# Interpolazione
# &
# Approssimazione

Una prima analisi del problema

Dott. Francesca Incensi: francesca.incensi3@unibo.it

## Interpolazione & Approssimazione

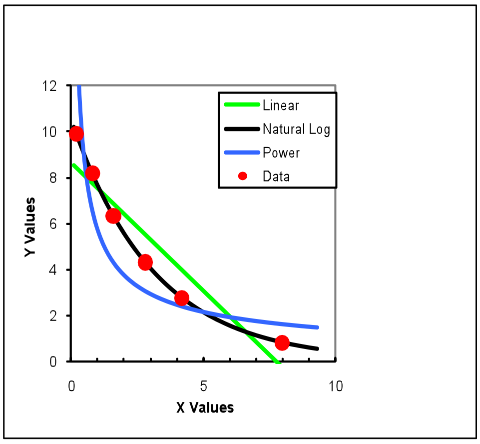
- **Metodi di interpolazione**
  - Interpolazione di Lagrange
  - Interpolazione di Newton
- **Metodi di approssimazione**
  - Approssimazione ai minimi quadrati

## *Interpolare o approssimare?*

- **Interpolazione** - passa per i punti dati.

  In caso di dati sperimentali si deve tener conto di un margine di errore.

- **Approssimazione** – si vuole determinare la curva che approssima i dati con il più piccolo errore.




## La necessità di interpolare o approssimare

Quando si presenta la necessità di approssimare?

Cosa vogliamo approssimare?

- La funzione da approssimare *f(x)* non è nota ma di essa si conoscono alcuni valori $y_i$ su un insieme di punti $x_i$ (caso discreto)

- La funzione è nota in forma analitica ma è una funzione complicata (caso continuo)

4

## Quale strategia usare?

- Pochi dati affidabili → **INTERPOLAZIONE**
- Dati sperimentali affetti da variabilità statistica → **APPROSSIMAZIONE ai MINIMI QUADRATI**
- Dati affetti da errore uniforme → **APPROSSIMAZINE UNIFORME**

Scelta della **DISTANZA** con cui si vuole approssimare (misura dell'errore)

5

## La INTERPOLAZIONE

COSTRUIRE UNA FUNZIONE CHE PASSI PER TUTTI I PUNTI ASSEGNATI.

Dati i punti **$(x_i,y_i)$, i=0,..,n** con

$$x_i \in R^m,\ m \geq 1, \quad x_i \neq x_j,\ per\ i \neq j$$

e una famiglia di funzioni

$$\phi(x;a_0,..,a_n) = y$$ **(modello matematico)**

si cercano i valori **$a_0,..a_n$** (parametri o gradi di libertà) tali che

$$\phi(x_i;a_0,..,a_n) = y_i \quad i = 0,..,n$$

6

# Modello matematico

- **Interpolazione lineare**

$$\phi(x;a_0,..,a_n) = a_0\phi_0(x) + a_1\phi_1(x) + .. + a_n\phi_n(x)$$

- Interpolazione polinomiale

$$\boldsymbol{\phi(x;a_0,..,a_n) = a_0 + a_1x + .. + a_nx^n}$$

- Interpolazione trigonometrica

$$\boldsymbol{\phi(x;a_0,..,a_n) = a_0 + a_1e^{xi} + a_2e^{2xi} .. + a_ne^{nxi}}$$

- **Interpolazione razionale**

$$\phi(x;a_0,..,a_n,b_0,..,b_m) = \frac{a_0 + a_1x + .. + a_nx^n}{b_0 + b_1x + .. + b_mx^m}$$

7

# Interpolazione Polinomiale

Risolubilità del problema di interpolazione mediante polinomi

**TEOREMA**

Dati gli *n+1* punti

$$(x_i, y_i), i = 0,..,n, \quad x_i \neq x_j, per\ i \neq j$$

**esiste** ed è **unico** il polinomio *p(x)* di grado *n* che verifica le condizioni:

$$p(x_i) = y_i, \quad i = 0,..,n$$

8

## Metodi di interpolazione polinomiale

- **Interpolazione di Lagrange:** un modo semplice ma oneroso (nella forma originale) di costruire il polinomio di interpolazione.
- **Interpolazione di Newton:** richiede differenze divise.

I due metodi producono due polinomi equivalenti (esistenza – unicità): la differenza tra i due consiste nell'approccio per ottenere i coefficienti.

