PERUGIA, GALN 2009

UN ALGORITMO PER IL CALCOLO DI RADICI PRIMARIE DI MATRICI

Ivan Gerace, **Federico Greco** & Bruno Iannazzo

Università degli Studi di Perugia

17 febbraio 2009

Radici Primarie e Principali

Sia p un intero positivo e sia A una matrice non-singolare in $\mathbb{C}^{n\times n}$.
Consideriamo l'equazione matriciale

$$X^p = A.$$

Definizione

Una soluzione $X \in \mathbb{C}^{n\times n}$ dell'equazione $X^p = A$

- è una **radice p-esima primaria** di A se X è esprimibile come polinomio di A;
- è una **radice p-esima principale** di A se X è primaria ed i suoi autovalori appartengono al settore del piano complesso

$$S_p = \{z \in \mathbb{C} \setminus \{0\} : |\arg(z)| \leq \pi/p\}.$$

RISULTATI NOTI:

- Una matrice A con s autovalori distinti ha esattamente p^s radici p-esime primarie.
- Se nessun autovalore di A è un numero reale non positivo, allora A ha una sola radice p-esima principale indicata con $A^{1/p}$.
  In particolare, se A è reale, allora $A^{1/p}$ è reale.

RISULTATI NOTI:

- Una matrice A con s autovalori distinti ha esattamente p^s radici p-esime primarie.
- Se nessun autovalore di A è un numero reale non positivo, allora A ha una sola radice p-esima principale indicata con $A^{1/p}$.
  In particolare, se A è reale, allora $A^{1/p}$ è reale.

APPLICAZIONI del calcolo di radici primarie di matrici:

- Calcolo di altre funzioni matriciali quali **Settore**, **Logaritmo** e **Media**;
- (Finanza) Modellizzazioni del rischio di credito (**Credit Risk Model**).

Algoritmi conosciuti (per matrici piene)

1. Algoritmi che usano la decomposizione di Schur

Björck-Hammarling[1983]	caso $p = 2$ ed $A \in \mathbb{C}^{n \times n}$
Higham[1986]	caso $p = 2$ ed $A \in \mathbb{R}^{n \times n}$ (utilizzando l'aritmetica reale)
Smith[2003]	caso p intero positivo (con aritmetica appropriata)

Algoritmi conosciuti (per matrici piene)

1. Algoritmi che usano la decomposizione di Schur

Björck-Hammarling[1983]	caso $p = 2$ ed $A \in \mathbb{C}^{n\times n}$
Higham[1986]	caso $p = 2$ ed $A \in \mathbb{R}^{n\times n}$ (utilizzando l'aritmetica reale)
Smith[2003]	caso p intero positivo (con aritmetica appropriata)

2. Algoritmi che usano iterazioni matriciali

Hoskins-Walton[1979]	caso A definita positiva
Iannazzo[2006]	caso p generico, A qualsiasi
Guo-Higham[2006]	caso p generico, A qualsiasi
Zietak-Laskiewicz[2009]	Miglioramenti nell'algoritmo

ALGORITMI CONOSCIUTI: CONFRONTO

Algoritmo	**Stabilità all'indietro**	**Operazioni**	**Memoria**
Schur	$\surd$	$O(pn^3)$	$O(pn^2)$
Iterazione	?	$O(n^3 \log p)$	$O(n^2)$

ALGORITMI CONOSCIUTI: CONFRONTO

Algoritmo	**Stabilità all'indietro**	**Operazioni**	**Memoria**
Schur	$\surd$	$O(pn^3)$	$O(pn^2)$
Iterazione	?	$O(n^3 \log p)$	$O(n^2)$

- *Iterazioni matriciali* **minor** numero di **operazioni**
  minor quantità di **memoria**
- *Decomposizione di Schur* **stabili** all'indietro.
- Smith[2003] (*Schur*) calcola tutte le radici primarie con lo stesso costo.

Algoritmo di Smith (caso $A \in \mathbb{R}^{n\times n}$)

Problema

Input: Un intero positivo p e una matrice $A \in \mathbb{R}^{n\times n}$ non-singolare.
Output: Una matrice X, radice p-esima primaria di A.

Algoritmo di Smith (caso $A \in \mathbb{R}^{n\times n}$)

Problema

Input: Un intero positivo p e una matrice $A \in \mathbb{R}^{n\times n}$ non-singolare.
Output: Una matrice X, radice p-esima primaria di A.

1. Si determina attraverso il metodo di Schur una matrice (quasi)-triangolare superiore $T \in \mathbb{R}^{n\times n}$ tale che $T = Q^*AQ$, per qualche $Q \in \mathbb{R}^{n\times n}$ ortogonale;

Algoritmo di Smith (caso $A \in \mathbb{R}^{n\times n}$)

Problema

Input: Un intero positivo p e una matrice $A \in \mathbb{R}^{n\times n}$ non-singolare.
Output: Una matrice X, radice p-esima primaria di A.

