Lezione 25

In questa lezione si conclude la descrizione della procedura per la ricerca di una fattorizzazione QR di una matrice e si discute un procedimento che utilizza tale procedura per la ricerca della soluzione di un sistema di equazioni lineari. Infine si approfondisce la nozione di *costo* di una procedura e lo si utilizza per confrontare i due procedimenti per la ricerca della soluzione di un sistema di equazioni lineari introdotti. Contestualmente si descrive un modello più realistico dell'insieme dei numeri di macchina. Le parti di testo in magenta sono asserti *omessi in classe*.

Riprendiamo la descrizione del procedimento di ricerca di una *fattorizzazione QR* di un'assegnata matrice iniziato nella Lezione scorsa.

- *Esempio* (continua).
  - *Secondo passo*
    Siano Ω e Θ le matrici determinate dal *Primo passo* e:
    $$\Delta = \mathrm{diag}(\| \omega_1 \|, \| \omega_2 \|, \| \omega_3 \|) \in \mathbb{R}^{3\times 3}$$
    Si ricordi che $\omega_1 \neq 0$ e $\omega_2 \neq 0$. *Se* anche $\omega_3 \neq 0$ allora Δ è invertibile e la coppia:
    $$U = \Omega\,\Delta^{-1} \quad , \quad T = \Delta\,\Theta$$
    è una fattorizzazione QR di A.

La procedura seguente formalizza il procedimento descritto:

- $[U, T] = \mathrm{GS}\,(A)$

  // A matrice $n \times n$ ad elementi reali.

  $\omega_1 = a_1$;

  `per` $k = 1, \ldots, n-1$ `ripeti`:
  `se` $\omega_k = 0$ `allora`: interrompi la costruzione;
  `altrimenti`:
  $d_k = \omega_k \cdot \omega_k$
  `per` $j = k+1, \ldots, n$ `ripeti`: $\theta_{kj} = \dfrac{a_j \cdot \omega_k}{d_k}$;
  $\omega_{k+1} = a_{k+1} - (\omega_1\theta_{1,k+1} + \cdots + \omega_k\theta_{k,k+1})$;

  `se` $\omega_n = 0$ `allora`: interrompi la costruzione;
  `altrimenti`:
  $d_n = \omega_n \cdot \omega_n$;
  $\Delta = \mathrm{diag}(\sqrt{d_1}, \ldots, \sqrt{d_n})$;
  $$U = (\omega_1, \ldots, \omega_n)\,\Delta^{-1}; T = \Delta \begin{bmatrix} 1 & & & \\ & \ddots & \theta_{ij} & \\ & 0 & \ddots & \\ & & & 1 \end{bmatrix}$$

  La procedura GS termina correttamente *se e solo se* applicata ad una matrice *invertibile*.

- *Osservazione*

  (1) La procedura GS (che realizza il *metodo di ortonormalizzazione di Gram-Schmidt*, da cui il nome) mostra che *qualunque matrice invertibile ammette fattorizzazione QR*. Si osservi che se U, T è una fattorizzazione QR di A ed $E \in \mathbb{R}^{n\times n}$ è una matrice diagonale tale che $|\, e_{11}\,| = 1, \ldots, |\, e_{nn}\,| = 1$ allora la coppia:
  $$U' = UE \quad , \quad T' = E\,T$$
  è a sua volta una fattorizzazione QR di A. Dunque: *la fattorizzazione QR non è unica.*

(2) Esistono altre procedure per la ricerca di una fattorizzazione QR di una matrice, più generali di GS e preferibili ad essa da un punto di vista numerico. Queste procedure terminano correttamente *in ogni caso* e dunque mostrano che *qualunque matrice $n \times n$ ammette fattorizzazione QR*. *Scilab* ha una funzione predefinita per la ricerca di una fattorizzazione QR di una matrice che utilizza il *metodo di Householder*:[1]

* `qr`

  Questa *funzione predefinita* restituisce *una fattorizzazione QR* di un'assegnata matrice A. Precisamente, se `A` è una matrice $n \times n$:

  ```
  [U,T] = qr(A)
  ```

  restituisce la coppia `U`,`T` di matrici $n \times n$ che *approssima* una fattorizzazione QR di `A`. Come già osservato `A` *può* non essere invertibile (vedere l'Esercizio 4).

