$.

Es: $p(z) = (z-1)^{10}$

In questo caso, ponendo $p_0(z) \equiv p(z), p_1(z) \equiv \frac{p_0'(z)}{10}$, si ottiene:

- $p_0(z) \equiv p(z) = (z-1)p_1(z)$, quindi $z = 1$ è radice;
- ponendo $p_2(z) \equiv \frac{p_1'(z)}{9}$, si ottiene $p_1(z) = (z-1)p_2(z)$, quindi $z = 1$ è radice;
- ponendo $p_3(z) \equiv \frac{p_2'(z)}{8}$, si ottiene $p_2(z) = (z-1)p_3(z)$, quindi $z = 1$ è radice;

  $\vdots$
- ponendo $p_9(z) \equiv \frac{p_8'(z)}{2}$, si ottiene $p_9(z) = (z-1)p_{10}(z) \equiv (z-1)$, quindi $z = 1$ è radice.
- In questo caso, $m = 1$ ad ogni passo intermedio.

Breakdown

Se l'algoritmo euclideo non termina regolarmente, allora:

$$\begin{aligned} p_0(z) &= (z-\alpha_1)p_1(z) - \beta_1 p_2(z) \\ &\vdots \\ p_{r-2}(z) &= (z-\alpha_{r-1})p_{r-1}(z) - \beta_{r-1}p_r(z) \\ p_{r-1}(z) &= (z-\alpha_r)p_r(z) + p_{r+1}(z), \end{aligned} \qquad (3.1)$$

con

$$\deg(p_i) = n-i, \qquad i = 0,\ldots,r < n,$$

e

$$\deg(p_{r+1}) < n-r-1.$$

In tal caso, parleremo di breakdown della procedura.

Breakdown

Posso tuttavia *switchare* alla procedura (2) senza normalizzazione, ottenendo:

$$\begin{aligned} p_r(z) &= q_{r+1}(z)p_{r+1}(z) - p_{r+2}(z) \\ &\vdots \\ p_{m-2}(z) &= q_{m-1}(z)p_{m-1}(z) - p_m(z) \\ p_{m-1}(z) &= q_m(z)p_m(z). \end{aligned} \tag{3.2}$$

La procedura (3.1)-(3.2) terminerà in al più n passi.

Breakdown

Riguardo alla procedura (3.1)-(3.2), ho due possibilità:

- $\deg(p_m(z)) = 0$: in questo caso $p(z)$ ha solo radici semplici. Pertanto il QR funziona “bene".
- $\deg(p_m(z)) > 0$: in questo caso, il polinomio

$$\frac{p_0(z)}{p_m(z)}$$

ha radici semplici.

Posso utilizzare il metodo QR per determinarne le radici in modo affidabile.

Riapplico l’intera procedura al polinomio $p_m(z)$.

Conclusioni

- La procedura (2.1), ovvero la (3.1)-(3.2), è concettualmente assai semplice.
- È agevolmente implementabile utilizzando strumenti di calcolo simbolico.
- Possono tuttavia aversi problemi legati all'utilizzo dell'aritmetica finita:

  può essere talora difficile distinguere, a causa degli errori di *round off*, la terminazione regolare da una condizione di *breakdown*.

L.Brugnano, D.Trigiante. Polynomial roots: the ultimate answer? *Lin. Alg. Appl.* 225 (1995) 207–219.

Sviluppi ulteriori

- È possibile rendere numericamente più affidabile la procedura descritta?

- Alcuni risultati sono stati ottenuti.

  L.Brugnano. Numerical implementation of a new algorithm for polynomials with multiple roots. *J. Difference Eq. Appl.* 1 (1995) 187–207.

- Recenti nuovi risultati (preliminari).

Esempio

$$p(z) \;=\; z^4 - 10z^3 + 35z^2 - 50z + 24.$$

Applicando l'algoritmo euclideo con $p_1(z) \;\equiv\; p'(z)/4$,, si ottengono i polinomi riportati nelle colonne della seguente matrice:

$$V \;=\; \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{15}{2} & 1 & 0 & 0 \\ \frac{35}{2} & -5 & 1 & 0 \\ -\frac{25}{2} & \frac{29}{5} & -\frac{5}{2} & 1 \end{pmatrix}.$$

La matrice V è una matrice triangolare inferiore con elementi diagonali unitari.