1. Si determina attraverso il metodo di Schur una matrice (quasi)-triangolare superiore $T \in \mathbb{R}^{n\times n}$ tale che $T = Q^*AQ$, per qualche $Q \in \mathbb{R}^{n\times n}$ ortogonale;
2. Si determina una radice p-esima primaria U della matrice T;

Algoritmo di Smith (caso $A \in \mathbb{R}^{n\times n}$)

Problema

Input: Un intero positivo p e una matrice $A \in \mathbb{R}^{n\times n}$ non-singolare.
Output: Una matrice X, radice p-esima primaria di A.

1. Si determina attraverso il metodo di Schur una matrice (quasi)-triangolare superiore $T \in \mathbb{R}^{n\times n}$ tale che $T = Q^*AQ$, per qualche $Q \in \mathbb{R}^{n\times n}$ ortogonale;
2. Si determina una radice p-esima primaria U della matrice T;
3. La soluzione cercata è $X = QUQ^*$

X è reale se A non ha autovalori negativi.

ALGORITMO DI SMITH: PUNTO 2

Sia $T \in \mathbb{R}^{n\times n}$ (quasi)-triangolare superiore. Per risolvere l'equazione matriciale

$$U^p = T$$

si definiscono ricorsivamente le seguenti p matrici:

$$\begin{cases} R^{(1)} = U \\ R^{(k)} = U \cdot R^{(k-1)} = U^k, & \text{per } k = 2, \ldots, p. \end{cases}$$

ALGORITMO DI SMITH: PUNTO 2

Sia $T \in \mathbb{R}^{n\times n}$ (quasi)-triangolare superiore. Per risolvere l'equazione matriciale

$$U^p = T$$

si definiscono ricorsivamente le seguenti p matrici:

$$\begin{cases} R^{(1)} = U \\ R^{(k)} = U \cdot R^{(k-1)} = U^k, & \text{per } k = 2, \ldots, p. \end{cases}$$

- $R^{(p)} = U^p = T$ è la matrice nota in partenza;
- Le matrici $R^{(k)}$ per $k = 1, \ldots, p-1$ devono essere calcolate;
- $R^{(1)} = U$ è la matrice cercata.

ALGORITMO DI SMITH: PUNTO 2

STEP 1

Sia l il numero dei suoi blocchi diagonali di T. Si calcolano e si memorizzano i blocchi diagonali $R_{jj}^{(1)} = U_{jj}, \quad \text{per } j = 1, \ldots, l.$

ALGORITMO DI SMITH: PUNTO 2

STEP 1

Sia l il numero dei suoi blocchi diagonali di T. Si calcolano e si memorizzano i blocchi diagonali $R_{jj}^{(1)} = U_{jj}, \quad \text{per } j = 1, \ldots, l.$

- U ha la stessa struttura di T perché è funzione di T.
- T_{jj} può indicare un blocco 1×1 o un blocco 2×2.
- Se T_{jj} è un blocco 1×1, allora U_{jj} è (una) radice p-esima. Se T_{jj} è un blocco 2×2, allora

$$U_{jj} = \alpha I_2 + \beta/\mu(T_{jj} - \theta I_2), \quad \text{per opportuni } \alpha, \beta, \mu, \theta \in \mathbb{R}.$$

- Per avere la radice principale devo scegliere la radice principale di ciascun blocco. Con scelte diverse si ottengono tutte le radici primarie.

ALGORITMO DI SMITH: PUNTO 2

STEP 2

Per $k = 1, \ldots, p-1$, si calcolano e si memorizzano i blocchi diagonali

$$R_{jj}^{(k)} = U_{jj}^{k} = (U_{jj})^{k}, \qquad \text{per } j = 1, \ldots, l.$$

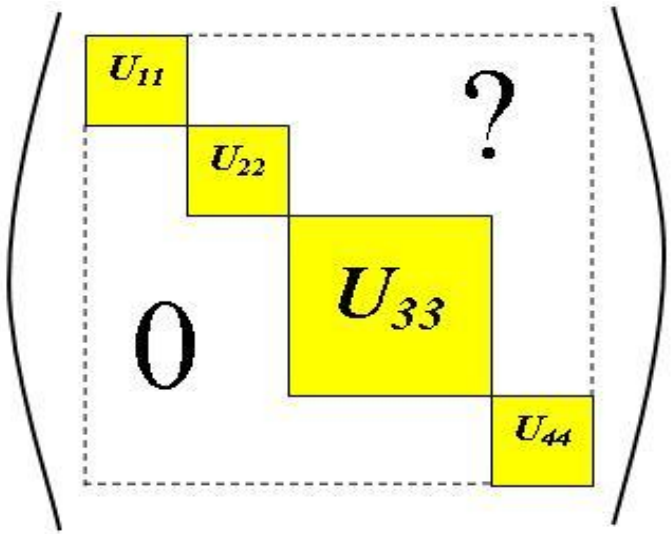
ALGORITMO DI SMITH: PUNTO 2

STEP 2

Per $k = 1, \ldots, p-1$, si calcolano e si memorizzano i blocchi diagonali

$$R_{jj}^{(k)} = U_{jj}^{k} = (U_{jj})^{k}, \qquad \text{per } j = 1, \ldots, l.$$

Al termine dello STEP 2 sono noti i blocchi diagonali di ogni $R^{(k)} = U^{k}$



ALGORITMO DI SMITH: PUNTO 2

STEP 3

Per $k = 1, \ldots, p-1$, si calcolano e si memorizzano i blocchi $R_{ij}^{(k)} = U_{ij}^{k}$ che appartengono alla prima sopradiagonale.