- *Procedimento di ricerca della soluzione di un sistema di equazioni lineari*

  Siano $A \in \mathbb{R}^{n\times n}$ e $b \in \mathbb{R}^n$. Per verificare l'invertibilità di A e cercare la soluzione del sistema $Ax = b$ si utilizza il procedimento seguente:

  (1) Si applica ad A una procedura di calcolo di una fattorizzazione QR;

  (2) Dette U, T le matrici generate dalla procedura:

  (2a) Si calcola $c = U^{\mathsf{T}}b$ soluzione del sistema $Ux = b$;

  (2b) *Se* T non invertibile *allora* il procedimento si arresta *altrimenti* si determina la soluzione x^* del sistema $Tx = c$ utilizzando la procedura di sostituzione all'indietro.

  Il procedimento è *soddisfacente*: *Se* A non è invertibile *allora* la matrice T è non invertibile ed il procedimento si arresta prematuramente, *altrimenti* il vettore finale x^* è la soluzione del sistema dato.

- *Uso del calcolatore*

  Utilizzando il calcolatore (che, si ricordi, opera in $F(\beta, m)$), il procedimento si modifica come segue: (0) I dati A e b del sistema si trasformano in $\hat{A}$ e $\hat{b}$, matrice e colonna di elementi $\mathrm{rd}(a_{ij}), \mathrm{rd}(b_i), i, j = 1, \ldots, n$; (1) Si applica ad $\hat{A}$ la procedura `qr`; (2) Dette $\hat{U}, \hat{T}$ le matrici determinate al passo precedente: si calcola il vettore $\hat{c} = \hat{U}^{\mathsf{T}} * \hat{b}$; (3) Se $\hat{T}$ invertibile: si applica la procedura `SI`, realizzazione di SI, alla coppia $\hat{T}, \hat{c}$ per il calcolo del vettore $\hat{x}$. Il vettore finale è utilizzato per approssimare la soluzione x^* del sistema $Ax = b$.

  Anche in questo caso per giudicare l'accuratezza di $\hat{x}$ come approssimazione di x^* occorre studiare il *condizionamento* del calcolo della soluzione del sistema $Tx = c$, ovvero indagare il *numero di condizionamento* della matrice T. Scelta in $\mathbb{R}^n$ la norma euclidea N_2, si studia:

  $$c_2(T) = \| T^{-1} \|_2 \, \| T \|_2$$

  Si ha:

  $$A = UT \quad \Rightarrow \quad T = U^{\mathsf{T}}A \qquad \text{e} \qquad A^{-1} = T^{-1}U^{\mathsf{T}} \quad \Rightarrow \quad T^{-1} = A^{-1}U$$

  Dunque:

  $$\| T \|_2 \leqslant \| U^{\mathsf{T}} \|_2 \, \| A \|_2 \quad \text{e} \quad \| T^{-1} \|_2 \leqslant \| A^{-1} \|_2 \, \| U \|_2$$

  Ma: per ogni matrice M ortogonale si ha $\| M \|_2 = 1$. Perciò:

  $$\| T \|_2 \leqslant \| A \|_2 \quad \text{e} \quad \| T^{-1} \|_2 \leqslant \| A^{-1} \|_2 \quad \Rightarrow \quad c_2(T) \leqslant c_2(A)$$

  In questo caso *la procedura di fattorizzazione QR produce un fattore destro con proprietà di condizionamento non peggiori di quelle della matrice iniziale.* Il procedimento di ricerca della soluzione del sistema di equazioni lineari esposto sopra è soddisfacente *anche* quando realizzato al calcolatore.

[1] Si veda, ad esempio: `https://en.wikipedia.org/wiki/QR_decomposition#Using_Householder_reflections`.

(G) Costo

La nozione di *costo*, già implicitamente adottata in precedenza, è quella "aritmetica:"

- *Definizione* (costo aritmetico).