Esempio

Consideriamo ora la matrice di compagnia C associata al polinomio $p(z)$:

$$C = \begin{pmatrix} 10 & 1 & 0 & 0 \\ -35 & 0 & 1 & 0 \\ 50 & 0 & 0 & 1 \\ -24 & 0 & 0 & 0 \end{pmatrix}.$$

Si ottiene: $V^{-1}CV = T^T$,

$$T = \begin{pmatrix} \frac{5}{2} & \frac{5}{4} & & \\ 1 & \frac{5}{2} & \frac{4}{5} & \\ & 1 & \frac{5}{2} & \frac{9}{20} \\ & & 1 & \frac{5}{2} \end{pmatrix} \equiv \begin{pmatrix} \alpha_1 & \beta_1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_3 \\ & & 1 & \alpha_4 \end{pmatrix}$$

Lanczos *connection*

Si può verificare che la precedente procedura equivale al metodo di Lanczos nonsimmetrico:

$$\begin{aligned} CV &= VT^T, \\ C^T W &= WT, \\ W^T V &= I. \end{aligned}$$

Se fissiamo

$$T = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & & \end{pmatrix}, \qquad \begin{aligned} V &= [\mathbf{v}_1, \mathbf{v}_2, \ldots], \\ W &= [\mathbf{w}_1, \mathbf{w}_2, \ldots], \end{aligned}$$

Lanczos *connection*

essa equivale a determinare vettori che soddisfano le seguenti relazioni di ricorenza a 3 termini:

$$\begin{aligned} C\mathbf{v}_i &= \mathbf{v}_{i-1} + \alpha_i \mathbf{v}_i + \beta_i \mathbf{v}_{i+1}, \\ C^T \mathbf{w}_i &= \beta_{i-1} \mathbf{w}_{i-1} + \alpha_i \mathbf{w}_i + \mathbf{w}_{i+1}, \end{aligned}$$

soggetti alla condizione di bi-ortogonalità:

$$\mathbf{w}_i^T \mathbf{v}_j = \begin{cases} 1, & i = j, \\ 0, & i \neq j. \end{cases}$$

Lanczos *connection*

Questi sono generati dalla procedura:

siano assegnati $\mathbf{v}_1$ e $\mathbf{w}_1$ (per convenzione, $\mathbf{v}_0 = \mathbf{w}_0 = 0$)

per $i = 1, 2, \ldots$

$\alpha_i = \mathbf{w}_i^T C \mathbf{v}_i$

$\mathbf{w}_{i+1} = C^T \mathbf{w}_i - \alpha_i \mathbf{w}_i - \beta_{i-1} \mathbf{w}_{i-1}$

$\tilde{\mathbf{v}}_{i+1} = C \mathbf{v}_i - \alpha_i \mathbf{v}_i - \mathbf{v}_{i-1}$

se $\|\tilde{\mathbf{v}}_{i+1}\| = 0$, trovato fattore, esci, fine se

$\beta_i = \mathbf{e}_{i+1}^T \tilde{\mathbf{v}}_{i+1}$

se $\beta_i = 0$, breakdown, esci, fine se

$\mathbf{v}_{i+1} = \tilde{\mathbf{v}}_{i+1} / \beta_i$

fine per

Lanczos *connection*

Riguardo alla scelta di $\mathbf{v}_1$ e $\mathbf{w}_1$:

- $\mathbf{v}_1$ contiene i coefficienti del polinomio

$$p_1(z) \equiv \frac{p'(z)}{n},$$

  per potenze decrescenti;

- $\mathbf{w}_1$ coincide con il primo versore, $\mathbf{e}_1$, di $\mathbb{R}^n$.

Lanczos *connection*

Tenendo conto che i vettori $\{\mathbf{v}_i\}$ sono le colonne di una matrice triangolare inferiore a diagonale unitaria, la relazione di ricorrenza diviene:

$$C\mathbf{v}_i \;=\; \mathbf{v}_{i-1} + \alpha_i \mathbf{v}_i + \beta_i \mathbf{v}_{i+1},$$

dove gli scalari α_i e β_i sono determinati imponendo che il nuovo vettore $\mathbf{v}_{i+1}$ sia la $(i+1)$-esima colonna di una matrice triangolare a diagonale unitaria. Ovvero:

$$\mathbf{e}_i^T \mathbf{v}_{i+1} \;=\; 0, \qquad \mathbf{e}_{i+1}^T \mathbf{v}_{i+1} \;=\; 1.$$

