

Esercitazione 7:

Fattorizzazione QR e Minimi quadrati.

Richiami di Teoria

Teorema: Data una matrice $A \in \mathbb{R}^{m \times n}$, ($m \geq n$), esiste una matrice ortogonale $Q \in \mathbb{R}^{m \times m}$ ($QQ^T = Q^TQ = I$) tale che

$$A = QR = Q \begin{pmatrix} R_1 \\ 0 \end{pmatrix},$$

dove $R \in \mathbb{R}^{m \times n}$ con $r_{i,j} = 0$, $i > j$ ($R_1 \in \mathbb{R}^{n \times n}$ è triangolare superiore).

Osservazione 0: La matrice Q è ortogonale e questo significa che la sua inversa coincide con la trasposta, cioè $Q^{-1} = Q^T$.

Osservazione 1: Se il rango di A è n (massimo), allora R_1 è non singolare.

Osservazione 2: La fattorizzazione QR è anche usata per risolvere sistemi lineari sovra- e sotto-determinati nel senso dei minimi quadrati (argomento non ancora trattato). Ma può essere usata anche per sistemi quadrati. Infatti si può scrivere

$$A = QR \implies Ax = QRx = b \iff \begin{cases} Qy = b \Rightarrow y = Q^T b \\ Rx = y \end{cases}$$

Osservazione 3: La routine MatLab che implementa la fattorizzazione QR è QR.

Esercizio 1

Fattorizzazione QR di una matrice.

Risolvere il sistema lineare $Ax = b$ mediante l'uso della fattorizzazione QR, dove

$$A = \begin{bmatrix} 4 & 1 & 0 & 1 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 1 & 2 & 1 & 4 \end{bmatrix}$$

e b è scelto in modo tale che il vettore soluzione x sia il vettore di tutti 1.

Richiami di Teoria

Riflettore elementare di Householder.

Definiamo un tipo speciale di matrice ortogonale e simmetrica H ($H^T = H^{-1} = H$) detta *matrice di riflessione di Householder*.

Sia $u \in \mathbb{R}^n$ un vettore normalizzato ($\|u\|_2 = 1$), allora è possibile definire la seguente matrice

$$H = I - 2uu^T$$

che risulta essere simmetrica ed ortogonale.

Interpretazione geometrica

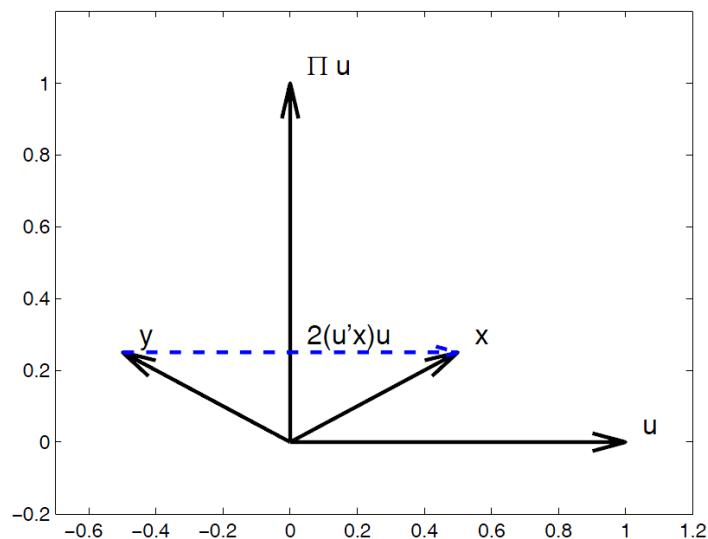
Moltiplicare una matrice di riflessione per un vettore è equivalente a riflettere tale vettore attraverso l'iperpiano perpendicolare ad u .

Sia Π l'iperpiano ortogonale ad u , sia x un vettore ed $y = Hx$ il vettore trasformato, ovvero si ha che

$$y = Hx = (I - 2uu^T)x = x - 2(u^T x)u$$

dove $u^T x$ è la componente di x nella direzione di u e $x - y = 2(u^T x)u$ per la regola del parallelogramma.

Sia veda la figura sottostante per un'interpretazione grafica nello spazio $\mathbb{R}^2$ con $u = e_1 = (1, 0)^T$ e $x = (1/2, 1/4)^T$.



