Rendiconti di Matematica, Serie VII
Volume 18, Roma (1998), 725-748

Una strategia di calcolo dello shift nell'algoritmo di Cholesky-Rutishauser per la fattorizzazione SVD

E. ANDRISANI – G. DI LENA

RIASSUNTO: *In questo lavoro si è introdotto un nuovo algoritmo per calcolare i valori singolari di una matrice bidiagonale con massima accuratezza relativa; tale algoritmo, testato su un'ampia classe di matrici, si è dimostrato competitivo rispetto al dqds* [8]. *Inoltre sono state individuate delle nuove tecniche per valutate lo shift, che permettono un ottimo criterio di splitting.*

Si è poi analizzata la stabilità numerica dell'algoritmo e si sono determinate le condizioni a cui devono soddisfare i termini della matrice bidiagonale affinché l'errore relativo non dipenda dall'ordine della matrice.

ABSTRACT: *In this paper we have introduced a new algorithm that computes the singular values of a bidiagonal matrix to maximal relative accuracy; this algorithm, tested on a wide class of matrices, is competitive compared to the dqds* [8]. *We have introduced some new techniques to determine the shift. Moreover they turn out to be an excellent criterion of splitting.*

Then we have analysed the numerical stability of the algorithm and we have determined the conditions that the terms of the bidiagonal matrix satisfy so that the relative error does not depend on the order of the matrix.

– Introduzione

La fattorizzazione SVD di una matrice è utilizzata sia per scopi teo-

KEY WORDS AND PHRASES: *Singular values – SVD – Cholesky LR methods.*
A.M.S. CLASSIFICATION: 65F15
Lavoro svolto nell'ambito del programma nazionale ex40% del M.U.R.S.T.: Analisi Numerica: Metodi e Software matematico.

rici che numerici. Esempi classici di applicazione sono: la determinazione del rango di una matrice, il problema lineare ai minimi quadrati semplice e totale, la teoria dei segnali. Un'ampia classe di metodi numerici per il calcolo dei valori singolari di una matrice trasformano questa, con un numero finito di operazioni, in una matrice bidiagonale avente gli stessi valori singolari. Poi, con procedimenti iterativi si genera una sequenza di matrici bidiagonali convergente ad una diagonale avente i valori singolari cercati. Il passaggio da una matrice all'altra si realizza mediante fattorizzazioni fondamentali, ed il criterio di stop consiste nell'approssimare elementi extradiagonali con lo zero. La routine del pacchetto Linpack che calcola i valori singolari di una matrice bidiagonale è basato sul classico algoritmo di Golub, che fa ricorso alla fattorizzazione QR. Questo algoritmo, a volte, calcola i valori singolari piccoli con accuratezza relativa troppo bassa. Demmel e Kahan hanno implementato una versione di detto algoritmo senza shift, ottenendo tutti i valori singolari in modo altamente accurato. L'algoritmo D.K. è implementato in una routine del Lapack. Successivamente Fernando e Parlett [8], hanno usato la fattorizzazione di Cholesky per generare la sequenza delle matrici bidiagonali dando origine ai ben noti algoritmi dqd senza shift e dqds con shift. In particolare gli autori fanno risalire le formule alla base dei loro algoritmi ai lavori di Rutishauser. Gli algoritmi proposti sono più efficienti dei precedenti, sia per i tempi di esecuzione che per l'accuratezza relativa. Tali risultati sono ben documentati sia a livello teorico che nella verifica sperimentale. L'algoritmo dqds è ritenuto il più efficiente nella classe dei metodi menzionati, e per tale motivo Parlett ha suggerito di aggiornare la routine di Demmel e Kahan del Lapack.

Questo lavoro, prende spunto da alcune maggiorazioni della variazione relativa dei valori singolari di due matrici, non considerate dai precedenti autori. In particolare per le matrici bidiagonali si riesce a calcolare l'errore relativo quando un elemento extradiagonale è approssimato con lo zero, ottenendo un criterio di splitting rigoroso ed efficiente. Si riesce, inoltre, utilizzando le maggiorazioni di cui sopra a minorare il più piccolo valore singolare della matrice, ad ottenere un semplice algoritmo per il calcolo dello shift. I calcoli per lo splitting e quelli per lo shift hanno una parte in comune utilizzabile per realizzare la fattorizzazione di Cholesky.

In tal modo si ottengono delle varianti degli algoritmi dqd e dqds, che a livello sperimentale risultano più efficienti. Alla formulazione del-

l'algoritmo proposto nel paragrafo 5 abbiamo dato il nome di algoritmo di Cholesky-Rutishauser con shift. Un'analisi della stabilità completa il lavoro.

La teoria della perturbazione è uno strumento essenziale per l'analisi degli algoritmi numerici. Infatti, permette di stabilire quando un problema è "ben condizionato", quando un algoritmo è "stabile" ed in ultima analisi dà una indicazione di attendibilità dei risultati numerici. Un modo classico di analizzare un problema o un algoritmo consiste nel trovare una relazione tra una variazione assoluta dei dati e una variazione assoluta dei risultati.

Per i valori singolari è noto [6] che comunque si considerino due matrici $A \in \Re^{n\times n}$ e $B \in \Re^{n\times n}$ risulta:

$$| \sigma_i(A+B) - \sigma_i(A) | \leq \sigma_1(B) \text{ per ogni } i = 1, \dots, n\,. \tag{1.1}$$

In altri termini, indicata con δA una perturbazione sulla matrice A, con $\sigma_1 \geq \dots \geq \sigma_n$ i suoi valori singolari e con $\sigma'_1 \geq \dots \geq \sigma'_n$ quelli di $A + \delta A$ si ha:

$$| \sigma'_i - \sigma_i | \leq \|\delta A\|_2 \text{ per ogni } i = 1, \dots, n \tag{1.2}$$

si può allora affermare che la perturbazione generata sui valori singolari da una perturbazione δA è limitata superiormente da $\|\delta A\|_2$.

Indicato con

$$\eta = \frac{\|\delta A\|_2}{\|A\|_2}$$

per $\sigma_i \approx \sigma_1 = \|A\|_2$ si ha:

$$\Big|\frac{\sigma_i - \sigma'_i}{\sigma_i}\Big| \leq \frac{\|\delta A\|_2}{|\,\sigma_i\,|} \approx \eta\,,$$

invece per $\sigma_i \approx \sigma_n = 1/\|A^{-1}\|_2$ si ha:

$$\Big|\frac{\sigma_i - \sigma'_i}{\sigma_i}\Big| \leq \frac{\eta\|A\|_2}{|\,\sigma_i\,|} \approx \eta\mu_2(A)$$

dove $\mu_2(A)$ indica il numero di condizione della matrice. Pertanto dalla (1.1) segue che i valori singolari non molto distanti da $\|A\|_2$ sono calcolati

con grande accuratezza relativa, mentre ciò non accade per i valori singolari piccoli. La possibilità di costruire algoritmi numerici con risultati "altamente accurati" per tutti i valori singolari, impone di sviluppare una teoria della perturbazione che ponga in relazione la variazione relativa dei risultati con la variazione relativa o la variazione assoluta dei dati.