9

## Interpolazione di Lagrange

Introduciamo in $\mathbf{P}_n$ un'opportuna base di funzioni:

$$\{L_i(x)\}_{i=0}^{n}$$

Ciascuna risolve un semplice problema di interpolazione

$$L_i(x_j) = \begin{cases} 0 & se \quad j \neq i \\ 1 & se \quad j = i \end{cases}$$

Il problema iniziale è risolto da

$$P_n(x) = L_0(x)y_0 + L_1(x)y_1 + \ldots + L_n(x)y_n =$$

$$= \sum_{i=0}^{n} L_i(x)y_i$$

10

# Interpolazione di Lagrange

Determiniamo le funzioni $\{L_i(x)\}_{i=0}^{n}$

Ciascuna ha grado al più n ed ha n zeri distinti, allora

$$L_i(x) = k(x - x_1)(x - x_2)\ldots(x - x_{i-1})(x - x_{i+1})\ldots(x - x_n) = k\prod_{\substack{j=0\\ j\neq i}}^{n}(x - x_j)$$

Valutiamo *k* considerando che $L_i(x_i) = 1$

$$\boldsymbol{L_i(x_i) = 1 = k\prod_{\substack{j=0\\ j\neq i}}^{n}(x_i - x_j) \quad da\ cui\ k = \frac{1}{\prod_{\substack{j=0\\ j\neq i}}^{n}(x_i - x_j)}}$$

quindi

$$\boxed{\boldsymbol{L_i(x) = \frac{\prod_{\substack{j=0\\ j\neq i}}^{n}(x - x_j)}{\prod_{\substack{j=0\\ j\neq i}}^{n}(x_i - x_j)}, \quad i = 0,.., n}}$$

11

# Esempio

Assegnati i punti:

| x | y |
|---|---|
| 1,1 | 10,6 |
| 1,7 | 15,2 |
| 3 | 20,3 |

Quali sono i coefficienti del polinomio di interpolazione di secondo grado?

$$P_2(x) = L_0(x)y_0 + L_1(x)y_1 + L_2(x)y_2$$

$$L_0(x) = \frac{(x-1.7)(x-3.0)}{(x_0 - x_1)(x_0 - x_2)} = \frac{(x-1.7)(x-3.0)}{(1.1-1.7)(1.1-3.0)} = \frac{1}{1.14}(x^2 - 4.7x + 5.1)$$

$$L_1(x) = \frac{(x-1.1)(x-3.0)}{(1.7-1.1)(1.7-3.0)} = -\frac{1}{0.78}(x^2 - 4.1x + 3.3)$$

$$L_2(x) = \frac{(x-1.1)(x-1.7)}{(3.0-1.1)(3.0-1.7)} = \frac{1}{2.47}(x^2 - 2.8x + 1.87)$$

12

# Esempio

Cosa succede se incrementiamo il numero dei punti da interpolare?

| x | y |
|---|---|
| 1,1 | 10,6 |
| 1,7 | 15,2 |
| 3 | 20,3 |
| 1,4 | 13,4 |
| 2,2 | 18,7 |

Dobbiamo ricalcolare tutti i $L_i(x)$!!

13

# Differenze divise

Assegnati i valori della funzione *f(x)* nei punti $x_0, x_1, .., x_n$ distinti dell'asse reale la differenza divisa di *f(x)* rispetto agli argomenti $x_0, x_1$ è definita da

$$F[x_0, x_1] = \frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{f(x_0) - f(x_1)}{x_0 - x_1} = F[x_1, x_0]$$

**DIFFERENZA DIVISA DI ORDINE 1**

14

# Differenze divise

**DIFFERENZA DIVISA DI ORDINE 2**

$$F[x_0, x_1, x_2] = \frac{F[x_1, x_2] - F[x_0, x_1]}{x_2 - x_0}$$

**DIFFERENZA DIVISA DI ORDINE *n***

$$F[x_0, x_1, ..., x_n] = \frac{F[x_1, ..., x_n] - F[x_0, ..., x_{n-1}]}{x_n - x_0}$$