Riflessione lungo il primo vettore canonico

Se $\mathbf{x} \in \mathbb{R}^n$ è un vettore non nullo, allora esiste sempre una matrice di riflessione di Householder $H = I - 2\mathbf{u}\mathbf{u}^T$ ed un numero reale α tali che

$$H\mathbf{x} = \alpha\mathbf{e}_1$$

dove $\mathbf{e}_1 = (1, 0, \dots, 0)^T$ è il primo vettore della base canonica di $\mathbb{R}^n$.
Il vettore $\mathbf{u}$ ed il valore α sono ottenuti come :

$$\alpha = \text{sign}(x_1)\|\mathbf{x}\|_2$$

$$\mathbf{v} = \mathbf{x} + \alpha\mathbf{e}_1$$

$$\mathbf{u} = \frac{\mathbf{v}}{\|\mathbf{v}\|_2}$$

$$H = I - 2\frac{\mathbf{v}\mathbf{v}^T}{\mathbf{v}^T\mathbf{v}} = I - 2\frac{\mathbf{v}\mathbf{v}^T}{\|\mathbf{v}\|_2^2} = I - 2\mathbf{u}\mathbf{u}^T$$

con x_1 prima componente di $\mathbf{x}$.

Esercizio 2

Scrivere una routine MatLab per calcolare il riflettore elementare di Householder.
L'algoritmo riceve in ingresso il vettore $\mathbf{x}$ e restituisce in uscita la matrice H .

```

H ← HRiflettore ( x )
1  α ← sign(x1)||x||2
2  v ← x + αe1
3  u ←  $\frac{\mathbf{v}}{\|\mathbf{v}\|_2}$ 
4  H ← I - 2uuT

```

Nota: Attenzione alla funzione `sign` di MatLab che restituisce 0 se in ingresso viene passato uno zero. $\mathbf{e}_1$ può essere generato come `eye(n,1)`.

Generare la matrice di riflessione H relativa alla prima colonna della matrice

$$A = \begin{pmatrix} 3 & -2 & 3 \\ 12 & 0 & 1 \\ -4 & 5 & 2 \end{pmatrix}$$

e calcolare HA ottenendo come risultato

$$HA = \begin{pmatrix} -13 & 2 & -1 \\ 0 & 3 & -2 \\ 0 & 4 & 3 \end{pmatrix}.$$

Fattorizzazione QR di una matrice A mediante uso dei riflettori di Householder.

Una matrice A di ordine $m \times n$ viene trasformata in una matrice triangolare superiore mediante l'applicazione successiva di trasformazioni elementari di Householder.

Ad esempio, per $m = 6$ e $n = 5$ supponiamo di aver già calcolato due matrici di trasformazione H_1 e H_2 tali che:

$$H_2 H_1 A = \begin{pmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & x & * & * \\ 0 & 0 & x & * & * \\ 0 & 0 & x & * & * \\ 0 & 0 & x & * & * \end{pmatrix}$$

ed allora cerchiamo una matrice di Householder $\tilde{H}_3$ di ordine 4×4 tale che

$$\tilde{H}_3 \begin{pmatrix} x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} y \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Allora poniamo quindi $H_3 = \text{diag}(I_2, \tilde{H}_3)$, ottenendo:

$$H_3 H_2 H_1 A = \begin{pmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & y & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & * & * \end{pmatrix}$$

Alla fine $H_n H_{n-1} \cdots H_1 A = R$ è una matrice triangolare superiore e ponendo $Q = H_1 \cdots H_n$ si ottiene $A = QR$.

Esercizio 3

Scrivere una routine MatLab per calcolare la fattorizzazione QR di una matrice generica A mediante l'uso dei riflettori elementari di Householder.

L'algoritmo riceve in ingresso la matrice A e restituisce in uscita le matrici Q ed R .

```
[Q, R] ← QRwithH ( A )
1  [m,n] ← size ( A )
2  // inizializzazione di R uguale ad A
3  R ← A
4  // inizializzazione di Q pari a identita' di dimensione m
5  Q ← Im
6  // calcolo quante matrici di Householder devo calcolare :
7  // minimo tra m-1 e n
8  d ← min(m - 1, n)
9  // ciclo sulle possibili matrici elementari
10 for k ← 1 to d
11     // estraggo vettore x della trasformazione elementare dalla matrice R
12     x ← (rk,k, rk+1,k, . . . , rm,k)T
13     // costruisco matrice elementare di Householder
14      $\tilde{P} \leftarrow \text{HRiflettore} ( x )$ 
15     // costruisco matrice H
16     H ← Im
17     Hk...m,k...m ←  $\tilde{H}$ 
18     // aggiorno R
19     R ← H · R
20     // aggiorno Q
21     Q ← Q · H
22 end
```

Come esempio si consideri la seguente matrice:

$$A = \begin{pmatrix} 1 & -1 & 4 \\ 1 & 4 & -2 \\ 1 & 4 & 2 \\ 1 & -1 & 0 \end{pmatrix}$$

Risultati

A =

1	-1	4
1	4	-2
1	4	2
1	-1	0

Q =

-0.5000	0.5000	-0.5000	-0.5000
-0.5000	-0.5000	0.5000	-0.5000
-0.5000	-0.5000	-0.5000	0.5000
-0.5000	0.5000	0.5000	0.5000

R =

-2.0000	-3.0000	-2.0000
-0.0000	-5.0000	2.0000
0.0000	0.0000	-4.0000
0.0000	-0.0000	0.0000

Richiami di Teoria

Minimi quadrati (retta di regressione).