Per questo scopo è utile considerare la fattorizzazione SVD della matrice

$$A = U\Sigma V^T$$

con U e V matrici ortogonali, e Σ la matrice diagonale dei valori singolari; e la rappresentazione della pseudoinversa

$$A^+ = V\Sigma^+ U^T\,;$$

risulta [4]:

TEOREMA 1. *Se $\sigma_k(AA^+B) = \sigma_k(B)$ per ogni k, allora:*

$$\sigma_k(A) = 0 \Rightarrow \sigma_k(B) = 0$$
$$\sigma_k(A) \neq 0 \Rightarrow \Big|\frac{\sigma_k(A) - \sigma_k(B)}{\sigma_k(A)}\Big| \leq \sigma_1(A^+(A-B))\,. \qquad \square$$

Ed in particolare denotando con $\text{range}(A) = \{y \in \Re^n | y = Ax, x \in \Re^n\}$ segue:

COROLLARIO 1. *I valori singolari di due matrici A e B tali che il* range(B) *è contenuto nel* range(A)*, verificano le seguenti relazioni:*

$$\sigma_k(A) = 0 \Rightarrow \sigma_k(B) = 0$$
$$\sigma_k(A) \neq 0 \Rightarrow \Big|\frac{\sigma_k(A) - \sigma_k(B)}{\sigma_k(A)}\Big| \leq \sigma_1(A^+(A-B))\,. \qquad \square$$

Analoghi risultati si ottengono sostituendo $\sigma_1(A^+(A-B))$ con $\sigma_1((A-B)A^+)$; per matrici invertibili vale l'unica:

$$\Big|\frac{\sigma_k(A) - \sigma_k(B)}{\sigma_k(A)}\Big| \leq \min\{\sigma_1(A^{-1}(A-B)),\ \sigma_1((A-B)A^{-1})\}\,.$$

1 – Un criterio di splitting

In generale gli algoritmi per il calcolo dei valori singolari di matrici bidiagonali hanno un criterio per considerare trascurabile un elemento extradiagonale sostituendolo con lo zero, senza che ciò comporti una rilevante variazione relativa sui valori singolari. In questo paragrafo si determina un algoritmo per maggiorare l'errore relativo quando un elemento extradiagonale di un'assegnata matrice bidiagonale è approssimato con lo zero.

Sia A la matrice bidiagonale così definita

$$\begin{pmatrix} a_1 & b_1 & & & & & \\ & \cdot & \cdot & & & O & \\ & & a_k & \cdot & b_k & & \\ & & & & \cdot & \cdot & \\ & O & & & & a_{n-1} & b_{n-1} \\ & & & & & & a_n \end{pmatrix}$$

i cui elementi diagonali si supporranno non nulli senza perdere in generalità, $E_{k,k+1}$ la matrice avente il solo elemento non nullo uguale a uno nel posto $(k, k+1)$ e B la matrice definita da:

$$B = A - b_k E_{k,k+1}\,.$$

La maggiorazione dell'errore relativo nel corollario 1 riferita al caso in esame vale:

$$\max_i \Big|\frac{\sigma_i(A) - \sigma_i(B)}{\sigma_i(A)}\Big| \le \|A^{-1} b_k E_{k,k+1}\|\,.$$

La matrice $A^{-1}E_{k,k+1}$ ha una sola colonna non nulla pari a $A^{-1}\mathbf{e}_k$ ove $\mathbf{e}_k = [0, \dots, 0, \underset{k}{1}, 0, \dots, 0]^T$, pertanto vale l'uguaglianza:

$$\|A^{-1}E_{k,k+1}\| = \|A^{-1}\mathbf{e}_k\|\,,$$

da cui si ha:

$$\Big|\frac{\sigma_i(A) - \sigma_i(B)}{\sigma_i(A)}\Big| \le \|b_k A^{-1}\mathbf{e}_k\|\,,\ \text{per ogni } i = 1, \dots, n\,. \tag{1.3}$$

Posto $r_k = \|b_k A^{-1}\mathbf{e}_k\|^2$ e $\mathbf{x} = b_k A^{-1}\mathbf{e}_k$, si ottiene il sistema:

$$\begin{cases} a_1 x_1 + b_1 x_2 & = 0 \\ \quad a_2 x_2 + b_2 x_3 & = 0 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \qquad a_k x_k + b_k x_{k+1} & = b_k \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \qquad\qquad\qquad\qquad a_n x_n = 0\,. \end{cases}$$

La soluzione del sistema risulta:

$$x_i = 0 \text{ per } i \geq k+1, \quad x_k = b_k/a_k, \quad x_{i-1} = -x_i b_{i-1}/a_{i-1} \text{ per } 1 < i \leq k\,.$$

I vettori $b_{k-1}A^{-1}\mathbf{e}_{k-1}$, $b_k A^{-1}\mathbf{e}_k$ hanno le prime $k-1$ componenti che differiscono per un fattore $\pm b_k/a_k$, il vettore $b_k A^{-1}\mathbf{e}_k$ ha una componente in più non nulla pari a b_k/a_k.

Pertanto si ha la seguente relazione:

$$r_k = \Big(\frac{b_k}{a_k}\Big)^2 (1 + r_{k-1}), \text{con } r_1 = \Big(\frac{b_1}{a_1}\Big)^2\,. \tag{1.4}$$

Un'altra maggiorazione relativa al corollario 1 si ottiene considerando la norma della matrice $b_k E_{k,k+1} A^{-1}$:

$$\Big|\frac{\sigma_i(A) - \sigma_i(B)}{\sigma_i(A)}\Big| \leq \|b_k E_{k,k+1} A^{-1}\| = \|b_k \mathbf{e}_{k+1}^T A^{-1}\| = \|b_k A^{-T}\mathbf{e}_{k+1}\|\,.$$

Analogamente al caso precedente ponendo $s_k = \|b_k A^{-T}\mathbf{e}_{k+1}\|^2$ si ha:

$$s_k = \Big(\frac{b_k}{a_{k+1}}\Big)^2 (1 + s_{k+1}), \text{ con } s_{n-1} = \Big(\frac{b_{n-1}}{a_n}\Big)^2\,.$$