15

# Schema delle differenze divise

$d_0$ $\quad$ $d_1$ $\quad$ $d_2$

$$\begin{array}{lllll}
x_0 & f(x_0) = F[x_0] & & & \\
x_1 & f(x_1) = F[x_1] \rightarrow & F[x_0, x_1] & & \\
x_2 & f(x_2) = F[x_2] \rightarrow & F[x_1, x_2] & F[x_0, x_1, x_2] & \\
x_3 & f(x_3) = F[x_3] \rightarrow & F[x_2, x_3] & F[x_1, x_2, x_3] & \\
.. & .. & .. & .. & .. \\
x_n & f(x_n) = F[x_n] \rightarrow & F[x_{n-1}, x_n] & F[x_{n-2}, x_{n-1}, x_n] & .. \quad F[x_0, ..., x_n]
\end{array}$$

16

## Differenze divise

Sia $f \in C^n[a,b]$ e siano $x_0, x_1, .., x_n$ punti distinti nell'intervallo [a,b], allora $\exists \xi \in [a,b]$ tale che :

$$F[x_0, ..., x_n] = \frac{f^n(\xi)}{n!}$$

17

## Differenze divise

**TEOREMA**

$$F[x_0, x_1, ..., x_n] = \sum_{j=0}^{n} \frac{f(x_j)}{\prod_{\substack{k=0 \\ k \neq i}}^{n} (x_j - x_k)} \qquad (*)$$

**Il coefficiente principale del polinomio di interpolazione di grado *n* che soddisfa la condizione** $p_n(x_i) = f(x_i) = y_i$ **per le formule di Lagrange assume la forma** $\sum_{j=0}^{n} \frac{f(x_j)}{\prod_{\substack{k=0 \\ k \neq i}}^{n} (x_j - x_k)}$

**coincide con la differenza divisa della funzione *f(x)* di ordine n rispetto agli argomenti $x_0, x_1, .., x_n$ per la (*).**

18

## Interpolazione di Newton

Questo metodo permette di costruire facilmente il polinomio di interpolazione di ***n+1* punti** a partire dal polinomio di interpolazione di un sottoinsieme di ***n* punti**

Il metodo di Newton utilizza le differenze divise $d_i$

$$P(x) = d_0 + d_1(x - x_0) + d_2(x - x_0)(x - x_1) + \dots$$

19

## Interpolazione di Newton

Se $p_n \in P_n$ è il polinomio di interpolazione dei punti

$$(x_i; f(x_i)), \qquad i = 0,\dots,n$$

allora la funzione

$$p_{n+1}(x) = p_n(x) + \prod_{j=0}^{n}(x - x_j)F[x_0,x_1,\dots,x_{n+1}]$$

è un polinomio in $P_{n+1}$ e soddisfa

$$p_{n+1}(x_i) = f(x_i), \quad i = 0,\dots,n+1$$

20

## Interpolazione di Newton

Iterando la formula si ottiene la **forma di Newton**:

$$p_{n+1}(x) = F[x_0] + (x - x_0)F[x_0,x_1] + \ldots +$$
$$+\prod_{j=0}^{n}(x - x_j)F[x_0,x_1,\ldots,x_{n+1}]$$

Polinomio che soddisfa le condizioni di interpolazione nei punti

$$(x_i; f(x_i)), \qquad i = 0,\ldots,n+1$$

21

## Interpolazione di Newton

La formula di Newton può essere scritta nella forma

$$p(x) = d_0 + (x - x_0)[d_1 + (x - x_1)[d_2 + \ldots$$
$$+(x - x_{n-2})[d_{n-1} + (x - x_{n-1})d_n]\ldots]]$$
$$d_0 = F[x_0] = f(x_0)$$
$$d_i = F[x_0,\ldots x_i], \quad i = 1,2,\ldots,n$$

Valutazione del polinomio con algoritmo tipo **Horner**

$$\boxed{\begin{aligned} & p = d(n) \\ & for \quad i = n-1,0,-1 \\ & \qquad p = p*(x - x(i)) + d(i) \\ & end \end{aligned}}$$

22

## Errore di interpolazione

Da cosa dipende l'errore di interpolazione?