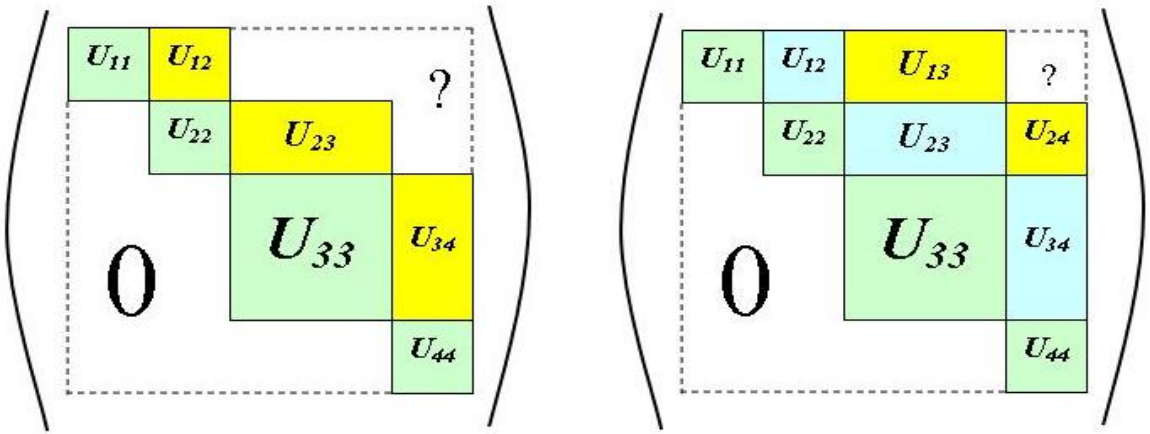
Si ripete la stessa operazione per la $(j-i)$-esima sopradiagonale, per $(j-i) = 2, \ldots, l-1$.

ALGORITMO DI SMITH: PUNTO 2

STEP 3

Per $k = 1, \ldots, p-1$, si calcolano e si memorizzano i blocchi $R_{ij}^{(k)} = U_{ij}^{k}$ che appartengono alla prima sopradiagonale.

Si ripete la stessa operazione per la $(j-i)$-esima sopradiagonale, per $(j-i) = 2, \ldots, l-1$.



Blocchi noti dopo STEP 3 per $(j-i) = 1$ e dopo STEP 3 per $(j-i) = 2$

COME AVVIENE LO STEP 3

Il blocco $R_{ij}^{(1)} = U_{ij}$ si determina risolvendo l'equazione di Sylvester:

$$\sum_{q=0}^{p-1} U_{ii}^{q} R_{ij}^{(1)} U_{jj}^{p-1-q} = T_{ij} - \sum_{k=1}^{p} U_{ii}^{p-k} B_{ij}^{(k)}$$

$B_{ij}^{(k)} = \sum_{\xi=i+1}^{j-1} R_{i\xi}^{(1)} R_{\xi j}^{(k-1)}$ è funzione di blocchi appartenenti a sopradiagonali precedentemente calcolate.

Ad esempio, nel caso $(j-i) = 1$, $B_{ij}^{(k)}$ è il blocco nullo.

COME AVVIENE LO STEP 3

Il blocco $R_{ij}^{(1)} = U_{ij}$ si determina risolvendo l'equazione di Sylvester:

$$\sum_{q=0}^{p-1} U_{ii}^{q} R_{ij}^{(1)} U_{jj}^{p-1-q} = T_{ij} - \sum_{k=1}^{p} U_{ii}^{p-k} B_{ij}^{(k)}$$

$B_{ij}^{(k)} = \sum_{\xi=i+1}^{j-1} R_{i\xi}^{(1)} R_{\xi j}^{(k-1)}$ è funzione di blocchi appartenenti a sopradiagonali precedentemente calcolate.

Ad esempio, nel caso $(j-i) = 1$, $B_{ij}^{(k)}$ è il blocco nullo.

Per $k = 2, \ldots, p-1$, il blocco $R_{ij}^{(k)}$ si calcola così:

$$R_{ij}^{(k)} = \sum_{q=0}^{k-1} U_{ii}^{q} R_{ij}^{(1)} U_{jj}^{k-1-q} + \sum_{q=0}^{k-1} U_{ii}^{q} B_{ij}^{(k-q)}.$$

ALGORITMO DI SMITH: COSTO

STEP	Operazione eseguita	ops	mem
1-2	Calcolo e memorizzazione degli $R_{jj}^{(k)}$	$O(pn)$	$O(pn)$
3	Calcolo dei $B_{ij}^{(k)}$	$O(pn^3)$	
3	Calcolo dei p termini e risoluzione delle $O(n^2)$ equazioni di Sylvester	$O(pn^2)$	
3	Calcolo e memorizzazione degli $R_{ij}^{(k)}$	$O(pn^2)$	$O(pn^2)$