Anche in questo caso $\sqrt{s_k}$ rappresenta l'errore relativo che si commette sostituendo b_k con lo zero. Una maggiorazione più economica dell'errore è data dal prossimo teorema:

TEOREMA 2. *Se* $|b_k/a_k| \leq r < 1$ *per* $k = 1 \ldots n-1$, *allora*

$$\|b_k A^{-1}\mathbf{e}_k\| \leq \Big|\frac{b_k}{a_k}\Big| \frac{1}{\sqrt{1-r^2}}\,.$$

DIM. Dalle relazioni (1.4) si ha:

$$r_k \leq r^2(1 + r_{k-1}) \text{ e } r_1 \leq r^2$$

da cui

$$r_k \leq \frac{r^2 - r^2 r^{2k}}{1 - r^2} \leq \frac{r^2}{1 - r^2}, \text{ per ogni } k\,.$$

Pertanto

$$\Big|\frac{\sigma_i(A) - \sigma_i(B)}{\sigma_i(A)}\Big| \leq \|b_k A^{-1}\mathbf{e}_k\| = \sqrt{r_k} = \Big|\frac{b_k}{a_k}\Big|\sqrt{1 + r_{k-1}} \leq \Big|\frac{b_k}{a_k}\Big|\Big(1 + \frac{r^2}{1 - r^2}\Big)^{1/2}$$

da cui la tesi. □

In particolare se $\forall\, k\ |b_k/a_k| \leq \sqrt{3}/2$ si ha: $\|b_k A^{-1}\mathbf{e}_k\| \leq 2|b_k/a_k|$.

NOTA. Assegnata una tolleranza *toll*, considerare trascurabile il termine b_k quando $|b_k/a_k| \leq$ toll è un criterio di splitting non sempre corretto.

DEMMEL e KAHAN in [5] hanno proposto un criterio corretto per effettuare lo splitting di una matrice bidiagonale. Con una leggera modifica algebrica tale criterio consiste nel considerare:

$$\begin{cases} \lambda_n = b_{n-1}/a_n \\ \lambda_{i-1} = \dfrac{a_i}{b_{i-1}}(1 + \lambda_i) \end{cases} \qquad \begin{cases} \mu_1 = b_1/a_1 \\ \mu_{i+1} = \dfrac{b_{i+1}}{a_{i+1}}(1 + \mu_i) \end{cases}$$

e provano che porre $b_k = 0$ equivale a commettere un errore relativo sui valori singolari non maggiore di $\min\{\lambda_k, \mu_k\}$.

Il criterio proposto afferma, invece, che l'errore relativo è non maggiore del $\min\{\sqrt{r_k}, \sqrt{s_k}\}$. Per un confronto tra i due criteri poniamo $\mu'_k = \sqrt{r_k}$; e si ha:

$$\mu'_1 = \mu_1 = b_1/a_1$$
$$\mu'_{i+1} = \frac{b_{i+1}}{a_{i+1}}(1 + (\mu'_i)^2)^{1/2}\,.$$

Poiché $(1 + (\mu'_i)^2)^{1/2} < 1 + \mu'_i$ segue $\mu'_i < \mu_i$.

Pertanto il criterio che utilizza le quantità (1.4) è più efficiente di quello suggerito da DEMMEL-KAHAN in [5].

Le quantità (1.4) si possono utilizzare per ottenere una maggiorazione migliore della (1.3); infatti considerate A e B matrici quadrate non singolari di ordine n, essendo $B = AA^{-1}B$ risulta [7]:

$$\sigma_k(A)\sigma_n(A^{-1}B) \leq \sigma_k(B) \leq \sigma_k(A)\sigma_1(A^{-1}B)$$

pertanto

$$\sigma_n(A^{-1}B) - 1 \leq \frac{\sigma_k(B) - \sigma_k(A)}{\sigma_k(A)} \leq \sigma_1(A^{-1}B) - 1\,.$$

Inoltre essendo

$$\sigma_n(A^{-1}B) = \sigma_n(I - A^{-1}(A - B)) \geq 1 - \sigma_1(A^{-1}(A - B))$$
$$\sigma_1(A^{-1}B) - 1 \leq \sigma_1(A^{-1}B - I) = \sigma_1(I - A^{-1}B) = \sigma_1(A^{-1}(A - B))\,,$$

si può concludere che

$$-\sigma_1(A^{-1}(A - B)) \leq \sigma_n(A^{-1}B) - 1 \leq \frac{\sigma_k(B) - \sigma_k(A)}{\sigma_k(A)} \leq$$
$$\leq \sigma_1(A^{-1}B) - 1 \leq \sigma_1(A^{-1}(A - B))$$

quindi

$$\Big|\frac{\sigma_k(A) - \sigma_k(B)}{\sigma_k(A)}\Big| \leq \max\{|\ \sigma_1(A^{-1}B) - 1\ |, |\ \sigma_n(A^{-1}B) - 1\ |\} \leq$$
$$\leq \sigma_1(A^{-1}(A - B))$$

e più in generale

$$\Big|\frac{\sigma_k(A) - \sigma_k(B)}{\sigma_k(A)}\Big| \leq \max_{i=1\ldots n}\{|\ \sigma_i(A^{-1}B) - 1\ |\} \leq \sigma_1(A^{-1}(A - B))\,. \tag{1.5}$$

È importante osservare come la nuova stima dell'errore relativo è ottimale quando i vettori singolari sinistri delle matrici A e B coincidono, mentre per la stima vista in precedenza ciò accade solo quando A e B hanno anche gli stessi vettori singolari destri. Infatti posto:

$$A = U_A\Sigma_A V_A^T \text{ e } B = U_B\Sigma_B V_B^T\,,$$

le fattorizzazioni SVD delle due matrici, risulta:

$$A^{-1}B = V_A\Sigma_A^{-1}U_A^T U_B\Sigma_B V_B^T \Rightarrow \sigma_k(A^{-1}B) = \sigma_k(\Sigma_A^{-1}U_A^T U_B\Sigma_B)$$

e se A e B hanno gli stessi vettori singolari sinistri, cioè $U_A = U_B$, si ha:

$$\sigma_k(A^{-1}B) = \sigma_k(\Sigma_A^{-1}\Sigma_B) = \frac{\sigma_k(B)}{\sigma_k(A)}\,.$$

Mentre:

$$\begin{aligned} A^{-1}(A-B) = & I - V_A\Sigma_A^{-1}U_A^T U_B\Sigma_B V_B^T = V_A(V_A^T V_B - \Sigma_A^{-1}U_A^T U_B\Sigma_B)V_B^T \Rightarrow \\ & \Rightarrow \sigma_k(A^{-1}(A-B)) = \sigma_k(V_A^T V_B - \Sigma_A^{-1}U_A^T U_B\Sigma_B) \end{aligned}$$

e dunque solo quando $U_A = U_B$ e $V_A = V_B$ risulterà

$$\sigma_k(A^{-1}(A-B)) = \sigma_k(I - \Sigma_A^{-1}\Sigma_B) = 1 - \frac{\sigma_k(B)}{\sigma_k(A)}\,.$$