  Sia ϕ un algoritmo, ovvero una sequenza *finita* di composizioni di *funzioni predefinite*. Si chiama *costo aritmetico* di ϕ *il numero* $C(\phi)$ *di pseudo-operazioni aritmetiche eseguite per calcolare* ϕ.[2]

Sia ϕ un algoritmo. Il costo di ϕ deve essere una quantità almeno approssimativamente proporzionale al *tempo* necessario al calcolatore per portare a termine il calcolo. La definizione adottata riesce nell'intento se:

(a) La maggior parte del tempo impiegato dal calcolatore per calcolare ϕ è speso nell'esecuzione di pseudo-operazioni aritmetiche.

(b) Il tempo necessario per eseguire ciascuna pseudo-operazione aritmetica è lo stesso, in particolare *non dipende dagli operandi*.

Il sussistere della condizione (a) *dipende da* ϕ. Ad esempio, se ϕ è la realizzazione di una procedura che calcola la norma infinito di un vettore il costo aritmetico è *zero* – per questo algoritmo *nessuna* delle funzioni predefinite calcolate è una pseudo-operazione aritmetica – ma non è vero che il calcolatore impiega tempo zero a calcolare ϕ. Nel seguito la definizione di costo sarà applicata solo ad algoritmi che calcolano *quasi esclusivamente* pseudo-operazioni aritmetiche.

La condizione (b), invece, *non dipende da* ϕ e secondo il modello di calcolatore presentato finora *non può essere verificata*. Si consideri, ad esempio, il calcolo in $F(2, 53)$ dello pseudo-prodotto $2^{b_1} \otimes 2^{b_2} = 2^{b_1+b_2}$. *Non è ragionevole* supporre che il tempo necessario per il calcolo sia indipendente da quali numeri interi b_1 e b_2 occorre sommare (calcolare la somma si due numeri interi *non può* richiedere un tempo *indipendente* dal numero di cifre necessario per rappresentare gli addendi – si pensi al solo tempo necessario per *leggere gli addendi e scrivere la somma*). In un modello *più realistico* di calcolatore, però, l'insieme dei numeri di macchina è il seguente:

- *Definizione* (numeri in virgola mobile con esponente limitato ed elementi denormalizzati)

  Siano β un numero intero maggiore di uno, m un numero intero positivo, b_{min} e b_{max} numeri interi tali che $b_{min} < b_{max}$. Si indica con:

  $$F(\beta, m, b_{min}, b_{max})$$

  l'insieme di tutti i numeri reali x tali che:

  $$x = (-1)^s \beta^b \, 0.c_1 \cdots c_m$$

  con $s \in \{0, 1\}$, b numero intero tale che $b_{min} \leqslant b \leqslant b_{max}$ e $c_1, \ldots, c_m$ cifre in base β *aritrarie*. Gli elementi dell'insieme che si ottengono scegliendo $c_1 \neq 0$ si dicono *normalizzati*. Gli elementi non nulli che si ottengono scegliendo $b = b_{min}$ e $c_1 = 0$ di dicono *denormalizzati*. $F(\beta, m, b_{min}, b_{max})$ è un insieme di *numeri in virgola mobile con esponente limitato ed elementi denormalizzati*.

Adottando questo modello, l'ipotesi (b) è verificata. Nell'Esercitazione vedremo come l'insieme dei numeri di macchina in *Scilab* sia fedelmente descritto dal nuovo modello.

Vediamo alcuni esempi di determinazione del costo di una procedura e poi confrontiamo i due procedimenti proposti per la ricerca della soluzione di un sistema di equazioni lineari.