Siano dati m punti (x_i, y_i) , $1 \leq i \leq m$.

Sia $p(x) = a_0 + a_1x$ il polinomio (lineare) di grado $n = 1$ con cui vogliamo approssimare i dati. Il problema dei minimi quadrati consiste nel rendere minima la seguente quantità (scarti al quadrato):

$$S = d(p, a_0, a_1) = \sum_{i=1}^m (p(x_i) - y_i)^2 = \sum_{i=1}^m (a_0 + a_1x_i - y_i)^2$$

La quantità S è la somma dei quadrati degli scarti verticali e risulta essere anche una funzione derivabile. Il minimo della funzione S è un punto di stazionarietà. Valgono perciò le seguenti condizioni (necessarie e sufficienti) dette *equazioni normali*:

$$\frac{\partial S}{\partial a_r} = 0 \quad r = 0, \dots, n$$

cioè

$$\frac{\partial S}{\partial a_r} = 2 \sum_{i=1}^m (a_0 + a_1x_i)x_i^r = 0 \quad r = 0, \dots, n$$

perciò avremo

$$\sum_{i=1}^m (a_0 + a_1x_i)x_i^r = \sum_{i=1}^m y_i x_i^r \quad r = 0, \dots, n$$

ed in forma matriciale

$$\begin{pmatrix} m & \sum_{i=1}^m x_i \\ \sum_{i=1}^m x_i & \sum_{i=1}^m x_i^2 \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^m y_i \\ \sum_{i=1}^m y_i x_i \end{pmatrix}.$$

Il sistema precedente può essere riscritto nella forma

$$A^T A a = A^T y$$

definendo

$$A = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_m \end{pmatrix}, \quad a = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} \quad \text{e} \quad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}.$$

Con queste definizioni il problema originario può essere formulato come segue:

$$\min_{a \in \mathbb{R}^2} \|Aa - y\|_2^2.$$

Nel caso generale di un'approssimazione mediante un polinomio della forma

$$p(x) = a_0 + a_1x + a_2x^2 + \dots + a_nx^n$$

la matrice A assume la forma

$$A = \begin{pmatrix} 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & x_m^2 & \dots & x_m^n \end{pmatrix}$$

e viene detta matrice di Vandermonde; Tale matrice risulta essere mal condizionata.

Soluzione del sistema delle equazioni normali¹

Utilizzando Cholesky

Nel caso generale $n > 1$, il sistema delle equazioni normali $A^T A \mathbf{x} = A^T \mathbf{b}$ con $A \in \mathbb{R}^{m \times n}$, essendo simmetrico e definito positivo (se rango di A massimo cioè punti x_i distinti) si può risolvere tramite la fattorizzazione di Cholesky della matrice $A^T A = L^T L$, da cui

$$A^T A \mathbf{x} = A^T \mathbf{b} \Leftrightarrow L(L^T \mathbf{x}) = A^T \mathbf{b} \Leftrightarrow \begin{cases} L \mathbf{y} = A^T \mathbf{b} & \Rightarrow \text{sost. avanti} \\ L^T \mathbf{x} = \mathbf{y} & \Rightarrow \text{sost. indietro} \end{cases}$$

In aritmetica finita la matrice $A^T A$ può essere malcondizionata (ed eventualmente con numero di condizionamento elevato) in quanto gli inevitabili errori di arrotondamento possono portare a matrici le cui colonne sono linearmente dipendenti. Quindi è preferibile utilizzare per la risoluzione numerica o la fattorizzazione QR o la fattorizzazione SVD.

Utilizzando QR

Un metodo che evita questo inconveniente perchè usa direttamente la matrice A è offerto dalla fattorizzazione QR , da cui

$$\|A\mathbf{x} - \mathbf{b}\|_2 = \|QR\mathbf{x} - \mathbf{b}\|_2 = \|Q(R\mathbf{x} - Q^T \mathbf{b})\|_2 = \|R\mathbf{x} - Q^T \mathbf{b}\|_2.$$

La matrice R ha la forma

$$R = \begin{pmatrix} R_1 \\ 0 \end{pmatrix} \begin{matrix} n \text{ righe} \\ m - n \text{ righe} \end{matrix}$$

con R_1 una matrice triangolare superiore non singolare in quanto A ha rango massimo. Ponendo $\mathbf{c} = Q^T \mathbf{b}$ e partizionandolo coerentemente con la partizione di R , ossia

$$\mathbf{c} = \begin{pmatrix} \mathbf{c}_1 \\ \mathbf{c}_2 \end{pmatrix} \begin{matrix} n \text{ componenti} \\ m - n \text{ componenti} \end{matrix}$$