Per A bidiagonale e $B = A - b_k E_{k,k+1}$, il calcolo di $\sigma_{1,n}(A^{-1}B)$ si realizza nel modo seguente:
posto $\mathbf{x} = b_k A^{-1}\mathbf{e}_k$ con $\|\mathbf{x}\|^2 = r_k$ si ha:

$$I - b_k A^{-1}E_{k,k+1} = I - \mathbf{x}\mathbf{e}_{k+1}^T$$

con $\mathbf{x}^T\mathbf{e}_{k+1} = 0$, inoltre per calcolare i quadrati dei valori singolari di $A^{-1}B$ si considera $(I - \mathbf{x}\mathbf{e}_{k+1}^T)^T(I - \mathbf{x}\mathbf{e}_{k+1}^T) = I - \mathbf{e}_{k+1}\mathbf{x}^T - \mathbf{x}\mathbf{e}_{k+1}^T + \|\mathbf{x}\|^2 E_{k+1,k+1}$, i cui autovalori sono $n-1$ pari ad uno ed i rimanenti ad $\sigma_{1,n}(A^{-1}B)$. Posto $\lambda \neq 1$, $\mathbf{z} \neq 0$, l'equazione agli autovalori risulta:

$$\mathbf{z} - \mathbf{x}^T\mathbf{z}\mathbf{e}_{k+1} - z_{k+1}\mathbf{x} + \|\mathbf{x}\|^2 z_{k+1}\mathbf{e}_{k+1} = \lambda\mathbf{z}$$

da cui

$$\begin{aligned} & z_i - z_{k+1}x_i = \lambda z_i && \text{per ogni } i = 1, \ldots, k \\ & z_{k+1} - \sum_{i=1}^{k} x_i z_i + \|\mathbf{x}\|^2 z_{k+1} = \lambda z_{k+1} && \text{per } i = k+1 \end{aligned}$$

ossia

$$\begin{aligned} & z_i = \frac{z_{k+1}}{1-\lambda}x_i && \text{per ogni } i = 1, \ldots, k \\ & \sum_{i=1}^{k} x_i z_i = (1-\lambda)z_{k+1} + \|\mathbf{x}\|^2 z_{k+1} && \text{per } i = k+1\,. \end{aligned}$$

Ora, sostituendo la prima equazione nella seconda, si ottiene la seguente equazione di secondo grado in λ:

$$\lambda^2 - (2 + \|\mathbf{x}\|^2)\lambda + 1 = 0\,.$$

In definitiva la formula per il calcolo di $\sigma_{1,n}(A^{-1}B)$ risulta:

$$\sigma_{1,n}(A^{-1}B) = \sqrt{\frac{(2 + \|\mathbf{x}\|^2) \pm \sqrt{(2 + \|\mathbf{x}\|^2)^2 - 4}}{2}} = \\ = \sqrt{\frac{(2 + r_k) \pm \sqrt{(2 + r_k)^2 - 4}}{2}}\,.$$

2 – Algoritmo di Cholesky-Rutishauser (shift nullo)

Il metodo basato sulla fattorizzazione di Cholesky per il calcolo dei valori singolari di una matrice bidiagonale $B_0 = B$, genera una sequenza di matrici secondo la relazione:

$$B_{i+1}^T B_{i+1} = B_i B_i^T\,. \tag{2.1}$$

In particolare, esplicitando la relazione (2.1) ed indicando con q_k e con e_k i quadrati degli elementi principali ed extradiagonali di B_i e con $\hat{q}_k$ ed $\hat{e}_k$ i quadrati degli elementi principali ed extradiagonali di B_{i+1}, si ottiene l'algoritmo dqd di Parlett [8]:

Algoritmo dqd:

$d_1 = q_1$
for $k = 1 : n - 1$
$\hat{q}_k = d_k + e_k$
$\hat{e}_k = e_k q_{k+1}/\hat{q}_k$
$d_{k+1} = d_k q_{k+1}/\hat{q}_k$
end
$\hat{q}_n = d_n\,.$

In [8] si è determinato un limite inferiore ed un limite superiore del più piccolo valore singolare della matrice bidiagonale. Tali limitazioni hanno

un ruolo fondamentale per poter stabilire quando eseguire lo splitting della matrice. Infatti si prova che:

$$\Big(\sum_{k=1}^{n}\frac{1}{d_k}\Big)^{-1/2} \le \sigma_n \le \min_k\{d_k^{1/2}\} \tag{2.2}$$

ove tali limiti vengono chiamati inf e sup.

Ora indicata con B' la matrice bidiagonale ottenuta dalla matrice B ponendo l'elemento extradiagonale $b_k = 0$ ed essendo per la (1.2) $|\sigma_i(B) - \sigma_i(B')| \le \|B - B'\|$, si ha per la (2.2):

$$\frac{|\,\sigma_i(B) - \sigma_i(B')\,|}{\sigma_i(B)} \le \frac{|\,b_k\,|}{\sigma_i(B)} \le \frac{|\,b_k\,|}{\sigma_n(B)} \le \frac{|\,b_k\,|}{\Big(\sum\limits_{K=1}^{n}\frac{1}{d_k}\Big)^{-1/2}} \le \text{eps}$$

e quindi in definitiva si considera come criterio di splitting la seguente relazione:

$$e_k \le \frac{\text{eps}^2}{\Big(\sum\limits_{K=1}^{n}\frac{1}{d_k}\Big)}\,.$$

Il nostro algoritmo, come il dqd, realizza la (2.1), ma a differenza di quest'ultimo è basato sulla formula ricorsiva (1.4) in modo da conglobare i calcoli per eseguire il criterio di splitting. Il nuovo algoritmo si può derivare più semplicemente modificando il dqd. In particolare:
per $k = 1$

$$d_1 = q_1 = e_1\frac{q_1}{e_1} = \frac{e_1}{r_1}$$
$$\hat{q}_1 = d_1 + e_1 = \frac{e_1}{r_1} + e_1 = \frac{e_1}{r_1}(1 + r_1)$$

per $k = 2$

$$d_2 = d_1\frac{q_2}{\hat{q}_1} = \frac{q_2}{(1 + r_1)} = e_2\frac{1}{\frac{e_2}{q_2}(1 + r_1)} = \frac{e_2}{r_2}$$

$$\hat{q}_2 = d_2 + e_2 = \frac{e_2}{r_2} + e_2 = \frac{e_2}{r_2}(1 + r_2)$$

al passo $k+1$-esimo

$$d_{k+1} = d_k \frac{q_{k+1}}{\hat{q}_k} = \frac{q_{k+1}}{(1+r_k)} = e_{k+1} \frac{1}{\frac{e_{k+1}}{q_{k+1}}(1+r_k)} = \frac{e_{k+1}}{r_{k+1}}$$

$$\hat{q}_{k+1} = d_{k+1} + e_{k+1} = \frac{e_{k+1}}{r_{k+1}} + e_{k+1} = \frac{e_{k+1}}{r_{k+1}}(1+r_{k+1})\,;$$

da cui:

$$\hat{q}_k = \frac{e_k}{r_k}(1+r_k) \qquad \hat{e}_k = e_k \frac{q_{k+1}}{\hat{q}_k}\,.$$

