Università degli Studi di Pisa

Facoltà di Scienze Matematiche, Fisiche e Naturali

Corso di Laurea in Matematica

Tesi di Laurea Specialistica
29 Giugno 2007

Metodi per la soluzione veloce di una classe di equazioni di Riccati algebriche

Candidato
Federico G. Poloni

Relatori
prof. Dario A. Bini
prof. Beatrice Meini

Controrelatore
prof. Luca Gemignani

⟨*Inserisci il tuo nome qui*⟩

Introduzione

In questo lavoro di tesi consideriamo l'equazione matriciale di secondo grado della forma

$$XCX + B - AX - XE = 0, \tag{1}$$

con $A \in \mathbb{R}^{m\times m}$, $X, B \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{n\times m}$, $E \in \mathbb{R}^{n\times n}$, nota come *equazione di Riccati algebrica non simmetrica* (NARE). In particolare, siamo interessati a studiare un caso particolare dell'equazione, derivante da un problema fisico nell'ambito della teoria del trasporto di neutroni, nel quale i coefficienti sono nella forma

$$\begin{aligned}
&B = ee^T, && C = qq^T,\\
&A = \Delta - eq^T, && E = D - qe^T\\
&D = \operatorname{diag}(d_1, \ldots, d_n), && \Delta = \operatorname{diag}(\delta_1, \ldots, \delta_n),\\
&d_i = \frac{1}{cx_i(1-\alpha)}, && \delta_i = \frac{1}{cx_i(1+\alpha)},\ i = 1, \ldots, n,\\
&e = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T, && q_i = \frac{w_i}{2x_i},\ i = 1, \ldots, n,
\end{aligned} \tag{2}$$

dove $0 \leqslant \alpha < 1$ e $0 < c \leqslant 1$ sono opportuni parametri reali.

Il caso simmetrico dell'equazione di Riccati algebrica (cioè quello in cui B e C sono hermitiane, e $A = D^H$) è ampiamente studiato (si veda per esempio la monografia [22]), mentre quello generale è meno noto, anche se recentemente è stato oggetto di diversi articoli in letteratura (per esempio [16, 12, 14, 4]). In particolare, il problema fisico che abbiamo esposto appare in letteratura come uno dei pochi esempi di equazioni di Riccati non simmetriche che emergono da contesti applicativi [16, 12].

Sotto opportune ipotesi sui coefficienti della (1), è possibile dimostrare l'esistenza di una soluzione non negativa X^* minimale secondo l'ordinamento termine a termine, che è quella che tipicamente si è interessati a calcolare. Gli algoritmi migliori sviluppati finora sono metodi iterativi che hanno un costo di $O(n^3)$ operazioni per passo e convergenza quadratica alla solu-

zione minimale non negativa [1] . Esaminiamo in particolare questi quattro algoritmi.

(a) il metodo di Newton [16],

(b) lo *structured doubling algorithm* [17],

(c) la riduzione ciclica [27],

(d) il metodo di Newton applicato all'iterazione di Lu [23] (solo per il problema (2)).

In particolare, (d) è il primo tentativo apparso in letteratura di costruire algoritmi specializzati per il problema (2). Tale algoritmo, pur velocizzando significativamente il calcolo della soluzione, mantiene una complessità di $O(n^3)$.

In questa tesi studiamo la struttura delle matrici coinvolte nella specializzazione degli algoritmi citati al problema fisico (2). In particolare, l'interesse primario dell'algebra lineare numerica è quello di riconoscere e sfruttare la struttura delle matrici coinvolte; poiché nel nostro problema le matrici hanno forti proprietà strutturali (rango 1, diagonale + rango 1), e in particolare dipendono da $O(n)$ parametri anziché da $O(n^2)$, è ragionevole aspettarsi che sia possibile arrivare ad algoritmi che riducano significativamente il costo computazionale rispetto al caso generale.

Riusciamo in effetti a sviluppare versioni strutturate dei quattro algoritmi esistenti, raggiungendo in tutti i casi un costo computazionale di $O(n^2)$ operazioni per passo (e mantenendo la convergenza quadratica). Gli algoritmi che proponiamo sono più veloci di quelli generali già per problemi di piccole dimensioni, e velocizzano notevolmente la soluzione al calcolatore dell'equazione di Riccati che risolve il problema fisico in esame. Mostriamo inoltre come sia possibile applicare agli algoritmi sviluppati la *tecnica di shift* [18] per accelerare la convergenza nei cosiddetti *casi critici* del problema (cioè, per il caso (2), quando $c = 1, \alpha = 0$). Gli esperimenti numerici condotti evidenziano l'efficacia dell'approccio proposto.

Come risultato supplementare, riusciamo a dimostrare alcune interessanti relazioni algebriche che legano gli algoritmi analizzati e forniscono nuovi spunti per l'analisi della convergenza e lo sviluppo di nuovi algoritmi. In particolare, proviamo che il metodo (d) calcola la stessa iterazione del metodo

[1] Vale la pena di notare che tutti gli algoritmi per la NARE implementabili su un calcolatore devono essere iterativi, in quanto la soluzione non è una funzione razionale dei coefficienti — per rendersene conto basta considerare il caso in dimensione 1, che è la comune equazione di secondo grado $ax^2 + bx + c = 0$.

(a) lavorando direttamente sui generatori della struttura di rango delle matrici coinvolte, e che il metodo (b) coincide essenzialmente con il metodo (c), a meno dell'applicazione di una trasformazione preliminare dell'equazione.

La tesi è così strutturata . Nel capitolo 1 introdurremo le notazioni e i teoremi di algebra lineare che andremo ad utilizzare nella trattazione; successivamente, nel capitolo 2, descriveremo i risultati di esistenza e gli algoritmi noti per la soluzione numerica delle NARE; nel capitolo 3 descriveremo alcuni algoritmi *ad–hoc* per il problema del trasporto di neutroni e come gli algoritmi generali possano essere adattati ad esso tenendo conto della struttura del problema, ed esporremo contestualmente le relazioni trovate tra i vari algoritmi. Infine, nei capitoli 4 e 5 esporremo brevemente i risultati numerici dell'implementazione dei metodi descritti e le conclusioni.

Capitolo 1

Risultati di algebra lineare

1.1 Matrici positive e M-matrici

Una matrice $P \in \mathbb{R}^{m\times n}$, è detta *positiva* se $P_{ij} > 0 \quad \forall i = 1,\dots,m, j = 1,\dots,n$; la notazione utilizzata per indicarlo è $P > 0$. Una matrice P diversa dalla matrice nulla è detta *non negativa* se $P_{ij} \geqslant 0 \quad \forall i = 1,\dots,m, j = 1,\dots,n$, e la notazione utilizzata è $P \geqslant 0$. Diremo più in generale che $A > B$ (risp. $A \geqslant B$) quando $A - B > 0$ (risp. $A - B \geqslant 0$).

Una matrice non negativa P è detta *riducibile* quando esiste una matrice di permutazione Π tale che

$$\Pi P \Pi^T = \begin{bmatrix} A & B \\ 0 & D \end{bmatrix},$$

con i blocchi A e D quadrati; cioè, quando P, a meno di permutazioni degli indici, è triangolare a blocchi. Le matrici positive hanno interessanti proprietà spettrali, come è provato dal seguente teorema [2].

Teorema 1.1 (Perron–Frobenius)**.** *Sia $P \in \mathbb{R}^{n\times n}$ positiva oppure non negativa irriducibile; allora,*

1. *P ha un autovalore strettamente positivo λ, detto* raggio di Perron *(o* valore di Perron*);*
2. *λ è il più grande (in modulo) tra gli autovalori di P (cioè $\lambda = \rho(P)$);*
3. *λ è un autovalore semplice;*
4. *l'autovettore v associato a λ (detto* vettore di Perron*) è strettamente positivo;*
5. *v è l'unico autovettore non negativo di P.*

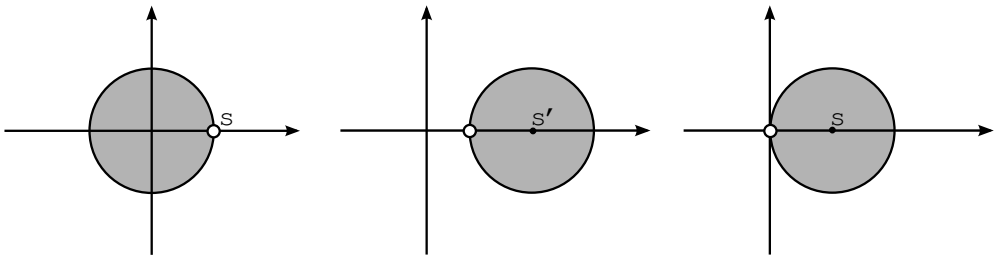

Figura 1.1: Gli spettri di una matrice positiva P con $\rho(P) = s$ (a sinistra), di una M-matrice non singolare $s'I - P$ (al centro), di una M-matrice singolare (a destra) $sI - P$, sono racchiusi all'interno delle aree grigie. Il cerchio bianco rappresenta la posizione del valore di Perron.

6. *$\lambda = \lambda(P)$ è strettamente crescente in ognuno degli elementi della matrice: cioè, se $Q \geqslant P$, allora $\lambda(Q) > \lambda(P)$.*

Risultati analoghi valgono per le matrici non negative *riducibili*; in particolare, tutte le disuguaglianze strette nel teorema appena enunciato diventano larghe.

Una matrice $Z \in \mathbb{R}^{n\times n}$ è detta *Z-matrice* se $Z_{ij} \leqslant 0$ per ogni $i \neq j$. Una Z-matrice M è detta *M-matrice* se esistono $s > 0, P \geqslant 0$ (con $s \in \mathbb{R}$, $P \in \mathbb{R}^{n\times n}$ e $\rho(P) \leqslant s$) tali che $M = sI_n - P$. In virtù del teorema 1.1, gli autovalori di una M-matrice sono contenuti nel disco

$$\{z \in \mathbb{C} : |z - s| \leqslant \rho(P)\},$$

(vedasi la figura 1.1 per un esempio) quindi in particolare hanno tutti parte reale non negativa, e $s - \rho(P)$ è l'autovalore con parte reale minore (detto *valore di Perron*). Una M-matrice è singolare se e solo se $s = \rho(P)$.

Valgono inoltre le seguenti proprietà [2]:

Teorema 1.2. *Sia Z una Z-matrice. Allora,*

1. *Se M è una M-matrice e $Z \geqslant M$, allora Z è una M-matrice non singolare;*

2. *Z è una M-matrice se e solo se esiste un vettore $v \geqslant 0$ (diverso dal vettore nullo) tale che $Zv \geqslant 0$;*

3. *Se Z è invertibile, Z è una M-matrice se e solo se $Z^1 \geqslant 0$;*

4. *Z è una M-matrice se e solo se tutti i suoi autovalori hanno parte reale non negativa.*

Ricordiamo che data una matrice M partizionabile come

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix},$$

con A e D quadrate, A è detta *sottomatrice principale di testa* di M, e $D - CA^{-1}B$ (quando A è invertibile) è detto *complemento di Schur* di A in M. Per le M-matrici valgono le seguenti proprietà.

Lemma 1.3. *Sia M una M-matrice; allora,*

1. *Le sottomatrici principali di testa di M e i loro complementi di Schur sono a loro volta M-matrici*
2. *Se M è non singolare, allora lo sono anche le sue sottomatrici principali di testa e i loro complementi di Schur. In particolare, il loro valore di Perron è maggiore o uguale di quello della matrice originale*

In virtù di queste proprietà, l'eliminazione di Gauss per M-matrici non singolari è numericamente ben condizionata senza bisogno di fare ricorso al pivoting; infatti, le sottomatrici principali di coda costruite durante l'eliminazione di Gauss sono i complementi di Schur di M, e il loro condizionamento (determinato dall'autovalore di modulo più piccolo, cioè dal valore di Perron) non è peggiore di quello della matrice originale.

1.2 Formula di Sherman–Morrison–Woodbury e sue applicazioni

La formula di Sherman–Morrison–Woodbury (in breve *SMW*) è la seguente utile identità [11, pag. 50].

Lemma 1.4. *Siano $D \in \mathbb{R}^{n\times n}$ e $C \in \mathbb{R}^{k\times k}$ matrici invertibili, e $U \in \mathbb{R}^{n\times k}$, $V \in \mathbb{R}^{k\times n}$. Allora $D - UCV$ è invertibile se e solo se lo è $C^{-1} - VD^{-1}U$, e in tal caso si ha*

$$(D - UCV)^{-1} = D^{-1} + D^{-1}U\left(C^{-1} - VD^{-1}U\right)^{-1}VD^{-1}.$$

Il seguente risultato lega tra loro la formula SMW e le M-matrici.

Lemma 1.5. *Siano D, C, U, V come dalle ipotesi del lemma 1.4, con $D, C \geqslant 0$ diagonali e invertibili, e $U, V \geqslant 0$. Allora, $D-UCV$ è una M-matrice (non singolare) se e solo se $C^{-1} - VD^{-1}U$ è una M-matrice (non singolare).*

Dimostrazione. Sia $C^{-1} - VD^{-1}U$ una M-matrice non singolare; allora, applicando la formula SMW abbiamo

$$(D - UCV)^{-1} = D^{-1} + D^{-1}U(C^{-1} - VD^{-1}U)^{-1}VD^{-1}.$$

Per il terzo punto del lemma 1.2, $(C^{-1} - VD^{-1}U)^{-1}$ è positiva; il membro di destra allora è non negativo perché somma di matrici non negative, quindi $(D - UCV)^{-1} \geqslant 0$, da cui, di nuovo per il terzo punto del lemma 1.2, la Z-matrice $D - UCV$ è una M-matrice. L'altra implicazione si dimostra in modo analogo, applicando la formula SMW alla matrice $C^{-1} - VD^{-1}U$. I risultati possono poi essere estesi alle M-matrici singolari con un semplice argomento di continuità. □

1.3 Operatori di displacement e matrici Cauchy-like

1.3.1 Matrici Cauchy-like e Trummer-like

Nel seguito faremo riferimento ad alcune proprietà delle matrici Cauchy-like. Una matrice $C \in \mathbb{R}^{n\times n}$ è detta *di Cauchy* se i suoi elementi sono della forma

$$C_{ij} = \frac{u_i v_j}{r_i - s_j},$$

per opportuni vettori u_i, v_i, r_i, s_i, $i = 1, \ldots, n$ tali che $r_i \neq s_j \ \forall i, j = 1, \ldots, n$. Definendo $R = \operatorname{diag}(r_1, \ldots, r_n)$ e $S = \operatorname{diag}(s_1, \ldots, s_n)$, chiaramente possiamo riscrivere la formula precedente come

$$RC - CS = uv^T.$$

L'operatore $\nabla_{R,S} : C \mapsto RC - CS$ è detto operatore di *displacement*, e u e v sono chiamati *generatori* di C.

Più in generale, date due matrici $R, S \in \mathbb{R}^{n\times n}$, definiamo l'*operatore di displacement* $\nabla_{R,S} : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ come $\nabla_{R,S}(C) = RC - CS$. Quando $r = \operatorname{rk}(\nabla_{R,S} C)$, diciamo che C ha *rango di displacement* r rispetto alla coppia (R, S).

Una matrice C è detta *Cauchy-like* (di rango r, dove solitamente r è basso rispetto alla dimensione n della matrice) se essa ha rango di displacement r rispetto a due matrici diagonali $R = \operatorname{diag}(r_1, \ldots, r_n)$ e $S = \operatorname{diag}(s_1, \ldots, s_n)$ tali che $r_i \neq s_j \ \forall i, j = 1, \ldots, n$. Segue dalle definizioni che le matrici Cauchy-like di rango 1 coincidono con le matrici di Cauchy, come definite poco sopra. Una matrice Cauchy-like di rango r è determinata da $2nr + 2n$ parametri

reali: basta infatti specificare i due vettori $\mathrm{diag}(R)$ e $\mathrm{diag}(S)$ che definiscono l'operatore di displacement, e due matrici $U \in \mathbb{R}^{n\times r}, V \in \mathbb{R}^{r\times n}$ tali che $RC - CS = UV$ (dette *generatori*).

L'operatore di displacement $\nabla_{R,S}$ associato a una matrice di Cauchy C (quindi con $R = \mathrm{diag}(r_1, \ldots, r_n)$, $S = \mathrm{diag}(s_1, \ldots, s_n)$) è invertibile; difatti, è semplice notare che $(\nabla_{R,S} C)_{ij} = (r_i - s_j)C_{ij}$; pertanto vale la seguente semplice formula esplicita per l'inverso di $\nabla_{R,S}$:

$$(\nabla_{R,S})^{-1} M = N \text{ dove } N_{ij} = \frac{M_{ij}}{r_i - s_j}.$$

Chiameremo inoltre una matrice T *Trummer-like*[1] quando essa ha rango di displacement basso rispetto alla coppia (D, D), per un'opportuna matrice diagonale $D = \mathrm{diag}(d_1, \ldots, d_n)$, con $d_1, \ldots, d_n$ distinti. L'operatore displacement associato $\nabla_{D,D}$ è singolare: infatti, data una qualunque altra matrice diagonale E, $DE - ED = 0$. D'altra parte, è semplice dimostrare (per esempio utilizzando i risultati della sezione 1.4) che il nucleo dell'operatore di displacement è costituito proprio dall'insieme delle matrici diagonali. Ne segue che una matrice Trummer-like T è determinata da $2rn + 2n$ parametri: i due vettori $\mathrm{diag}(D)$ e $\mathrm{diag}(T)$, e due matrici $U \in \mathbb{R}^{n\times r}, V \in \mathbb{R}^{r\times n}$ tali che $DT - TD = UV$.

1.3.2 Prime proprietà degli operatori di displacement

Gli operatori di displacement godono di proprietà formali molto simili a quelle dell'usuale derivata.

Lemma 1.6. *Valgono le seguenti identità, dove tutte le variabili che compaiono sono matrici $n \times n$*

1. *Se D ed M sono diagonali, $\nabla_{D,D} M = 0$.*

2. *(Linearità)*
$$\nabla_{R,S}(M + N) = \nabla_{R,S}(M) + \nabla_{R,S}(N);$$

3. *(Displacement del prodotto)*
$$\nabla_{R,T}(MN) = \nabla_{R,S}(M)N + M\,\nabla_{S,T}(N)$$

[1] Il nome scelto deriva dal fatto che il problema di moltiplicare velocemente una matrice Trummer-like per un vettore è noto in letteratura come *problema di Trummer*, si veda [9] e i riferimenti in esso riportati per ulteriori dettagli.

con la sua generalizzazione a k termini

$$\begin{aligned}\nabla_{R_0,R_k}(M_1M_2\cdots M_k) = &\nabla_{R_0,R_1}(M_1)M_2\cdots M_n + \cdots + \\ &M_1\cdots M_j\,\nabla_{R_j,R_{j+1}}(M_{j+1})M_{j+2}\cdots M_k + \cdots + \\ &M_1M_2\cdots M_{k-1}\,\nabla_{R_{k-1},R_k}(M_k);\end{aligned}$$

4. (Displacement dell'inversa)

$$\nabla_{R,S}(M^{-1}) = -M^{-1}\,\nabla_{S,R}(M)M^{-1}.$$

1.3.3 Algoritmi per matrici Cauchy-like e Trummer-like

Poiché i generatori permettono di ricostruire in tempo al più $O(r)$ ognuna delle entrate di una matrice Cauchy-like o Trummer-like, è semplice sviluppare un algoritmo per il prodotto matrice–vettore in tempo $O(rn^2)$: è sufficiente ricostruire la matrice a partire dai suoi generatori, e quindi applicare il normale algoritmo per il prodotto matrice–vettore. Analogamente, il prodotto di una Cauchy-like o di una Trummer-like per una matrice $n \times s$ costa $O(rn^2 + sn^2)$, visto che i due passaggi di ricostruzione della matrice e di moltiplicazione sono indipendenti. Presentiamo nell'algoritmo 1.1 una semplice implementazione in Octave/Matlab dell'algoritmo.

Più interessante è il problema di risolvere velocemente sistemi lineari con matrice Cauchy-like o Trummer-like. Nel caso Cauchy, il problema è stato trattato da Gohberg, Kailath e Olshevsky [10] (algoritmo GKO). L'idea è quella di sfruttare il fatto che il complemento di Schur di una matrice Cauchy-like è Cauchy-like, e i suoi generatori sono facilmente calcolabili a partire da quelli della matrice originale. In dettaglio, se le matrici coinvolte sono partizionate (con i primi blocchi di dimensione 1 e i secondi di dimensione $n-1$) come

$$\begin{bmatrix} r_1 & \\ & R_2 \end{bmatrix}\begin{bmatrix} c_{11} & c_{12} \\ c_{21} & C_{22} \end{bmatrix} - \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & C_{22} \end{bmatrix}\begin{bmatrix} s_1 & \\ & S_2 \end{bmatrix} = \begin{bmatrix} u_1 \\ U_2 \end{bmatrix}\begin{bmatrix} v_1 & V_2 \end{bmatrix},$$

allora il complemento di Schur $\widehat{C} = C_{22} - c_{21}c_{11}c_{12}$ soddisfa l'equazione di displacement

$$R_1\widehat{C} - \widehat{C}S_2 = (U_2 - \frac{1}{c_{11}}c_{21}u_1)(V_2 - \frac{1}{c_{11}}c_{12}v_1).$$

Utilizzando questo fatto, possiamo effettuare l'eliminazione di Gauss sulla matrice C in tempo $O(rn^2)$: ad ogni passo, non abbiamo bisogno di calcolare esplicitamente il complemento di Schur, ma solamente i suoi generatori,

```
function y = camv(r, s, u, v, x)
% returns y = C*x, where C satisfies
% diag(r)*C - C*diag(S) = u*v
% x may be a vector or a matrix
  n = size(u, 1);
  C = (u*v) ./ (r*ones(1, n) - ones(n, 1)*s);
  y = C*x;
endfunction
function y = trmv(d, dg, u, v, x)
% returns y = T*x, where T satisfies
% diag(d)*T - T*diag(d) = u * v
% and diag(T) = dg
  n = size(u, 1);
  T = (u*v) ./ (d*ones(1, n) - ones(n, 1)*d + eye(n));
  for i = 1:n
    T(i,i) = dg(i);
  endfor
  y = T*x;
endfunction
```

Algoritmo 1.1: Prodotto MV per matrici Cauchy-like e Trummer-like

in tempo $O(rn)$. In questo modo, si può effettuare l'eliminazione di Gauss di una matrice Cauchy-like con costo $O(rn^2)$. Combinando con l'usuale algoritmo di sostituzione all'indietro, possiamo risolvere un sistema lineare in tempo $O(rn^2)$. Analogamente, la soluzione simultanea di s sistemi lineari con la stessa matrice può essere effettuata in $O(rn^2 + sn^2)$ (l'eliminazione di Gauss viene effettuata una volta sola, la sostituzione all'indietro s volte). Nell'algoritmo può essere facilmente introdotto il pivoting parziale per migliorarne la stabilità numerica; tuttavia ciò non sarà necessario nella nostra trattazione perché i sistemi che risolveremo avranno come matrici M-matrici non singolari.