Lanczos *connection*

La procedura che si ottiene, infine, è:

assegnato $\mathbf{v}_1$ e posto $\mathbf{v}_0 = 0$, (4)

per $i = 1, 2, \ldots$

$\alpha_i = \mathbf{e}_i^T (C\mathbf{v}_i - \mathbf{v}_{i-1})$

$\tilde{\mathbf{v}}_{i+1} = C\mathbf{v}_i - \alpha_i \mathbf{v}_i - \mathbf{v}_{i-1}$

se $\|\tilde{\mathbf{v}}_{i+1}\| = 0$, trovato fattore, esci, fine se

$\beta_i = \mathbf{e}_{i+1}^T \tilde{\mathbf{v}}_{i+1}$

se $\beta_i = 0$, breakdown, esci, fine se

$\mathbf{v}_{i+1} = \tilde{\mathbf{v}}_{i+1} / \beta_i$

fine per

I vettori $\{\mathbf{w}_i\}$ non sono più necessari.

Lanczos *connection*

Nel caso di uscita dal ciclo prima di aver costruito per intero la matrice V, si ha:

- Nel caso in cui si è trovato un fattore, il vettore $\mathbf{v}_i$ conterrà, dalla posizione i-esima, i coefficienti di tale fattore, cui si riapplica la procedura vista.
  Le radici distinte di $p(z)$ sono gli autovalori della matrice tridiagonale $i \times i$:

$$T_i = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{i-1} \\ & & 1 & \alpha_i \end{pmatrix}.$$

Lanczos *connection*

- Nel caso di breakdown:
  - questa condizione è noto possa verificarsi durante il metodo di Lanczos nonsimmetrico;
  - è anche noto che possa verificarsi nella tridiagonalizzazione di una matrice non hermitiana;
  - nell'ambito dei metodi iterativi, si pone ad esso rimedio mediante tecniche di *look ahead*;
  - equivalenti a trasformare la matrice assegnata in una matrice non tridiagonale ma tridiagonale a blocchi;
  - questo significa che i vettori $\{\mathbf{v}_i\}$ non soddisfano più, almeno localmente, una relazione a 3 termini.

Breakdown

In questo caso, introduciamo i seguenti *breakdown vectors*, definiti come segue:

- siano $\mathbf{v}_1, \ldots, \mathbf{v}_i$ generati dalla procedura (4), con $\mathbf{v}_j$ avente i primi $j-1$ elementi nulli e l'elemento j-esimo uguale a 1, $j = 1, \ldots, i$;
- sia $\mathbf{e}_{i+1}^T \tilde{\mathbf{v}}_{i+1} = 0$ ma $\tilde{\mathbf{v}}_{i+1} \neq 0$;
- sia $\mathbf{e}_{i+1+k}^T \tilde{\mathbf{v}}_{i+1} \equiv \beta_i$ la prima componente non nulla di $\tilde{\mathbf{v}}_{i+1}$;
- definiamo: $$\mathbf{v}_{i+1+k} = \tilde{\mathbf{v}}_{i+1}/\beta_i,$$ $$\mathbf{v}_{i+j} = C\mathbf{v}_{i+j+1}, \quad j = k, k-1, \ldots, 1;$$
- osserviamo che i vettori fino ad ora generati, $[\mathbf{v}_1, \ldots, \mathbf{v}_{i+k+1}]$, definiscono ancora una matrice (rettangolare) triangolare inferiore con diagonale unitaria;

Breakdown

- definiamo, ora, la relazione a $k+3$ termini che permetterà di generare $\mathbf{v}_{i+k+2}$:

$$C\mathbf{v}_{i+1} = \mathbf{v}_i + \sum_{j=0}^{k} \gamma_j \mathbf{v}_{i+1+j} + \beta_{i+k+1}\mathbf{v}_{i+k+2},$$

- i coefficienti γ_j e β_{i+k+1} sono determinati imponendo, nell'ordine, le condizioni:

$$\begin{aligned} \mathbf{e}_{i+1+j}^T \mathbf{v}_{i+k+2} &= 0, \qquad j = 0, \ldots, k, \\ \mathbf{e}_{i+k+2}^T \mathbf{v}_{i+k+2} &= 1. \end{aligned}$$

- i rimanenti vettori, $\mathbf{v}_{i+k+3}, \ldots, \mathbf{v}_n$, sono generati nuovamente dalla (4).