Queste ultime insieme alle (1.4) costituiscono il nuovo algoritmo che abbiamo denominato *algoritmo di Cholesky-Rutishauser*:

ALGORITMO CR:

$t = 1$

for $k = 1 : n-1$

$r_k = (e_k/q_k)t; \quad t = 1 + r_k;$

$\hat{q}_k = (e_k/r_k)t; \quad \hat{e}_k = e_k q_{k+1}/\hat{q}_k;$

end

$$\hat{q}_n = \frac{e_{n-1}}{r_{n-1}} \frac{q_n}{\hat{q}_{n-1}}\,;$$

3 – Stabilità dell'algoritmo di Cholesky-Rutishauser (shift nullo)

In aritmetica esatta la formula ricorsiva dell'algoritmo CR ha la seguente espressione:

$$r_1 = \frac{e_1}{q_1}; \qquad r_{k+1} = \frac{e_{k+1}}{q_{k+1}}(1+r_k)\,. \tag{3.1}$$

Il calcolo dei termini r_k comporta ad ogni passo tre operazioni: una moltiplicazione, una somma ed una divisione. Pertanto indicati con ν, μ e η gli errori relativi sulle tre operazioni e con ε la precisione della macchina, la formula ricorsiva (3.1) in aritmetica finita assume questa nuova espressione:

$$\tilde{r}_{k+1} = \frac{e_{k+1}}{q_{k+1}}(1+\tilde{r}_k)(1+\nu_k)(1+\mu_k)(1+\eta_k) \text{ con } \max\{|\,\nu_k\,|, |\,\mu_k\,|, |\,\eta_k\,|\} < \varepsilon\,.$$

Posto $(1+\nu_k)(1+\mu_k)(1+\eta_k) = (1+\delta_k)$, si ha:

$$\begin{aligned} \tilde{r}_1 &= \frac{e_1}{q_1}(1+\delta_1) \qquad\qquad \text{con } \mid \delta_1 \mid \leq \varepsilon \\ \tilde{r}_{k+1} &= \frac{e_{k+1}}{q_{k+1}}(1+\tilde{r}_k)(1+\delta_{k+1}) \quad \text{con } \mid \delta_{k+1} \mid \leq 3\varepsilon\,. \end{aligned} \tag{3.2}$$

LEMMA 1. *Se $e_k/q_k \leq \frac{1}{2}1/1+3\varepsilon$ per $k = 1 \ldots n-1$, allora*

$$r_k \leq \frac{1}{1+6\varepsilon} \quad e \quad \frac{r_k}{1+r_k}(1+3\varepsilon) \leq \frac{1}{2}\,.$$

DIM. Posto $r = 1/2(1+3\varepsilon)$ risulta per ogni k: $e_k/q_k \leq r < 1$ cioè $(b_k/a_k)^2 \leq r < 1$ il che implica $|b_k/a_k| \leq \sqrt{r} < 1$ e ciò comporta per il teorema 2 che $\sqrt{r_k} \leq |\frac{b_k}{a_k}|\frac{1}{\sqrt{1-r}}$. Passando ai quadrati si ottiene

$$r_k \leq \frac{e_k}{q_k}\frac{1}{1-r} < \frac{r}{1-r} = \frac{1}{1+6\varepsilon}$$

e quindi

$$\frac{r_k}{1+r_k}(1+3\varepsilon) \leq \frac{\frac{r}{1-r}}{1+\frac{r}{1-r}}(1+3\varepsilon) = \frac{1}{2}\,. \qquad \square$$

TEOREMA 3. *In assenza di underflow ed overflow, se $3n\varepsilon < 1$, si ha:*

$$\left|\frac{r_k - \tilde{r}_k}{r_k}\right| \leq \frac{3n\varepsilon}{1-3n\varepsilon}\,.$$

DIM. Posto:

$$p_k = \frac{r_k - \tilde{r}_k}{r_k} \tag{3.3}$$

si ottiene al primo passo $p_1 = -\delta_1$, essendo $\tilde{r}_1 = \frac{e_1}{q_1}(1+\delta_1)$, mentre al passo $k+1$-esimo risulta:

$$p_{k+1} = \frac{r_{k+1} - \tilde{r}_{k+1}}{r_{k+1}} = \frac{\frac{e_{k+1}}{q_{k+1}}(1+r_k) - \frac{e_{k+1}}{q_{k+1}}(1+\tilde{r}_k)(1+\delta_{k+1})}{\frac{e_{k+1}}{q_{k+1}}(1+r_k)} =$$

$$= 1 - \frac{(1+\tilde{r}_k)(1+\delta_{k+1})}{(1+r_k)}$$

dalla (3.3), essendo $\tilde{r}_k = r_k(1-p_k)$, si ha:

$$p_{k+1} = 1 - \frac{1+r_k(1-p_k)}{(1+r_k)}(1+\delta_{k+1}) = 1 - \frac{1+r_k-p_kr_k}{(1+r_k)} + \\ - \delta_{k+1}\frac{1+r_k-p_kr_k}{(1+r_k)} = p_k\Big(\frac{r_k}{1+r_k}\Big) - \delta_{k+1}\Big(1-p_k\Big(\frac{r_k}{1+r_k}\Big)\Big) = \\ = p_k\Big(\frac{r_k}{1+r_k}\Big)(1+\delta_{k+1}) - \delta_{k+1}\,.$$

Indicato con $\rho_{k+1} = |p_{k+1}|$, risulta:

$$\rho_{k+1} = \mid p_{k+1} \mid = \Big|p_k\Big(\frac{r_k}{1+r_k}\Big)(1+\delta_{k+1}) - \delta_{k+1}\Big| \le \Big|p_k\Big(\frac{r_k}{1+r_k}\Big)(1+\delta_{k+1})\Big| + \\ + \mid \delta_{k+1} \mid \le \rho_k\Big(\frac{r_k}{1+r_k}\Big)(1+3\varepsilon) + 3\varepsilon\,;$$

da cui la relazione di ricorrenza:

$$\rho_{k+1} \le \Big(\frac{r_k}{1+r_k}\Big)(1+3\varepsilon)\rho_k + 3\varepsilon\,. \tag{3.4}$$