¹Sistema con un numero maggiore di righe rispetto alle incognite quindi si tratta di un sistema sovra-determinato.

abbiamo

$$R\mathbf{x} - \mathbf{c} = \begin{pmatrix} R_1\mathbf{x} - \mathbf{c}_1 \\ -\mathbf{c}_2 \end{pmatrix} \begin{matrix} n \text{ componenti} \\ m - n \text{ componenti} \end{matrix}.$$

Quindi il problema di minimo diventa

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}\|_2^2 &= \min_{x \in \mathbb{R}^n} \|R\mathbf{x} - \mathbf{c}\|_2^2 = \min_{x \in \mathbb{R}^n} [\|R_1\mathbf{x} - \mathbf{c}_1\|_2^2 + \|\mathbf{c}_2\|_2^2] \\ &= \|\mathbf{c}_2\|_2^2 + \min_{x \in \mathbb{R}^n} \|R_1\mathbf{x} - \mathbf{c}_1\|_2^2. \end{aligned}$$

Poiché R_1 è non singolare, la soluzione x^* del sistema lineare

$$R_1\mathbf{x} = \mathbf{c}_1$$

è tale che

$$\min_{x \in \mathbb{R}^n} \|R_1\mathbf{x} - \mathbf{c}_1\|_2^2 = \|R_1\mathbf{x}^* - \mathbf{c}_1\|_2^2 = 0.$$

Ne segue che x^* è soluzione del problema di minimo ed inoltre

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 = \|\mathbf{c}_2\|_2^2.$$

Osservazione: Se la matrice A non ha rango massimo, la matrice R_1 ottenuta ha almeno un elemento diagonale nullo e quindi non è possibile calcolare la soluzione del sistema lineare $R_1\mathbf{x} = \mathbf{c}_1$. Questa difficoltà viene superata applicando il metodo QR con la tecnica del massimo pivot per colonne (QR con pivoting) e associare alla variabile il cui elemento è zero (o vicino a zero) il valore zero.

Utilizzando SVD

Teorema: Data una matrice $A \in \mathbb{R}^{m \times n}$, esiste una fattorizzazione della forma

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

con

- U una matrice unitaria di dimensioni $m \times m$;
- Σ una matrice diagonale di dimensioni $m \times n$ che contiene i valori singolari $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_p$ con $p = \min(m, n)$ della matrice A (i valori singolari σ_i e gli autovalori di λ_i di $A^T A$, ordinati in modo decrescente, sono legati dalla seguente relazione $\sigma_i = \sqrt{\lambda_i}$);
- V una matrice unitaria di dimensioni $n \times n$;
- il comando MatLab che esegue la SVD è `svd`.

Utilizzando la SVD abbiamo

$$\|A\mathbf{x} - \mathbf{b}\|_2 = \|U\Sigma V^T\mathbf{x} - \mathbf{b}\|_2 = \|U(\Sigma V^T\mathbf{x} - U^T\mathbf{b})\|_2 =$$

per cui ponendo $y = V^T x$ e $d = U^T b$ si ha

$$\|U(\Sigma V^T\mathbf{x} - U^T\mathbf{b})\|_2 = \|\Sigma\mathbf{y} - \mathbf{d}\|_2.$$

La matrice Σ è della forma

$$\Sigma = \begin{pmatrix} \text{diag}(\sigma_i) \\ 0 \end{pmatrix}$$

per cui partizionando d_i nella forma

$$d = \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}$$

coerentemente con Σ si ha:

$$\min_{x \in \mathbb{R}^n} \|A\mathbf{x} - \mathbf{b}\|_2^2 = \|\Sigma\mathbf{y} - \mathbf{d}\|_2^2 = \|\mathbf{d}_2\|_2^2 + \min_{y \in \mathbb{R}^n} \|\text{diag}(\sigma_i)\mathbf{y} - \mathbf{d}_1\|_2^2$$

da cui $y_i = d_i/\sigma_i$ e quindi $x = Vy$.

Esercizio 4

Siano dati le seguenti coppie di valori.

x_i	-5	-3	1	3	4	6	8
$y_i = f(x_i)$	18	7	0	7	16	50	67

Si vuole calcolare la retta di regressione utilizzando :

1. la soluzione delle equazioni normali attraverso Cholesky;
2. mediante fattorizzazione QR della matrice;
3. mediante fattorizzazione SVD della matrice;
4. il comando `\` di MatLab o Octave.

La soluzione procede nel seguente modo:

- Scrivere una routine che costruisca la matrice A della retta di regressione ai minimi quadrati.

L'algoritmo riceve in ingresso il vettore x e restituisce in uscita la matrice A .