Breakdown

In questo modo:

- la matrice $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$, è ancora triangolare inferiore a diagonale unitaria;
- la matrice C risulta essere simile ad una matrice tridiagonale a blocchi $(T')^T$.

$$T' = \left[\begin{array}{ccccc|ccccc|ccccc}
\alpha_1 & \beta_1 & & & & & & & & & & & & & \\
1 & \ddots & \ddots & & & & & & & & & & & & \\
 & \ddots & \ddots & \beta_{i-1} & & & & & & & & & & & \\
 & & 1 & \alpha_i & & 0 & \dots & 0 & \beta_i & & & & & & \\
\hline
 & & & 1 & & \gamma_0 & \gamma_1 & \dots & \gamma_k & & \beta_{i+k+1} & & & & \\
 & & & & & 1 & 0 & & & & & & & & \\
 & & & & & & \ddots & \ddots & & & & & & & \\
 & & & & & & & 1 & 0 & & & & & & \\
\hline
 & & & & & & & & 1 & & \alpha_{i+k+2} & \beta_{i+k+2} & & & \\
 & & & & & & & & & & 1 & \ddots & \ddots & & \\
 & & & & & & & & & & & \ddots & \ddots & \beta_{n-1} & \\
 & & & & & & & & & & & & 1 & \alpha_n &
\end{array}\right]$$

Es: $p(z) = z^6 + z^5 + z - 1.$

$$T' = \left(\begin{array}{cc|cc|cc} -\frac{1}{6} & \frac{5}{36} & & & & \\ 1 & -\frac{5}{6} & 0 & -6 & & \\ \hline & & 1 & -\frac{2}{5} & -\frac{29}{25} & \frac{642}{125} \\ & & & 1 & 0 & \\ \hline & & & & 1 & -\frac{19}{15} \quad -\frac{10}{9} \\ & & & & & 1 \quad \frac{5}{3} \end{array}\right),$$

$$V = \left(\begin{array}{cc|c|cc} 1 & & & & \\ 5/6 & 1 & & & \\ \hline 0 & 0 & 1 & & \\ 0 & 0 & -2/5 & 1 & \\ \hline 0 & -6 & -1 & -2/5 & 1 \\ 1/6 & 37/5 & 0 & -1 & -5/3 \quad 1 \end{array}\right).$$

Condizionamento

Esaminiamo le proprietà di stabilità della successione $\{\mathbf{v}_i\}$ generata dalla procedura (4).

Per semplicità supponiamo che non avvengano *breakdown*.

Evidentemente, la successione appartiene allo spazio di Krylov generato da $\mathbf{v}_1$ e C.

Consideriamo la matrice di Krylov:

$$
\begin{aligned}
B &= \begin{pmatrix} \mathbf{v}_1 & C\mathbf{v}_1 & \dots & C^{n-1}\mathbf{v}_1 \end{pmatrix} \\
&= \begin{pmatrix} V\mathbf{e}_1 & CV\mathbf{e}_1 & \dots & C^{n-1}V\mathbf{e}_1 \end{pmatrix} \\
&= V \begin{pmatrix} \mathbf{e}_1 & T^T\mathbf{e}_1 & \dots & (T^T)^{n-1}\mathbf{e}_1 \end{pmatrix} \\
&\equiv VG\,. \qquad (5)
\end{aligned}
$$

B = VG

- Si tratta della fattorizzazione LU della matrice B.
- Il condizionamento della procedura (4) dipenderà dal condizionamento della fattorizzazione:

$$(B + \delta B) = (V + \delta V)(G + \delta G),$$

$$\delta V = \delta B\, G^{-1} - V\, \delta G\, G^{-1},$$

$$\frac{\|\delta V\|}{\|V\|} \leq \kappa(B)\frac{\|\delta B\|}{\|B\|} + \kappa(G)\frac{\|\delta G\|}{\|G\|}.$$

Condizionamento

- Si tratta di minimizzare il numero di condizionamento di B e G.