Nel caso di matrici Trummer-like, l'algoritmo descritto in [10] può essere adattato con poco sforzo. Infatti, ad ogni passo dell'eliminazione di Gauss, possiamo aggiornare la diagonale come se stessimo procedendo a un'eliminazione di Gauss tradizionale, e aggiornare implicitamente il resto della matrice come dall'algoritmo GKO. Presentiamo come riferimento una semplice implementazione nell'algoritmo 1.2

Poiché le matrici Cauchy-like e Trummer-like sono definiti da un numero di generatori che cresce linearmente con n (quando r è indipendente da n), ha senso chiedersi se è possibile eseguire le operazioni riportate con complessità computazionale minore. In effetti esistono in letteratura algoritmi approssimati per il prodotto Cauchy-like–vettore [9] e per la soluzione di sistemi lineari [25] in tempo $O(n \log^2 n)$ (algoritmi di tipo *superfast*). Tali algoritmi sono approssimati, quindi il loro utilizzo negli algoritmi che esporremo potrebbe causare un eccessivo accumulo di errori numerici, e quindi una perdita di accuratezza nella soluzione, o addirittura la non convergenza di alcuni algoritmi. Inoltre, un'analisi completa della loro stabilità esiste solo per particolari scelte delle matrici diagonali R e S che compaiono nella definizione di $\nabla_{R,S}$. Pertanto in questa tesi ci limitiamo a trattare il problema attraverso l'utilizzo degli algoritmi esatti. Una trattazione ulteriore sarebbe necessaria per determinare se i metodi superfast possono essere applicati o meno nell'implementazione degli algoritmi qui presentati.

1.4 Prodotto di Kronecker e equazione di Sylvester

Siano $A = (a_{ij}) \in \mathbb{R}^{m \times n}$ e $B = (b_{kl}) \in \mathbb{R}^{p \times q}$; definiamo $A \otimes B$ come la matrice $mp \times nq$ tale che

$$(A \otimes B)_{(i-1)p+k,(j-1)q+l} = a_{ij} b_{kl},$$

```
function x = trsv(d, dg, u, v, b)
% returns x = T^{-1} b, where T satisfies
% diag(d)*T - T*diag(d) = u*v
% and diag(T) = dg
% precondition: Gaussian elimination without pivoting
%  is stable for T
  n = size(u, 1);
  U = zeros(n);    %matrix U of the LU factorization
  l = zeros(1, n); %active column of L of the LU factorization
  x = b;
  for k = 1:n
    % generates a column of L and solves L^{-1}*b on-the-fly
    l(k+1:n) = (u(k+1:n,:)*v(:,k) / dg(k)) ./ (d(k+1:n)-d(k));
    x(k+1:n) = x(k+1:n) - l(k+1:n)*b(k);
    if(abs(dg(k)) < 1.d-10) warn "Near-to-singular matrix";
    % generates a row of U
    U(k,k) = dg(k);
    U(k,k+1:n) = (u(k,:)*v(:,k+1:n)) ./ (d(k)-d(k+1:n));
    %updates the generators to generators of the Schur complement
    u(k+1:n,:) = u(k+1:n,:) - l(k+1:n)*u(k,:);
    v(:,k+1:n) = v(:,k+1:n) - v(:,k)*U(k,k+1:n) / dg(k);
    %updates the diagonal
    dg(k+1:n) = dg(k+1:n) - l(k+1:n)*U(k,k+1:n);
  endfor
endfunction
```

Algoritmo 1.2: Soluzione di un sistema lineare con matrice Trummer-like

cioè la matrice a blocchi $p \times q$ che ha nel blocco in posizione (i, j) la matrice $a_{ij}B$. Indicheremo inoltre, per ogni m, n, con vec l'operatore $\mathbb{R}^{m\times n} \mapsto \mathbb{R}^{mn}$ tale che per ogni matrice $M = (m_{ij}) \in \mathbb{R}^{m\times n}$

$$(\operatorname{vec} M)_{(i-1)n+j} = m_{ij},$$

cioè l'operatore che prende le colonne di M e le concatena in modo da formare un unico vettore.

Diverse utili identità matriciali coinvolgono l'operatore vec e il prodotto di Kronecker [19].

Lemma 1.7. *Per tutte le matrici A, B, C, D per cui le operazioni indicate hanno senso, si ha*

1. $(A \otimes B)(C \otimes D) = (AC \otimes BD)$
2. $(A \otimes B)^{-1} = (A^{-1} \otimes B^{-1})$
3. $\operatorname{vec}(ABC) = (C^T \otimes A) \operatorname{vec} B$

L'equazione

$$AX - XB = C, \quad X, C \in \mathbb{R}^{m\times n}, \quad A \in \mathbb{R}^{m\times m}, \quad B \in \mathbb{R}^{n\times n} \tag{1.1}$$

(o anche $\nabla_{A,B}(X) = C$, utilizzando la notazione della sezione 1.3) è detta *equazione di Sylvester*. Utilizzando le proprietà appena descritte, siamo in grado di riscriverla come

$$(I_n \otimes A - B^T \otimes I_m) \operatorname{vec} X = \operatorname{vec} C,$$

cioè un sistema lineare di dimensione $mn \times mn$. Risolvere esplicitamente il sistema lineare non è solitamente un buon algoritmo risolutivo per l'equazione di Sylvester, soprattutto per l'elevato costo computazionale ($O(m^3n^3)$ in assenza di strutture); tuttavia, attraverso questa forma siamo in grado di studiare più facilmente le proprietà delle soluzioni. L'esistenza e l'unicità della soluzione dipendono dalla singolarità o meno della matrice $I_n \otimes A - B^T \otimes I_m$. In particolare, quando $m = n$ e quindi tutte le matrici coinvolte sono quadrate, possiamo esprimere la condizione di singolarità attraverso gli autovalori di A e B:

Lemma 1.8. *Siano A e B matrici quadrate $n \times n$; l'equazione* (1.1) *ha una e una sola soluzione se e solo se A e B non hanno autovalori comuni.*

Dimostrazione. Siano $A = UMU^{-1}$ e $B^T = VNV^{-1}$ le forme canoniche di Jordan rispettivamente di A e B^T. Allora,

$$I \otimes A - B^T \otimes I = (U \otimes V)(I \otimes M - N \otimes I)(U \otimes V)^{-1}.$$

Il termine di destra ha gli stessi autovalori della matrice $I \otimes M - N \otimes I$, che è triangolare superiore per la struttura della forma canonica di Jordan. In particolare, gli elementi sulla diagonale sono tutte le somme della forma $\lambda_i - \mu_j$, dove $(\lambda_i)_{i=1,\dots,n}$ e $(\mu_j)_{j=1,\dots,n}$ sono gli autovalori di A e B^T, contati con la loro molteplicità. In particolare, $I \otimes A - B^T \otimes I$ è singolare se e solo se almeno uno degli elementi sulla diagonale è nullo, cioè quando esistono i e j tali che $\lambda_i = -\mu_j$, cioè quando λ_i è un autovalore comune a A e a B. □

Per quanto riguarda la soluzione numerica dell'equazione di Sylvester, un algoritmo comunemente usato è il metodo di Bartels–Stewart [1], che si basa sulla riduzione in forma di Schur di A e B attraverso il metodo QR.

Capitolo 2

Forma della soluzione e algoritmi risolutivi per le equazioni di Riccati

2.1 Prime proprietà delle soluzioni

Si dice *equazione di Riccati algebrica non simmetrica* (o in breve *NARE*, dall'inglese) l'equazione matriciale della forma

$$XCX - AX - XE + B = 0 \tag{2.1}$$

con $B, C \in \mathbb{R}^{m\times n}, A \in \mathbb{R}^{m\times m}, E \in \mathbb{R}^{n\times n}, C \in \mathbb{R}^{n\times m}$. All'equazione possiamo associare le seguenti matrici $2n \times 2n$

$$\mathcal{H} = \begin{bmatrix} E & -C \\ B & -A \end{bmatrix}, \quad \mathcal{J} = \begin{bmatrix} I_n & 0 \\ 0 & -I_n \end{bmatrix}, \quad \mathcal{M} = \mathcal{J}\mathcal{H} = \begin{bmatrix} E & -C \\ -B & A \end{bmatrix}; \tag{2.2}$$

la matrice $\mathcal{H}$, in particolare, è detta *Hamiltoniana* dell'equazione. Notiamo che, se S è una soluzione dell'equazione, allora

$$\mathcal{H}\begin{bmatrix} I_n \\ S \end{bmatrix} = \begin{bmatrix} I_n \\ S \end{bmatrix}(E - CS), \tag{2.3}$$

cioè, Span $\begin{bmatrix} I_n \\ S \end{bmatrix}$ è un sottospazio invariante per $\mathcal{H}$[1]. Viceversa, se Span $\begin{bmatrix} U \\ V \end{bmatrix}$ è un sottospazio invariante per $\mathcal{H}$ di dimensione n e U è invertibile, allora $S = VU^{-1}$ è una soluzione dell'equazione di Riccati. Se $\mathcal{H}$ è diagonalizzabile,

[1] Indichiamo qui con Span M lo spazio vettoriale generato dalle colonne della matrice M.

i suoi sottospazi invarianti sono quelli generati dagli autovettori, quindi le soluzioni dell'equazione di Riccati sono tutte della forma

$$S = \begin{bmatrix} v_1 & v_2 & \cdots & v_n \end{bmatrix} \begin{bmatrix} u_1 & u_2 & \cdots & u_n \end{bmatrix}^{-1},$$

dove i vettori $\begin{bmatrix} u_i \\ v_i \end{bmatrix}$ sono autovettori di $\mathcal{H}$. In particolare, nel caso in cui $\mathcal{H}$ ha autovalori semplici ci sono al più $\binom{2n}{n}$ soluzioni.

La strada di calcolare gli autovettori di $\mathcal{H}$ però spesso non è praticabile per la soluzione numerica dell'equazione di Riccati, in quanto il condizionamento del problema agli autovettori per $\mathcal{H}$ può essere molto peggiore rispetto a quello della NARE (si veda per esempio [14] per un'analisi del condizionamento della NARE).

2.2 Equazioni di Riccati e M-matrici

Un caso frequente nelle applicazioni è quello in cui $\mathcal{M}$, come definita nella (2.2), è una M-matrice non singolare oppure singolare irriducibile. In tal caso, è possibile dimostrare che esiste una soluzione non negativa S tale che per ogni altra soluzione non negativa S' si abbia $S' \geqslant S$ (*soluzione minimale non negativa*) [26, 16, 12, 13].

Teorema 2.1. *Siano i coefficienti della* (2.1) *tali da rendere* $\mathcal{M}$ *una* M*-matrice non singolare oppure singolare irriducibile. Valgono allora i seguenti risultati.*

1. *La NARE* (2.1) *ha una soluzione minimale non negativa* S*.*
2. $E - CS$ *e* $A - SC$ *sono* M*-matrici irriducibili.*
3. *si ha la seguente decomposizione di* $\mathcal{H}$*:*

$$\begin{bmatrix} I & 0 \\ -S & I \end{bmatrix} \begin{bmatrix} E & -C \\ B & -A \end{bmatrix} \begin{bmatrix} I & 0 \\ S & I \end{bmatrix} = \begin{bmatrix} E - CS & -C \\ 0 & -(A - SC) \end{bmatrix}.$$

   In particolare, gli autovalori di $\mathcal{H}$ *sono autovalori di* $E - CS$ *o di* $-(A - SC)$*; i primi hanno parte reale non negativa, mentre i secondi hanno parte reale non positiva.*

4. *Detta* T *la soluzione minimale non negativa della NARE*

$$XBX - XA - DX + C = 0, \tag{2.4}$$

si ha, analogamente alla (2.3)*,*

$$\begin{bmatrix} E & -C \\ B & -A \end{bmatrix} \begin{bmatrix} T \\ I \end{bmatrix} = - \begin{bmatrix} T \\ I \end{bmatrix} (A - BT). \tag{2.5}$$

Nel caso in cui $\mathcal{M}$ sia singolare irriducibile, inoltre, otteniamo dal teorema di Perron–Frobenius che esistono due vettori positivi $u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$ e $v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$ tali che

$$u^T \mathcal{M} = 0, \quad \mathcal{M} v = 0, \text{ e quindi } (u^T \mathcal{J})\mathcal{H} = 0, \mathcal{H} v = 0. \tag{2.6}$$

Dalla quantità $\mu := u^T \mathcal{J} v = u_1^T v_1 - u_2^T v_2$ (detta *drift*) dipendono alcune proprietà dell'equazione. In particolare, adattando la terminologia consuetamente usata nelle catene di Markov [19, 6], l'equazione di Riccati è detta

- *positiva ricorrente*, se $\mu > 0$;
- *nulla ricorrente*, se $\mu = 0$;
- *transiente*, se $\mu < 0$.

Il caso nullo ricorrente corrisponde a un autovalore zero di $\mathcal{H}$ con molteplicità algebrica 2, e richiede particolare attenzione nella formulazione degli algoritmi che tratteremo, in quanto spesso le proprietà di convergenza risultano indebolite. Chiameremo tale caso *caso critico*, e tutti gli altri ($\mathcal{M}$ non singolare, oppure $\mathcal{M}$ singolare irriducibile transiente o positiva ricorrente) *casi non critici*.

2.3 Iterazioni funzionali

Supponiamo che i coefficienti dell'equazione di Riccati siano tali che $\mathcal{M}$, come definita nell'equazione (2.2), sia una M-matrice. In questo caso possiamo applicare con successo una classe di iterazioni funzionali per trovare una soluzione $S \geqslant 0$ dell'equazione [16].

Siano $A = A_1 - A_2$ e $E = E_1 - E_2$ due *splitting regolari* di A ed E, cioè due decomposizioni tali che A_1 ed E_1 siano M-matrici non singolari, e $A_2 \geqslant 0$, $E_2 \geqslant 0$. [2] Allora, possiamo definire la seguente iterazione funzionale

$$X_{k+1} = \nabla^{-1}_{A_1, -E_1}(X_k C X_k + B + A_2 X_k + X_k E_2). \tag{2.7}$$

[2] Si noti che, poiché $\begin{bmatrix} E_1 & 0 \\ 0 & A_1 \end{bmatrix} \geqslant \begin{bmatrix} E & -C \\ -B & A \end{bmatrix}$, per il punto 1 del teorema 1.2, se $\mathcal{M}$ è una M-matrice basta imporre che E_1 e A_1 siano Z-matrici per poter dimostrare che sono in realtà M-matrici non singolari.

Si noti che l'operatore $\nabla_{A_1,-E_1}$ è invertibile in virtù del teorema 1.8, perché A_1 ed E_1, essendo M-matrici non singolari, hanno solo autovalori con parte reale strettamente positiva. I punti fissi della (2.7) sono le soluzioni della NARE, perché

$$S = \nabla^{-1}_{A_1,-E_1}(SCS+B+A_2S+SE_2) \iff A_1S+SE_1 = SCS+B+A_2S+SE_2;$$

inoltre, con un'opportuna scelta dei valori iniziali, le iterate convergono alla soluzione, come provato dal seguente risultato.

Teorema 2.2 ([16, 14])**.** *Siano i coefficienti della* (2.1) *tali da rendere* $\mathcal{M}$ *una* M*-matrice non singolare oppure singolare irriducibile,* $A = A_1 - A_2$*,* $E = E_1 - E_2$ *splitting regolari, e* $X_0 = 0$ *il punto di partenza dell'iterazione* (2.7)*. Allora,*

1. $X_k \geqslant X_{k+1}$ *per ogni* $k = 0, 1, \ldots$*;*
2. *Se* $X_0 \geqslant S$*, l'iterazione* (2.7) *converge crescendo a* S*.*

Le iterazioni funzionali di questo tipo hanno costo $O(\max(m,n)^3)$ per passo e convergenza lineare alla soluzione nel caso non critico. Nel caso critico la convergenza risulta sublineare.

2.4 Metodo di Newton

Un metodo più efficiente delle iterazioni funzionali per la ricerca della soluzione è il metodo di Newton [16]. La derivata secondo Fréchet della funzione $\mathrm{R}(X) := XCX - AX - XE + B$ in X è la funzione $\mathrm{R}'_X : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ definita da

$$\mathrm{R}'_X(Z) = -(A - XC)Z - Z(E - CX).$$

Il metodo di Newton applicato all'equazione (2.1) è allora

$$X^{(k+1)} = X^{(k)} - (\mathrm{R}'_{X^{(k)}})^{-1}\,\mathrm{R}(X^{(k)}), \quad k = 0, 1, \ldots$$

ed è ben definito se $\mathrm{R}'_{X^{(k)}}$ è invertibile per ogni k. Utilizzando le definizioni delle mappe R e R'_X, possiamo riscrivere più esplicitamente la relazione precedente come l'equazione di Sylvester in $X^{(k+1)}$

$$\begin{aligned}(A - X^{(k)}C)(X^{(k+1)} - X^{(k)}) + (X^{(k+1)} - X^{(k)})(E - CX^{(k)}) \\ = X^{(k)}CX^{(k)} - AX^{(k)} - X^{(k)}E + B, \quad (2.8)\end{aligned}$$

ovvero, svolgendo i calcoli,

$$X^{(k+1)} = \nabla^{-1}_{A-X^{(k)}C,-(E-CX^{(k)})}(B - X^{(k)}CX^{(k)}). \tag{2.9}$$

Il seguente risultato prova che nel caso di nostro interesse il metodo di Newton è ben definito e convergente:

Teorema 2.3. *([16, 14]) Siano i coefficienti della* (2.1) *tali da rendere* $\mathcal{M}$ *una* M*-matrice non singolare oppure singolare irriducibile, e consideriamo il metodo di Newton con* $X^{(0)} = 0$*. Allora,*

1. *La matrice*

$$M_{X^{(k)}} := I \otimes (A - X^{(k)}C) + (E - CX^{(k)})^T \otimes I,$$

detta con lieve abuso di notazione matrice Jacobiana*, è una* M*-matrice non singolare per ogni* $k \geqslant 0$*; quindi in particolare il metodo è sempre ben definito;*

2. *la successione* $X^{(k)}$ *converge crescendo alla soluzione minimale non negativa della NARE.*

Il metodo di Newton, come è noto dai risultati classici, ha convergenza quadratica nel caso in cui la derivata della funzione calcolata nella soluzione minimale S, cioè R'_S, sia una matrice non singolare. Utilizzando il lemma 1.8, è possibile dimostrare che questa ipotesi è verificata nel caso non critico. Nel caso critico, il metodo di Newton ha convergenza lineare [16]. Il metodo di Newton ha costo $O(\max(m,n)^3)$ per passo; tuttavia, la costante nascosta nella notazione $O(\cdot)$ è particolarmente onerosa; per esempio, la versione con matrici quadrate ($m = n$) implementata utilizzando il metodo di Bartels–Stewart per la soluzione dell'equazione di Sylvester richiede $62n^3$ operazioni per passo.

2.5 Riduzione a un'equazione unilaterale e riduzione ciclica

Ramaswami [27] ha notato che S è la soluzione minimale non negativa della (2.1) se e solo se la matrice

$$\mathcal{S} = \begin{bmatrix} I - tE + tCS & 0 \\ S & 0 \end{bmatrix}$$

è la soluzione minimale non negativa dell'equazione matriciale

$$\mathcal{A}_1 Y^2 + \mathcal{A}_0 Y + \mathcal{A}_{-1} = 0 \tag{2.10}$$

con

$$\mathcal{A}_{-1} = \begin{bmatrix} I - tE & 0 \\ tB & 0 \end{bmatrix}, \quad \mathcal{A}_0 = \begin{bmatrix} -I & tC \\ 0 & -I - tA \end{bmatrix}, \quad \mathcal{A}_1 = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}, \tag{2.11}$$

e $t \in \mathbb{R}$ è tale che $1/t \geq \max\{e_{i,i}, a_{i,i} : \quad i = 1, \ldots, n\}$. Per questa classe di equazioni sono noti in letteratura svariati algoritmi risolutivi (si veda per esempio il libro [6]). Richiamiamo brevemente il più classico, noto come *riduzione ciclica* (CR) [6, 3]. Siano $\mathcal{A}_i^{(0)} = \mathcal{A}_i$, $i = -1, 0, 1$ e $\widehat{\mathcal{A}}^{(0)} = \mathcal{A}_0$, e definiamo le successioni

$$\begin{aligned} \mathcal{A}_0^{(k+1)} &= \mathcal{A}_0^{(k)} - \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \mathcal{A}_1^{(k)} - \mathcal{A}_1^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)}, \quad \mathcal{K}^{(k)} = \left(\mathcal{A}_0^{(k)}\right)^{-1}, \\ \mathcal{A}_{-1}^{(k+1)} &= -\mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)}, \\ \mathcal{A}_1^{(k+1)} &= -\mathcal{A}_1^{(k)} \mathcal{K}^{(k)} \mathcal{A}_1^{(k)}, \\ \widehat{\mathcal{A}}_0^{(k+1)} &= \widehat{\mathcal{A}}_0^{(k)} - \mathcal{A}_1^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)}. \end{aligned} \tag{2.12}$$

Vale il seguente risultato di convergenza [27, 4].