ALGORITMO DI SMITH: COSTO

STEP	Operazione eseguita	ops	mem
1-2	Calcolo e memorizzazione degli $R_{jj}^{(k)}$	$O(pn)$	$O(pn)$
3	Calcolo dei $B_{ij}^{(k)}$	$O(pn^3)$	
3	Calcolo dei p termini e risoluzione delle $O(n^2)$ equazioni di Sylvester	$O(pn^2)$	
3	Calcolo e memorizzazione degli $R_{ij}^{(k)}$	$O(pn^2)$	$O(pn^2)$

COSTO TOTALE ALGORITMO:

PUNTI	ops	memoria
1 e 3	$O(n^3)$	$O(n^2)$
2	$O(pn^3)$	$O(pn^2)$

OSSERVAZIONI SU SMITH[2003]

Supponiamo di voler risolvere

$$X^{2^k} = A.$$

Se applichiamo Smith[2003] per $p = 2^k$ abbiamo un totale di $O(2^k n^3)$ operazioni.

Osservazioni su Smith[2003]

Supponiamo di voler risolvere

$$X^{2^k} = A.$$

Se applichiamo Smith[2003] per $p = 2^k$ abbiamo un totale di $O(2^k n^3)$ operazioni.

Se applichiamo k volte Smith[2003] per $p = 2$ abbiamo un totale di $O(kn^3) = O(n^3 \log p)$ operazioni, come nel caso del miglior algoritmo iterativo.

Nel caso $X^p = A$ con p primo questa osservazione non può essere ripetuta. Questo spinge a cercare un algoritmo che usi il metodo di Schur e abbia un totale di $O(n^3 \log p)$ operazioni per ogni p.

Algoritmo GGI

Problema

Input: Un intero positivo p e una matrice $A \in \mathbb{R}^{n \times n}$ non-singolare.
Output: Una matrice X, radice p-esima primaria di A.

Algoritmo GGI

Problema

Input: Un intero positivo p e una matrice $A \in \mathbb{R}^{n\times n}$ non-singolare.
Output: Una matrice X, radice p-esima primaria di A.

1. Si determina attraverso il metodo di Schur una matrice (quasi)-triangolare superiore $T \in \mathbb{R}^{n\times n}$ tale che $T = Q^*AQ$, per qualche $Q \in \mathbb{R}^{n\times n}$ ortogonale;
2. Si determina una radice p-esima primaria U dell'equazione $U^p = T$ attraverso il calcolo di al più $2\lfloor \log_2 p \rfloor$ matrici intermedie;
3. La soluzione cercata è $X = QUQ^*$

ALGORITMO GGI; SCHEMA PUNTO 2

STEP 1-2

Sia l il numero dei blocchi diagonali di T. Per $q = 1, \ldots, p-1$, si calcolano e si memorizzano i blocchi diagonali

$$U_{jj}^{q} = (U_{jj})^{q}, \qquad \text{per } j = 1, \ldots, l.$$

ALGORITMO GGI; SCHEMA PUNTO 2

STEP 1-2

Sia l il numero dei blocchi diagonali di T. Per $q = 1, \ldots, p-1$, si calcolano e si memorizzano i blocchi diagonali

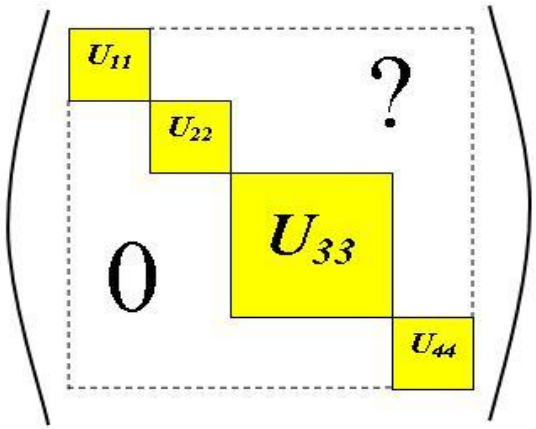
$$U_{jj}^{q} = (U_{jj})^{q}, \qquad \text{per } j = 1, \ldots, l.$$

STEP 3

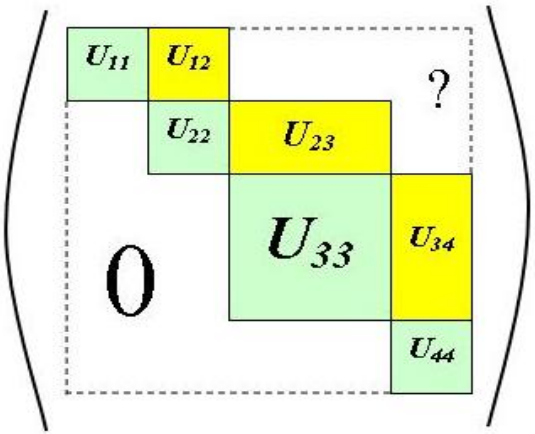
Si calcolano e si memorizzano i blocchi della prima sopradiagonale di U e di $O(\log_2 p)$ matrici intermedie che ora definiremo.