- Regolarità della funzione
- Disposizione dei punti di interpolazione sull'asse delle ascisse

Rappresentazione dell'errore

$$E(x) = f(x) - p(x) = \frac{\omega_n(x)}{(n+1)!} f^{(n+1)}(\xi)$$

$$\omega_n(x) = \prod_{j=0}^{n} (x - x_j)$$



## Interpolazione di Newton

- Calcola i coefficienti $d_k$ del polinomio di Newton con l'algoritmo per le differenze divise
- Valuta il polinomio di Newton per ogni punto x nell'intervallo di visualizzazione utilizzando l'algoritmo di Horner opportunamente modificato
- Visualizza il polinomio di Newton e i punti interpolati (plot)

Rispetto alla formula di Lagrange la complessità computazionale si riduce di oltre la metà:

$$\underbrace{\frac{n(n+1)}{2}}_{DIF.DIVISE} + \underbrace{n}_{HORNER} = \frac{n(n+3)}{2}$$

# Il fenomeno di Runge

Non è in generale vero che al crescere del numero dei punti di interpolazione e quindi anche del grado $k$ del polinomio di interpolazione, la successione dei polinomi $p_k$ converga a $f(x)$, per $x_i$ **equidistanti.**

**FUNZIONE DI RUNGE:**

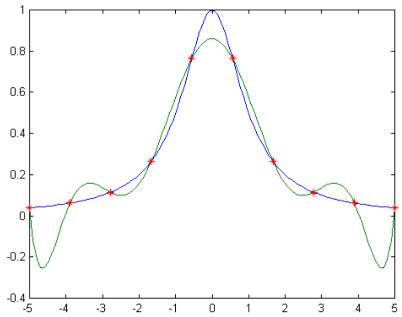
$$f(x) = \frac{1}{1+x^2}, \quad [a,b] = [-5,5]$$

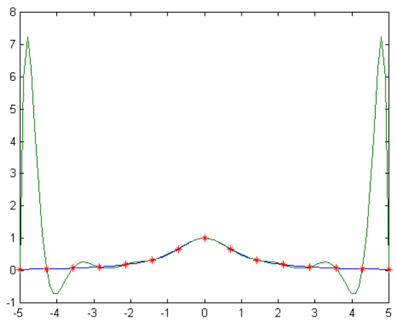
**scegliamo punti di interpolazione EQUIDISTANTI**

25

# Il fenomeno di Runge

$$f(x) = \frac{1}{1+x^2}, \quad [a,b] = [-5,5]$$




Grado 9

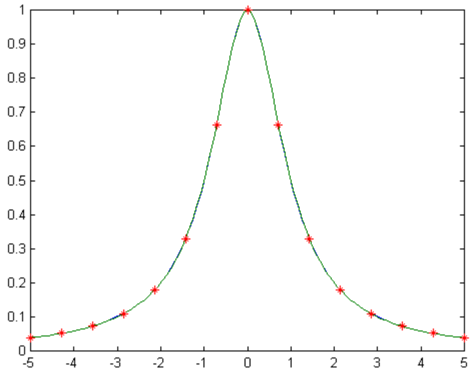
Grado 15

**L'errore aumenta all'estremità dell'intervallo**

26

# Il fenomeno di Runge

$$f(x) = \frac{1}{1+x^2}, \quad [a,b] = [-5,5]$$




Spline di interpolazione di grado 3

27

# Quale *approssimazione* usare?

- Dati sperimentali affetti da variabilità statistica → **APPROSSIMAZIONE ai MINIMI QUADRATI**
- Dati affetti da errore uniforme → **APPROSSIMAZINE UNIFORME**

Scelta della **DISTANZA** con cui si vuole approssimare (misura dell'errore)

28

# Distanza (norma)

Approssimare i punti **($x_i$,$y_i$),i=0,..,m**,con la funzione ***$f_n(x)$***

- **Approssimazione ai minimi quadrati**

  $f_n(x)$ vuole minimizzare la quantità:

$$\|y - f_n\|_2 = \sum_{i=0}^{m} w_i (y_i - f_n(x_i))^2$$

- **Approssimazione uniforme**

  $f_n(x)$ vuole minimizzare la quantità:

$$\|y - f_n\|_\infty = \max_{i=0,..,m} |y_i - f_n(x_i)|$$

# Approssimazione ai minimi quadrati: caso discreto

Assegnati i punti ($x_i$,$y_i$), i=0,..,m, scelte n+1 funzioni base

$$\Phi_0, \Phi_1, .., \Phi_n, \ (n << m)$$

si vuole approssimare i punti tramite la funzione

$$f_n(x) = a_0\Phi_0 + a_1\Phi_1 + .. + a_n\Phi_n,$$

individuata con il criterio dei minimi quadrati, ossia i coefficienti $a_i$ sono determinati in modo che il **residuo**

$$S = \sum_{i=0}^{m} (y_i - f_n(x_i))^2$$ risulti minimo

## Forma matriciale

Si vuole determinare il vettore a che renda minimo il residuo:

$$S = \|Ha - y\|_2^2$$

$$H = \begin{bmatrix} \Phi_0(x_0) & .. & \Phi_n(x_0) \\ .. & .. & .. \\ .. & .. & .. \\ .. & .. & .. \\ \Phi_0(x_m) & .. & \Phi_n(x_m) \end{bmatrix}, \quad y = \begin{bmatrix} y_0 \\ .. \\ y_m \end{bmatrix}$$