$A \leftarrow \text{CostruisciA}(x)$

1 $n \leftarrow \text{length}(x)$

2 *// affianco vettore di tutti 1 il vettore x per colonna*

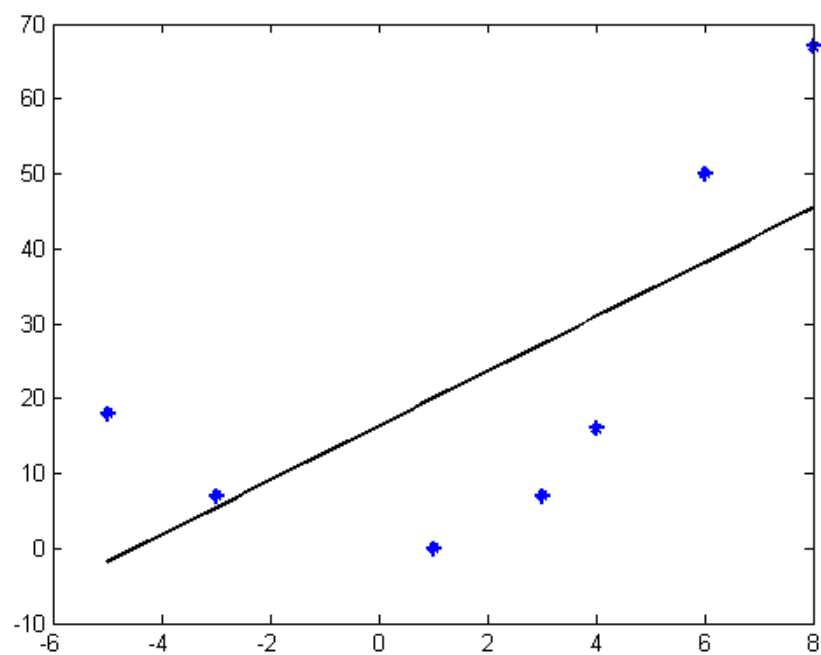
3 $A \leftarrow [1_n | x]$

- (Risoluzione delle equazioni normali) Dato $K = A^T A$ e $F = A^T y$ si deve risolvere il sistema lineare 2×2 $Ka = F$ sfruttando la fattorizzazione di Cholesky di K .

Attenzione: La routine MatLab `chol` restituisce una matrice triangolare superiore, i.e. L^T .

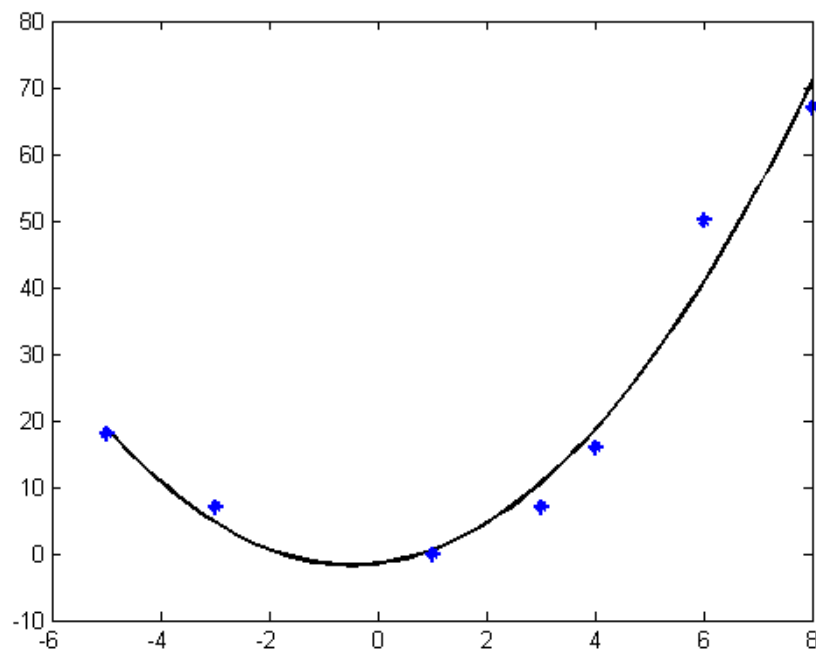
- (Risoluzione mediante QR) Dato la fattorizzazione QR di A si deve risolvere il sistema $R_1 a = c_1$ di dimensione 2×2 , precisamente
- (Risoluzione mediante SVD) Data la fattorizzazione SVD di A si deve calcolare $d = U^T y$, $\hat{y}_i = d_i / \sigma_i$ ed infine $x = V \hat{y}$.
- (Risoluzione mediante comando MatLab) si deve risolvere il sistema $Aa = y$.
- Visualizzare i risultati ottenuti (anche graficamente).

```
1 //calcola fattorizzazione QR
2  $[Q, R] \leftarrow \text{QRwithH} ( A )$ 
3  $\mathbf{c} \leftarrow Q^T \cdot y$ 
4  $\mathbf{c}_1 \leftarrow (c_1, \dots, c_n)^T$ 
5  $R_1 \leftarrow R_{1 \dots n, 1 \dots n}$ 
6 //calcolo soluzione di  $R_1 \mathbf{x} = \mathbf{c}_1$  con sostituzione all'indietro
7  $\mathbf{a} \leftarrow \text{RisolviSistemaTriSup} ( R_1, \mathbf{c}_1 )$ 
```