- *Esempio*

  - *Prodotto riga per colonna*

    Sia $\texttt{prc}_n$ la realizzazione del *prodotto riga per colonna* $a^{\mathsf{T}}b$, con $a, b \in \mathbb{R}^n$, ottenuta sostituendo ciascuna operazione aritmetica con la corrispondente pseudo-operazione e specificando l'ordine di composizione delle pseudo-operazioni. La corrispondenza biunivoca

[2] Il costo va valutato *nel caso peggiore*: il numero di pseudo-operazioni aritmetiche eseguite per calcolare $\phi(x)$ potrebbe dipendere da x.

tra operazioni aritmetiche e pseudo-operazioni così stabilita rende possibile determinare il costo di $\texttt{prc}_n(a,b)$ calcolando il numero di operazioni aritmetiche in $a^\mathsf{T}b$. Dette a_i, b_i le componenti di a, b si ha:

$$a^\mathsf{T}b = a_1b_1 + \cdots + a_nb_n$$

Il calcolo di $a^\mathsf{T}b$ richiede n prodotti e $n-1$ somme. Allora:

$$C(\texttt{prc}_n) = 2n - 1$$

– *Prodotto matrice per colonna*

Sia $\texttt{pmc}_n$ la realizzazione del *prodotto matrice per colonna* Ab, con $A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n$. Si ragiona come nel caso del prodotto riga per colonna, ovvero si calcola il numero di operazioni aritmetiche in Ab. Dette $r_1, \ldots, r_n$ le righe di A si ha:

$$Ab = \begin{bmatrix} r_1b \\ \vdots \\ r_nb \end{bmatrix}$$

Ciascuna delle n componenti del vettore Ab è un prodotto riga per colonna. Il calcolo di Ab richiede quindi n^2 prodotti e $n(n-1)$ somme. Allora:

$$C(\texttt{pmc}_n) = n\,C(\texttt{prc}_n) = 2n^2 - n$$

– *Prodotto matrice triangolare per colonna*

Sia $\texttt{pmtc}_n$ la realizzazione del *prodotto matrice triangolare per colonna* Tc, con $T \in \mathbb{R}^{n\times n}$ matrice triangolare, $c \in \mathbb{R}^n$. Si ragiona come nel caso del prodotto matrice per colonna. Dette $r_1, \ldots, r_n$ le righe di T si ha:

$$Tc = \begin{bmatrix} r_1c \\ \vdots \\ r_nc \end{bmatrix}$$

Ciascuna delle n componenti del vettore Tc è un prodotto riga per colonna. Questa volta, però, ciascuna componente ha un costo diverso. Supponendo, ad esempio, T triangolare superiore ed evitando di calcolare operazioni con risultato noto a priori (per ogni $x \in \mathbb{R}$ si ha $0\,x = 0$ e $0 + x = x$) si ottiene:

$$C(\texttt{pmtc}_n) = C(\texttt{prc}_n) + C(\texttt{prc}_{n-1}) + \cdots + C(\texttt{prc}_1) = 2(1 + 2 + \cdots + n) - n = n^2$$

Il costo di $\texttt{pmtc}_n$ è circa *metà* del costo di $\texttt{pmc}_n$.

– *Procedura* SI

Come già sappiamo (Lezione 17) il calcolo di $\mathrm{SI}(T, c)$ per $T \in \mathbb{R}^{n\times n}$ e $c \in \mathbb{R}^n$ richiede n divisioni e $\frac{1}{2}\,n(n-1)$ prodotti e somme. Allora:

$$C(\texttt{SI}) = n^2$$

Il costo è uguale a quello di $\texttt{pmtc}_n$. Gli stessi risultati si ottengono per `SA`.

– *Procedura* EGPP

Il calcolo di $\texttt{EGPP}(A)$ per $A \in \mathbb{R}^{n\times n}$ risulta richiedere:

$$\sum_{k=1}^{n-1} k = \tfrac{1}{2}\,n(n-1) \text{ divisioni e } \sum_{k=1}^{n-1} k^2 = \tfrac{1}{6}\,(n-1)n(2n-1) \text{ prodotti e somme}$$

In totale:

$$C(\texttt{EGPP}) = \tfrac{2}{3}\,n^3 - \tfrac{1}{2}\,n^2 - \tfrac{1}{6}\,n$$

Si osservi che il calcolo di $\texttt{EGPP}(A)$ richiede anche *confronti*, in numero trascurabile rispetto alle pseudo-operazioni aritmetiche.