## Approssimazione ai minimi quadrati

In forma matriciale

$$\min S = \|Ha - y\|_2^2$$

**Equazioni normali** $\longrightarrow$

$$H^T Ha = H^T y$$

$$H = \begin{bmatrix} \Phi_0(x_0) & .. & \Phi_n(x_0) \\ .. & .. & .. \\ \Phi_0(x_m) & .. & \Phi_n(x_m) \end{bmatrix}, \quad y = \begin{bmatrix} y_0 \\ .. \\ y_m \end{bmatrix}$$

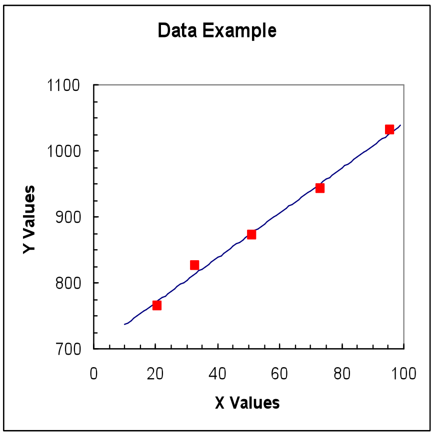
**Attenzione al MALCONDIZIONAMENTO**: $K(H^T H) = K(H)^2$

Il risultato di questo è che errori di arrotondamento nella risoluzione del sistema possono causare grandi errori nell'approssimante dei dati.

# Esempio : lineare (n=1)

Vogliamo approssimare i dati:

| X | Y |
|---|---|
| 20,5 | 765 |
| 32,7 | 826 |
| 51 | 873 |
| 73,2 | 942 |
| 95,7 | 1032 |




Determinare la retta

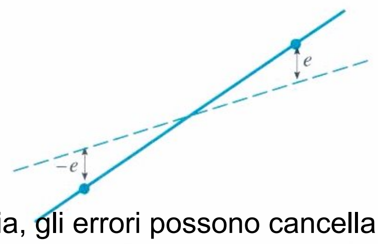
$$f_1(x)=ax+b$$

che ***meglio*** approssima i dati

# Approssimazione ai minimi quadrati

Minimizzare la differenza fra i punti dati e  i punti della funzione approssimante.

L'errore può essere definito come:




$$\boxed{e_k = \left(f_1(x_k) - y_k\right)}$$

Tuttavia, gli errori possono cancellarsi l'un l'altro e ancora avere una pessima approssimazione.

**La soluzione è minimizzare la somma dei quadrati.  Questo porta alla soluzione ai minimi quadrati.**

$$\boxed{S = \sum (e_k)^2}$$

## Approssimazione lineare ai minimi quadrati

Vogliamo determinare una funzione approssimante di primo grado (retta):

$$\boxed{f_1(x_i) = a\,x_i \;+\; b, \quad i = 1,\dots,m}$$

L'errore è definito come:

$$e_i = f(x_i) - y_i$$

$$(x_i, y_i) - \textit{punti}\ \text{dati}$$

$$(x_i, f(x_i)) - \textit{punti}\ \text{approssimati}$$

## Approssimazione lineare ai minimi quadrati

La somma degli errori è data da:

$$\boxed{S = e_1^2 + e_2^2 + e_3^2 + \dots}$$

Sostituiamo l'espressione dell'errore:

$$\boxed{S = \sum_{i=1}^{m} e_i^2 = \sum_{i=1}^{m} \left(y_i - f(x_i)\right)^2 = \sum_{i=1}^{m} \left(y_i - \left(a x_i \;+\; b\right)\right)^2}$$

## Approssimazione lineare ai minimi quadrati

Come minimizziamo l'errore?