- Si può dimostrare che:

$$\kappa(B) \leq \kappa(C)^{n-1},$$

$$\kappa(G) \leq \kappa(T)^{n-1}.$$

- Si tratta, pertanto, di minimizzare il numero di condizionamento di C e T, compatibilmente con i vincoli del problema.

Bilanciamento della matrice C

Supponiamo, per semplicità, che il polinomio

$$p(z) = z^n + c_{n-1}z^{n-1} + \cdots + c_1 z + c_0$$

sia monico e a coefficienti tutti non nulli.

Definiamo la matrice diagonale

$$D = \begin{pmatrix} 1 & & & \\ & c_{n-1} & & \\ & & \ddots & \\ & & & c_1 \end{pmatrix}.$$

Bilanciamento della matrice C

Consideriamo

$$\hat{C} \equiv D^{-1}CD = \begin{pmatrix} -b_{n-1} & b_{n-1} & & \\ \vdots & & \ddots & \\ -b_1 & & & b_1 \\ -b_0 & & & \end{pmatrix},$$

$b_i = c_i/c_{i-1}$, al posto di

$$C = \begin{pmatrix} -c_{n-1} & 1 & & \\ \vdots & & \ddots & \\ -c_1 & & & 1 \\ -c_0 & & & \end{pmatrix}.$$

Bilanciamento della matrice C

Utilizzando la norma $\|\cdot\|_\infty$, si ottengono le seguenti stime per i rispettivi numeri di condizione:

$$\kappa(\hat{C}) \;\leq\; 4\,\frac{\max_i |b_i|}{\min_i |b_i|},$$

$$\kappa(C) \;\leq\; 1 + \max_i |c_i| \left(1 + \frac{1 + \max_i |c_i|}{|c_0|}\right).$$

Bilanciamento della matrice C

Esaminiamo alcuni particolari casi significativi:

- Tutte le radici di $p(z)$ sono *clusterizzate* intorno a ξ, con $|\xi| \gg 1$. Si ottiene:

$$\kappa(C) \approx 2|\xi|^n, \qquad \kappa(\hat{C}) \approx 4n^2.$$

- Tutte le radici di $p(z)$ sono *clusterizzate* intorno a ξ, con $|\xi| \ll 1$. Si ottiene

$$\kappa(C) \approx n|\xi|^{1-n}, \qquad \kappa(\hat{C}) \approx 4n^2.$$

Bilanciamento della matrice C

- r radici di $p(z)$ sono *clusterizzate* intorno a ξ_1, con $|\xi_1| \gg 1$, e $n-r$ radici sono prossime a ξ_2, con $|\xi_2| \ll 1$. Si ottiene:

$$\kappa(C) \approx \frac{|\xi_1|^r}{|\xi_2|^{n-r}}, \qquad \kappa(\hat{C}) \approx n^2 \frac{|\xi_1|}{|\xi_2|}.$$

- Esempio: $p(z) = (z+20)^7 + 1.$

  Si ottiene: $\kappa(C) \approx 1.7 \cdot 10^9, \qquad \kappa(\hat{C}) \approx 1.4 \cdot 10^2.$

Bilanciamento della matrice C

Nel caso in cui alcuni dei c_j siano nulli, si procede come segue. Si supponga, per semplicità, che

$$\begin{aligned} 0 &\neq c_{n-1}, \ldots, c_i, \\ 0 &= c_{i-1} = \ldots = c_{i-k+1}, \\ 0 &\neq c_{i-k}, \ldots, c_n. \end{aligned}$$

Allora, si pone $D = \mathrm{diag}(d_j)$, dove:

$$\begin{aligned} d_{n-j+1} &= c_j, \qquad j = 1, \ldots, i,\ i+k, \ldots, n, \\ d_{n-j+1} &= d_{n-j+2}\zeta, \qquad j = i+1, \ldots, i+k-1, \\ \zeta &= (c_{i-k}/c_i)^{1/k}. \end{aligned}$$

Bilanciamento della matrice C

Questa strategia fornisce, infatti, il miglior bilanciamento, nel caso di un unico *cluster* di radici.