Esercizio 5

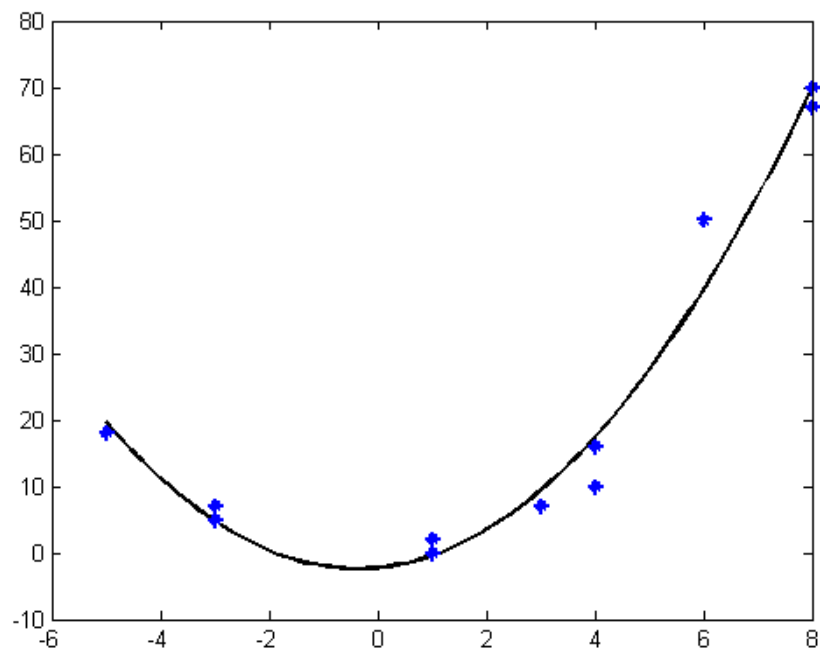
Ripetere l'esercizio precedente calcolando la parabola (polinomio di grado 2) che meglio approssima i dati. In questo caso la matrice A è della forma $[1_n | x | x.^2]$ con $n = 3$ parametri.



Esercizio 6

Ripetere l'esercizio precedente sul seguente insieme di dati (anche con x ripetute)

x_i	-5	-3	-3	1	1	3	4	4	6	8	8
$y_i = f(x_i)$	18	7	5	0	2	7	16	10	50	67	70



Esercizio 7²

L'equazione differenziale

$$\frac{dN}{dt} = -\lambda N.$$

la cui soluzione analitica è

$$N(t) = N_0 e^{-\lambda t}.$$

viene utilizzata per modellare fenomeni in cui una quantità (con N_0 quantità iniziale) decade e questo decadimento è proporzionale (con valore costante λ) alla quantità rimasta.

Tipicamente dalle misure di laboratorio si hanno i valori di t_i e $N(t_i)$ cioè la quantità rimasta oppure t_i e $N(t_i)/N_0$ [%] cioè la percentuale di riduzione della quantità dopo un certo intervallo di tempo.

Lo scopo dell'approssimazione, in questo caso, è trovare i valore λ e N_0 che meglio descrivono (nel senso dei minimi quadrati) il fenomeno fisico.

Per ricondurre il problema della ricerca dei parametri N_0 e λ di un modello non lineare (in questo caso esponenziale) possiamo considerare, in questo particolare caso, la versione in cui viene applicato il logaritmo, cioè

$$\log(N(t)) = \log(N_0) - \lambda t$$

che diventa un problema di interpolazione polinomiale quando i dati y_i vengono trasformati mediante il logaritmo.

Quindi per risolvere il problema eseguiamo i seguenti passi:

- definiamo il nostro modello polinomiale attraverso $a_0 = \log(N_0)$ e $a_1 = -\lambda$;
- eseguiamo il logaritmo dei dati, cioè $\log(y_i)$;
- calcoliamo il polinomio (retta) di miglior approssimazione con il comando `polyfit`;
- calcoliamo N_0 e λ dai coefficienti del polinomio.