Teorema 2.4. *Consideriamo la riduzione ciclica* (2.12) *applicata all'equazione unilaterale* (2.10) *con i coefficienti come in* (2.11) *derivanti da un'equazione di Riccati algebrica con $\mathcal{M}$ M-matrice non singolare oppure singolare irriducibile. Allora,*

1. *le successioni $\mathcal{A}_{-1}^{(k)}$ e $\mathcal{A}_1^{(k)}$ sono convergenti; nel caso non critico, almeno una di esse converge quadraticamente a zero;*

2. *la successione*
$$\mathcal{S}^{(k)} = -\left(\widehat{\mathcal{A}}^{(k)}\right)^{-1} \mathcal{A}_{-1}$$
converge a $\mathcal{S}$; in particolare, il blocco $(2,1)$ converge alla soluzione minimale non negativa S dell'equazione di Riccati. La convergenza è quadratica nel caso non critico, e lineare nel caso critico.

La riduzione ciclica utilizzata per trovare la soluzione dell'equazione di Riccati ha costo $O(\max(m,n)^3)$ per passo. La costante moltiplicativa nascosta nella notazione $O(\cdot)$ è circa metà di quella del metodo di Newton; nel caso $m = n$, ad esempio, il costo è $\frac{68}{3}n^3$ operazioni per passo.

Un altro modo di esprimere la riduzione ciclica è la formulazione funzionale dell'algoritmo fornita in [6]. Sia $\varphi^{(k)}(z) = z\mathcal{A}_1^{(k)} + \mathcal{A}_0^{(k)} + z^{-1}\mathcal{A}_{-1}^{(k)}$ e

$\psi^{(k)}(z) = \varphi^{(k)}(z)^{-1}$, dove $z \in \mathbb{C}$ e $\psi^{(k)}(z)$ è definita per tutti i valori di z che non annullano $\det \varphi^{(i)}(z)$. Si può allora facilmente verificare la seguente equazione [6].

$$\psi^{(i+1)}(z^2) = \frac{1}{2}(\psi^{(i)}(z) + \psi^{(i)}(-z)). \tag{2.13}$$

2.6 Structured Doubling Algorithm

Lo *structured doubling algorithm* (SDA), introdotto in [17], è un metodo di tipo diverso da quelli elencati finora per il calcolo della soluzione minimale non negativa di un'equazione di Riccati. L'algoritmo può essere enunciato tramite la seguente iterazione

$$\begin{aligned} E_{k+1} &= E_k(I - G_kH_k)^{-1}E_k, \\ F_{k+1} &= F_k(I - H_kG_k)^{-1}F_k, \\ G_{k+1} &= G_k + E_k(I - G_kH_k)^{-1}G_kF_k, \\ H_{k+1} &= H_k + F_k(I - H_kG_k)^{-1}H_kE_k, \end{aligned} \tag{2.14}$$

a partire dai valori iniziali

$$\begin{aligned} E_0 &= I - 2\gamma V^{-1}, \\ F_0 &= I - 2\gamma W^{-1}, \\ G_0 &= 2\gamma(E + \gamma I)^{-1}CW^{-1}, \\ H_0 &= 2\gamma W^{-1}B(E + \gamma I)^{-1}, \end{aligned} \tag{2.15}$$

con

$$W = A + \gamma I - B(E + \gamma I)^{-1}C, \quad V = E + \gamma I - C(A + \gamma I)^{-1}B.$$

Si hanno i seguenti risultati di convergenza.

Teorema 2.5 ([17, 15])**.** *Siano i coefficienti della* (2.1) *tali da rendere* $\mathcal{M}$ *una M-matrice, e* $\gamma \geqslant \max(\mathrm{diag}(A)), \max(\mathrm{diag}(E))$*. Allora, nel caso non critico,*

1. *Ad ogni passo* k*,* $I - G_kH_k$ *e* $I - H_kG_k$ *sono M-matrici non singolari. Quindi in particolare l'iterazione SDA è ben definita ad ogni passo;*

2. *Le due successioni* $\{E_k\}$ *e* $\{F_k\}$ *sono limitate, e almeno una di esse converge quadraticamente a zero;*

3. *La successione* $\{H_k\}$ *converge quadraticamente alla soluzione minimale non negativa* S *della* (2.1)*, e la successione* $\{G_k\}$ *converge quadraticamente alla soluzione minimale non negativa* T *dell'equazione* (2.4)*.*

Nel caso critico, l'iterazione è ben definita, $\{E_k\}$ e $\{F_k\}$ sono limitate, e $\{H_k\}$ e $\{G_k\}$ convergono rispettivamente a S e T, ma non necessariamente quadraticamente.

L'iterazione stabilita dallo SDA ha costo $O(\max(n,m)^3)$ per passo. Per $m = n$, il costo è $\frac{64}{3}n^3$ operazioni per passo.

Una semplice interpretazione dello SDA segue facendo alcune modifiche all'approccio presentato in [17] e nei lavori in esso citati.

Teorema 2.6. *Sia*

$$\mathcal{H}_\gamma := (\mathcal{H} + \gamma I)^{-1}(\mathcal{H} - \gamma I),$$

con $\mathcal{H}$ come definita in (2.2)*, e*

$$\mathcal{L}_k := \begin{bmatrix} E_k & 0 \\ -H_k & I \end{bmatrix}, \quad \mathcal{U}_k = \begin{bmatrix} I & -G_k \\ 0 & F_k \end{bmatrix}.$$

Allora, ad ogni passo k dello SDA vale la seguente relazione.

$$\mathcal{H}_\gamma^{2^k} = \mathcal{U}_k^{-1}\mathcal{L}_k.$$

Dimostrazione. Dimostriamo la tesi per induzione su k. Al passo $k = 0$, valgono le seguenti fattorizzazioni, detti $E_\gamma := E + \gamma I$ e $A_\gamma := A + \gamma I$.

$$\mathcal{H} + \gamma I = \begin{bmatrix} E_\gamma & 0 \\ B & I \end{bmatrix} \begin{bmatrix} I & -E_\gamma^{-1}C \\ 0 & -W + 2\gamma I \end{bmatrix},$$
$$\mathcal{H} - \gamma I = \begin{bmatrix} I & -C \\ 0 & -A_\gamma \end{bmatrix} \begin{bmatrix} V - 2\gamma I & 0 \\ -A_\gamma^{-1}B & I \end{bmatrix}.$$

Inoltre, come si può verificare svolgendo esplicitamente i prodotti e utilizzando la formula SMW per il calcolo di V^{-1}, vale

$$\begin{bmatrix} E_\gamma & 0 \\ B & I \end{bmatrix}^{-1} \begin{bmatrix} I & -C \\ 0 & -A_\gamma \end{bmatrix}$$
$$= \begin{bmatrix} E_\gamma^{-1} & -E_\gamma^{-1}C \\ -BE_\gamma^{-1} & -W \end{bmatrix} = \begin{bmatrix} I & -E_\gamma^{-1}CW^{-1} \\ 0 & -W^{-1} \end{bmatrix}^{-1} \begin{bmatrix} V^{-1} & 0 \\ W^{-1}BE_\gamma^{-1} & I \end{bmatrix}.$$

Possiamo allora scrivere

$$\begin{aligned}
\mathcal{H}_\gamma &= (\mathcal{H} + \gamma I)^{-1}(\mathcal{H} - \gamma I) \\
&= \begin{bmatrix} I & -E_\gamma^{-1}C \\ 0 & -W + 2\gamma I \end{bmatrix}^{-1} \begin{bmatrix} E_\gamma & 0 \\ B & I \end{bmatrix}^{-1} \begin{bmatrix} I & -C \\ 0 & -A_\gamma \end{bmatrix} \begin{bmatrix} V - 2\gamma I & 0 \\ -A_\gamma^{-1}B & I \end{bmatrix} \\
&= \begin{bmatrix} I & -E_\gamma^{-1}C \\ 0 & -W + 2\gamma I \end{bmatrix}^{-1} \begin{bmatrix} I & -E_\gamma^{-1}CW^{-1} \\ 0 & -W^{-1} \end{bmatrix}^{-1} \begin{bmatrix} V^{-1} & 0 \\ W^{-1}BE_\gamma^{-1} & I \end{bmatrix} \begin{bmatrix} V - 2\gamma I & 0 \\ -A_\gamma^{-1}B & I \end{bmatrix} \\
&= \begin{bmatrix} I & -2\gamma E_\gamma^{-1}CW^{-1} \\ 0 & I - 2\gamma W^{-1} \end{bmatrix}^{-1} \begin{bmatrix} I - 2\gamma V^{-1} & 0 \\ -2\gamma W^{-1}BE_\gamma^{-1} & I \end{bmatrix}.
\end{aligned}$$

che era quanto richiesto dal passo base dell'induzione, in accordo con la (2.15). Ora, supponendo la tesi valida per tutti gli $h \leqslant k$, possiamo calcolare esplicitamente

$$
\begin{aligned}
\mathcal{H}_\gamma^{2^{k+1}} &= \mathcal{H}_\gamma^{2^k}\mathcal{H}_\gamma^{2^k} = \begin{bmatrix} I & -G_k \\ 0 & F_k \end{bmatrix}^{-1} \begin{bmatrix} E_k & 0 \\ -H_k & I \end{bmatrix} \begin{bmatrix} I & -G_k \\ 0 & F_k \end{bmatrix}^{-1} \begin{bmatrix} E_k & 0 \\ -H_k & I \end{bmatrix} \\
&= \begin{bmatrix} I & -G_k \\ 0 & F_k \end{bmatrix}^{-1} \begin{bmatrix} I & -E_kG_k(I-H_kG_k)^{-1} \\ 0 & F_k(I-H_kG_k)^{-1} \end{bmatrix}^{-1} \begin{bmatrix} E_k + G_k(I-H_kG_k)^{-1}H_k & 0 \\ -F_k(I-H_kG_k)^{-1}H_k & I \end{bmatrix} \begin{bmatrix} E_k & 0 \\ -H_k & I \end{bmatrix} \\
&= \begin{bmatrix} I & -(G_k + E_kG_k(I-H_kG_k)^{-1})F_k \\ 0 & F_k(I-H_kG_k)^{-1}F_k \end{bmatrix}^{-1} \begin{bmatrix} E_k^2 + E_kG_k(I-H_kG_k)^{-1}H_kE_k & 0 \\ -(H_k + F_k(I-H_kG_k)^{-1}H_kE_k) & I \end{bmatrix} \\
&= \begin{bmatrix} I & -(G_k + E_k(I-G_kH_k)^{-1})G_kF_k \\ 0 & F_k(I-H_kG_k)^{-1}F_k \end{bmatrix}^{-1} \begin{bmatrix} E_k(I-G_kH_k)^{-1}E_k & 0 \\ -(H_k + F_k(I-H_kG_k)^{-1}H_kE_k) & I \end{bmatrix} \\
&= \begin{bmatrix} I & -G_{k+1} \\ 0 & F_{k+1} \end{bmatrix}^{-1} \begin{bmatrix} E_{k+1} & 0 \\ -H_{k+1} & I \end{bmatrix}.
\end{aligned}
$$

quindi anche il passo induttivo è completato. □

In questa chiave di lettura, il metodo SDA appare molto simile al metodo QR per il calcolo degli autovalori e autovettori di una matrice (per un'introduzione al metodo QR, si veda ad esempio [11]). Ad ogni passo, infatti, partiamo da una fattorizzazione UL (a blocchi) della matrice $\mathcal{H}_\gamma^{2^k}$, e andiamo a calcolare la fattorizzazione UL del "prodotto scambiato" $\widehat{\mathcal{U}}_k^{-1}\widehat{\mathcal{L}}_k = \mathcal{L}_k\mathcal{U}_k^{-1}$ per ottenere le nuove matrici $\mathcal{L}_{k+1} = \widehat{\mathcal{L}}_k\mathcal{L}_k$ e $\mathcal{U}_{k+1} = \widehat{\mathcal{U}}_k\mathcal{U}_k$.

Inoltre, la trasformazione preliminare $\mathcal{H} \mapsto (\mathcal{H} + \gamma I)^{-1}(\mathcal{H} - \gamma I)$ lascia invariati i sottospazi invarianti (da cui dipendono le soluzioni della NARE), mentre trasforma gli autovalori secondo la relazione $\lambda \mapsto (\lambda + \gamma)^{-1}(\lambda - \gamma)$. Poiché gli autovalori inizialmente erano n nel semipiano positivo e n nel semipiano negativo (teorema 2.1), dopo la trasformazione le loro immagini sono n all'interno del cerchio unitario e n al di fuori. Durante lo SDA, ad ogni passo la matrice E_k va a tener conto degli autovalori di $\mathcal{H}_\gamma^{2^k}$ che stanno all'interno del cerchio unitario, mentre la matrice F_k^{-1} di quelli al di fuori. Allora, sia E_k che F_k ad ogni passo hanno solo autovalori di modulo minore o uguale a 1, e quindi restano limitate. L'efficacia dello SDA deriva dal fatto che riesce a calcolare in modo implicito la fattorizzazione UL di una matrice con alcuni autovalori molto grandi e altri molto piccoli (in modulo) senza mai dover lavorare esplicitamente con numeri grandi o invertire matrici mal condizionate, e quindi senza perdere di precisione.

2.7 Tecnica di shift

Nel caso critico, come abbiamo visto, gli algoritmi fin qui introdotti hanno proprietà di convergenza nettamente peggiori. Pertanto, è stato introdotto in letteratura [18, 6, 15] un metodo per trasformare una NARE di tipo nullo ricorrente in una di tipo non nullo ricorrente, noto come *tecnica di shift*.

Supponiamo di essere nel caso critico, e sia v l'autovettore destro relativo all'autovalore zero di $\mathcal{M}$, come nella (2.6), e siano $\eta \in \mathbb{R}$ e $p \in \mathbb{R}^{2n}$; consideriamo la matrice

$$\widetilde{\mathcal{H}} := \begin{bmatrix} \widetilde{E} & -\widetilde{C} \\ \widetilde{B} & -\widetilde{A} \end{bmatrix} = H + \eta v p^T; \tag{2.16}$$

Essa definisce implicitamente una nuova equazione di Riccati

$$X\widetilde{C}X - \widetilde{A}X - X\widetilde{E} + \widetilde{B} = 0 \tag{2.17}$$

Valgono i seguenti risultati.

Teorema 2.7 ([15])**.** *Sia $XCX - AX - XE + B = 0$ una NARE nulla ricorrente, con i coefficienti tali che $\mathcal{M}$ sia una M-matrice non singolare oppure singolare irriducibile, e sia* (2.17) *l'equazione di Riccati definita dai coefficienti definiti dalla* (2.16)

1. *L'equazione* (2.17) *ha la stessa soluzione minimale non negativa S della* (2.1)*;*

2. *La matrice hamiltoniana $\widetilde{\mathcal{H}}$ associata alla* (2.17) *ha un autovalore nullo di molteplicità algebrica 1; in particolare, questo implica che l'equazione* (2.17) *è non critica;*

Poiché la (2.17) è non critica, possiamo applicare ad essa i metodi usuali per il calcolo della soluzione, recuperando le proprietà di convergenza che venivano a cadere per la (2.1) nel caso nullo ricorrente, ed ottenendo così la soluzione al problema originale in modo più efficiente. Il calcolo del vettore v, necessario per applicare lo shift, equivale a trovare il nucleo della matrice $\mathcal{M}$ (che sappiamo essere 1-dimensionale, ad esempio per il teorema di Perron–Frobenius), quindi può essere effettuato nel caso generale in tempo $O(n^3)$. Si noti tuttavia che la matrice $\widetilde{\mathcal{M}}$ associata alla (2.17) in generale *non* è una M-matrice, quindi l'applicabilità dei metodi sopra enunciati non è sempre assicurata.

Capitolo 3

Specializzazione degli algoritmi esistenti al problema

3.1 L'equazione del trasporto *one-group* per neutroni

Il problema che intendiamo trattare in questo lavoro di tesi nasce da un problema fisico: siamo interessati a modellizzare il comportamento di un flusso di neutroni all'interno di un materiale schermante omogeneo, come una lastra di piombo. Tale problema sorge per esempio nella progettazione di metodi per schermare efficacemente le emissioni di sostanze radioattive. L'equazione da cui deriva, una variazione dell'equazione del trasporto per neutroni nell'approssimazione *one-group* [21], nasce dalla modellizzazione matematica di questo problema e assume la forma

$$\left\{(\mu+\alpha)\frac{\partial}{\partial x}+1\right\}\varphi(x,\mu)=\frac{c}{2}\int_{-1}^{1}\varphi(x,\omega)d\omega, \tag{3.1}$$

$$\varphi(0,\mu)=f(\mu),\ \mu>-\alpha,\ |\mu|\leqslant 1, \tag{3.2}$$

$$\lim_{x\to\infty}\varphi(x,\mu)=0, \tag{3.3}$$

dove $\varphi(x,\mu)$ è il flusso di neutroni uscente dalla posizione x e diretto secondo l'angolo μ; v è la velocità dei neutroni (che supponiamo costante); c ($0\leqslant c\leqslant 1$) è il numero medio di neutroni che emergono da una collisione (cioè, $1-p$, dove p è la probabilità che il neutrone venga assorbito da un atomo del materiale); α (*angular shift*, $0\leqslant\alpha<1$) è un parametro aggiuntivo derivante da una modifica del modello introdotta da Coron [7] e Ganapol [8]. Quest'ultimo parametro non ha un immediata interpretazione fisica, ma

ha il ruolo di semplificare la trattazione del problema e di velocizzare alcuni degli algoritmi risolutivi.

Attraverso una tecnica risolutiva nota come *invariant embedding*, è possibile ridurre il problema alla seguente equazione integrale [20]:

$$\left(\frac{1}{\mu+\alpha}+\frac{1}{\nu-\alpha}\right)X(\mu,\nu) = c\left(1+\frac{1}{2}\int_{-\alpha}^{1}\frac{X(\omega,\nu)}{\omega+\alpha}d\omega\right)\left(1+\frac{1}{2}\int_{\alpha}^{1}\frac{X(\mu,\omega)}{\omega-\alpha}d\omega\right). \quad (3.4)$$

Possiamo risolvere numericamente questa equazione discretizzando i due integrali nel membro di destra. Siano $\{w_i\}$, $\{x_i\}$ i pesi e i nodi di una quadratura gaussiana a n punti su $[0,1]$, cioè

$$\int_0^1 f(x)dx \approx \sum_{i=1}^{n} w_i f(x_i),$$

con $0 < x_n < x_{n-1} < \cdots < x_1 < 1$, $\sum_{i=1}^{n} w_i = 1$; trasformiamo questi parametri in pesi e nodi sugli intervalli $[-\alpha,1]$ e $[\alpha,1]$ per ottenere

$$\begin{aligned} x_k^- &= (1+\alpha)x_k - \alpha, w_k^- = w_k(1+\alpha) && k=1,\ldots,n,\\ x_k^+ &= (1-\alpha)x_k + \alpha, w_k^- = w_k(1-\alpha) && k=1,\ldots,n. \end{aligned}$$

L'equazione (3.4), valutata in $(\mu,\nu) = (x_i^-, x_j^+)$, $\forall i,j \in \{1,\ldots,n\}$ e discretizzata con l'approsimazione introdotta, diventa allora

$$\left(\frac{1}{x_i^-+\alpha}+\frac{1}{x_j^+-\alpha}\right)X(x_i^-,x_j^+) = c\left(1+\frac{1}{2}\sum_{k=1}^{n} w_k^- \frac{X(x_k^-,x_j^+)}{x_k^-+\alpha}\right)\left(1+\frac{1}{2}\sum_{k=1}^{n} w_k^+ \frac{X(x_i^-,x_k^+)}{x_k^+-\alpha}\right).$$

Riscrivendo in forma matriciale, otteniamo

$$\Delta X + XD = (Xq+e)(e^T+q^TX), \quad (3.5)$$

o anche

$$XCX - AX - XE + B = 0 \quad (3.6)$$

ponendo $(X)_{ij} = X(x_i^-, x_j^+)$ e

$$\begin{aligned} B &= ee^T, & C &= qq^T, \\ A &= \Delta - eq^T, & E &= D - qe^T \\ D &= \operatorname{diag}(d_1, \ldots, d_n), & \Delta &= \operatorname{diag}(\delta_1, \ldots, \delta_n), \\ d_i &= \frac{1}{cx_i(1-\alpha)}, & \delta_i &= \frac{1}{cx_i(1+\alpha)}, \\ e &= \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T, & q_i &= \frac{w_i}{2x_i}. \end{aligned} \tag{3.7}$$

È possibile inoltre dimostrare (si vedano [21] e i riferimenti in esso citati) che la soluzione di interesse fisico è quella minimale non negativa.

3.2 Convergenza e applicabilità degli algoritmi generali

Le dimostrazioni di convergenza per gli algoritmi esposti nel capitolo precedente avevano come ipotesi che la matrice $\mathcal{M}$ fosse una M-matrice non singolare o singolare irriducibile. Pertanto, per assicurare l'applicabilità dei metodi esposti, dimostriamo innanzitutto che per il problema (3.5) la matrice $\mathcal{M}$ risulta avere tali proprietà, come già notato in [12].