Il procedimento per per la ricerca della soluzione di un sistema di equazioni lineari che utilizza la fattorizzazione LR ha dunque costo:

$$C(\texttt{EGPP}) + C(\texttt{SA}) + C(\texttt{SI}) = \tfrac{2}{3}\, n^3 + \text{ termini di grado inferiore in } n$$

Un calcolo analogo per il procedimento per per la ricerca della soluzione di un sistema di equazioni lineari che utilizza la fattorizzazione QR porta ad un costo:[3]

$$C(\texttt{qr}) + C(\texttt{pmc}_n) + C(\texttt{SI}) = \tfrac{4}{3}\, n^3 + \text{ termini di grado inferiore in } n$$

che è circa *il doppio* del precedente. Si osservi che si è scelto di esprimere il costo dei procedimenti mostrando esplicitamente solo *il termine dominante* al crescere di n. In entrambi i casi *il termine dominante è generato dalla procedura di ricerca della fattorizzazione.*

- *Osservazione* (calcolo della matrice inversa)

  La funzione predefinita `inv` di *Scilab* determina un'approssimazione della matrice inversa di una data matrice invertibile. Sia $A \in \mathbb{R}^{n\times n}$ una matrice invertibile. Dette $e_1, \ldots, e_n$ le colonne della matrice identica, il calcolo di $Y = \texttt{inv}(A)$ avviene, sostanzialmente, in questo modo:

  - $[S, D, P] = \mathrm{EGPP}(A)$
  - `per` $k = 1, \ldots, n$ `ripeti`:
    $c = \mathrm{SA}(S, Pe_k)$;
    $y_k = \mathrm{SI}(D, c)$;
  - $Y = (y_1 \ldots, y_n)$.

  Dunque si calcolano le n colonne della matrice inversa come soluzione degli n sistemi lineari $Ay = e_1, \ldots, Ay = e_n$. Tutti i sistemi hanno *la stessa matrice* e per la soluzione degli n sistemi è sufficiente calcolare *una sola volta* la fattorizzazione di A. Questo fa sì che il termine dominante del costo del procedimento sia ancora un multiplo di n^3.

Esercizi

---

1. Dimostrare che la coppia U, T definita nel *Secondo passo* è una fattorizzazione QR di A.
2. Sia $A \in \mathbb{R}^{3\times 3}$. Dimostrare che se $\mathrm{GS}(A)$ termina prematuramente allora A non è invertibile.
3. Sia:
   $$A = \begin{bmatrix} 0 & 2 \\ 1 & -1 \end{bmatrix}$$
   Determinare $\mathrm{GS}(A)$ e dedurre dal risultato due diverse fattorizzazioni QR di A.
4. Sia $A \in \mathbb{R}^{n\times n}$ la matrice nulla. Per $n = 5$ verificare che la procedura GS applicata ad A termina prematuramente mentre $\texttt{qr}(A)$ determina una fattorizzazione QR *esatta*.
5. Dimostrare, utilizzando la definizione di norma indotta, che: *se M è una matrice ortogonale allora* $\| M \|_2 = 1$.
6. Verificare che il calcolo della matrice H_k nel passo k-esimo di $\mathrm{EGPP}(A)$ richiede $n-k$ divisioni e che il calcolo del prodotto $H_k P_k A^{(k)}$ richiede $(n-k)^2$ prodotti e somme.
7. Verificare che il calcolo di $\mathrm{EGPP}(A)$ richiede non più di $\frac{1}{2}\, n(n-1)$ confronti. Determinare l'errore relativo commesso approssimando il numero di pseudo-operazioni aritmetiche e confronti richiesto dal calcolo di $\mathrm{EGPP}(A)$ con il numero delle sole pseudo-operazioni aritmetiche.
8. Sia $\texttt{ata}_n$ la realizzazione del prodotto $A^{\mathsf{T}}A$ con $A \in \mathbb{R}^{n\times n}$. Determinare $C(\texttt{ata}_n)$. Si tenga conto che la matrice $A^{\mathsf{T}}A$ è *simmetrica*.

---

[3] Si veda la pagina di Wikipedia citata in precedenza. Si osservi che il calcolo della fattorizzazione QR richiede anche $n-1$ radici quadrate.