Esempio:

$$p(z) = x^{10} - 1024.$$

Si ottiene:

$$C = \begin{pmatrix} 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \\ 1024 & & & \end{pmatrix}, \qquad \hat{C} = 2\begin{pmatrix} 0 & 1 & & \\ \vdots & & \ddots & \\ 0 & & & 1 \\ 1 & & & \end{pmatrix},$$

$$\kappa(C) = 1024, \qquad \kappa(\hat{C}) = 1.$$

Bilanciamento della matrice T

Dopo lo scalamento di C, la fattorizzazione

$$CV = VT^T$$

diviene:

$$(D^{-1}CD)\,D^{-1}V \equiv \hat{C}D^{-1}V = D^{-1}VT^T.$$

Da questa si ottiene, introducendo la matrice diagonale D_1:

$$\hat{C}\hat{V} \equiv \hat{C}(D^{-1}VD_1) = (D^{-1}VD_1)\,(D_1^{-1}T^TD_1) \equiv \hat{V}\hat{T}^T.$$

La matrice D_1 è definita in modo da rendere $|\hat{T}|$ simmetrica.

Codice BTR

Quanto fino ad ora esposto, è stato implementato nel codice Matlab BTR.

Esso è disponibile al sito

http://www.math.unifi.it/~brugnano

Alla voce Ricerca e, quindi, Codici di calcolo.

Una nuova versione del codice Matlab, BTR2, è anche disponibile.

L.Brugnano. Numerical implementation of a new algorithm for polynomials with multiple roots. *J. Difference Eq. Appl.* 1 (1995) 187–207.

$$p(z) = (z - 3.14)^{10}$$

```
roots(p)
ans =
   3.2922e+000 +4.9864e-002i
   3.2922e+000 -4.9864e-002i
   3.2330e+000 +1.2981e-001i
   3.2330e+000 -1.2981e-001i
   3.1386e+000 +1.5889e-001i
   3.1386e+000 -1.5889e-001i
   3.0462e+000 +1.2718e-001i
   3.0462e+000 -1.2718e-001i
   2.9901e+000 +4.8236e-002i
   2.9901e+000 -4.8236e-002i
```

$$p(z) = (z - 3.14)^{10}$$

```
[r,m,b,info] = btr(p)

r =
    3.140000000000001e+000
m =
    10
b =
    []
info =
    0
```

$$p(z) = (z-1)^9(z-2)$$

```
roots(p)
ans =
   2.000000000005336e+000
   1.0325e+000 +1.2229e-002i
   1.0325e+000-1.2229e-002i
   1.0163e+000 +3.0104e-002i
   1.0163e+000 -3.0104e-002i
   9.9315e-001 +3.2798e-002i
   9.9315e-001 -3.2798e-002i
   9.7438e-001 +2.0616e-002i
   9.7438e-001 -2.0616e-002i
   9.6736e-001
```

$$p(z) = (z-1)^9(z-2)$$

```
[r,m,b,info] = btr(p)

r =
    9.999999999999876e-001
    1.999999999999890e+000
m =
    9
    1
b =
    []
info =
    0
```

Sviluppi ulteriori

Consideriamo nuovamente la fattorizzazione di Lanczos non simmetrica:

$$CV = VT^T.$$

Se fattorizziamo

$$V = QR,$$

allora:

$$CQ = Q(RT^T R^{-1}) \equiv QH,$$

con H matrice di Hessemberg superiore.

Si ottiene la fattorizzazione di Arnoldi,

$$CQ = QH, \qquad Q^T Q = I.$$

Metodo di Arnoldi

La fattorizzazione di Arnoldi ha alcuni vantaggi, rispetto a quella di Lanczos non simmetrico:

- non è soggetta a *breakdown* ;
- utilizza una trasformazione ortogonale di C;
- scegliendo

$$\mathbf{q}_1 = \mathbf{v}_1/\|\mathbf{v}_1\|,$$

la procedura si arresta non appena sono determinate le radici distinte.

Tuttavia:

- si perde la struttura tridiagonale della matrice trasformata, passando a quella (più ingombrante) di Hessemberg.