Lemma 3.1. *Per l'equazione di Riccati* (3.5)*, la matrice $\mathcal{M}$ è una M-matrice irriducibile. La matrice è singolare solo se $c = 1$, e nulla ricorrente (caso critico) solo se $c = 1$ e $\alpha = 0$).*

Dimostrazione. Il risultato segue facilmente dal teorema 1.5: infatti, dobbiamo semplicemente verificare che

$$0 \leqslant 1 - \begin{bmatrix} e^T & q^T \end{bmatrix} \begin{bmatrix} D^{-1} & 0 \\ 0 & \Delta^{-1} \end{bmatrix} \begin{bmatrix} q \\ e \end{bmatrix},$$

che, ricordando le definizioni, equivale a

$$1 \geqslant \sum_{i=1}^{n} \frac{c(1-\alpha)}{2} c_i + \sum_{i=1}^{n} \frac{c(1+\alpha)}{2} w_i = c.$$

L'uguaglianza, che per il lemma 1.4 equivale alla singolarità della matrice, si ha se e solo se $c = 1$. Quando $\mathcal{M}$ è singolare, è semplice verificare che i suoi vettori di Perron destro e sinistro sono

$$u^T = \begin{bmatrix} u_1^T & u_2^T \end{bmatrix} = \begin{bmatrix} e^T & q^T \end{bmatrix} \begin{bmatrix} D^{-1} & 0 \\ 0 & \Delta^{-1} \end{bmatrix}, \quad v = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} D^{-1} & 0 \\ 0 & \Delta^{-1} \end{bmatrix} \begin{bmatrix} q \\ e \end{bmatrix},$$

e quindi il *drift* è

$$\sum_{i=1}^{n} \frac{c^2 x_i (1+\alpha)^2}{2} w_i - \sum_{i=1}^{n} \frac{c^2 x_i (1-\alpha)^2}{2} w_i,$$

che si annulla se e solo se $\alpha = 0$. □

3.3 Equazioni di Riccati con $\mathcal{M}$ diagonale più rango 1

Tratteremo nel seguito direttamente il caso leggermente più generale in cui la matrice $\mathcal{M}$ è una M-matrice diagonale più rango 1. Introduciamo allo scopo una notazione il più simile possibile a quella della sezione 3.1. Siano d, δ, e, q, $\widetilde{e}$, $\widetilde{q}$ vettori positivi tali che

$$\begin{bmatrix} e^T & q^T \end{bmatrix} \begin{bmatrix} D^{-1} & 0 \\ 0 & \Delta^{-1} \end{bmatrix} \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} \leqslant 1,$$

con $D = \operatorname{diag}(d)$ e $\Delta = \operatorname{diag}(\delta)$; allora, con una dimostrazione analoga a quella del lemma 3.1, si ha che

$$\mathcal{M} = \begin{bmatrix} D & 0 \\ 0 & \Delta \end{bmatrix} - \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix}$$

è una M-matrice irriducibile. Definiamo

$$\begin{aligned} &XCX + B - XE - AX = 0, \\ &B = \widetilde{e} e^T, \qquad C = \widetilde{q} q^T, \\ &A = \Delta - \widetilde{e} q^T, \quad E = D - \widetilde{q} e^T \end{aligned} \tag{3.8}$$

l'equazione di Riccati associata a $\mathcal{M}$.

Questo caso permette di trattare sia l'equazione di Riccati del problema (3.5) che quella modificata che origina dall'applicazione della tecnica di shift.

Teorema 3.2. *Sia $\widetilde{\mathcal{M}}$ la matrice associata all'equazione di Riccati ottenuta applicando la tecnica di shift all'equazione* (3.5)*, scegliendo $p^T = \begin{bmatrix} e^T & q^T \end{bmatrix}$ e $0 \leqslant \eta \leqslant d_1$. Allora, $\widetilde{\mathcal{M}}$ è una M-matrice diagonale più rango 1.*

Dimostrazione. Dalla (2.16) otteniamo

$$\begin{aligned}\widetilde{\mathcal{H}} =&\mathcal{H} + \eta \begin{bmatrix} D^{-1}q \\ \Delta e \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix} \\ =& \begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix} - \begin{bmatrix} q \\ -e \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix} + \eta \begin{bmatrix} D^{-1}q \\ \Delta^{-1}e \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix} \\ =& \begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix} - \begin{bmatrix} (I-\eta D^{-1})q \\ -(I+\eta\Delta^{-1})e \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix},\end{aligned}$$

e

$$\widetilde{\mathcal{M}} = \mathcal{J}\widetilde{\mathcal{H}} = \begin{bmatrix} D & 0 \\ 0 & \Delta \end{bmatrix} - \begin{bmatrix} (I-\eta D^{-1})q \\ (I+\eta\Delta^{-1})e \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix}$$

quindi $\widetilde{\mathcal{M}}$ e $\widetilde{\mathcal{H}}$ sono diagonali più rango 1. Inoltre, ricordando che nel caso critico $u^T\mathcal{J}v = 0$, abbiamo

$$u^T\widetilde{\mathcal{M}} = u^T\mathcal{M} + \eta u^T\mathcal{J}vp^T = 0,$$

da cui segue che $\widetilde{\mathcal{M}}$ è una M-matrice per il punto 2 del teorema 1.2. □

3.4 Iterazione funzionale di Lu

Il primo tentativo di costruire un algoritmo specifico per il problema del trasporto di neutroni è dovuto a Lin-Zhang Lu [24, 23]. Esponiamo tale algoritmo nel caso dell'equazione (3.8), leggermente più generale di quello trattato da Lu. Siano

$$u = X\widetilde{q} + \widetilde{e}, \quad v = X^Tq + e. \tag{3.9}$$

Possiamo riscrivere la (3.8) come

$$\nabla_{D,-\Delta} X = uv^T; \tag{3.10}$$

sostituendo l'espressione di X data dalla (3.10) nella (3.9), otteniamo il seguente sistema di equazioni non lineari che determinano u e v.

$$\begin{cases} u = \nabla^{-1}_{D,-\Delta}(uv^T)\widetilde{q} + \widetilde{e}, \\ v = \left(\nabla^{-1}_{D,-\Delta}(uv^T)\right)^T q + e. \end{cases} \tag{3.11}$$

Possiamo risolvere questo sistema di equazioni non lineari o impostando un'iterazione funzionale del tipo

$$\begin{cases} u^{(k+1)} = \nabla^{-1}_{D,-\Delta}(u^{(k)}v^{(k)T})\widetilde{q} + \widetilde{e}, \\ v^{(k+1)} = \left(\nabla^{-1}_{D,-\Delta}(u^{(k)}v^{(k)T})\right)^T q + e, \end{cases} \tag{3.12}$$

oppure applicando il metodo di Newton per trovare uno zero della funzione

$$f\left(\begin{bmatrix} u \\ v \end{bmatrix}\right) = \begin{bmatrix} \nabla_{D,-\Delta}^{-1}(u^{(k)}v^{(k)T})\widetilde{q} + \widetilde{e} \\ \left(\nabla_{D,-\Delta}^{-1}(u^{(k)}v^{(k)T})\right)^T q + e \end{bmatrix}. \tag{3.13}$$

Calcolando esplicitamente lo Jacobiano di f, il metodo di Newton può essere convenientemente espresso come l'iterazione $\{\widehat{u}^{(k)}\}$, $\{\widehat{v}^{(k)}\}$, dove

$$\begin{bmatrix} \widehat{u}^{(k+1)} \\ \widehat{v}^{(k+1)} \end{bmatrix} = (\widehat{\mathcal{R}}^{(k)})^{-1} \begin{bmatrix} \widetilde{e} - \widehat{H}^{(k)}\widehat{v}^{(k)} \\ e - \widehat{K}^{(k)}\widehat{u}^{(k)} \end{bmatrix}, \tag{3.14}$$

e le matrici $\widehat{\mathcal{R}}^{(k)}$, $\widehat{H}^{(k)}$ e $\widehat{K}^{(k)}$ sono definite come

$$\widehat{\mathcal{R}}^{(k)} := I_{2n} - \begin{bmatrix} \widehat{G}^{(k)} & \widehat{H}^{(k)} \\ \widehat{K}^{(k)} & \widehat{L}^{(k)} \end{bmatrix}, \tag{3.15}$$

$$\begin{aligned} \widehat{G}^{(k)} &:= \operatorname{diag}(\widehat{g}_i^{(k)}), & \widehat{g}_i^{(k)} &= \textstyle\sum_{l=1}^{n} \frac{\widehat{v}_l^{(k)}\widetilde{q}_l}{d_l+\delta_i}, \\ \widehat{H}^{(k)} &:= (\widehat{h}_{ij}^{(k)}), & \widehat{h}_{ij}^{(k)} &= \tfrac{\widehat{u}_i^{(k)}\widetilde{q}_j}{d_j+\delta_i}, \\ \widehat{K}^{(k)} &:= (\widehat{\kappa}_{ij}^{(k)}), & \widehat{\kappa}_{ij}^{(k)} &= \tfrac{\widehat{v}_i^{(k)}q_j}{d_i+\delta_j}, \\ \widehat{L}^{(k)} &:= \operatorname{diag}(\widehat{l}_i^{(k)}), & \widehat{l}_i^{(k)} &= \textstyle\sum_{l=1}^{n} \frac{\widehat{u}_l^{(k)}q_l}{d_i+\delta_l}. \end{aligned} \tag{3.16}$$

Cioè $\widehat{G}^{(k)}$ e $\widehat{L}^{(k)}$ sono diagonali, e $\widehat{H}^{(k)}$ e $\widehat{K}^{(k)}$ sono Cauchy-like, con equazioni di displacement

$$\Delta\widehat{H}^{(k)} + \widehat{H}^{(k)}D = u^{(k)}\widetilde{q}^T, \quad D\widehat{K}^{(k)} + \widehat{K}^{(k)}\Delta = v^{(k)}q^T.$$

Valgono le seguenti proprietà di convergenza.

Teorema 3.3 ([24, 23])**.** *Per l'equazione* (3.8)*,*

1. *Le sequenze generate dall'iterazione funzionale* (3.12)*, a partire da* $u^{(0)} = v^{(0)} = 0$*, convergono crescendo a* u *e* v *come definiti in* (3.9)*. La convergenza è lineare nel caso non critico.*

2. *Le sequenze generate dal metodo di Newton* (3.14)*, a partire da* $\widehat{u}^{(-1)} = \widehat{v}^{(-1)} = 0$*, convergono crescendo a* u *e* v*. La convergenza è quadratica nel caso non critico.*

L'iterazione (3.12) si può implementare facilmente con costo $O(n^2)$ per passo. Anche il metodo di Newton (3.14) può essere implementato in tempo $O(n^2)$ per passo, facendo uso dell'eliminazione di Gauss veloce per matrici Trummer-like.

Teorema 3.4. *([5]) L'iterazione* (3.14) *può essere implementata in costo* $O(n^2)$ *per passo.*

Dimostrazione. Il calcolo esplicito dei coefficienti di $\widehat{G}^{(k)}$, $\widehat{H}^{(k)}$, $\widehat{K}^{(k)}$, $\widehat{L}^{(k)}$ può essere effettuato in tempo $O(n^2)$ attraverso le formule (3.15). Il calcolo del termine noto del sistema (3.14) può essere effettuato in tempo $O(n^2)$ in quanto richiede solamente due prodotti matrice–vettore (con $\widehat{H}^{(k)}$ e $\widehat{K}^{(k)}$ rispettivamente) e un'addizione tra vettori.

Ora, la matrice $\widehat{\mathcal{R}}^{(k)}$ è Trummer-like con rango di displacement 2: difatti, si ha

$$\begin{bmatrix} \Delta & 0 \\ 0 & -D \end{bmatrix} \mathcal{R} - \mathcal{R} \begin{bmatrix} \Delta & 0 \\ 0 & -D \end{bmatrix} = \begin{bmatrix} 0 & -u^{(k)}\widetilde{q}^T \\ v^{(k)}q^T & 0 \end{bmatrix},$$

e il termine di destra ha chiaramente rango 2. Possiamo allora utilizzare l'algoritmo 1.2 per risolvere il sistema lineare che compare nella (3.14) in tempo $O(n^2)$ invece che $O(n^3)$. □

Si noti che l'algoritmo può essere ulteriormente velocizzato sfruttando il fatto che $\widehat{\mathcal{R}}^{(k)}$ ha due blocchi diagonali (e quindi semplici da invertire): attraverso un'eliminazione di Gauss a blocchi possiamo portare il sistema nella forma

$$\begin{bmatrix} I - \widehat{G}^{(k)} & -\widehat{H}^{(k)} \\ 0 & I - \widehat{L}^{(k)} - \widehat{K}^{(k)}(I - \widehat{G}^{(k)})^{-1}\widehat{H}^{(k)} \end{bmatrix} \begin{bmatrix} \widehat{u}^{(k+1)} \\ \widehat{v}^{(k+1)} \end{bmatrix} = \begin{bmatrix} \widetilde{e} - \widehat{H}^{(k)}\widehat{v}^{(k)} \\ e - \widehat{K}^{(k)}\widehat{u}^{(k)} - K^{(k)}(I - \widehat{G}^{(k)})^{-1}(\widetilde{e} - \widehat{H}^{(k)}\widehat{v}^{(k)}) \end{bmatrix},$$

Ora, la matrice $\widehat{S}^{(k)}$ che compare nel blocco $(2,2)$ è il complemento di Schur di $I - \widehat{G}^{(k)}$ in $\widehat{\mathcal{R}}^{(k)}$ e quindi, come già notato nella sezione 1.3.3, ha anch'esso struttura displacement. Difatti, si ha

$$\begin{aligned} \nabla_{D,D}\,\widehat{S}^{(k)} &= \nabla_{D,D}\left(I - \widehat{L}^{(k)} - \widehat{K}^{(k)}(I - \widehat{G}^{(k)})^{-1}\widehat{H}^{(k)}\right) \\ &= -\nabla_{D,-\Delta}\left(\widehat{K}^{(k)}\right)(I - \widehat{G}^{(k)})^{-1}\widehat{H}^{(k)} \\ &\quad - \widehat{K}^{(k)}(I - \widehat{G}^{(k)})^{-1}\nabla_{-\Delta,D}\left(\widehat{H}^{(k)}\right) \\ &= -u\widetilde{q}^T(I - \widehat{G}^{(k)})^{-1}\widehat{H}^{(k)} + \widehat{K}^{(k)}(I - \widehat{G}^{(k)})^{-1}vq^T, \end{aligned}$$

quindi $\widehat{S}^{(k)}$ ha rango di displacement 2 rispetto a (D, D).

3.5 Metodo di Newton strutturato

Per il problema (3.8), anche il metodo di Newton può essere implementato in sole $O(n^2)$ operazioni per passo, come provato in [5]. Abbiamo visto che il metodo di Newton richiede a ogni passo la soluzione dell'equazione (2.9), cioè, nella notazione con il prodotto di Kronecker,

$$\operatorname{vec} X^{(k+1)} = \left(I \otimes (A - X^{(k)}C) + (E - CX^{(k)})^T \otimes I\right)^{-1} \operatorname{vec}(B - X^{(k)}CX^{(k)}).$$

La matrice da invertire (detta, con lieve abuso di notazione, *matrice Jacobiana*) assume nel caso del problema (3.8) la forma

$$M_{X^{(k)}} = D^T \otimes I_n + I_n \otimes \Delta - (e + X^{(k)T}q)\widetilde{q}^T \otimes I_n - I_n \otimes (\widetilde{e} + X^{(k)}\widetilde{q})q^T.$$

Ponendo $\mathfrak{D} := D^T \otimes I_n + I_n \otimes \Delta$, $u^{(k)} := \widetilde{e} + X^{(k)}\widetilde{q}$, $v^{(k)} := e + X^{(k)T}q$, e

$$U^{(k)} := \begin{bmatrix} v^{(k)} \otimes I_n & I_n \otimes u^{(k)} \end{bmatrix}, \quad V := \begin{bmatrix} \widetilde{q}^T \otimes I_n \\ I_n \otimes q^T \end{bmatrix}, \tag{3.17}$$

possiamo riscrivere la $M_{X^{(k)}}$ più semplicemente come

$$M_{X^{(k)}} = \mathfrak{D} - U^{(k)}V. \tag{3.18}$$

Poiché $U^{(k)} \in \mathbb{R}^{n^2 \times 2n}$ e $V \in \mathbb{R}^{2n \times n^2}$, l'inversione di $M_{X^{(k)}}$ può essere ricondotta all'inversione di una matrice $2n \times 2n$ usando la formula SMW:

$$M_{X^{(k)}}^{-1} = \mathfrak{D}^{-1} + \mathfrak{D}^{-1}U^{(k)}(I_{2n} - V\mathfrak{D}^{-1}U^{(k)})^{-1}V\mathfrak{D}^{-1}. \tag{3.19}$$

Utilizzando la formula, l'iterazione di Newton diventa allora

$$\operatorname{vec} X^{(k+1)} = \left(\mathfrak{D}^{-1} + \mathfrak{D}^{-1}U(I_{2n} - V\mathfrak{D}^{-1}U^{(k)})^{-1}V\mathfrak{D}^{-1}\right)(\widetilde{e}e^T - X^{(k)}\widetilde{q}q^TX^{(k)}). \tag{3.20}$$

Il punto delicato qui è l'inversione della matrice

$$\mathcal{R}^{(k)} = I_{2n} - V\mathfrak{D}^{-1}U^{(k)};$$

difatti, le altre operazioni necessarie per l'implementazione del metodo di Newton possono tutte essere effettuate con costo $O(n^2)$ operazioni: $\mathfrak{D}$ è una matrice diagonale di dimensione $n^2 \times n^2$, quindi il prodotto matrice–vettore con $\mathfrak{D}^{-1}$ costa $2n^2$ operazioni, e le identità

$$(v^T \otimes I_n)\operatorname{vec}(M) = Mv,$$
$$(I_n \otimes v^T)\operatorname{vec}(M) = M^Tv.$$

function X=fastnewton$(D, \Delta, e, q, \widetilde{e}, \widetilde{q})$
k=0;
$X^{(0)}$=0
do
Calcola $tmp_1 = \widetilde{e}e^T - (X^{(k)}\widetilde{q})(q^T X^{(k)})$
Calcola $tmp_2 = V\mathfrak{D}^{-1}\,\mathrm{vec}(tmp_1)$
Calcola gli elementi sulla diagonale e i generatori di displacement di $\mathcal{R}^{(k)}$ utilizzando (3.22)
Risolvi $\mathcal{R}^{(k)}tmp_3 = tmp_2$ utilizzando l'algoritmo 1.2
Calcola $\mathrm{vec}\, X^{(k+1)} = \mathfrak{D}^{-1}(\mathrm{vec}(tmp_1) + Utmp_3)$
k=k+1
while(criterio di stop)
Restituisci $X = X^{(k)}$
end function

Algoritmo 3.1: Algoritmo di Newton strutturato

(che discendono dal lemma 1.7) ci consentono di effettuare anche le altre operazioni in tempo $O(n^2)$.

Sviluppando esplicitamente i prodotti, otteniamo

$$\mathcal{R}^{(k)} = I_{2n} - \begin{bmatrix} G^{(k)} & H^{(k)} \\ K^{(k)} & L^{(k)} \end{bmatrix} \tag{3.21}$$

con

$$\begin{aligned} &G^{(k)} = \mathrm{diag}(g_i^{(k)}), && g_i^{(k)} = \textstyle\sum_{l=1}^{n} \frac{v_l^{(k)}\widetilde{q}_l}{d_l+\delta_i}, \\ &H^{(k)} = (h_{ij}^{(k)}), && h_{ij}^{(k)} = \frac{u_i^{(k)}\widetilde{q}_j}{d_j+\delta_i}, \\ &K^{(k)} = (\kappa_{ij}^{(k)}), && \kappa_{ij}^{(k)} = \frac{v_i^{(k)}q_j}{d_i+\delta_j}, \\ &L^{(k)} = \mathrm{diag}(l_i^{(k)}), && l_i^{(k)} = \textstyle\sum_{l=1}^{n} \frac{u_l^{(k)}q_l}{d_i+\delta_l}, \\ &u^{(k)} = X^{(k)}\widetilde{q} + \widetilde{e}, \\ &v^{(k)} = X^{(k)T}q + e. \end{aligned} \tag{3.22}$$

Si noti il fatto che queste equazioni sono formalmente identiche alle (3.15). Si ha quindi, analogamente alla dimostrazione del teorema 3.4, che $\mathcal{R}^{(k)}$ è Trummer-like con rango di displacement 1, e perciò la soluzione di un sistema lineare $\mathcal{R}^{(k)}x = b$ si può effettuare con $O(n^2)$ operazioni aritmetiche. Riportiamo più esplicitamente il procedimento nell'algoritmo 3.1.

3.6 Relazione tra il metodo di Lu e il metodo di Newton strutturato

Si osservi che l'iterazione del metodo di Newton (2.9) per la NARE (3.8) può essere scritta come

$$\begin{aligned}\nabla_{\Delta,-D}\, X^{(k+1)} =& \Delta X^{(k+1)} + X^{(k+1)} D \\ =& (X^{(k+1)}\widetilde{q} + \widetilde{e})(e^T + q^T X^{(k+1)}) \\ & - (X^{(k)}\widetilde{q} - X^{(k+1)}\widetilde{q})(q^T X^{(k)} - q^T X^{(k+1)})\end{aligned}$$

che ci mostra che $X^{(k+1)}$ è Cauchy-like con rango di displacement 2. Questa proprietà vale per tutte le iterate $X^{(k)}$, $k \geqslant 1$, del metodo di Newton, ottenute a partire da qualsiasi matrice iniziale $X^{(0)}$. Quindi ha senso chiedersi se i calcoli dell'algoritmo di Newton non sono in qualche senso "sovrabbondanti", visto che potremmo memorizzare e aggiornare solo i generatori displacement delle iterate anziché l'intera matrice. Ponendo infatti $u^{(k)} = X^{(k)}\widetilde{q} + \widetilde{e}$, $v^{(k)} = X^{(k)T} q + e$, avremmo ad ogni passo k

$$X^{(k)} = \nabla_{\Delta,-D}\left((u^{(k)} - u^{(k-1)})(v^{(k)} - v^{(k-1)})^T - u^{(k)} v^{(k)T}\right),$$

e quindi solo i vettori $u^{(k)}$ e $v^{(k)}$ sono rilevanti per ricostruire le iterate. Sorprendentemente, l'algoritmo generato da questa ottimizzazione non è un nuovo algoritmo, ma l'algoritmo di Newton (3.14) applicato all'iterazione di Lu.

Teorema 3.5. *[5] Siano* $\{\widehat{u}^{(k)}\}$, $\{\widehat{v}^{(k)}\}$, $k \geqslant -1$ *le iterate generate dall'algoritmo di Lu* (3.14) *per il problema* (3.8) *a partire da* $\widehat{u}^{(-1)} = v^{(-1)} = 0$, *e* $\{X^{(k)}\}$, $k \geqslant 0$ *le iterate del metodo di Newton* (2.9) *a partire da* $X^{(0)} = 0$ *per lo stesso problema. Allora, per tutti i* $k \geqslant 0$, *si ha*

$$\widehat{u}^{(k)} = X^{(k)}\widetilde{q} + \widetilde{e}, \quad \widehat{v}^{(k)} = X^{(k)T} q + e.$$

Dimostrazione. Dimostreremo il risultato per induzione su k. È semplice verificare dalle definizioni che $\widehat{\mathcal{R}}^{(-1)} = I_{2n}$, e quindi $\widehat{u}^{(0)} = \widetilde{e}$, $\widehat{v}^{(0)} = e$; quindi il passo base $k = 0$ è verificato[1].