Un esempio di fenomeno fisico che può essere descritto da un decadimento di tipo esponenziale è l'andamento dell'altezza della schiuma della birra³ dove la legge di decadimento esponenziale può essere derivata dall'ipotesi che la variazione della schiuma sia proporzionale al volume della schiuma presente mediante una relazione del tipo $dV = -(V/\tau)dt = -\lambda dt$ con $\lambda = \tau$.

Nella tabella seguente sono riportati i valori medi relativi alla birra "Budweiser Budvar"

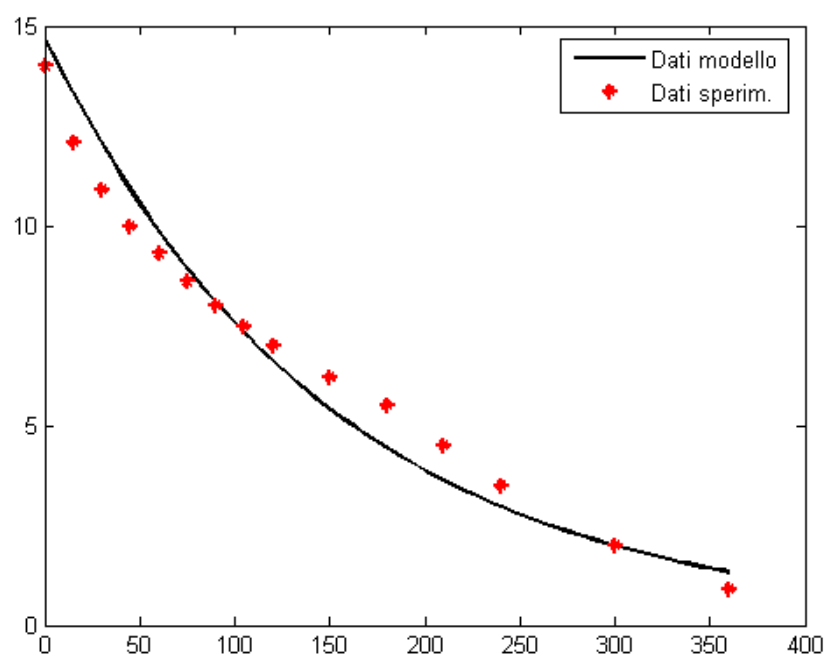
$t_i[s]$	0	15	30	45	60	75	90	105	120	150	180	210	240	300	360
$h_i[cm]$	14.0	12.1	10.9	10.0	9.3	8.6	8.0	7.5	7.0	6.2	5.5	4.5	3.5	2.0	0.9

dove è riportato a diversi istanti temporali t_i [s] il valore dell'altezza della schiuma h_i in [cm]. Calcolare il valore di λ e N_0 .

²Facoltativo.

³Per maggiori dettagli si veda "Demonstration of the exponential decay law using beer froth", di A. Leike, in European Journal of Physics, 23 (2002) pag. 2126.

Risultati



Differenze tra soluzione con QR e SVD

Nel risolvere il sistema delle equazioni normali nel caso in cui la matrice A ha rango massimo sia la fattorizzazione QR che la SVD portano ai medesimi risultati. In questo caso si preferisce utilizzare la fattorizzazione SVD in quanto è quella numericamente più stabile.

Invece, la differenza tra QR e SVD si può apprezzare quando si richiede di risolvere, nel senso dei minimi quadrati, un sistema sotto-determinato, cioè un sistema con un numero maggiore di incognite rispetto al numero di equazioni. Questo significa che ci sono più soluzioni che producono il minimo.

Sia nel caso della QR che nella SVD si fissano a zero le variabili che non compaiono nella soluzione del sistema (si noti che nella fattorizzazione non si fissano le x a zero ma, nella nostra notazione le y) questo si traduce nel seguente fatto: la fattorizzazione QR realizza la soluzione con più zeri, cioè quella più sparsa mentre la fattorizzazione SVD si può dimostrare che realizza la soluzione x quella di norma minima ($\|x\|_2^2$) cioè quella più vicina al vettore nullo.

Ad esempio se consideriamo il seguente sistema lineare

$$\begin{cases} x_1 + 2x_2 + x_3 + x_4 = 2 \\ x_1 - 2x_2 + x_3 - x_4 = 0 \end{cases}$$

ed utilizzando le seguenti istruzioni MatLab

```
A = [1 2 1 1; 1 -2 1 -1];
b = [2 0]'
```

```
% QR con pivoting e versione economica
```

```
% -----
[Q,R,E] = qr(A,0)
c = Q'*b
x_qr = zeros(4,1);
x_qr(E(1:2)) = R(1:2,1:2)\c
```

```
% SVD in versione economica
```

```
% -----
[U,S,V] = svd(A,0)
d = U'*b
z = zeros(4,1);
z(1:2) = d./diag(S)
x_svd = V*z
norm(x_qr)^2
norm(x_svd)^2
```

si hanno i seguenti risultati

```
x_qr =
    1.0000
    0.5000
         0
         0
```

```
% con norma 1.2500
x_svd =
    0.5000
    0.4000
    0.5000
    0.2000
% con norma 0.7000
```

Esercizio 8⁴

Un'interessante applicazione della SVD è alla compressione delle immagini.