Si ripete la stessa operazione per la $(j-i)$-esima sopradiagonale, per $(j-i) = 2, \ldots, l-1$.

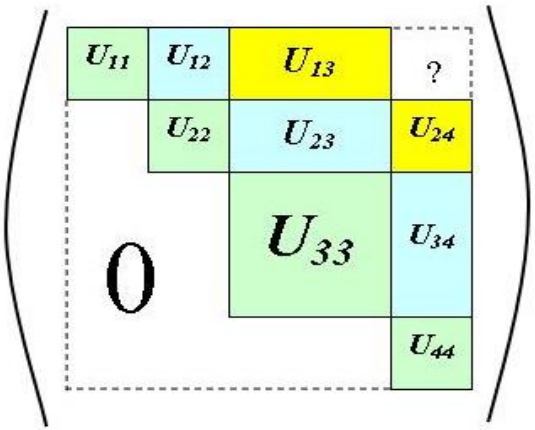
ALGORITMO GGI; SCHEMA PUNTO 2



STEP 1-2



STEP 3 per $(j-i)=1$



STEP 3 per $(j-i)=2$

Decomposizione di p in base 2

p lo possiamo scrivere in modo unico come $\sum_{h=0}^{m} 2^{c_h}$, per

- $c_0 = \lfloor \log_2 p \rfloor$
- $m \geq 0$
- $c_1, \ldots, c_m$ oppotuni interi tali che $\lfloor \log_2 p \rfloor > c_1 > \ldots > c_m \geq 0$.

Decomposizione di p in base 2

p lo possiamo scrivere in modo unico come $\sum_{h=0}^{m} 2^{c_h}$, per

- $c_0 = \lfloor \log_2 p \rfloor$
- $m \geq 0$
- $c_1, \ldots, c_m$ oppotuni interi tali che $\lfloor \log_2 p \rfloor > c_1 > \ldots > c_m \geq 0$.

Esempi

- $8 = 2^3$. In questo caso $m = 0$ e $c_0 = 3$.

Decomposizione di p in base 2

p lo possiamo scrivere in modo unico come $\sum_{h=0}^{m} 2^{c_h}$, per

- $c_0 = \lfloor \log_2 p \rfloor$
- $m \geq 0$
- $c_1, \ldots, c_m$ oppotuni interi tali che $\lfloor \log_2 p \rfloor > c_1 > \ldots > c_m \geq 0$.

Esempi

- $8 = 2^3$. In questo caso $m = 0$ e $c_0 = 3$.
- $11 = 8 + 2 + 1 = 2^3 + 2^1 + 2^0$.
  In questo caso $m = 2$, $c_0 = 3$, $c_1 = 1$, e $c_2 = 0$.

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

Consideriamo dapprima il caso $p = 2^{c_0}$.
Definiamo ricorsivamente le matrici intermedie:

$$\begin{cases} V^{(0)} = U \\ V^{(k)} = V^{(k-1)} \cdot V^{(k-1)} = U^{2^k} \end{cases} \qquad \text{per } k = 1, \ldots, c_0.$$

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

Consideriamo dapprima il caso $p = 2^{c_0}$.
Definiamo ricorsivamente le matrici intermedie:

$$\begin{cases} V^{(0)} = U \\ V^{(k)} = V^{(k-1)} \cdot V^{(k-1)} = U^{2^k} \end{cases} \qquad \text{per } k = 1, \ldots, c_0.$$

- Le matrici sono $c_0 = \lfloor \log_2 p \rfloor$, invece delle 2^{c_0} usate da Smith[2003];
- $V^{(c_0)} = U^p = T$ è la matrice nota in partenza;
- Le matrici $V^{(k)}$, per $k = 0, , \ldots, c_0 - 1$, devono essere calcolate;
- $V^{(0)} = U$ è la matrice cercata.

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

Caso $p = 8$

In questo caso $c_0 = 3$. Le matrici intermedie sono le seguenti:

$$V^{(0)} = U$$
$$V^{(1)} = V^{(0)} \cdot V^{(0)} = U^2$$
$$V^{(2)} = V^{(1)} \cdot V^{(1)} = U^4$$
$$V^{(3)} = V^{(2)} \cdot V^{(2)} = U^8 = T$$

Sono sufficienti 3 matrici interemdie invece delle 7 usate da Smith[2003].

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

[GGI, 2009]: Consideriamo la $(j-i)$-esima sopradiagonale. Il blocco $V_{ij}^{(0)} = U_{ij}$ si determina risolvendo l'equazione di Sylvester:

$$\sum_{q=0}^{p-1} U_{ii}^{q} V_{ij}^{(0)} U_{jj}^{p-1-q} = T_{ij} - \sum_{(r;s;t)\in A_{c_0}} U_{ii}^{r} B_{ij}^{(s)} U_{jj}^{t}.$$

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

[GGI, 2009]: Consideriamo la $(j-i)$-esima sopradiagonale. Il blocco $V_{ij}^{(0)} = U_{ij}$ si determina risolvendo l'equazione di Sylvester:

$$\sum_{q=0}^{p-1} U_{ii}^{q} V_{ij}^{(0)} U_{jj}^{p-1-q} = T_{ij} - \sum_{(r;s;t)\in A_{c_0}} U_{ii}^{r} B_{ij}^{(s)} U_{jj}^{t}.$$

- $B_{ij}^{(s)} = \sum_{\xi=i+1}^{j-1} V_{i\xi}^{(s-1)} V_{\xi j}^{(s-1)}$ è funzione di blocchi appartenenti a sopradiagonali precedentemente calcolate.