Supponiamo ora per induzione che $\widehat{u}^{(k)} = X^{(k)}\widetilde{q} + \widetilde{e} = u^{(k)}$, $\widehat{v}^{(k)} = X^{(k)T} q + e = v^{(k)}$. Le equazioni (3.22) e (3.15) definiscono allora le stesse matrici;

[1] Notiamo anche che questo fatto ci permette di "risparmiare" un'iterazione del metodo di Lu come originariamente esposto in [23], perché possiamo usare come valore iniziale $u^{(0)} = \widetilde{e}$, $v^{(0)} = e$ anziché $\widehat{u}^{(-1)} = v^{(-1)} = 0$.

utilizzando questa equivalenza, nel resto della dimostrazione, per semplificare la notazione, tralasceremo il simbolo "hat" e l'indice (k).

Vale la relazione

$$V\mathfrak{D}^{-1}\operatorname{vec}\mathrm{R}(X) = V\mathfrak{D}^{-1}\operatorname{vec}(uv^T) - V\operatorname{vec}X = \begin{bmatrix}\widetilde{e}-(I-G)u\\ e-(I-L)v\end{bmatrix},$$

in virtù di

$$\begin{aligned}\mathfrak{D}^{-1}\operatorname{vec}(XD+\Delta X) &= \operatorname{vec}X,\\ V\mathfrak{D}^{-1}\operatorname{vec}(uv^T) &= \begin{bmatrix}Gu\\ Lv\end{bmatrix},\end{aligned}$$

che possono essere facilmente verificate dalle definizioni di $\mathfrak{D}$, G e L, dove V è la matrice definita in (3.17). Ora, partendo dall'equazione (2.8) ed esprimendo l'iterazione nella forma con il prodotto di Kronecker, otteniamo

$$\operatorname{vec}(X^{(k+1)} - X) = M_X^{-1}\operatorname{vec}\mathrm{R}(X).$$

Applicando l'operatore V a entrambi i lati dell'iterazione di Newton, ed esprimendo M_X come dalla (3.19), si ha

$$\begin{aligned}V\operatorname{vec}(X^{(k+1)} - X) =& V\mathfrak{D}^{-1}\operatorname{vec}\mathrm{R}(X)\\ &+ V\mathfrak{D}^{-1}U(I_{2n} - V\mathfrak{D}^{-1}U)^{-1}V\mathfrak{D}^{-1}\operatorname{vec}\mathrm{R}(X)\\ =&(I_{2n} + V\mathfrak{D}^{-1}U(I_{2n} - V\mathfrak{D}^{-1}U)^{-1})V\mathfrak{D}^{-1}\operatorname{vec}\mathrm{R}(X)\\ =&(I_{2n} - V\mathfrak{D}^{-1}U)^{-1}V\mathfrak{D}^{-1}\operatorname{vec}\mathrm{R}(X),\end{aligned} \tag{3.23}$$

dove l'ultima uguaglianza utilizza l'identità $I + M(I-M)^{-1} = (I-M)^{-1}$ con $M = V\mathfrak{D}^{-1}U$. Ricordiamo ora la definizione $R = (I_{2n} - V\mathfrak{D}^{-1}U)$, e l'iterazione di Lu (3.14), che assume la forma

$$\begin{bmatrix}\widehat{u}^{(k+1)}\\ \widehat{v}^{(k+1)}\end{bmatrix} = R^{-1}\begin{bmatrix}\widetilde{e}-Hu\\ e-Kv\end{bmatrix}.$$

Ora, calcoliamo esplicitamente

$$\begin{aligned}\mathcal{R}\begin{bmatrix}\widehat{u}^{k+1}-u\\ \widehat{v}^{k+1}-v\end{bmatrix} &= \begin{bmatrix}\widetilde{e}-Hu\\ e-Kv\end{bmatrix} - \left(I_{2n} - \begin{bmatrix}G & H\\ K & L\end{bmatrix}\right)\begin{bmatrix}u\\ v\end{bmatrix} =\\ &\begin{bmatrix}\widetilde{e}-Hv\\ e-Ku\end{bmatrix} - \begin{bmatrix}u\\ v\end{bmatrix} + \begin{bmatrix}Gu+Hv\\ Ku+Lv\end{bmatrix} = \begin{bmatrix}\widetilde{e}-(I-G)u\\ e-(I-L)v\end{bmatrix} = V\mathfrak{D}^{-1}\operatorname{vec}\mathrm{R}(X),\end{aligned}$$

e sostituiamo in (3.23) per ottenere

$$V \operatorname{vec}(X^{(k+1)} - X) = \begin{bmatrix} \widehat{u}^{k+1} - u \\ \widehat{v}^{k+1} - v \end{bmatrix}.$$

Infine, ricordando la definizione di V data in (3.17), troviamo

$$\begin{bmatrix} \widehat{u}^{k+1} - u \\ \widehat{v}^{k+1} - v \end{bmatrix} = V \operatorname{vec}(X^{(k+1)} - X) = \begin{bmatrix} X^{(k+1)}\widetilde{q} - X\widetilde{q} \\ X^{(k+1)T}q - X^T q \end{bmatrix} = \begin{bmatrix} X^{(k+1)}\widetilde{q} + \widetilde{e} - u \\ X^{(k+1)T}q + e - v \end{bmatrix}$$

da cui possiamo concludere che $\widehat{u}^{k+1} = X^{(k+1)}\widetilde{q} + \widetilde{e}$, $\widehat{v}^{k+1} = X^{(k+1)T}q + e$. □

3.7 Stabilità numerica degli algoritmi di Newton e Lu

Nel provare la stabilità numerica dei due algoritmi qui presentati, l'unico punto problematico è l'inversione della matrice $\mathcal{R}^{(k)}$. Dimostriamo pertanto che il condizionamento di $\mathcal{R}^{(k)}$ non è peggiore del condizionamento della matrice Jacobiana $M_X^{(k)}$ del metodo di Newton non strutturato. Per brevità omettiamo gli indici (k), e ricordiamo che si ha

$$\mathcal{R} = I_{2n} - V\mathfrak{D}^{-1}U, \quad M_X = \mathfrak{D} - UV,$$

con $\mathfrak{D} \geqslant 0$ diagonale e $U, V > 0$, e che M_X è una M-matrice non singolare. Supponiamo che M_X sia ben condizionata in norma-1, cioè che $\|\mathfrak{D}\|_1$, $\|U\|_1$, $\|V\|_1$, $\left\|M_X^{-1}\right\|_1$ siano limitate dall'alto. Allora, innanzitutto si ha

$$0 \leqslant \mathfrak{D}^{-1} \leqslant (\mathfrak{D} - UV)^{-1},$$

disuguaglianze tra matrici positive, quindi anche $\|\mathfrak{D}^{-1}\|_1$ è limitata. In più, abbiamo

$$\mathcal{B} := \begin{bmatrix} \mathfrak{D} & -U \\ -V & I \end{bmatrix}, \mathcal{B}^{-1} = \begin{bmatrix} (\mathfrak{D} - UV)^{-1} & 0 \\ V(\mathfrak{D} - UV)^{-1} & I \end{bmatrix} \begin{bmatrix} I & U \\ 0 & I \end{bmatrix},$$

da cui segue che anche $\mathcal{B}$ è una M-matrice (perché ha inversa positiva), e che la sua inversa è limitata in norma-1. Ma $\mathcal{R} = I - V\mathfrak{D}^{-1}U$ è il complemento di Schur di $\mathfrak{D}$ in $\mathcal{B}$, e pertanto la sua inversa $\mathcal{R}^{-1}$ è una sottomatrice di $\mathcal{B}^{-1}$. Da questo segue che $\|\mathcal{R}^{-1}\|_1 \leqslant \|\mathcal{B}^{-1}\|_1$, e quindi anche $\mathcal{R}$ è ben condizionata.

Un'altra possibile fonte di instabilità numerica potrebbe essere l'eccessiva crescita dei coefficienti dei generatori durante l'eliminazione gaussiana con l'algoritmo 1.2. Casi di *generators growth* sono stati riportati in alcuni

casi con l'utilizzo dell'algoritmo GKO [28], andando esplicitamente a scegliere matrici iniziali con generatori mal condizionati. Questo non accade nel nostro caso, visto che la norma dei generatori è limitata, e non abbiamo mai incontrato casi di crescita significativa dei coefficienti durante gli esperimenti numerici.

3.8 Riduzione ciclica strutturata

Nella descrizione di questo algoritmo e del successivo, consideriamo direttamente il caso più generale in cui $\mathcal{M}$ è una M-matrice diagonale più rango r. Le notazioni sono formalmente le stesse del caso precedente, cioè

$$\begin{aligned} &XCX + B - XE - AX = 0, \\ &B = \widetilde{e}e^T, \qquad C = \widetilde{q}q^T, \\ &A = \Delta - \widetilde{e}q^T, \quad E = D - \widetilde{q}e^T; \end{aligned} \tag{3.24}$$

con questa volta $e, q, \widetilde{e}, \widetilde{q} > 0$ in $\mathbb{R}^{n\times r}$ e $D = \mathrm{diag}(d)$, $\Delta = \mathrm{diag}(\delta)$ con $d, \delta > 0$ in $\mathbb{R}^n$.

3.8.1 Struttura a blocchi

È semplice notare come durante la riduzione ciclica applicata con matrici iniziali della forma (2.11), non tutti i blocchi si riempiono.

Teorema 3.6. *Siano $\mathcal{A}_{-1}^{(k)}, \mathcal{A}_0^{(k)}, \widehat{\mathcal{A}}_0^{(k)}, \mathcal{A}_1^{(k)}$, $k \geqslant 0$, le iterate generate dalla riduzione ciclica con matrici iniziali come in* (2.11)*. Allora,*

1. *Le iterate hanno la forma*

$$\mathcal{A}_{-1}^{(k)} = \begin{bmatrix} * & 0 \\ * & 0 \end{bmatrix}, \mathcal{A}_0^{(k)} = \begin{bmatrix} -I & * \\ * & * \end{bmatrix}, \widehat{\mathcal{A}}_0^{(k)} = \begin{bmatrix} -I & tC \\ * & -I - tA \end{bmatrix}, \mathcal{A}_1^{(k)} = \begin{bmatrix} 0 & 0 \\ 0 & * \end{bmatrix},$$

   dove $*$ indica un generico blocco $n \times n$.

2. *I blocchi $(2,1)$ di $\mathcal{A}_0^{(k)}$ e di $\widehat{\mathcal{A}}_0^{(k)}$ coincidono.*

Dimostrazione. Il risultato può essere dimostrato per induzione, notando come i blocchi nulli si combinano facendo i prodotti di matrici che determinano l'iterazione CR. In particolare, il secondo punto segue dal fatto che, nelle formule (2.12) per il calcolo di $\mathcal{A}_0^{(k+1)}$ e $\widehat{\mathcal{A}}_0^{(k+1)}$, il termine $-\mathcal{A}_{-1}^{(k)}\mathcal{K}^{(k)}\mathcal{A}_1^{(k)}$ modifica solamente il blocco $(2,1)$, mentre il termine $-\mathcal{A}_1^{(k)}\mathcal{K}^{(k)}\mathcal{A}_{-1}^{(k)}$ modifica solamente la seconda colonna a blocchi. □

3.8.2 Struttura di rango

Applicando la riduzione in forma unilaterale (2.11) alla NARE (3.24), otteniamo

$$\varphi^{(0)}(z) = \begin{bmatrix} (I-D)z^{-1} - I & 0 \\ 0 & zI - (I+\Delta) \end{bmatrix} + \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix};$$

utilizzando la formula SMW (Lemma (1.4)) per l'inversione di $\varphi^{(0)}(z)$, si ha

$$\psi^{(0)}(z) = \mathcal{Z}(z) + \mathcal{Z}(z) \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} r(z) \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix} \mathcal{Z}(z)$$

dove

$$\mathcal{Z}(z) := \begin{bmatrix} (I-D)z^{-1} - I & 0 \\ 0 & zI - (I+\Delta) \end{bmatrix}^{-1},$$
$$r(z) := \left(I_r + \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix} \mathcal{Z}(z) \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} \right)^{-1}.$$

Ora, notiamo che

$$\mathcal{D}\mathcal{Z}(z) = \mathcal{Z}(z)\mathcal{D} = \begin{bmatrix} zI & 0 \\ 0 & -I \end{bmatrix} + zZ(z) \text{ dove } \mathcal{D} := \begin{bmatrix} I-D & 0 \\ 0 & I+\Delta \end{bmatrix};$$

questo ci permette di calcolare

$$\begin{aligned} \nabla_{\mathcal{D},\mathcal{D}}\, \psi^{(0)}(z) = &- \mathcal{D}\mathcal{Z}(z) \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} r(z) \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix} \mathcal{Z}(z) \\ &+ \mathcal{Z}(z) \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} r(z) \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix} \mathcal{Z}(z)\mathcal{D} \\ = &\begin{bmatrix} -z\widetilde{q} \\ \widetilde{e} \end{bmatrix} r(z) \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix} Z(z) + \mathcal{Z}(z) \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} r(z) \begin{bmatrix} e^T & -q^T \end{bmatrix}. \end{aligned}$$

Ponendo

$$\begin{aligned} \widetilde{s}^{(0)}(z) &:= -zr(z) \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix} \mathcal{Z}(z), \\ \widetilde{t}^{(0)}(z) &:= r(z) \begin{bmatrix} z^{-1}e^T & q^T \end{bmatrix} \mathcal{Z}(z), \\ \widetilde{u}^{(0)}(z) &:= \mathcal{Z}(z) \begin{bmatrix} \widetilde{q} \\ \widetilde{e} \end{bmatrix} r(z), \end{aligned}$$

otteniamo

$$\nabla_{\mathcal{D},\mathcal{D}}\, \psi^{(0)}(z) = \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} \widetilde{s}^{(0)}(z) + \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} \widetilde{t}^{(0)}(z) + \widetilde{u}^{(0)}(z) \begin{bmatrix} e^T & -q^T \end{bmatrix}.$$

A questo punto, utilizzando (2.13) e la linearità dell'operatore di displacement $\nabla_{\mathcal{D},\mathcal{D}}$, è semplice dimostrare per induzione che

$$\nabla_{\mathcal{D},\mathcal{D}}\,\psi^{(k)}(z) = \begin{bmatrix}\widetilde{q}\\ 0\end{bmatrix}\widetilde{s}^{(k)}(z) + \begin{bmatrix}0\\ \widetilde{e}\end{bmatrix}\widetilde{t}^{(k)}(z) + \widetilde{u}^{(k)}(z)\begin{bmatrix}e^T & -q^T\end{bmatrix}, \tag{3.25}$$

per tutti i $k \geqslant 0$, dove

$$\begin{aligned}\widetilde{s}^{(k+1)}(z^2) &:= \frac{1}{2}(\widetilde{s}^{(k)}(z) + \widetilde{s}^{(k)}(-z)),\\ \widetilde{t}^{(k+1)}(z^2) &:= \frac{1}{2}(\widetilde{t}^{(k)}(z) + \widetilde{t}^{(k)}(-z)),\\ \widetilde{u}^{(k+1)}(z^2) &:= \frac{1}{2}(\widetilde{u}^{(k)}(z) + \widetilde{u}^{(k)}(-z)).\end{aligned}$$

Sicché, $\psi^{(k)}(z)$ ha rango di displacement 3 per tutti i $k \geqslant 0$. Anche $\varphi^{(k)}(z)$ ha rango di displacement 3, perché utilizzando il punto 4 del Lemma 1.6 si ha

$$\nabla_{\mathcal{D},\mathcal{D}}\,\varphi^{(k)}(z) = \nabla_{\mathcal{D},\mathcal{D}}\left(\psi^{(k)}(z)^{-1}\right) = -\varphi^{(k)}(z)\left(\nabla_{\mathcal{D},\mathcal{D}}\,\psi^{(k)}(z)\right)\varphi^{(k)}(z).$$

Possiamo provare però un risultato più preciso sulla struttura di $\varphi^{(k)}(z)$.

Teorema 3.7. *Sia $\varphi^{(k)}(z) = z\mathcal{A}_1^{(k)} + \mathcal{A}_0^{(k)} + z^{-1}\mathcal{A}_{-1}^{(k)}$ la successione generata dal metodo CR applicato alle* (2.11) *per il problema* (3.24)*. Allora,*

$$\begin{aligned}\nabla_{\mathcal{D},\mathcal{D}}\,\mathcal{A}_{-1}^{(k)} =& \mathcal{A}_{-1}^{(k)}\begin{bmatrix}\widetilde{q}\\ 0\end{bmatrix}s_0^{(k)} + \mathcal{A}_0^{(k)}\begin{bmatrix}0\\ \widetilde{e}\end{bmatrix}t_{-1}^{(k)} + u_0\begin{bmatrix}e^T & -q^T\end{bmatrix}\mathcal{A}_{-1}^{(k)},\\ \nabla_{\mathcal{D},\mathcal{D}}\,\mathcal{A}_0^{(k)} =& \mathcal{A}_{-1}^{(k)}\begin{bmatrix}\widetilde{q}\\ 0\end{bmatrix}s_1^{(k)} + \mathcal{A}_0^{(k)}\begin{bmatrix}\widetilde{q}\\ 0\end{bmatrix}s_0^{(k)} + \mathcal{A}_0^{(k)}\begin{bmatrix}0\\ \widetilde{e}\end{bmatrix}t_0^{(k)}\\ &+ \mathcal{A}_1^{(k)}\begin{bmatrix}0\\ \widetilde{e}\end{bmatrix}t_{-1}^{(k)} + u_0\begin{bmatrix}e^T & -q^T\end{bmatrix}\mathcal{A}_0^{(k)},\\ \nabla_{\mathcal{D},\mathcal{D}}\,\mathcal{A}_1^{(k)} =& \mathcal{A}_0^{(k)}\begin{bmatrix}\widetilde{q}\\ 0\end{bmatrix}s_1^{(k)} + \mathcal{A}_1^{(k)}\begin{bmatrix}0\\ \widetilde{e}\end{bmatrix}t_0^{(k)} + u_0\begin{bmatrix}e^T & -q^T\end{bmatrix}\mathcal{A}_1^{(k)},\end{aligned} \tag{3.26}$$

dove

$$\begin{aligned}s_0^{(0)} &:= \begin{bmatrix}e^T & 0\end{bmatrix}, & s_0^{(k+1)} &:= s_0^{(k)} - s_1^{(k)}\mathcal{K}^{(i)}\mathcal{A}_{-1}^{(k)},\\ s_1^{(0)} &:= \begin{bmatrix}0 & q^T\end{bmatrix}, & s_1^{(k+1)} &:= -s_1^{(k)}\mathcal{K}^{(i)}\mathcal{A}_1^{(k)},\\ t_{-1}^{(0)} &:= -\begin{bmatrix}e^T & 0\end{bmatrix}, & t_{-1}^{(k+1)} &:= -t_{-1}^{(k)}\mathcal{K}^{(i)}\mathcal{A}_{-1}^{(k)},\\ t_0^{(0)} &:= -\begin{bmatrix}0 & q^T\end{bmatrix}, & t_0^{(k+1)} &:= t_0^{(k)} - t_{-1}^{(k)}\mathcal{K}^{(i)}\mathcal{A}_1^{(k)},\\ u_0 &= -\begin{bmatrix}\widetilde{q}\\ \widetilde{e}\end{bmatrix}.\end{aligned} \tag{3.27}$$

Dimostrazione. Dimostreremo il risultato per induzione. Il passo base è una semplice verifica; per quanto riguarda il passo induttivo, per brevità presentiamo solo il calcolo relativo ad $\mathcal{A}_{-1}^{(k)}$; i passaggi da effettuare negli altri due casi sono molto simili. Oltre ai normali passaggi algebrici, ci serve solo notare che $\mathcal{A}_0^{(k)}\mathcal{K}^{(k)} = \mathcal{K}^{(k)}\mathcal{A}_0^{(k)} = I$, per come è definita $\mathcal{K}^{(k)}$, e che

$$\mathcal{A}_{-1}^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} = \mathcal{A}_1^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} = 0,$$

per via della posizione dei blocchi di zeri in $\mathcal{A}_{-1}^{(k)}$ e $\mathcal{A}_1^{(k)}$, (vedi Teorema 3.6).

$$\begin{aligned}
&\nabla_{\mathcal{D},\mathcal{D}}\, \mathcal{A}_{-1}^{(k+1)} = \nabla_{\mathcal{D},\mathcal{D}} \left(-\mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \right) \\
=& - \nabla_{\mathcal{D},\mathcal{D}} \left(\mathcal{A}_{-1}^{(k)} \right) \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} - \mathcal{A}_{-1}^{(k)} \nabla_{\mathcal{D},\mathcal{D}} \left(\mathcal{K}^{(k)} \right) \mathcal{A}_{-1}^{(k)} - \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \nabla_{\mathcal{D},\mathcal{D}} \left(\mathcal{A}_{-1}^{(k)} \right) \\
=& - \nabla_{\mathcal{D},\mathcal{D}} \left(\mathcal{A}_{-1}^{(k)} \right) \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} + \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \nabla_{\mathcal{D},\mathcal{D}} \left(\mathcal{A}_0^{(k)} \right) \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \\
&- \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \nabla_{\mathcal{D},\mathcal{D}} \left(\mathcal{A}_{-1}^{(k)} \right) \\
=& - \left(\mathcal{A}_{-1}^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} s_0^{(k)} + \mathcal{A}_0^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} t_{-1}^{(k)} + u_0 \begin{bmatrix} e & -q \end{bmatrix} \mathcal{A}_{-1}^{(k)} \right) \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \\
&+ \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \left(\mathcal{A}_{-1}^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} s_1^{(k)} + \mathcal{A}_0^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} s_0^{(k)} + \mathcal{A}_0^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} t_0^{(k)} \right. \\
&\left. + \mathcal{A}_1^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} t_{-1}^{(k)} + u_0 \begin{bmatrix} e & -q \end{bmatrix} \mathcal{A}_0^{(k)} \right) \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \\
&- \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \left(\mathcal{A}_{-1}^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} s_0^{(k)} + \mathcal{A}_0^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} t_{-1}^{(k)} + u_0 \begin{bmatrix} e & -q \end{bmatrix} \mathcal{A}_{-1}^{(k)} \right) \\
=& - \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} \left(s_0^{(k)} - s_1^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \right) \\
&+ \left(\mathcal{A}_0^{(k)} - \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \mathcal{A}_1^{(k)} \right) \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} \left(-t_{-1}^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \right) - u_0 \begin{bmatrix} e & -q \end{bmatrix} \mathcal{A}_{-1}^{(k)} \mathcal{K}^{(k)} \mathcal{A}_{-1}^{(k)} \\
=& \mathcal{A}_{-1}^{(k+1)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} s_0^{(k+1)} + \mathcal{A}_0^{(k+1)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix} t_{-1}^{(k+1)} + u_0 \begin{bmatrix} e & -q \end{bmatrix} \mathcal{A}_{-1}^{(k+1)},
\end{aligned}$$

□

Si noti che le equazioni di displacement appena dimostrate possono essere trovate imponendo che

$$\begin{aligned}
\widetilde{s}^{(k)}(z)\varphi^{(k)}(z) &= s_0^{(k)} + s_1^{(k)} z, \\
\widetilde{t}^{(k)}(z)\varphi^{(k)}(z) &= t_0^{(k)} + t_{-1}^{(k)} z^{-1}, \\
\varphi^{(k)}(z)\widetilde{u}^{(k)}(z) &= u_0
\end{aligned}$$

nell'equazione (3.25).