Semplificando ulteriormente si ha:

$$\rho_1 \le \varepsilon; \qquad \rho_{k+1} \le (1+3\varepsilon)\rho_k + 3\varepsilon\,;$$

mediante semplice verifica per induzione si prova:

$$\rho_k \le (1+3\varepsilon)^k - 1 \quad \text{per } k = 1, \dots, n-1\,. \tag{3.5}$$

In definitiva per la (3.5) e per l'ipotesi fatta segue la tesi:

$$\rho_k \le (1+3\varepsilon)^k - 1 \le \frac{3n\varepsilon}{1-3n\varepsilon}\,. \qquad \square$$

COROLLARIO 2. *Se $e_k/q_k \le r = 1/2(1+3\varepsilon)$ per $k = 1 \dots n-1$, si ha:*

$$\Big|\frac{r_k - \tilde{r}_k}{r_k}\Big| \le 6\varepsilon\,.$$

DIM. Per l'ipotesi fatta e per il lemma 1, la (3.4) si semplifica:

$$\rho_1 \le \varepsilon; \qquad \rho_{k+1} \le \frac{1}{2}\rho_k + 3\varepsilon\,;$$

mediante semplice verifica per induzione si prova:

$$\rho_k \le 6\varepsilon\,. \qquad \square$$

NOTA. In generale l'errore di arrotondamento dipende dalla dimensione della matrice bidiagonale; mentre una condizione del tipo

$$\frac{e_k}{q_k} \leq \frac{1}{2} \text{ per } k = 1, \dots, n-1$$

verificata asintoticamente in virtù della convergenza del metodo, rende l'errore indipendente dalla dimensione della matrice. Per avere un'idea di quando tale condizione si conserva da una iterata all'altra si può tenere conto che da

$$\frac{q_{k+1}}{q_k} \leq 1, \qquad \frac{e_k}{q_k} \leq \frac{1}{2}, \quad \text{per } k = 1, \dots, n-1$$

segue

$$\frac{\hat{e}_k}{\hat{q}_k} \leq \frac{1}{2}, \quad \text{per } k = 1, \dots, n-1\,.$$

Infatti si ha:

$$\frac{\hat{e}_k}{\hat{q}} = \frac{q_{k+1}}{e_k}\Big(\frac{r_k}{1+r_k}\Big)^2 = \frac{q_{k+1}}{e_k}\left(\frac{\dfrac{e_k}{q_k}(1+r_{k-1})}{1+\dfrac{e_k}{q_k}(1+r_{k-1})}\right)^2 =$$

$$= \frac{q_{k+1}}{q_k}\frac{e_k}{q_k}\frac{1}{\Big(\dfrac{e_k}{q_k}+\dfrac{1}{1+r_{k-1}}\Big)^2}$$

ed essendo $e_k/q_k \leq 1/(1+r_{k-1})$, segue l'asserto. □

4 – Una strategia di calcolo dello shift

L'introduzione dello shift nell'algoritmo CR mira a migliorare la convergenza, in particolare data la matrice bidiagonale B e lo shift

$$\tau \leq \sigma_n(B) \tag{4.1}$$

la nuova matrice $\widehat{B}$ si ottiene ponendo

$$\widehat{B}^T\widehat{B} = BB^T - \tau^2 I\,.$$

Una implementazione ottimale è data dall'algoritmo dqds di Parlett:

ALGORITMO dqds:

$$d_1 = q_1 - \tau^2$$

for $k = 1 : n-1$

$$\hat{q}_k = d_k + e_k$$

$$\hat{e}_k = e_k q_{k+1}/\hat{q}_k$$

$$d_{k+1} = d_k(q_{k+1}/\hat{q}_k) - \tau^2$$

end

$$\hat{q}_n = d_n\,.$$

In [8] FERNANDO e PARLETT suggeriscono di calcolare lo shift, utilizzando le quantità d_k ottenute al passo precedente, secondo la formula:

$$\tau^2 = \frac{1}{\dfrac{1}{d_{k^*-1}} + \dfrac{1}{d_{k^*}} + \dfrac{1}{d_{k^*+1}}} - \bar{\tau}^2 \tag{4.2}$$

ove k^* è l'indice tale che $d_{k^*} = \min_k\{d_k\}$; le quantità d_k, $\bar{\tau}^2$ provengono dal passo precedente.

Poiché τ è una sottostima di $\sigma_n(B)$, la (4.1) non è garantita a-priori. La validità di τ come shift è assicurata solo a-posteriori, dopo aver eseguito il passo dqds e verificato che:

$$d_k \geq 0 \quad \text{per ogni } k\,,$$

qualora tale condizione non è soddisfatta si rigetta lo shift sostituendolo con un valore più piccolo aggiungendo uno o più termini nella somma che è presente al denominatore nell'espressione (4.2).

Il criterio di splitting, nell'implementazione di Parlett, è basato sulla verifica:

$$e_k \leq \mathrm{eps}^2 \cdot s^2$$

ove s^2 è la somma di tutti gli shift utilizzati dall'algoritmo.

La nostra stima dello shift fa ricorso alle quantità r_k ottenute mediante la (1.4), utilizzate sia per un passo dell'algoritmo CR che per stabilire eventuali splitting. Nell'algoritmo CR gli elementi diagonali tendono ai valori singolari, mentre quelli extradiagonali tendono a zero: in particolare il $\min_k\{a_k\}$ tende al più piccolo valore singolare. Poiché

$$\min_k\{a_k\} \geq \sigma_n(B)$$

tale minimo non può essere utilizzato come shift.