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

[GGI, 2009]: Consideriamo la $(j-i)$-esima sopradiagonale. Il blocco $V_{ij}^{(0)} = U_{ij}$ si determina risolvendo l'equazione di Sylvester:

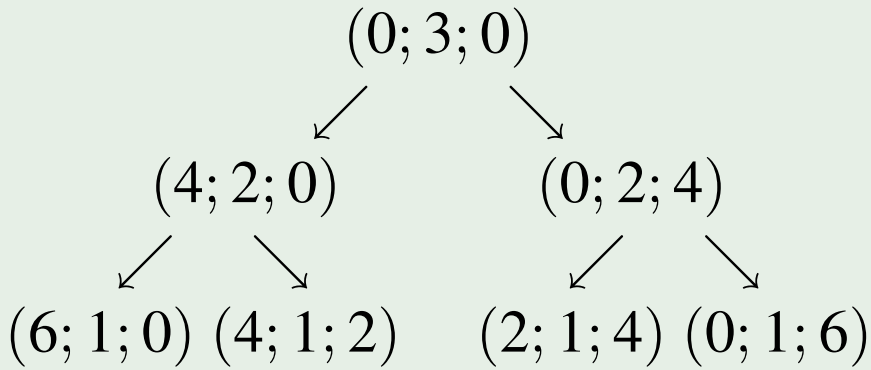
$$\sum_{q=0}^{p-1} U_{ii}^{q} V_{ij}^{(0)} U_{jj}^{p-1-q} = T_{ij} - \sum_{(r;s;t)\in A_{c_0}} U_{ii}^{r} B_{ij}^{(s)} U_{jj}^{t}.$$

- $B_{ij}^{(s)} = \sum_{\xi=i+1}^{j-1} V_{i\xi}^{(s-1)} V_{\xi j}^{(s-1)}$ è funzione di blocchi appartenenti a sopradiagonali precedentemente calcolate.
- A_{c_0} è un insieme di $2^{c_0} - 1$ terne più facile da visualizzare che da descrivere.

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

Caso $p = 8$; $(c_0 = 3)$

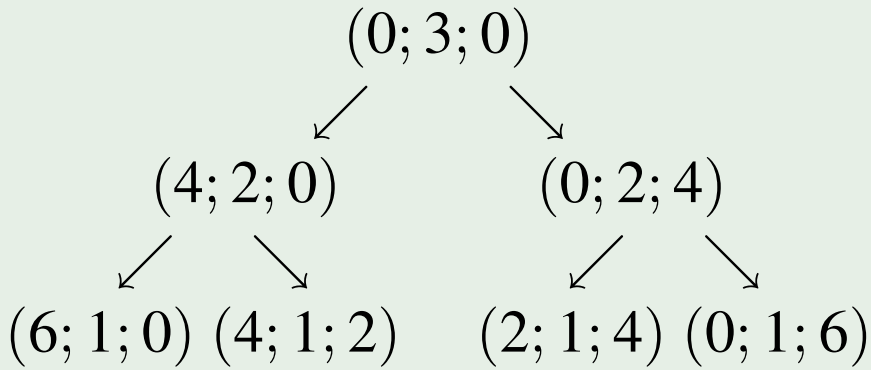
L'insieme A_3 è fatto così: (NB: $r + 2^s + t = 2^3, \quad \forall (r,s,t) \in A_3$)



Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = 2^{c_0}$

Caso $p = 8$; $(c_0 = 3)$

L'insieme A_3 è fatto così: (NB: $r + 2^s + t = 2^3, \quad \forall (r,s,t) \in A_3$)



REGOLE per A_{c_0}:

- $(0; c_0; 0) \in A_{c_0}$;
- Se $(r; s; t) \in A_{c_0}$ e $s > 1$, allora $(r + 2^{s-1}; s - 1; t) \in A_{c_0}$ e $(r; s - 1; t + 2^{s-1}) \in A_{c_0}$.

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = \sum 2^{c_h}$

Passiamo al caso $p = \sum_{h=0}^{m} 2^{c_h}$.

Definiamo ricorsivamente le matrici intermedie:

$$\begin{cases} V^{(0)} = U \\ V^{(k)} = V^{(k-1)} \cdot V^{(k-1)} = U^{2^k} \end{cases} \qquad \text{per } k = 1, \ldots, c_0.$$

$$\begin{cases} W^{(0)} = V^{(c_0)} \\ W^{(h)} = W^{(h-1)} \cdot V^{(c_h)} = U^{p-2^{c_h+1}-\ldots-2^{c_m}} \end{cases} \qquad \text{per } h = 1, \ldots, m.$$

$V^{(0)}, \ldots, V^{(c_0)} = U^{2^{c_0}}$ sono come nel caso 2^{c_0}.
Successivamente si moltiplica $W^{(0)} = V^{(c_0)}$ per $V^{(c_1)}, \ldots, V^{(c_m)}$.