Le matrici $n \times r$ $s_0^{(k)}$, $s_1^{(k)}$, $t_0^{(k)}$ e $t_{-1}^{(k)}$ hanno un'interpretazione più chiara in termini di blocchi della matrice.

Teorema 3.8. *Siano $s_0^{(k)}$, $s_1^{(k)}$, $t_0^{(k)}$ e $t_{-1}^{(k)}$ le successioni definite da* (3.27)*. Allora, per tutti i $k \geqslant 0$,*

1. $s_0^{(k)} = \begin{bmatrix} -e^T & q^T \end{bmatrix} \mathcal{A}_0^{(k)} \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}$;
2. $s_1^{(k)} = \begin{bmatrix} -e^T & q^T \end{bmatrix} \mathcal{A}_1^{(k)}$;
3. *il blocco* $(2,2)$ *di* $\mathcal{A}_0^{(k)}$ *è* $-I - t\Delta - t\widetilde{e}t_0^{(k)}$;
4. *il blocco* $(2,1)$ *di* $\mathcal{A}_{-1}^{(k)}$ *è* $-t\widetilde{e}t_{-1}^{(k)}$.

Il teorema qui sopra può essere dimostrato per induzione.

3.8.3 L'algoritmo strutturato

Le relazioni di struttura di displacement (3.26) ci permettono di sviluppare un algoritmo strutturato per l'iterazione CR, con costo computazionale $O(n^2 r)$. Grazie agli algoritmi 1.1 e 1.2, infatti, possiamo eseguire prodotti e soluzioni di sistemi lineari che coinvolgono $\mathcal{A}_{-1}^{(k)}$, $\mathcal{A}_0^{(k)}$ e $\mathcal{A}_1^{(k)}$ velocemente utilizzando solamente i generatori di queste matrici.

Esplicitamente, ad ogni passo k dell'algoritmo, teniamo in memoria soltanto le otto matrici generatrici

$$\begin{aligned} &\mathcal{A}_{-1}^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix}, \quad \mathcal{A}_0^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix}, \quad \mathcal{A}_0^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix}, \quad \mathcal{A}_1^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix}, \quad t_0^{(k)}, \quad t_{-1}^{(k)}, \\ &\begin{bmatrix} e^T & -q^T \end{bmatrix} \mathcal{A}_{-1}^{(k)}, \quad \begin{bmatrix} e^T & -q^T \end{bmatrix} \mathcal{A}_0^{(k)}, \quad \begin{bmatrix} e^T & -q^T \end{bmatrix} \mathcal{A}_1^{(k)}, \end{aligned} \tag{3.28}$$

(le prime quattro di dimensione $n \times r$, le altre quattro di dimensione $r \times n$) e le aggiorniamo al valore di k successivo. Notiamo che $s_0^{(k)}$ e $s_1^{(k)}$ si possono ricostruire dai generatori qui sopra utilizzando la caratterizzazione data nel Teorema 3.8. Poiché il blocco $(1,1)$ di $\mathcal{A}_{-1}^{(k)}$ e il blocco $(2,2)$ di $\mathcal{A}_1^{(k)}$ sono matrici Trummer-like, oltre ai generatori abbiamo bisogno di calcolare esplicitamente anche le loro diagonali principali. Per fare questo, notiamo che una matrice Trummer-like si può scrivere come $\bar{D} + \text{Trummer}(\mathcal{D}, U, V)$, dove $\bar{D}$ è la sua diagonale principale e $\text{Trummer}(\mathcal{D}, U, V)$ è l'unica matrice

Trummer-like rispetto a $\nabla_{\mathcal{D},\mathcal{D}}$ con generatori U, V e tutti zeri sulla diagonale principale. Di conseguenza, con $T = \mathcal{A}_{-1}^{(k)}$ si ha

$$\mathcal{A}_{-1}^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} - \text{Trummer}(\mathcal{D}, U, V) \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} = \bar{D} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix}. \tag{3.29}$$

Il membro di sinistra dell'uguaglianza può essere calcolato a partire dalle matrici generatrici (3.28), quindi per l'uguaglianza sappiamo quanto vale $\bar{D} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix}$. Per come sono messi i blocchi nulli di $\mathcal{A}_{-1}^{(k)}$, solo le prime n componenti di $\bar{D}$ sono diverse da zero, ed esse possono essere calcolate come

$$\bar{D}_{ii} = \frac{\left(\bar{D} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} \right)_{i1}}{\widetilde{q}_{i1}}. \tag{3.30}$$

Possiamo calcolare la diagonale del blocco $(2,2)$ di $\mathcal{A}_1^{(k)}$ lavorando allo stesso modo a partire da $\mathcal{A}_1^{(k)} \begin{bmatrix} 0 \\ \widetilde{e} \end{bmatrix}$.

Quindi, a ogni passo dell'algoritmo calcoliamo innanzitutto le due diagonali principali, $s_0^{(k)}$ e $s_1^{(k)}$; quindi andiamo ad aggiornare le otto matrici generatrici: $t_0^{(k)}$ e $t_{-1}^{(k)}$ con le formule (3.27), le altre direttamente dalle formule che definiscono l'iterazione CR (2.12), ad esempio

$$\mathcal{A}_{-1}^{(k+1)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} = -\mathcal{A}_1^{(k)} \mathcal{K}^{(k)} \left(\mathcal{A}_1^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} \right),$$

e analogamente per le altre cinque. Tutti i prodotti di matrici che coinvolgono $\mathcal{K}^{(k)}$ si possono effettuare attraverso l'Algoritmo 1.2 su $\mathcal{A}_0^{(k)}$ (difatti, si aveva $\mathcal{K}^{(k)} = \left(\mathcal{A}_0^{(k)} \right)^{-1}$, e la diagonale di $\mathcal{A}_0^{(k)}$ è completamente nota grazie al Teorema 3.6 e al punto 3 del teorema 3.8. Poiché applichiamo questi algoritmi per moltiplicare le matrici strutturate per matrici $n \times r$ (o le loro versioni trasposte a matrici $r \times n$), o per risolvere sistemi lineari con termine costante della stessa dimensione, $s = r$ e quindi il costo computazionale totale è $O(n^2 r)$. Le operazioni da eseguire sono descritte brevemente nell'Algoritmo 3.2

La scelta più immediata per il criterio di stop sarebbe quella di calcolare ad ogni passo l'iterata $X^{(k)}$ e quindi calcolare esplicitamente il residuo dell'equazione di Riccati (3.24); tuttavia, seguire questa strada comporta calcoli abbastanza dispendiosi. Il Teorema 2.4 fornisce un'altra possibilità per il criterio di stop. Poiché insieme alla convergenza si ha che anche $\mathcal{A}_{-1}^{(k)}$ e

function X=fastcr$(D,\Delta,e,q,\widetilde{e},\widetilde{q})$
k=0;
inizializza le matrici generatrici
(con (3.27) e (2.12))
do
calcola le diagonali di $\mathcal{A}_{-1}^{(k)}$ e $\mathcal{A}_{1}^{(k)}$
(con (3.29) e (3.30))
calcola $s_0^{(k)}$ and $s_1^{(k)}$
calcola le matrici generatrici al passo $k+1$
k=k+1
while(criterio di stop)
costruisci i generatori di $\widehat{\mathcal{A}}_0^{(k)}$
(con il Teorema (3.6))
calcola X=il blocco $(2,1)$ di $-\left(\widehat{\mathcal{A}}^{(k)}\right)^{-1}\mathcal{A}_{-1}^{(0)}$
end function

Algoritmo 3.2: Riduzione ciclica strutturata

$\mathcal{A}_1^{(k)}$ convergono a un limite finito, ci basta verificare che le norme delle due matrici

$$\mathcal{A}_{-1}^{(k+1)}\begin{bmatrix}\widetilde{q}\\0\end{bmatrix}-\mathcal{A}_{-1}^{(k)}\begin{bmatrix}\widetilde{q}\\0\end{bmatrix},\quad \mathcal{A}_{1}^{(k+1)}\begin{bmatrix}0\\\widetilde{e}\end{bmatrix}-\mathcal{A}_{1}^{(k)}\begin{bmatrix}0\\\widetilde{e}\end{bmatrix}$$

siano sufficientemente piccole. Questo può essere verificato velocemente, in quanto le matrici di cui abbiamo bisogno sono due delle matrici generatrici che vengono già calcolate ad ogni passo. Nel caso non critico, un'altra scelta praticabile è controllare che

$$\min\left(\left\|\mathcal{A}_{-1}^{(k)}\begin{bmatrix}\widetilde{q}\\0\end{bmatrix}\right\|_1,\left\|\mathcal{A}_{1}^{(k)}\begin{bmatrix}0\\\widetilde{e}\end{bmatrix}\right\|_1\right)<\varepsilon,$$

poiché almeno una tra $\mathcal{A}_{-1}^{(k)}$ e $\mathcal{A}_{1}^{(k)}$ converge a zero, sempre per il Teorema 2.4.

Si noti anche che l'algoritmo può essere ancora leggermente accelerato tenendo conto della posizione dei blocchi di zeri, evitando di calcolare esplicitamente i prodotti che li coinvolgono e che sappiamo essere zero. In questo modo, riduciamo tutti i calcoli a calcoli con matrici $n\times n$ e $n\times r$. In questa forma, i blocchi $n\times r$ (o $r\times n$) delle matrici generatrici di cui dobbiamo tenere traccia sono dieci.

3.9 SDA strutturato

Passiamo ora a descrivere come anche lo structured doubling algorithm può essere implementato in $O(n^2 r)$ per le equazioni di Riccati con $\mathcal{M}$ diagonale più rango r (3.24). Notiamo innanzitutto che per ogni k le matrici $\mathcal{H}_\gamma^{2^k}$ e $\mathcal{H}$ commutano, in quanto $\mathcal{H}_\gamma^{2^k}$ è una funzione razionale di $\mathcal{H}$. Si ha allora

$$\mathcal{H}_\gamma^{2^k}\left(\begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix} - \begin{bmatrix} \widetilde{q} \\ -\widetilde{e} \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix}\right) = \left(\begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix} - \begin{bmatrix} \widetilde{q} \\ -\widetilde{e} \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix}\right)\mathcal{H}_\gamma^{2^k},$$

o anche, riordinando i termini,

$$\begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix}\mathcal{H}_\gamma^{2^k} - \mathcal{H}_\gamma^{2^k}\begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix} = \begin{bmatrix} \widetilde{q} \\ -\widetilde{e} \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix}\mathcal{H}_\gamma^{2^k} - \mathcal{H}_\gamma^{2^k}\begin{bmatrix} \widetilde{q} \\ -\widetilde{e} \end{bmatrix} \begin{bmatrix} e^T & q^T \end{bmatrix}, \tag{3.31}$$

il che implica che $\mathcal{H}_\gamma^{2^k}$ ha rango di displacement 2 rispetto a

$$\mathcal{D} = \begin{bmatrix} D & 0 \\ 0 & -\Delta \end{bmatrix}$$

per ogni $k \geqslant 0$. Possiamo arrivare semplicemente anche a un risultato sulla struttura dei blocchi E_k, F_k, G_k, H_k. Dalla fattorizzazione dimostrata nel teorema 2.6

$$\mathcal{H}_\gamma^{2^k} = \mathcal{U}_k^{-1}\mathcal{L}_k = \begin{bmatrix} I & G_k F_k^{-1} \\ 0 & F_k^{-1} \end{bmatrix}\begin{bmatrix} E_k & 0 \\ -H_k & I \end{bmatrix}$$

è semplice verificare che

$$\begin{aligned} \begin{bmatrix} I & -G_k \end{bmatrix}\mathcal{H}_\gamma^{2^k} &= \begin{bmatrix} E_k & 0 \end{bmatrix}, \\ \begin{bmatrix} 0 & F_k \end{bmatrix}\mathcal{H}_\gamma^{2^k} &= \begin{bmatrix} -H_k & I \end{bmatrix}, \\ \mathcal{H}_\gamma^{2^k}\begin{bmatrix} I \\ H_k \end{bmatrix} &= \begin{bmatrix} E_k \\ 0 \end{bmatrix}, \\ \mathcal{H}_\gamma^{2^k}\begin{bmatrix} 0 \\ F_k \end{bmatrix} &= \begin{bmatrix} G_k \\ I \end{bmatrix}. \end{aligned} \tag{3.32}$$

Moltiplicando a sinistra per $\begin{bmatrix} 0 & F_k \end{bmatrix}$ e a destra per $\begin{bmatrix} 0 \\ F_k \end{bmatrix}$ la (3.31) otteniamo

$$-F_k\Delta + \Delta F_k = -F_k\widetilde{e}(e^T + q^T G_k) + (H_k\widetilde{q} + \widetilde{e})q^T F_k.$$

Allo stesso modo, moltiplicando la (3.31) a sinistra per $\begin{bmatrix} 0 & F_k \end{bmatrix}$ o per $\begin{bmatrix} I & -G_k \end{bmatrix}$ e a destra per $\begin{bmatrix} I \\ H_k \end{bmatrix}$ o per $\begin{bmatrix} 0 \\ F_k \end{bmatrix}$ in tutte le quattro combinazioni, otteniamo

rispettivamente

$$\begin{aligned} DE_k - E_kD &= (\widetilde{q} + G_k\widetilde{e})e^TE_k - E_k\widetilde{q}(e^T + q^TH_k), \\ \Delta F_k - F_k\Delta &= (H_k\widetilde{q} + \widetilde{e})q^TF_k - F_k\widetilde{e}(e^T + q^TG_k), \\ DG_k + G_k\Delta &= (\widetilde{q} + G_k\widetilde{e})(e^T + q^TG_k) - E_k\widetilde{q}q^TF_k, \\ \Delta H_k + H_kD &= (H_k\widetilde{q} + \widetilde{e})(e^T + q^TH_k) - E_k\widetilde{q}q^TF_k, \end{aligned} \tag{3.33}$$

che forniscono un'espressione esplicita di E_k e F_k come matrici Trummer-like e di G_k e H_k come matrici Cauchy-like.

L'idea dell'algoritmo, similmente al caso della riduzione ciclica, è di tenere in memoria le otto matrici

$$\begin{matrix} E_k\widetilde{q}, & F_k\widetilde{e}, & (H_k\widetilde{q} + \widetilde{e}), & (\widetilde{q} + G_k\widetilde{e}), \\ e^TE_k, & q^TF_k, & (e^T + q^TH_k), & (e^T + q^TG_k), \end{matrix} \tag{3.34}$$

e calcolare ad ogni iterazione come si modificano secondo le equazioni che definiscono lo SDA (2.14), effettuando tutti i calcoli grazie agli algoritmi per matrici Cauchy-like e Trummer-like della sezione 1.3.3. Per utilizzare effettivamente tali algoritmi, abbiamo però bisogno di ricavare esplicitamente le diagonali principali di E_k e F_k. Queste possono essere calcolate utilizzando la stessa idea della sezione 3.8.3: poiché già conosciamo, per esempio, il valore di $E_k\widetilde{q}$ e i generatori Trummer-like U, V di E_k, possiamo scrivere

$$E_k\widetilde{q} = \mathrm{diag}(E_k)\widetilde{q} + \mathrm{Trummer}(U, V)\widetilde{q}, \tag{3.35}$$

ricavare da questa equazione $\mathrm{diag}(E_k)\widetilde{q}$, e infine calcolare

$$(E_k)_{ii} = \frac{(\mathrm{diag}(E_k)\widetilde{q})_{i1}}{\widetilde{q}_{i1}}. \tag{3.36}$$

Il calcolo della diagonale principale di F_k avviene analogamente a partire da $F_k\widetilde{e}$. Per l'inversione delle due matrici $I - G_kH_k$ e $I - H_kG_k$ serve inoltre conoscere anche le diagonali principali di queste due matrici. Queste si possono ricavare calcolando esplicitamente i coefficienti di G_k e H_k a partire dai loro generatori displacement (in tempo $O(n^2r)$, come nell'Algoritmo 1.1), e andando a calcolare solo le entrate diagonali dei prodotti di matrici G_kH_k e H_kG_k: ognuno degli n elementi diagonali $(G_kH_k)_{ii}$ si calcola con un prodotto scalare tra la riga i-esima di G_k e la colonna i-esima di H_k, quindi in tempo $O(n)$; complessivamente, il tempo necessario per quest'ultimo passo è quindi $O(n^2)$. Riassumiamo brevemente le operazioni da effettuarsi nell'Algoritmo 3.3.

```
function X=fastsda(D,Δ,e,q,ẽ,q̃)
 k=0;
 calcola le matrici generatrici (3.34) al passo 0
 do
  calcola le diagonali di E_k e F_k (con (3.35) e (3.36))
  calcola le diagonali di I − G_k H_k e I − H_k G_k
  calcola le matrici generatrici al passo k+1
  k=k+1
 while(criterio di stop)
 restituisci X=H_k
end function
```

Algoritmo 3.3: Structred doubling algorithm strutturato

Come criterio di stop, analogamente al caso della riduzione ciclica, possiamo controllare per esempio che

$$\max\left(\left\|E_{k+1}\widetilde{q}-E_k\widetilde{q}\right\|_1,\left\|F_{k+1}\widetilde{e}-F_k\widetilde{e}\right\|_1\right)<\varepsilon,$$

oppure, nel caso non critico, che

$$\min\left(\left\|E_k\widetilde{q}\right\|_1,\left\|F_k\widetilde{e}\right\|_1\right)<\varepsilon.$$

3.10 Relazioni tra SDA e CR

La somiglianza in superficie dei calcoli e degli algoritmi necessari per lo sviluppo delle versioni veloci dello SDA e della CR in realtà è indice di un collegamento più profondo tra questi due algoritmi, come l'analisi del caso particolare dell'equazione (3.5) ha permesso di scoprire. Il risultato che intendiamo provare in questa sezione è il seguente.

Teorema 3.9. *Lo* structured doubling algorithm *è formalmente uguale alla riduzione ciclica applicata all'equazione unilaterale*

$$\begin{bmatrix}E_0 & 0\\ 0 & 0\end{bmatrix}+\begin{bmatrix}-I & G_0\\ H_0 & -I\end{bmatrix}X+\begin{bmatrix}0 & 0\\ 0 & F_0\end{bmatrix}X^2=0. \tag{3.37}$$

Dimostrazione. Supponiamo per semplicità di essere nel caso in cui $\mathcal{M}$ è una M-matrice non singolare, e $\mathcal{H}$ è diagonalizzabile. In virtù dei risultati della sezione 2.1, se S è la soluzione minimale non negativa della NARE (2.1), vale

$$\mathcal{H}\begin{bmatrix}I\\ S\end{bmatrix}=\begin{bmatrix}I\\ S\end{bmatrix}R,$$

dove $R = E - CS$ è una M-matrice. È semplice provare per induzione che vale anche

$$\mathcal{H}^k \begin{bmatrix} I \\ S \end{bmatrix} = \begin{bmatrix} I \\ S \end{bmatrix} R^k,$$

o più in generale per una funzione $f(Y)$ esprimibile come serie di potenze in Y si ha

$$f(\mathcal{H}) \begin{bmatrix} I \\ S \end{bmatrix} = \begin{bmatrix} I \\ S \end{bmatrix} f(R). \tag{3.38}$$

Applichiamo questo risultato a $\mathcal{C}_\gamma : Y \mapsto (Y + \gamma I)^{-1}(Y - \gamma I)$, dove γ è come definito nella sezione 2.6. Poiché la $\mathcal{C}_\gamma$, vista come funzione $\mathbb{C} \to \mathbb{C}$, manda il semipiano positivo nell'interno del cerchio unitario e il semipiano negativo nell'esterno, per i risultati sugli autovalori di $\mathcal{H}$ e di R della sezione 2.2, $\mathcal{H}_\gamma := \mathcal{C}_\gamma(\mathcal{H})$ ha n autovalori dentro il cerchio unitario e n fuori, e gli autovalori di $R_\gamma := \mathcal{C}_\gamma(R)$ sono gli n autovalori dentro al cerchio unitario di $\mathcal{H}_\gamma$. Ora, utilizziamo la scomposizione del caso $k = 0$ del Teorema 2.6, e abbiamo

$$\begin{bmatrix} E_0 & 0 \\ -H_0 & I \end{bmatrix} \begin{bmatrix} I \\ S \end{bmatrix} = \begin{bmatrix} I & -G_0 \\ 0 & F_0 \end{bmatrix} \begin{bmatrix} R_\gamma \\ SR_\gamma \end{bmatrix}, \tag{3.39}$$

con E_0, F_0, G_0, H_0 definiti come in (2.15). Utilizzando questa relazione è semplice verificare che

$$\mathcal{S} := \begin{bmatrix} R_\gamma & 0 \\ SR_\gamma & 0 \end{bmatrix}$$

è una soluzione della (3.37), in quanto eseguendo i prodotti a blocchi della (3.37) e della (3.39) si ottengono formalmente le stesse uguaglianze tra matrici.

Abbiamo ora bisogno di un risultato più generale sulle equazioni matriciali e sulla riduzione ciclica, dimostrato in [6].