L'idea alla base della nostra tecnica di shift consiste nell'utilizzare una matrice che ha il $\min_k\{a_k\}$ pari al minimo valore singolare. In particolare se il minimo si ha in corrispondenza di $k = 1$ si considera la matrice di partenza B a cui si sostituisce b_1 con lo zero; per $1 < k < n$ si pone zero b_{k-1} e b_k; per $k = n$ si pone zero solo b_{n-1}. Alla matrice bidiagonale B, associamo la matrice B' ottenuta da B sostituendo b_{n-1} con lo zero, facendo ricorso alla (1.5) si ha:

$$\sigma_n(B) \geq \frac{\sigma_n(B')}{\sigma_1(B^{-1}B')}$$

e l'analoga

$$\sigma_n(B) \geq \frac{\sigma_n(B')}{1 + \sigma_1(B^{-1}(B - B'))}\,.$$

Se

$$\sigma_n(B') \geq a_n \tag{4.3}$$

ipotesi verificata asintoticamente nell'algoritmo CR, e calcolando $\sigma_1(B^{-1}B')$ e $\sigma_1(B^{-1}(B - B'))$ in termini delle quantità (1.4), si hanno semplici formule per stimare lo shift:

$$\tau^2 = \frac{q_n}{\sqrt{\dfrac{(2 + r_{n-1}) + \sqrt{(2 + r_{n-1})^2 - 4}}{2}}} \quad \text{e} \quad \tau^2 = \frac{q_n}{1 + \sqrt{r_{n-1}}}\,.$$

Poiché la (4.3) non può essere verificata a-priori, si seleziona l'indice k^*:

$q_{k^*} = \min_k\{q_k\}$ e si determina lo shift nel seguente modo:

$$
\begin{aligned}
\tau^2 &= \frac{q_1}{\Big(\frac{(2+r_1)+\sqrt{(2+r_1)^2-4}}{2}\Big)} && \text{se} \quad k^* = 1 \\
\tau^2 &= \frac{q_{k^*}}{\Big(\frac{(2+r_{k^*})+\sqrt{(2+r_{k^*})^2-4}}{2}\Big)\Big(\frac{(2+r_{k^*-1})+\sqrt{(2+r_{k^*-1})^2-4}}{2}\Big)} && \text{se } 1 < k^* < n \\
\tau^2 &= \frac{q_n}{\Big(\frac{(2+r_{n-1})+\sqrt{(2+r_{n-1})^2-4}}{2}\Big)} && \text{se} \quad k^* = n
\end{aligned}
\tag{4.4}
$$

oppure

$$
\begin{aligned}
\tau^2 &= \frac{q_1}{(1+\sqrt{r_1})^2} && \text{se } k^* = 1 \\
\tau^2 &= \frac{q_{k^*}}{(1+\sqrt{r_{k^*}})^2(1+\sqrt{r_{k^*-1}})^2} && \text{se } 1 < k^* < n \\
\tau^2 &= \frac{q_n}{(1+\sqrt{r_{n-1}})^2} && \text{se } k^* = n\,.
\end{aligned}
\tag{4.5}
$$

Anche in questo caso τ è solo una stima inferiore di $\sigma_n(B)$, e la sua validità è ottenuta alla stessa maniera del passo dqds del Parlett.

5 – Implementazione e prove numeriche

L'implementazione numerica prevede le seguenti fasi:

1. Calcolo ricorsivo delle quantità r_k e conseguente verifica di splitting; l'eventuale verificarsi dello splitting sospende il calcolo dei termini r_k successivi.
2. Completamento di un passo dell'algoritmo CR relativamente al blocco isolato dallo splitting.
3. Calcolo dello shift secondo le formule (4.5).
4. Calcolo di un passo della fattorizzazione di Cholesky con shift mediante l'algoritmo dqds; nel caso lo shift utilizzato non risulta idoneo, si procede ad una correzione aggiungendo altri fattori al denominatore nella (4.5) e si ripete il passo.

Le prove numeriche hanno evidenziato che le formule (4.5) pur essendo meno precise delle (4.4) risultano più efficienti complessivamente. Ciò è stato da noi attribuito al fatto che non sempre è verificata l'ipotesi (4.3), per cui si ottiene una stima dello shift che non soddisfa la relazione (4.1). Inoltre si è osservato che l'alternanza di un passo con shift, ed uno senza, sembra migliorare il processo iterativo, creando un effetto regolarizzante sulle matrici. La parte critica dell'algoritmo è il passo 4. che prevede il rigetto dello shift ed il suo ricalcolo; l'alternanza con un passo senza shift ha consentito una previsione dello shift valida quasi sempre al primo tentativo.

L'algoritmo proposto, da noi denominato *algoritmo di Cholesky-Rutishauser con shift*, è stato testato su dieci classi di matrici bidiagonali [5]-[8] e confrontato con l'algoritmo dqds con shift di Parlett. Per quanto riguarda la precisione dei valori singolari calcolati, la variazione percentuale è stata dell'ordine di grandezza della precisione utilizzata, a prova che entrambi gli algoritmi calcolano i valori singolari con la stessa precisione.

Le differenze si hanno nei tempi di esecuzione. Pertanto per ogni classe di matrici mostriamo gli speedup dei tempi d'esecuzione e gli speedup dei costi computazionali[(1)].

Le classi considerate sono:

1. Queste matrici hanno gli elementi diagonali ordinati dal più grande al più piccolo. Tutte le matrici hanno l'elemento 1 nella posizione (1,1) e ogni elemento sopradiagonale $(i, i+1)$ uguaglia il suo vicino (i, i) sulla diagonale. Di queste matrici quattro sono 10×10 e hanno un multiplo costante fra gli elementi adiacenti sulla diagonale e sopradiagonale: 10^{10}, 10^5, 10^2, 10. Le restanti, invece, sono 20×20 e si ottengono dalle prime quattro ripetendo ogni elemento due volte.
2. Questa classe è identica alla classe 1 eccetto l'ordine inverso degli elementi sulla diagonale e sopradiagonale.
3. Le matrici di questa classe hanno dimensione 20×20 e 40×40, e si ottengono unendo le matrici della classe 1 con quelle della classe 2.
4. Le matrici di questa classe hanno dimensione 20×20 e 40×40, e si ottengono unendo le matrici della classe 2 con quelle della classe 1.

[(1)]Il costo computazionale di una radice quadrata è stato valutato come rapporto tra il tempo di esecuzione di una radice e quello di una moltiplicazione del calcolatore utilizzato per i test.

5. Tale classe è costituita da un'unica matrice 41×41 le cui diagonali sono ordinate come nella classe 1 in cui il rapporto fra gli elementi adiacenti è 0.79 ed ogni elemento sopradiagonale è identico all'elemento diagonale sotto di esso.
6. Le matrici 20×20 appartenenti a questa classe sono state generate lasciando che ogni elemento della bidiagonale sia un numero casuale del tipo $r \times 10^i$, dove r è un numero uniformemente distribuito fra -0.5 e 0.5 ed i è un intero casuale uniformemente distribuito fra 0 e -15, 0 e -10, 0 e -5. Qualora i è nullo si ottiene una matrice i cui elementi sono uniformemente distribuiti fra -0.5 e 0.5.
7. La matrice considerata è la matrice bidiagonale di Toeplitz di ordine 100×100 con $a_i = 1$ e $b_i = 2$ per ogni i.
8. Le matrici di questa classe sono caratterizzate dall'aver scelto gli elementi diagonali a_i e gli elementi della sopradiagonale b_i nel seguente modo:

$$a_{i-1} = \beta a_i \text{ e } b_i = a_i \text{ con } a_n = 1$$

   over β è un intero ed n l'ordine della matrice.
9. Le matrici di questa classe sono ottenute dalle matrici della classe 8 invertendo l'ordine degli elementi.
10. Tre matrici sono 10×10 con elemento 1 sulla sopradiagonale e una costante rispettivamente uguale a 10^{-2}, 10^{-4}, 10^{-8}. Le altre tre matrici 20×20 si ottengono unendo due coppie di ciascuna delle prime tre matrici e ponendo l'elemento centrale sopradiagonale $B_{10,11}$ uguale a 10^{-15} volte il valore dell'elemento diagonale.