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = \sum 2^{c_h}$

$$\begin{cases} V^{(0)} = U \\ V^{(k)} = V^{(k-1)} \cdot V^{(k-1)} = U^{2^k} \end{cases} \qquad \text{per } k = 1, \ldots, c_0.$$

$$\begin{cases} W^{(0)} = V^{(c_0)} \\ W^{(h)} = W^{(h-1)} \cdot V^{(c_h)} = U^{p-2^{c_h+1}-\ldots-2^{c_m}} \end{cases} \qquad \text{per } h = 1, \ldots, m.$$

- Le matrici sono $c_0 + m = O(\log_2 p)$, invece delle p usate da Smith[2003];
- $W^{(m)} = U^p = T$ è la matrice nota in partenza;
- Tutte le $V^{(k)}$ etutte le $W^{(h)}$ devono essere calcolate;
- $V^{(0)} = U$ è la matrice cercata.

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = \sum 2^{c_h}$

Caso $p = 11$

In questo caso $m = 2$, $c_0 = 3$, $c_1 = 1$, e $c_2 = 0$. Segue che

$$
\begin{aligned}
V^{(0)} &= U \\
V^{(1)} &= V^{(0)} \cdot V^{(0)} = U^2 \\
V^{(2)} &= V^{(1)} \cdot V^{(1)} = U^4 \\
V^{(3)} &= V^{(2)} \cdot V^{(2)} = U^8 = W^{(0)} \\
W^{(1)} &= W^{(0)} \cdot V^{(c_1)} = V^{(3)} \cdot V^{(1)} = U^{10} \\
W^{(2)} &= W^{(1)} \cdot V^{(c_2)} = W^{(1)} \cdot V^{(0)} = U^{11} = T
\end{aligned}
$$

Sono sufficienti 5 matrici interemdie invece delle 10 usate da Smith[2003].

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = \sum 2^{c_h}$

[GGI, 2009]: Consideriamo la $(j-i)$-esima sopradiagonale. Il blocco $V_{ij}^{(0)} = U_{ij}$ si determina risolvendo l'equazione di Sylvester:

$$\sum_{q=0}^{p-1} U_{ii}^{q} V_{ij}^{(0)} U_{jj}^{p-1-q} = T_{ij} - \sum_{h=1}^{m} C_{ij}^{(h)} U_{jj}^{p-2^{c_0}-\ldots-2^{c_h}} +$$

$$- \sum_{\{h:c_h \neq 0\}} U_{ii}^{p-2^{c_h}-\ldots-2^{c_m}} \left[\sum_{(r;s;t)\in A_{c_h}} U_{ii}^{r} B_{ij}^{(s)} U_{jj}^{t} \right] U_{jj}^{2^{c_h+1}+\ldots+2^{c_m}}.$$

Calcolo del blocco $V_{ij}^{(0)} = U_{ij}$; caso $p = \sum 2^{c_h}$

[GGI, 2009]: Consideriamo la $(j-i)$-esima sopradiagonale. Il blocco $V_{ij}^{(0)} = U_{ij}$ si determina risolvendo l'equazione di Sylvester:

$$\sum_{q=0}^{p-1} U_{ii}^{q} V_{ij}^{(0)} U_{jj}^{p-1-q} = T_{ij} - \sum_{h=1}^{m} C_{ij}^{(h)} U_{jj}^{p-2^{c_0}-\ldots-2^{c_h}} + \\ - \sum_{\{h:c_h \neq 0\}} U_{ii}^{p-2^{c_h}-\ldots-2^{c_m}} \left[\sum_{(r;s;t)\in A_{c_h}} U_{ii}^{r} B_{ij}^{(s)} U_{jj}^{t} \right] U_{jj}^{2^{c_h+1}+\ldots+2^{c_m}}.$$

- Gli insiemi A_{c_h} ed i blocchi $B_{ij}^{(s)}$ già li conosciamo.
- Anche $C_{ij}^{(h)} = \sum_{\xi=i+1}^{j-1} W_{i\xi}^{(h-1)} V_{\xi j}^{(c_h)}$ è funzione di blocchi appartenenti a sopradiagonali precedentemente calcolate.

ALGORITMO GGI: COME AVVIENE LO STEP 3

Si determinano i blocchi $V_{ij}^{(0)} = U_{ij}$ appartenenti alla prima sopradiagonale risolvendo l'equazione di Sylvester.