Teorema 3.10. *Sia* $\varphi(z) = z^{-1}\mathcal{A}_{-1} + \mathcal{A}_0 + z\mathcal{A}_1$, *con* $\mathcal{A}_i \in \mathbb{C}^{m\times m}$, $i = -1, 0, 1$. *Supponiamo che il polinomio* $\det(z\varphi(z))$ *abbia* m *zeri di modulo minore di 1 e* m *zeri di modulo maggiore di 1 (contando eventuali zeri all'infinito se il grado è minore di* $2m$*), e che esista una soluzione* $\mathcal{S}$ *dell'equazione matriciale* (2.10) *tale che* $\rho(\mathcal{S}) < 1$. *Allora,*

1. $\mathcal{S}$ *è l'unica soluzione della* (2.10) *tale che* $\rho(\mathcal{S}) \leqslant 1$*;*
2. *Applicando la riduzione ciclica alla* (2.10)*, la successione*

$$\mathcal{S}^{(k)} = -\left(\widehat{\mathcal{A}}^{(k)}\right)^{-1} \mathcal{A}_{-1}$$

converge quadraticamente a $\mathcal{S}$.

3. *Detta* $\mathcal{W} = \mathcal{A}_1\mathcal{S} + \mathcal{A}_0$*, la matrice* $\mathcal{T} = -\mathcal{A}_1\mathcal{W}$ *è l'unica soluzione dell'equazione*
$$\mathcal{A}_1 + X\mathcal{A}_0 + X^2\mathcal{A}_{-1} = 0 \tag{3.40}$$
tale che $\rho(\mathcal{T}) \leqslant 1$.

4. *Vale la seguente fattorizzazione, nota come* fattorizzazione canonica*:*
$$\varphi(z) = (I - z\mathcal{T})\mathcal{W}(I - z^{-1}\mathcal{S}), \quad \textit{per } |z| = 1. \tag{3.41}$$

Dimostriamo innanzitutto l'applicabilità del risultato al nostro caso. Il polinomio
$$\det\left(\begin{bmatrix} E_0 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -I & G_0 \\ H_0 & -I \end{bmatrix} z + \begin{bmatrix} 0 & 0 \\ 0 & F_0 \end{bmatrix} z^2\right) = 0 \tag{3.42}$$
ha in totale $4n$ zeri. Poiché il termine costante e quello quadratico hanno entrambi una colonna di zeri, è semplice verificare che n degli zeri del polinomio sono in 0 e n all'infinito. I restanti $2n$ zeri sono esattamente gli autovalori di $\mathcal{H}_\gamma$: difatti, se (λ, u) è una coppia autovalore–autovettore di $\mathcal{H}_\gamma$ allora si ha
$$\begin{bmatrix} E_0 & 0 \\ -H_0 & I \end{bmatrix} u = \lambda \begin{bmatrix} I & -G_0 \\ 0 & F_0 \end{bmatrix} u,$$
e, moltiplicando quest'ultima a sinistra per $\begin{bmatrix} I & 0 \\ 0 & -\lambda I \end{bmatrix}$, si ottiene
$$\begin{bmatrix} E_0 & 0 \\ \lambda H_0 & -\lambda I \end{bmatrix} u = \begin{bmatrix} \lambda I & -\lambda G_0 \\ 0 & \lambda^2 F_0 \end{bmatrix} u,$$
o anche, semplicemente riordinando i termini,
$$\left(\begin{bmatrix} E_0 & 0 \\ 0 & 0 \end{bmatrix} + \begin{bmatrix} -I & G_0 \\ H_0 & -I \end{bmatrix} \lambda + \begin{bmatrix} 0 & 0 \\ 0 & F_0 \end{bmatrix} \lambda^2\right) u = 0.$$

Poiché allora è applicabile il teorema 3.10, la CR applicata all'equazione (3.37) converge alla $\mathcal{S}$. È semplice notare che, per come sono disposti i blocchi di zeri nella CR, le iterate restano sempre nella forma
$$\mathcal{A}_{-1}^{(k)} = \begin{bmatrix} E_k & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathcal{A}_0^{(k)} = \begin{bmatrix} -I & G_k \\ H_k & -I \end{bmatrix}, \quad \mathcal{A}_1^{(k)} = \begin{bmatrix} 0 & 0 \\ 0 & F_k \end{bmatrix},$$
e che le relazioni che legano due iterate successive sono
$$\begin{aligned} E_{k+1} &= E_k(I - G_kH_k)^{-1}E_k, \\ F_{k+1} &= F_k(I - H_kG_k)^{-1}F_k, \\ G_{k+1} &= G_k + E_k(I - G_kH_k)^{-1}G_kF_k, \\ H_{k+1} &= H_k + F_k(I - H_kG_k)^{-1}H_kE_k, \end{aligned}$$

cioè esattamente le stesse relazioni dello SDA (2.14).

Possiamo inoltre dimostrare utilizzando solo le proprietà della riduzione ciclica che $\lim H_k = X$. Notiamo innanzitutto che, con le notazioni del teorema 3.10, si ha

$$\mathcal{W} = \begin{bmatrix} 0 & 0 \\ 0 & F_0 \end{bmatrix} \begin{bmatrix} R_\gamma & 0 \\ SR_\gamma & 0 \end{bmatrix} + \begin{bmatrix} -I & G_0 \\ H_0 & -I \end{bmatrix} = \begin{bmatrix} -I & G_0 \\ S & -I \end{bmatrix},$$

dove abbiamo utilizzato la relazione $S = H_0 + F_0 S R_\gamma$, che deriva dalla seconda riga a blocchi della (3.39), e

$$\mathcal{T} = -\begin{bmatrix} 0 & 0 \\ 0 & F_0 \end{bmatrix} \begin{bmatrix} -I & G_0 \\ S & -I \end{bmatrix}^{-1} = \begin{bmatrix} 0 & 0 \\ TS & T \end{bmatrix},$$

con $T = F(I - SG_0)^{-1}$. La (3.41) allora diventa

$$\varphi(z) = \left(I - z \begin{bmatrix} 0 & 0 \\ TS & T \end{bmatrix} \right) \begin{bmatrix} -I & G_0 \\ S & -I \end{bmatrix} \left(I - z^{-1} \begin{bmatrix} R_\gamma & 0 \\ SR_\gamma & 0 \end{bmatrix} \right).$$

Invertendo entrambi i lati dell'equazione (per $|z| = 1$) si ha

$$\psi(z) := \varphi(z)^{-1} = \left(\sum_{j \geqslant 0} z^{-j} \begin{bmatrix} R_\gamma^j & 0 \\ SR_\gamma^j & 0 \end{bmatrix} \right) \mathcal{W}^{-1} \left(\sum_{j \geqslant 0} z^j \begin{bmatrix} 0 & 0 \\ T^j S & T^j \end{bmatrix} \right).$$

Il termine noto della serie $\psi(z) = \sum_{i \in \mathbb{Z}} \psi_i z^i$ allora è

$$\begin{aligned} \psi_0 =& \sum_{j \geqslant 0} \begin{bmatrix} R_\gamma^j & 0 \\ SR_\gamma^j & 0 \end{bmatrix} \mathcal{W}^{-1} \begin{bmatrix} 0 & 0 \\ T^j S & T^j \end{bmatrix} \\ =& \mathcal{W}^{-1} + \begin{bmatrix} I \\ S \end{bmatrix} \left(\sum_{j \geqslant 1} \begin{bmatrix} R_\gamma^j & 0 \end{bmatrix} \mathcal{W}^{-1} \begin{bmatrix} 0 \\ T^j \end{bmatrix} \right) \begin{bmatrix} S & I \end{bmatrix} \end{aligned} \tag{3.43}$$

Ora, grazie alla formulazione (2.13) della CR, otteniamo facilmente che le iterate della CR sono

$$\psi^{(k)}(z) = \sum_{i \in \mathbb{Z}} \psi_{2^k i} z^i;$$

in particolare, il termine noto ψ_0 resta invariato, e gli altri coefficienti delle $\psi^{(k)}(z)$ sono di volta in volta coefficienti di indice sempre maggiore della $\psi(z)$. In particolare, dalla convergenza della CR (o dall'analiticità di $\psi(z)$ in un intorno della circonferenza unitaria, si veda [6] per maggiori dettagli) è possibile dedurre che gli altri coefficienti convergono uniformemente a zero, cioè che

$$\lim_{k \to \infty} \psi^{(k)}(z) = \psi_0, \quad \text{per } |z| = 1.$$

Ne segue che

$$\lim_{k\to\infty} \varphi^{k)}(z) = \psi_0^{-1}, \quad \text{per } |z| = 1.$$

Confrontando i termini costanti, si ha

$$\lim_{k\to\infty} \begin{bmatrix} -I & G_k \\ H_k & -I \end{bmatrix} = \psi_0^{-1}. \tag{3.44}$$

Dalla (3.43) possiamo stabilire un risultato parziale sulla struttura della ψ_0^{-1}; indicando

$$P := \sum_{j\geqslant 1} \begin{bmatrix} R_\gamma^j & 0 \end{bmatrix} \mathcal{W}^{-1} \begin{bmatrix} 0 \\ F_0^j \end{bmatrix},$$

abbiamo

$$\psi_0^{-1} = \mathcal{W}^{-1} + \begin{bmatrix} I \\ S \end{bmatrix} P \begin{bmatrix} S & I \end{bmatrix};$$

applicando la formula SMW otteniamo

$$\psi_0 = \mathcal{W} + \mathcal{W} \begin{bmatrix} I \\ S \end{bmatrix} \widehat{P}^{-1} \begin{bmatrix} S & I \end{bmatrix} \mathcal{W}, \tag{3.45}$$

dove $\widehat{P}$ è la matrice da invertire che compare nella formula SMW, che non siamo qui interessati a calcolare esplicitamente. Poiché

$$\mathcal{W} \begin{bmatrix} I \\ S \end{bmatrix} = \begin{bmatrix} * \\ 0 \end{bmatrix}, \quad \begin{bmatrix} S & I \end{bmatrix} \mathcal{W} = \begin{bmatrix} 0 & * \end{bmatrix}$$

(le $*$ di nuovo indicano blocchi che non siamo interessati a calcolare esplicitamente), il secondo addendo nel termine di destra della (3.45) va a modificare solo il blocco $(1, 2)$, per cui abbiamo

$$\psi_0^{-1} = \begin{bmatrix} -I & * \\ S & -I \end{bmatrix}.$$

In particolare questa relazione combinata alla (3.44) dimostra che $\lim_{k\to\infty} H_k = S$. □

Partendo dalla (2.5) e operando nello stesso modo, si arriva a

$$\begin{bmatrix} E_0 & 0 \\ -H_0 & I \end{bmatrix} \begin{bmatrix} T \\ I \end{bmatrix} = \begin{bmatrix} I & -G_0 \\ 0 & F \end{bmatrix} \begin{bmatrix} T \\ I \end{bmatrix} U,$$

con $U = \mathcal{C}_\gamma(-A + BT)$, da cui si può dimostrare che

$$\mathcal{T} = \begin{bmatrix} 0 & 0 \\ TU^{-1} & U^{-1} \end{bmatrix}$$

è una soluzione con $\rho(\mathcal{T}) < 1$ dell'equazione unilaterale

$$\begin{bmatrix} E_0 & 0 \\ 0 & 0 \end{bmatrix} X^2 + \begin{bmatrix} -I & G_0 \\ H_0 & -I \end{bmatrix} X + \begin{bmatrix} 0 & 0 \\ 0 & F_0 \end{bmatrix} = 0,$$

arrivare ad una fattorizzazione canonica diversa di $\varphi(z^{-1})$ e dimostrare con un ragionamento analogo che $\lim_{k\to\infty} G_k = T$.

3.11 Possibili varianti degli algoritmi

Nell'analisi condotta, possiamo notare che le uniche proprietà della funzione $\mathcal{C}_\gamma$ che abbiamo usato sono che $\mathcal{C}_\gamma(\mathcal{H})$ abbia n autovalori all'interno del cerchio unitario e n fuori. Gli stessi risultati possono essere raggiunti con una qualunque altra funzione che trasformi lo *splitting* degli autovalori di $\mathcal{H}$ in uno splitting rispetto alla circonferenza unitaria, necessario per la convergenza della riduzione ciclica. Possiamo interpretare alla luce di questo risultato la riduzione di Ramaswami come composta di due parti distinte: l'applicazione della funzione $f(Z) = I - tZ$ alla matrice $\mathcal{H}$, che ci dà

$$f(\mathcal{H}) = \begin{bmatrix} I - tE & tC \\ -tB & I + tA \end{bmatrix} = \begin{bmatrix} \widehat{E} & -\widehat{C} \\ \widehat{B} & -\widehat{A} \end{bmatrix}, \tag{3.46}$$

e una riduzione a un'equazione unilaterale sostanzialmente diversa da quella dello SDA applicata all'equazione di Riccati

$$X\widehat{C}X - \widehat{A}X - X\widehat{E} + \widehat{B} = 0,$$

che ha le stesse soluzioni dell'equazione originaria (perché l'applicazione di una funzione razionale preserva i sottospazi invarianti di $\mathcal{H}$) ma presenta uno splitting degli autovalori di $\mathcal{H}$ rispetto alla circonferenza unitaria. Nel dettaglio, tale riduzione è quella data dall'equivalenza

$$\begin{bmatrix} \widehat{E} & -\widehat{C} \\ \widehat{B} & -\widehat{A} \end{bmatrix} \begin{bmatrix} I \\ X \end{bmatrix} = \begin{bmatrix} I \\ X \end{bmatrix} f(R)$$
$$\Updownarrow \tag{3.47}$$
$$\begin{bmatrix} \widehat{E} & 0 \\ -\widehat{B} & 0 \end{bmatrix} + \begin{bmatrix} -I & -\widehat{C} \\ 0 & \widehat{A} \end{bmatrix} \begin{bmatrix} f(R) & 0 \\ X & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} f(R) & 0 \\ X & 0 \end{bmatrix}^2 = 0$$

Da questo punto di vista, è chiaro che i due passaggi sono indipendenti. In particolare, possiamo immaginare la riduzione di Ramaswami e lo SDA come appartenenti a una stessa classe di algoritmi definiti da

- un metodo di shift per creare uno splitting degli autovalori rispetto alla circonferenza unitaria;
- un metodo per ridurre l'equazione di Riccati a un'equazione unilaterale;
- un metodo per risolvere l'equazione unilaterale (CR o altri algoritmi analoghi).

Come primo esempio, proponiamo un algoritmo che utilizza la riduzione unilaterale dello SDA (che fornisce un costo per passo leggermente minore rispetto a quello della CR) e il metodo di shift della riduzione di Ramaswami. Nei dettagli, partiamo dalla (3.46), facciamone una fattorizzazione UL formalmente analoga alla (3.39)

$$f(\mathcal{H}) = \begin{bmatrix} E_t & tC \\ -tB & A_t \end{bmatrix} = \begin{bmatrix} I & -tCA_t^{-1} \\ 0 & A_t^{-1} \end{bmatrix}^{-1} \begin{bmatrix} E_t - t^2 CA_t^{-1}B & 0 \\ -tA_t^{-1}B & I \end{bmatrix},$$

con $E_t := I - tE$ e $A_t := I + tA$, e applichiamo la riduzione (3.37); in questo modo otteniamo un algoritmo con regole di update uguali alle (2.14) (e quindi lo stesso costo per passo dello SDA), e valori iniziali

$$\begin{aligned} \widehat{E}_0 &= E_t - t^2 CA_t^{-1}B, \\ \widehat{F}_0 &= A_t^{-1}, \\ \widehat{G}_0 &= tCA_t^{-1}, \\ \widehat{H}_0 &= -tA_t^{-1}B. \end{aligned}$$

Tale algoritmo ha valori iniziali leggermente più semplici da calcolare rispetto a quelli dello SDA e dovrebbe quindi essere lievemente più veloce e meglio condizionato. Analogamente allo SDA, tale algoritmo può essere applicato per la soluzione di tutte le equazioni di Riccati in cui $\mathcal{M}$ sia una M-matrice non singolare oppure singolare irriducibile, e ha convergenza quadratica nei casi non critici.

Capitolo 4

Implementazione e risultati numerici

4.1 Primo confronto tra gli algoritmi

Come complemento allo sviluppo teorico, abbiamo effettivamente implementato e testato gli algoritmi descritti nel capitolo precedente. Il linguaggio scelto per l'implementazione è stato il Fortran 90. Gli esperimenti sono stati condotti sul server del gruppo di ricerca di analisi numerica dell'università di Pisa; un riassunto della configurazione del computer utilizzato per gli esperimenti è dato in Tabella 4.1. Gli algoritmi testati sono i seguenti:

SDA Lo *structured doubling algorithm* come descritto nella sezione 2.6;

CR La riduzione ciclica, come descritta nella sezione 2.5, accelerata tenendo conto della posizione dei blocchi di zeri come dimostrato nel Teorema 3.6;

Lu L'algoritmo di Newton applicato all'iterazione di Lu (3.14), implemen-

processore	4 × Intel Xeon 2.8 Ghz
cache size	512 KByte per processore
memoria RAM	6 GByte
Sistema operativo	SuSE Linux 9.1 (kernel 2.6.5-bigsmp)
Compilatore FORTRAN	Lahey/Fujitsu Fortran 95 compiler L6.20c
Opzioni compilatore	`-o2 -tp4`

Tabella 4.1: Configurazione del sistema usato per gli esperimenti

tato come descritto nell'articolo originale di Lu [23] con l'uso della funzione LAPACK95 `la_gesv` per la soluzione del sistema lineare;

sSDA La versione strutturata dello SDA descritta nella sezione 3.9;

sCR La versione strutturata della CR descritta nella sezione 3.8;

sLu La versione strutturata dell'algoritmo di Lu applicato all'iterazione di Newton, implementata con l'uso dell'Algoritmo 1.2 per la soluzione del sistema lineare.

I primi due algoritmi sono algoritmi generici per la soluzione di NARE non strutturate; il terzo è un algoritmo *ad-hoc* per il problema derivante dall'equazione del trasporto di neutroni (3.5). I successivi tre sono le versioni specializzate alla struttura del problema (3.5) sviluppate in questa tesi. I primi tre algoritmi hanno complessità $O(n^3)$ per passo, gli ultimi tre $O(n^2)$ per passo.

L'equazione utilizzata per il test è l'equazione (3.5), con le tre scelte di parametri $(\alpha, c) = (0.5, 0.5)$, (caso non critico) $(\alpha, c) = (10^{-8}, 1 - 10^{-6})$ (vicino al caso critico), $(\alpha, c) = (0, 1)$ (caso critico). Nel caso critico, è stata utilizzata la tecnica di shift con $\eta = d_1$. I risultati riportati nei grafici sono il tempo totale di esecuzione e il residuo relativo dell'equazione, calcolato come

$$Res = \frac{\left\| \mathrm{R}(\widetilde{X}) \right\|_1}{\max(\left\| \widetilde{X}\widetilde{q} + \widetilde{e} \right\|_1, \left\| e^T + q^T \widetilde{X} \right\|_1)},$$

dove $\widetilde{X}$ è la soluzione calcolata. Nei grafici, 1 significa che la convergenza non è stata raggiunta ($Res \geqslant 1$).

Come si può notare, il metodo **sLu** è notevolmente più veloce degli altri metodi e ha una buona accuratezza; esso è quindi sicuramente il metodo da preferire per la soluzione delle equazioni di Riccati. Per $n = 2048$, per esempio, **sLu** è circa 50 volte più veloce di **Lu**, il più competitivo degli algoritmi non strutturati.