Classe 1

MATRICE Fattore	Dimensione	SPEEDUP Flops	SPEEDUP Time
10	10	1.402	1.549
10^2	10	1.630	1.888
10^5	10	1.892	2.119
10^{10}	10	2.542	2.707
10	20	1.407	1.426
10^2	20	1.535	1.320
10^5	20	1.776	1.386
10^{10}	20	1.938	1.324

Classe 2

MATRICE Fattore-Dimensione		SPEEDUP Flops	SPEEDUP Time
10	10	1.483	1.628
10^2	10	1.558	1.574
10^5	10	1.208	1.314
10^{10}	10	2.293	2.046
10	20	1.357	1.382
10^2	20	1.327	1.248
10^5	20	1.390	1.230
10^{10}	20	1.626	1.418

Classe 3

MATRICE Fattore-Dimensione		SPEEDUP Flops	SPEEDUP Time
10	20	1.604	1.695
10^2	20	1.761	1.711
10^5	20	2.129	1.985
10^{10}	20	1.465	1.368
10	40	1.680	1.606
10^2	40	1.805	1.632
10^5	40	2.038	1.686
10^{10}	40	1.544	1.286

Classe 4

MATRICE Fattore-Dimensione		SPEEDUP Flops	SPEEDUP Time
10	20	1.703	1.521
10^2	20	1.773	1.485
10^5	20	2.009	1.523
10^{10}	20	2.099	1.558
10	40	1.310	1.341
10^2	40	1.309	1.323
10^5	40	1.270	1.255
10^{10}	40	1.183	1.141

Classe 5

SPEEDUP Flops	SPEEDUP Time
1.445	1.679

Classe 6

MATRICE $r-i$		SPEEDUP Flops	SPEEDUP Time
$-0.5:0.5$	$0:-15$	1.528	1.432
$-0.5:0.5$	$0:-15$	1.961	1.658
$-0.5:0.5$	$0:-10$	1.384	1.245
$-0.5:0.5$	$0:-10$	1.529	1.366
$-0.5:0.5$	$0:-5$	1.292	1.299
$-0.5:0.5$	$0:-5$	1.628	1.595
$-0.5:0.5$	0	1.415	1.517
$-0.5:0.5$	0	1.587	1.701

Classe 7

SPEEDUP Flops	SPEEDUP Time
1.794	2.121

Classe 8

MATRICE Fattore-Dimensione		SPEEDUP Flops	SPEEDUP Time
60	20	1.932	2.082
60	40	2.855	2.832
60	80	4.700	4.396
2	20	1.424	1.613
2	40	1.558	1.774
2	80	1.964	2.172

Classe 9

MATRICE Fattore-Dimensione		SPEEDUP Flops	SPEEDUP Time
60	20	1.613	1.546
60	40	1.540	1.447
60	80	1.854	1.676
2	20	1.599	1.678
2	40	1.593	1.607
2	80	1.666	1.603

Classe 10

MATRICE Fattore	Dimensione	SPEEDUP Flops	SPEEDUP Time
10^{-2}	10	1.795	2.021
10^{-4}	10	2.043	2.311
10^{-8}	10	2.416	2.625
10^{-2}	20	1.914	2.033
10^{-4}	20	2.040	2.217
10^{-8}	20	2.515	2.690

BIBLIOGRAFIA

[1] K. Vince Fernando – Beresford N. Parlett: *Implict Cholesky algoritms for singular values and vectors of triangular matrices*, Numerical Linear Algebra Appl. (1995), 507-531.

[2] P. Deift – J. Demmel – L.-C. Li – C. Tomei: *The Bidiagonal Singular Value Decomposition and Hamiltonian Mechanics*, Siam J. Numer. Anal., **28** (1991), 1463-1516.

[3] J. Demmel, W. Kahan: *Accurate singular values of bidiagonal matrices*, Siam J. Sci. Statistic. Comput, **11** (1990) 873-192.

[4] G. Di Lena – R. I. Peluso – G. Piazza: *Teoria della Perturbazione Relativa dei Valori Singolari di una Matrice*, Rapporto n. 24/92, Dipartimento di Matematica, Università degli studi di Bari, Italia.

[5] J. Demmel – W. Kahan: *Computing Small Singular Value of Bidiagonal Matrices with Garanted High Relative Accuracy*, Lapack Working Note # 3.

[6] G. H. Golub – C. F. Loan: *Matrix Computations*, 3rd ed., The Johns Hopkins University Press Baltimore, Maryland, 1996.

[7] R. A. Horn – C. A. Johnson: *Topics in Matrix Analysis*, Cambridge University 1991.

[8] K. Vince Fernando – Beresford N. Parlett: *Accurate Singular Values and Differential qd Algorithms*, Numer. Math., **67** (1994), 191-229.

[9] Beresford N. Parlett: *The New qd Algorithms*, Acta Numerica (1995), 459-491.

[10] E. Andrisani – G. Di Lena: *Tecniche di calcolo dello shift nell'algoritmo di Cholesky-Rutishauser per la fattorizzazione SVD*, Rapporto n. 20/97, Dipartimento di Matematica, Università degli studi di Bari, Italia.

[11] G. Di Lena – R. I. Peluso – G. Piazza: *Results on Relative Perturbation of the Singular Values of a Matrix*, BIT 33 (1993), 647-653.

[12] R. A. Horn – C. A. Johnson: *Matrix Analysis*, Cambridge University 1985.

[13] J. Demmel: *Applied Numerical Linear Algebra*, Berkeley, University of California, Siam 1997.

Lavoro pervenuto alla redazione il 9 febbraio 1998
ed accettato per la pubblicazione il 12 ottobre 1998.
Bozze licenziate il 17 dicembre 1998

Indirizzo degli Autori:

E. Andrisani – Via N. Columella, 5 – 75100 Matera, Italy

G. Di Lena – Dipartimento Interuniversitario di Matematica – Via E. Orabona, 4 – 70125 Bari, Italy