Metodo di Arnoldi

$\mathbf{q}_1 = \mathbf{v}_1/\|\mathbf{v}_1\|$
per i = 1,. . . ,n
    $\mathbf{u} = C\mathbf{q}_i$
    per k = 1,. . . ,i
        $h_{ki} = \mathbf{q}_k^T\mathbf{u}$
        $\mathbf{u} = \mathbf{u} - h_{ki}\mathbf{q}_k$
    fine per
    $\beta = \|\mathbf{u}\|$
    se $i = n$ oppure $\beta \leq tol$
        trovato fattore, esci
    fine se
    $h_{i+1,i} = \beta$
    $\mathbf{q}_{i+1} = \mathbf{u}/\beta$
fine se

Es: $p(z) = z^6 + z^5 + z - 1.$

$$T' = \left(\begin{array}{cc|cc|cc} -\frac{1}{6} & \frac{5}{36} & & & & \\ 1 & -\frac{5}{6} & 0 & -6 & & \\ \hline & & 1 & -\frac{2}{5} & -\frac{29}{25} & \frac{642}{125} \\ & & & 1 & 0 & \\ \hline & & & & 1 & -\frac{19}{15} \quad -\frac{10}{9} \\ & & & & & 1 \quad \frac{5}{3} \end{array}\right),$$

$$H = \begin{pmatrix} 0.0000 & 0.0806 & -0.0298 & -0.4422 & -0.4173 & 1.0050 \\ 1.0000 & -0.6774 & -0.3043 & -0.2750 & 0.5199 & -0.9190 \\ 0 & 0.8720 & -0.1515 & -0.0614 & -0.0827 & -0.1477 \\ 0 & 0 & 0.9499 & -0.0786 & -0.0522 & -0.0572 \\ 0 & 0 & 0 & 0.8721 & -0.2760 & 0.4802 \\ 0 & 0 & 0 & 0 & 1.0197 & 0.1836 \end{pmatrix}.$$

Es: $p(z) = (z^4 - 1)^4$.

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix},$$

```
eig(H)
ans =
    -1.00000000000000
     0.00000000000000 + 1.00000000000000i
     0.00000000000000 - 1.00000000000000i
     1.00000000000000
```

Futuri sviluppi

- Valutare la possibilità di utilizzare la fattorizzazione di Arnoldi per definire una nuova procedura numericamente più stabile, riguardo ai problemi derivanti dall'utilizzo dell'aritmetica finita;
- nuove tecniche di precondizionamento del problema;
- . . . . . .

Un'ultimo esempio

Polinomio derivante da un modello del primo ordine per il cambiamento dei periodi di rotazione e rivoluzione della Terra (Spedicato, 2003):

$$\beta_4 y^4 + \beta_2 y^2 + \beta_1 y + \beta_0 = 0,$$

$$\begin{aligned}
\beta_0 &= -\frac{5}{4} R^2 \eta^{5/3}, \\
\beta_1 &= \frac{5}{2} R^2 \eta^{4/3} + r^2 \eta^{1/3} \omega, \\
\beta_2 &= -\frac{5}{4} R^2 \eta - \frac{1}{2} r^2 \eta - r^2 \omega, \\
\beta_4 &= \frac{1}{2} r^2 \eta^{1/3}.
\end{aligned}$$

Un'ultimo esempio

In unità CGS:

$$\begin{aligned} R &= 1.497766 \cdot 10^{13}, \\ r &= 6.3 \cdot 10^{8}, \\ \omega &= 2\pi/86400, \\ \eta &= \omega/365.24, \end{aligned}$$

In particolare:

$$y^* \equiv \eta^{1/3} \approx 5.839324324411193 \cdot 10^{-3},$$

è sempre soluzione.

Un'ultimo esempio

Utilizzando la function btr2, si ottiene:

```
r = btr2(p)

r =
    -2.19506651700l013e+002
     5.83932281909938e-003
     5.83932281918015e-003
     2.19494973054462e+002
```

(risultati analoghi si ottengono con btr)

Le due radici più piccole sono riconosciute come una doppia.

Prime 6 cifre coincidenti (e corrette nel risultato).

Un'ultimo esempio

È interessante vedere i risultati riguardo y^* quando $r \to 0$:

- $r = 10^2$: $y^* = 5.839324324411193e-003$,
- $r = 10^1$: $y^* = 5.839324324411193e-003$,
- $r = 10^0$: $y^* = 5.839324324411193e-003$,
- $r = 10^{-1}$: $y^* = 5.839324324411192e-003$,
- $r = 10^{-2}$: $y^* = 5.839324324411192e-003$,
- $r = 10^{-3}$: $y^* = 5.839324324411193e-003$,
- $r = 10^{-6}$: $y^* = 5.839324324411193e-003$,
- $r = 0$: $y^* = 5.839324324411193e-003$ (doppia).

Ora, tutte le cifre sono praticamente corrette.