ALGORITMO GGI: COME AVVIENE LO STEP 3

Si determinano i blocchi $V_{ij}^{(0)} = U_{ij}$ appartenenti alla prima sopradiagonale risolvendo l'equazione di Sylvester.
Si determinano $V_{ij}^{(k)}$, per $k = 1, \ldots, c_0$ e $W_{ij}^{(h)}$, per $h = 1, \ldots, c_{m-1}$. Per esempio, posto $p' = p - 2^{c_{h+1}} - \ldots - 2^{c_m}$,

$$W_{ij}^{(h)} = \sum_{q=0}^{p'-1} U_{ii}^{q} V_{ij}^{(0)} U_{jj}^{p'-1-q} + \sum_{h'=1}^{h} C_{ij}^{(h')} U_{jj}^{p'-2^{c_0}-\ldots-2^{c_{h'}}} +$$

$$+ \sum_{\{h' : c_{h'} \neq 0\}} U_{ii}^{p'-2^{c_{h'}}-\ldots-2^{c_h}} \left[\sum_{(r;s;t) \in A_{c_{h'}}} U_{ii}^{r} B_{ij}^{(s)} U_{jj}^{t} \right] U_{jj}^{2^{c_{h'+1}}+\ldots+2^{c_h}}.$$

Poi si passa alla sopradiagonale successiva.

Osservazioni sul calcolo di $V_{ij}^{(0)} = U_{ij}$

Nell'equazione di Sylvester

$$\sum_{q=0}^{p-1} U_{ii}^{q} V_{ij}^{(0)} U_{jj}^{p-1-q} = T_{ij} - \sum_{h=1}^{m} C_{ij}^{(h)} U_{jj}^{p-2^{c_0}-\ldots-2^{c_h}} +$$

$$- \sum_{\{h:c_h \neq 0\}} U_{ii}^{p-2^{c_h}-\ldots-2^{c_m}} \left[\sum_{(r;s;t)\in A_{c_h}} U_{ii}^{r} B_{ij}^{(s)} U_{jj}^{t} \right] U_{jj}^{2^{c_h+1}+\ldots+2^{c_m}},$$

il termine noto ha $O(p)$ addendi perché la prima sommatoria ha m addendi e la seconda sommatoria ha al più $p-m-1$ addendi

$$\sum_{\{h:c_h\neq 0\}} |A_{c_h}| \leq \sum_{h=0}^{m} |A_{c_h}| = \sum_{h=0}^{m} (2^{c_h}-1) = \sum_{h=0}^{m} 2^{c_h} - (m+1) = p-m-1.$$

ALGORITMO GGI: COSTO

STEP	Operazione eseguita	ops	mem
1-2	Calcolo e memorizzazione degli U_{jj}^{q}	$O(pn)$	$O(pn)$
3	Calcolo dei $B_{ij}^{(k)}$ e dei $C_{ij}^{(h)}$	$O(n^3 \log_2 p)$	
3	Calcolo dei p termini e risoluzione delle $O(n^2)$ equazioni di Sylvester	$O(pn^2)$	
3	Calcolo e memorizzazione dei $V_{ij}^{(k)}$ e dei $W_{ij}^{(h)}$	$O(n^2 \log_2 p)$	$O(n^2 \log_2 p)$

ALGORITMO GGI: COSTO

STEP	Operazione eseguita	ops	mem
1-2	Calcolo e memorizzazione degli U_{jj}^{q}	$O(pn)$	$O(pn)$
3	Calcolo dei $B_{ij}^{(k)}$ e dei $C_{ij}^{(h)}$	$O(n^3 \log_2 p)$	
3	Calcolo dei p termini e risoluzione delle $O(n^2)$ equazioni di Sylvester	$O(pn^2)$	
3	Calcolo e memorizzazione dei $V_{ij}^{(k)}$ e dei $W_{ij}^{(h)}$	$O(n^2 \log_2 p)$	$O(n^2 \log_2 p)$

COSTO TOTALE ALGORITMO:

PUNTI	ops	memoria
1 e 3	$O(n^3)$	$O(n^2)$
2	$O(n^3 \log_2 p + pn^2)$	$O(n^2 \log_2 p + pn)$

COSTO: CONFRONTO CON GLI ALTRI ALGORITMI

Algoritmo	**Stabilità all'indietro**	**Operazioni**	**Memoria**
Schur Smith	$\surd$	$O(pn^3)$	$O(pn^2)$
Schur GGI	?	$O(n^3 \log_2 p + pn^2)$	$O(n^2 \log_2 p + pn)$
Iterazione	?	$O(n^3 \log p)$	$O(n^2)$

GGI vs. Smith: sperimentazioni al volo

MATRICI TEST: A matrice 4×4 con 3 blocchi diagonali
B matrice 10×10 con 6 blocchi diagonali

Matrice	p	GGI		Smith	
		tempo	residuo	tempo	residuo
A	73	0.1710	$2.70 \cdot 10^{-13}$	0.3750	$2.70 \cdot 10^{-13}$
A	277	0.2500	$1.02 \cdot 10^{-12}$	3.3590	$1.02 \cdot 10^{-12}$
A	1009	0.6250	$3.99 \cdot 10^{-12}$	42.700	$3.99 \cdot 10^{-12}$
B	73	0.2810	$7.48 \cdot 10^{-13}$	1.4370	$7.51 \cdot 10^{-13}$

PROSSIMO PASSO: Controllo della Stabilità all'indietro.

Questa non è l'ultima slide