Data una immagine o matrice A di dimensione $m \times n$ la vogliamo approssimare rimuovendo informazione ridondante o meno significativa. L'idea è di sfruttare la fattorizzazione SVD ed eliminare i valori singolari piccoli; questo può essere spiegato dal fatto che i valori singolari sono ordinati in modo decrescente e valori singolari piccoli contribuiscono "poco" alla formazione della matrice risultante finale.

Matematicamente vogliamo approssimare una matrice A con una matrice $\tilde{A}$ di rango inferiore, diciamo r misurando l'errore in norma di Frobenius cioè $\|A - \tilde{A}\|_F$. Grazie al teorema dimostrato da Eckart-Young (1936) per far questo basta considerare la decomposizione SVD della matrice A e considerare solo i primi r valori singolari. L'errore che si commette è

$$\|A - \tilde{A}\|_F = \sqrt{\sum_{i=r+1}^n \sigma_i^2}.$$

Nella figura sottostante è rappresentata la fattorizzazione SVD in "versione economica" dove sono mostrate solamente le matrici coinvolte nel risultato finale se si considerano solamente un numero limitato di valori singolari.

$$\begin{pmatrix} 0.5176 & 0.9830 & 0.2731 \\ 0.0278 & 0.9837 & 0.6412 \\ 0.1697 & 0.5947 & 0.9756 \\ 0.0110 & 0.8249 & 0.7826 \end{pmatrix} = \begin{pmatrix} -0.4603 & 0.8165 & 0.3340 & 0.0991 \\ -0.5291 & 0.0241 & -0.6147 & -0.5845 \\ -0.4963 & -0.5148 & 0.6505 & -0.2561 \\ -0.5117 & -0.2602 & -0.2957 & 0.7635 \end{pmatrix} \begin{pmatrix} 2.1900 & 0 & 0 & 0 \\ 0 & 0.6502 & 0 & 0 \\ 0 & 0 & 0.3114 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} -0.1566 & 0.5123 & 0.8444 \\ -0.7718 & 0.4700 & -0.4283 \\ -0.6163 & -0.7188 & 0.3218 \end{pmatrix}'$$

A = U Σ V'

Figura 1: Fattorizzazione SVD ridotta

Attenzione che la scelta di r deve essere tale che il costo per la memorizzazione di U , Σ e V ridotte alle prime r componenti sia inferiore al costo di memorizzazione la matrice completa A .

Il seguente codice mostra una bozza di codice MatLab per questa tipologia di applicazioni. L'immagine test per eccellenza in queste applicazioni è quella di lena.

⁴Facoltativo.

```
close all
clear all
clc

[A,map]= imread('lena.gif');           % caricamento immagine
B = im2double(A, 'indexed');           % conversione in numeri reali

figure;
imshow(B,map);                         % visualizzazione

[U,S,V]=svd(B);                       % decomposizione ai valori singolari

k = 128;                               % numero di valori singolari che vogliamo tenere
C = zeros(size(B));
for j=1:k
    C = C + S(j,j)*U(:,j)*V(:,j)';
end

figure;
imshow(C,map);                         % visualizza immagine
```

Si chiede di provare il codice per vedere come si comporta l'algoritmo sui primi valori singolari.

Risultati

Nella prima figura vengono mostrati il valore dei singoli valori singolari.

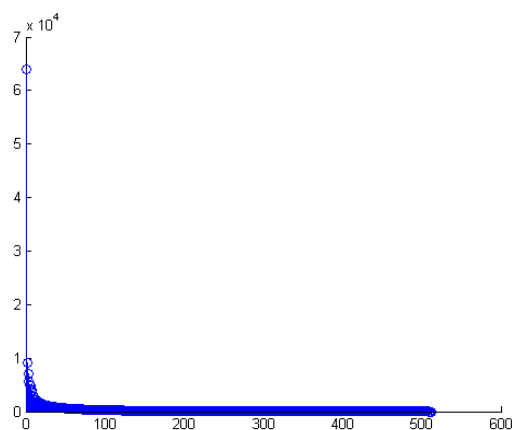


Figura 2: Valore dei valori singolari

Nella figura successiva vengono mostrate (da sinistra verso destra e dall'alto verso il basso) le prime 16 immagini ottenute dai primi 16 valori singolari.



Figura 3: Immagini da 1 a 16 valori singolari



Figura 4: Confronto tra immagine originale (in alto) e l'immagine con i primi 128 valori singolari (in basso).