Dai grafici emerge il fatto che i tre metodi strutturati hanno complessità $O(n^2)$ anziché $O(n^3)$, sotto forma della diversa pendenza delle rette dei tempi. Per grandi dimensioni, anche gli algoritmi **sCR** e **sSDA** diventano più veloci del più competitivo tra i metodi non strutturati. Tuttavia, nei casi quasi critico e critico essi presentano notevoli problemi di stabilità, che li portano ad avere una pessima accuratezza o eventualmente a non convergere del tutto. La causa di questi problemi è probabilmente dovuta al metodo suggerito

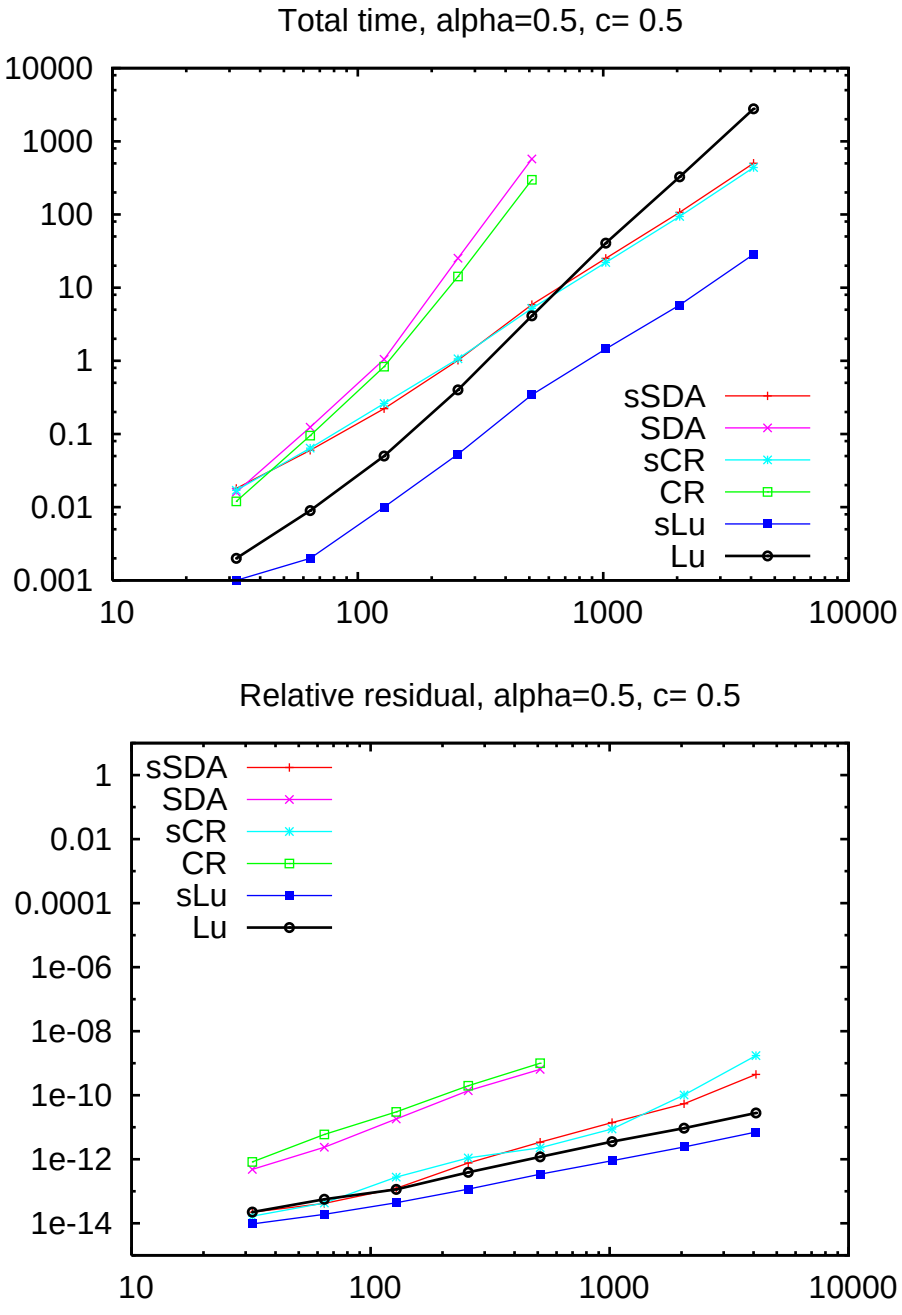

Figura 4.1: Caso non critico

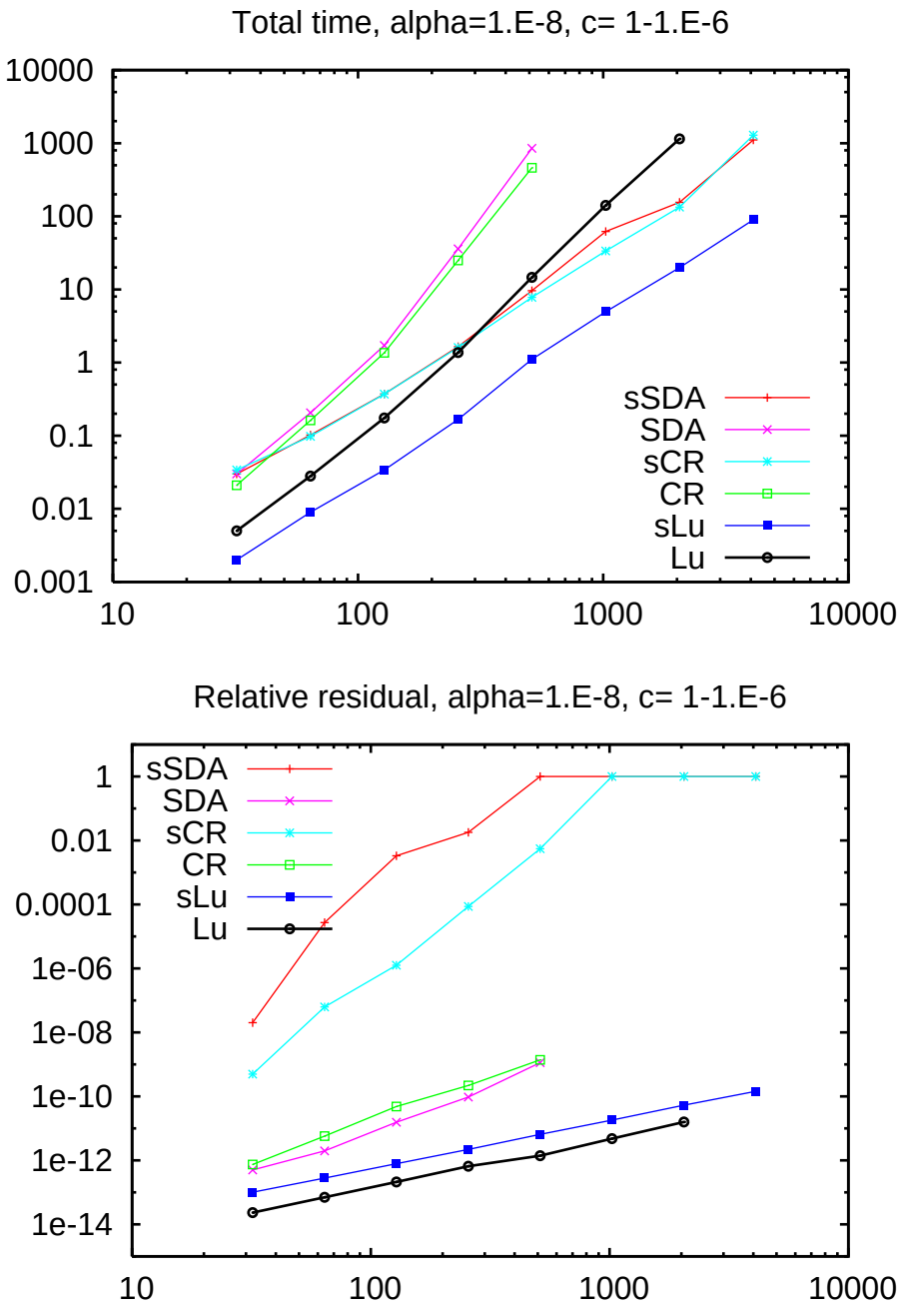

Figura 4.2: Caso quasi critico

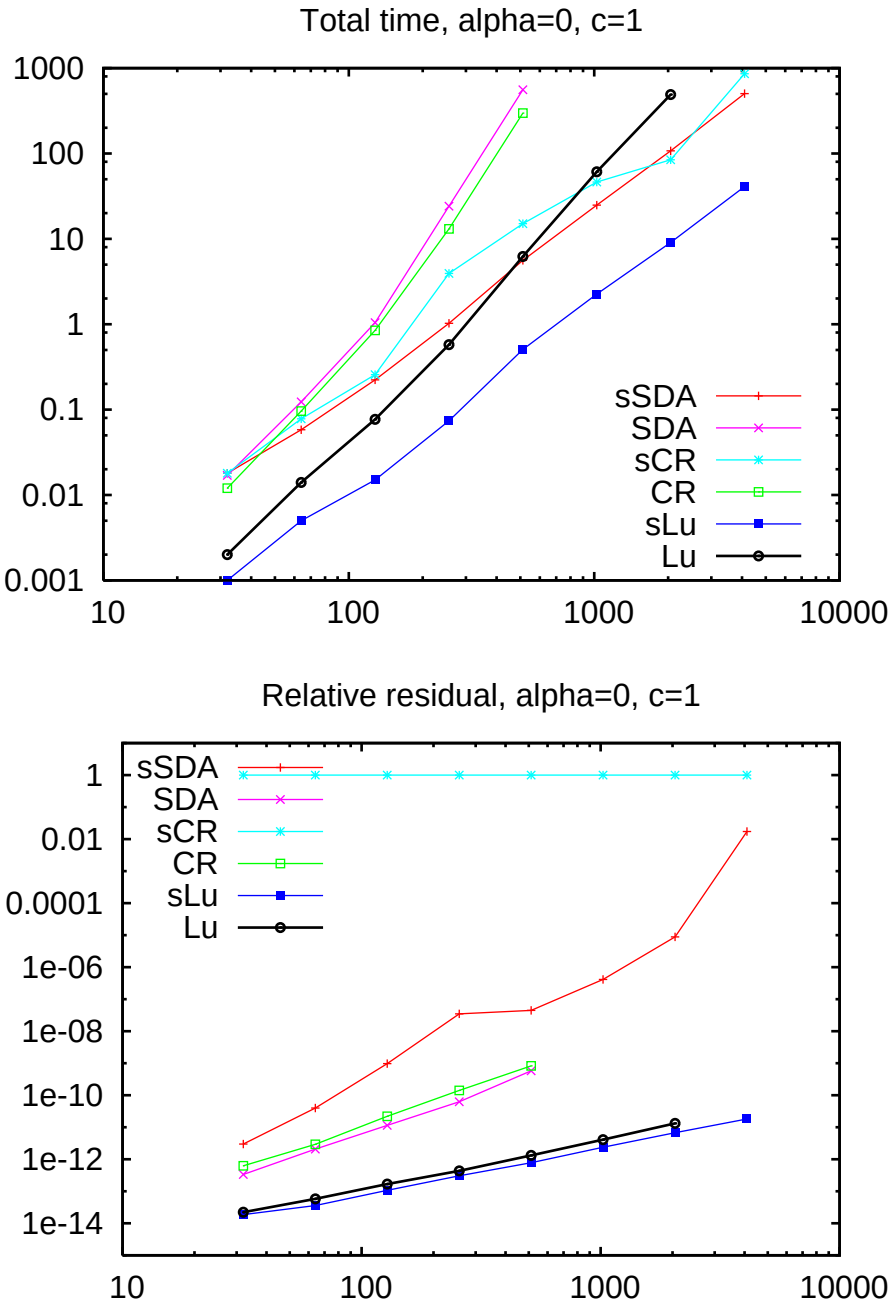

Figura 4.3: Caso critico

	$\alpha = 0.5,\ c = 0.5$	
n	Lu	sLu
32	$4.8 \cdot 10^{-16}$ (4)	$2.3 \cdot 10^{-16}$ (5)
256	$1.6 \cdot 10^{-15}$ (4)	$4.0 \cdot 10^{-16}$ (5)

	$\alpha = 0,\ c = 1$		
n	Lu	sLu	sLu+shift
32	$5.2 \cdot 10^{-8}$ (25)	$4.2 \cdot 10^{-8}$ (26)	$4.4 \cdot 10^{-16}$ (6)
256	$4.6 \cdot 10^{-8}$ (25)	$8.0 \cdot 10^{-8}$ (25)	$1.2 \cdot 10^{-15}$ (6)

Tabella 4.2: Errore relativo (e in parentesi il numero di passi iterativi) dei metodi **Lu**, **sLu** e, nel caso critico, **sLu+shift**

per l'update delle diagonali delle matrici Trummer-like coinvolte. Difatti, il metodo suggerito prevede di calcolare per esempio (equazione (3.29))

$$\mathcal{A}_{-1}^{(k)} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} - \text{Trummer}(\mathcal{D}, U, V) \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix} = \bar{D} \begin{bmatrix} \widetilde{q} \\ 0 \end{bmatrix};$$

nei casi vicino al critico, la sottrazione da effettuare al membro di sinistra è tra due valori abbastanza vicini. Questo causa un'apprezzabile perdita di precisione ad ogni iterazione. A differenza del metodo di Lu, nello SDA e nella CR gli errori si accumulano tra un passo e il successivo, in quanto i dati del problema intervengono solo nelle matrici iniziali al passo 0, quindi la perdita di accuratezza si accumula tutta sul risultato finale. Sarebbe necessario trovare un metodo diverso per l'*update* delle diagonali per rendere stabili questi algoritmi anche nei casi vicini al critico.

4.2 Efficacia del metodo di shift ed accuratezza della soluzione

Visti i risultati dei test precedenti, ci concentriamo qui sui soli algoritmi **Lu** e **sLu**. Nei due casi test $n = 32$ e $n = 256$, abbiamo confrontato la soluzione calcolata dai due metodi con quella calcolata in quadrupla precisione, che assumiamo come soluzione esatta. Riportiamo nella Tabella 4.2 l'errore relativo

$$\frac{\left\| \widetilde{X} - X \right\|_1}{\|X\|_1},$$

dove $\widetilde{X}$ è la soluzione calcolata dai due metodi e X è la soluzione calcolata in quadrupla precisione.

Gli esperimenti evidenziano come l'accuratezza raggiunta dai due metodi sia sostanzialmente la stessa, e riesca a scendere ai livelli della precisione di macchina $\varepsilon \approx 10^{-16}$. Nel caso critico, i metodi non shiftati raggiungono un'accuratezza di circa $O(\sqrt{\varepsilon})$, in accordo con l'analisi proposta da Guo e Higham [14]. L'applicazione della tecnica di shift non solo permette di raggiungere piena accuratezza nella soluzione calcolata, ma permette anche di raggiungere convergenza quadratica anziché lineare, e quindi di abbassare sostanzialmente il numero di passi richiesti. Per $n = 512$, per esempio, **sLu+shift** è cinque volte più veloce di **sLu** e 80 volte più veloce di **Lu**.

Capitolo 5

Conclusioni e spunti di ricerca

I risultati raggiunti sono interessanti sia dal punto di vista computazionale che da quello teorico. Dal punto di vista computazionale, l'algoritmo **sLu** descritto sopra fornisce un metodo iterativo per la soluzione dell'equazione (3.5) più veloce di quelli a noi noti in letteratura [23, 17, 13]. L'efficacia si manifesta già per basse dimensioni, ma cresce ulteriormente al crescere della dimensione n delle matrici coinvolte, in quanto la complessità è di $O(n^2)$ per passo anziché $O(n^3)$. Gli algoritmi **sSDA** e **sCR** sono più lenti e presentano seri problemi di stabilità vicino al caso critico. Tuttavia, vale la pena di notare che entrambi sono stati sviluppati direttamente per il problema più generale delle equazioni di Riccati (3.24) in cui $\mathcal{M}$ è una M-matrice diagonale più rango r. In particolare, presentano una complessità di $O(rn^2)$, da confrontare con la complessità di $O(r^2n^2)$ che si otterrebbe sviluppando la versione per lo stesso problema degli algoritmi di Newton e Newton–Lu. Pertanto, SDA e CR dovrebbero diventare più veloci degli algoritmi di tipo Newton in problemi del tipo (3.24) con $r \approx 10 - 15$. Sarebbe necessario effettuare ulteriori sforzi per trovare una versione dei due algoritmi che sia stabile anche per casi vicino al critico. Va inoltre esaminata la possibilità di utilizzare all'interno degli algoritmi qui sviluppati le tecniche con costo quasi-lineare per le operazioni con matrici con struttura di rango, come proposto nella parte finale della sezione 1.3.3.

Dal punto di vista teorico, l'analisi condotta ha permesso di riconoscere che gli algoritmi in esame non sono quattro algoritmi scorrelati, ma sono a due a due intimamente legati. Questo si riflette innanzitutto sull'analisi teorica, ad esempio in quanto i risultati di convergenza di un algoritmo possono essere più facilmente collegati a quelli dell'altro e le dimostrazioni possono essere unificate. Particolarmente rilevante è poi il rapporto scoperto tra SDA e CR, in quanto questo risultato non riguarda solo il problema che stiamo studiando, ma è applicabile a tutte le equazioni di Riccati algebriche e alle

altre equazioni matriciali a cui è applicabile lo SDA. Ricondurre un algoritmo nuovo come lo SDA a un algoritmo noto da tempo e più studiato quale la CR aiuta a comprenderne più a fondo la struttura e le proprietà. Il nuovo punto di vista consente di elaborare nuove varianti dei due algoritmi che varrebbe la pena di analizzare in modo più approfondito.

Appendice A

Notazioni utilizzate

$\otimes$	Prodotto di Kronecker tra matrici (vedi sezione 1.4)
$\mathrm{diag}(d_1, \ldots, d_n)$	Matrice diagonale D di dimensione $n \times n$ definita da $D_{ii} = d_i$, $D_{ij} = 0$ per $i \neq j$
$\mathrm{diag}(d)$	quando $d = \begin{bmatrix} d_1 & d_2 & \cdots & d_n \end{bmatrix}^T$ è un vettore, la matrice $\mathrm{diag}(d_1, \ldots, d_n)$
$\mathrm{diag}(M)$	quando $M \in \mathbb{R}^{n \times n}$ è una matrice quadrata, il vettore formato dagli elementi diagonali di M: $\mathrm{diag}(M) = \begin{bmatrix} M_{11} & M_{22} & \cdots & M_{nn} \end{bmatrix}^T$
I_n	Matrice identità di dimensione $n \times n$. Quando è evidente dal contesto, l'indice n è omesso
$\mathbb{R}^{m \times n}$	Algebra delle matrici $m \times n$ a coefficienti reali
$\mathrm{Span}\, M$	Immagine della matrice $m \times n$ M, cioè $\mathrm{Span}\, M = \{w \in \mathbb{R}^m \mid \exists v \in \mathbb{R}^n \text{ t.c. } Mv = w\}$
$\rho(M)$	$\rho(M) = \max\{\lvert\lambda\rvert \mid \lambda \text{ autovalore di } M\}$, raggio spettrale della matrice quadrata M
$\mathrm{rk}(M)$	Rango della matrice M
$\mathrm{Sp}(M)$	$\mathrm{Sp}(M) = \{\lambda \mid \lambda \text{ autovalore di } M\}$, spettro della matrice quadrata M
M^T	trasposta della matrice M: $(M^T)_{ij} = M_{ji}$

Bibliografia

[1] R. H. Bartels and G. W. Stewart, *Solution of the equation $AX + XB = C$*, Commun. ACM, 15 (1972), pp. 820–826.

[2] A. Berman and R. J. Plemmons, *Nonnegative matrices in the mathematical sciences*, vol. 9 of Classics in Applied Mathematics, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1994. Revised reprint of the 1979 original.

[3] D. A. Bini, L. Gemignani, and B. Meini, *Computations with infinite Toeplitz matrices and polynomials*, Linear Algebra Appl., 343/344 (2002), pp. 21–61. Special issue on structured and infinite systems of linear equations.

[4] D. A. Bini, B. Iannazzo, G. Latouche, and B. Meini, *On the solution of algebraic Riccati equations arising in fluid queues*, Linear Algebra Appl., 413 (2006), pp. 474–494.

[5] D. A. Bini, B. Iannazzo, and F. Poloni, *A fast Newton's method for a nonsymmetric algebraic Riccati equation*, tech. rep., Dipartimento di Matematica, Università di Pisa, Pisa, Italy, 2007. Submitted for publication.

[6] D. A. Bini, G. Latouche, and B. Meini, *Numerical methods for structured Markov chains*, Numerical Mathematics and Scientific Computation, Oxford University Press, New York, 2005.

[7] F. Coron, *Computation of the asymptotic states for linear half space kinetic problems*, Transport Theory Stat. Phys., 19 (1990), pp. 89–114.

[8] B. D. Ganapol, *An investigation of a simple transport model*, Transport Theory Stat. Phys., 21 (1992), pp. 1–37.

[9] A. Gerasoulis, *A fast algorithm for the multiplication of generalized Hilbert matrices with vectors*, Math. Comp., 50 (1988), pp. 179–188.

[10] I. Gohberg, T. Kailath, and V. Olshevsky, *Fast Gaussian elimination with partial pivoting for matrices with displacement structure*, Math. Comp., 64 (1995), pp. 1557–1576.

[11] G. H. Golub and C. F. Van Loan, *Matrix computations*, Johns Hopkins Studies in the Mathematical Sciences, Johns Hopkins University Press, Baltimore, MD, third ed., 1996.

[12] C.-H. Guo, *Nonsymmetric algebraic Riccati equations and Wiener-Hopf factorization for M-matrices*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 225–242 (electronic).

[13] ——, *Efficient methods for solving a nonsymmetric algebraic Riccati equation arising in stochastic fluid models*, J. Comput. Appl. Math., 192 (2006), pp. 353–373.

[14] C.-H. Guo and N. J. Higham, *Iterative solution of a nonsymmetric algebraic Riccati equation*, Numer. Linear Algebra Appl., 12 (2005), pp. 191–200.

[15] C.-H. Guo, B. Iannazzo, and B. Meini, *On the doubling algorithm for a (shifted) nonsymmetric algebraic Riccati equation*, tech. rep., Dipartimento di Matematica, Università di Pisa, Pisa, Italy, May 2006. To appear in SIMAX.

[16] C.-H. Guo and A. J. Laub, *On the iterative solution of a class of nonsymmetric algebraic Riccati equations*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 376–391 (electronic).

[17] X.-X. Guo, W.-W. Lin, and S.-F. Xu, *A structure-preserving doubling algorithm for nonsymmetric algebraic Riccati equation*, Numer. Math., 103 (2006), pp. 393–412.

[18] C. He, B. Meini, and N. H. Rhee, *A shifted cyclic reduction algorithm for quasi-birth-death problems*, SIAM J. Matrix Anal. Appl., 23 (2001/02), pp. 673–691 (electronic).

[19] L. Hogben, ed., *Handbook of linear algebra*, Discrete Mathematics and its Applications (Boca Raton), Chapman & Hall/CRC, Boca Raton, FL, 2007. Associate editors: Richard Brualdi, Anne Greenbaum and Roy Mathias.

[20] J. Juang, C. L. Hsing, and P. Nelson, *Global existence, asymptotics and uniqueness for the reflection kernel of the angularly shifted transport equation*, Math. Models Methods Appl. Sci., 5 (1995), pp. 239–251.

[21] J. Juang and W.-W. Lin, *Nonsymmetric algebraic Riccati equations and Hamiltonian-like matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 228–243 (electronic).

[22] P. Lancaster and L. Rodman, *Algebraic Riccati equations*, Oxford Science Publications, The Clarendon Press Oxford University Press, New York, 1995.

[23] L.-Z. Lu, *Newton iterations for a non-symmetric algebraic Riccati equation*, Numer. Linear Algebra Appl., 12 (2005), pp. 191–200.

[24] ——, *Solution form and simple iteration of a nonsymmetric algebraic Riccati equation arising in transport theory*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 679–685 (electronic).

[25] P.-G. Martinsson, V. Rokhlin, and M. Tygert, *A fast algorithm for the inversion of general Toeplitz matrices*, Comput. Math. Appl., 50 (2005), pp. 741–752.

[26] H.-B. Meyer, *The matrix equation $AZ + B - ZCZ - ZD = 0$*, SIAM J. Appl. Math., 30 (1976), pp. 136–142.

[27] V. Ramaswami, *Matrix analytic methods for stochastic fluid flows*, in Proceedings of the 16th International Teletraffic Congress, Elsevier Science, Edinburg, 1999, pp. 19–30.

[28] D. R. Sweet and R. P. Brent, *Error analysis of a fast partial-pivoting method for structured matrices*, vol. 2563 of Advanced Signal Processing Algorithms, Proceedings of SPIE, 1995, pp. 266–280.

Indice