**Minimo di una funzione in due variabili (a,b):**

calcoliamo la derivata parziale di S rispetto ad entrambe le variabili e le poniamo uguali a zero.

$$\boxed{\begin{cases} \dfrac{\partial S}{\partial \mathbf{a}} = 0 \\ \dfrac{\partial S}{\partial \mathbf{b}} = 0 \end{cases}}$$

## Approssimazione lineare ai minimi quadrati

Rispetto alla prima variabile, a

$$\frac{\partial S}{\partial \mathbf{a}} = 0 = \sum_{i=1}^{m} (2)(y_i - \mathbf{a}x_i - \mathbf{b})(-x_i)$$

$$-\sum_{i=1}^{m} x_i y_i + \sum_{i=1}^{m} \mathrm{a}x_i^2 + \sum_{i=1}^{m} \mathrm{b}x_i = 0$$

$$\boxed{\sum_{i=1}^{m} ax_i^2 + \sum_{i=1}^{m} bx_i = \sum_{i=1}^{m} x_i y_i \qquad [1]}$$

## Approssimazione lineare ai minimi quadrati

Rispetto alla seconda variabile, b

$$\frac{\partial S}{\partial \mathbf{b}} = \mathbf{0} = \sum_{i=1}^{m} (2)(y_i - \mathbf{a}x_i \; - \; \mathbf{b})(-1)$$

$$-\sum_{i=1}^{m} y_i + \sum_{i=1}^{m} \mathrm{a}x_i + \sum_{i=1}^{m} \mathrm{b} = 0$$

$$\boxed{\sum_{i=1}^{m} \mathrm{a}x_i + \sum_{i=1}^{m} \mathrm{b} = \sum_{i=1}^{m} y_i \qquad [2]}$$

## Approssimazione lineare ai minimi quadrati

Il sistema diventa allora

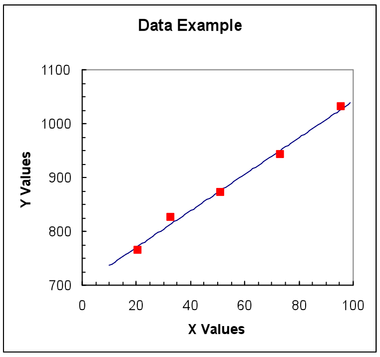
$$\begin{cases} \sum_{i=1}^{m} \mathrm{a}x_i^2 + \sum_{i=1}^{m} \mathrm{b}x_i = \sum_{i=1}^{m} x_i y_i & [1] \\ \sum_{i=1}^{m} \mathrm{a}x_i + \sum_{i=1}^{m} \mathrm{b} = \sum_{i=1}^{m} y_i & [2] \end{cases}$$

# Approssimazione lineare ai minimi quadrati

$$\begin{bmatrix} \sum_{i=1}^{m} x_i^2 & \sum_{i=1}^{m} x_i \\ \sum_{i=1}^{m} x_i & m \end{bmatrix} \begin{Bmatrix} a \\ b \end{Bmatrix} = \begin{Bmatrix} \sum_{i=1}^{m} x_i y_i \\ \sum_{i=1}^{m} y_i \end{Bmatrix}$$

# Esempio

| X | Y |
|---|---|
| 20.5 | 765 |
| 32.7 | 826 |
| 51 | 873 |
| 73.2 | 942 |
| 95.7 | 1032 |




$$a = \frac{5(254932.5) - 273.1(4438)}{5(18607.27) - (273.1)^2} = 3.395$$

$$b = \frac{18607.27(4438) - 273.1(254932.5)}{5(18607.27) - (273.1)^2} = 702.2$$

La retta che meglio approssima i dati nel senso dei minimi quadrati è:

$$\boxed{y = 3.395x + 702.2}$$

## Approssimazione ai minimi quadrati – pol. 2° grado

Come minimizzare l'errore per un'approssimazione ***$f_2(x)$*** polinomio di secondo grado?

$$\boxed{f_2(x) = \mathrm{a}x^2 + \mathrm{b}x + \mathrm{c}}$$

**Minimo di una funzione in tre variabili (a,b,c):**

calcoliamo le derivate parziali di S rispetto alle variabili e le poniamo uguali a zero.

$$\boxed{\begin{array}{c} \dfrac{\partial S}{\partial \mathbf{a}} = \mathbf{0} \\ \dfrac{\partial S}{\partial \mathbf{b}} = \mathbf{0} \\ \dfrac{\partial S}{\partial \mathbf{c}} = \mathbf{0} \end{array}}$$

## Approssimazione ai minimi quadrati

Il sistema che ne deriva in forma matriciale è:

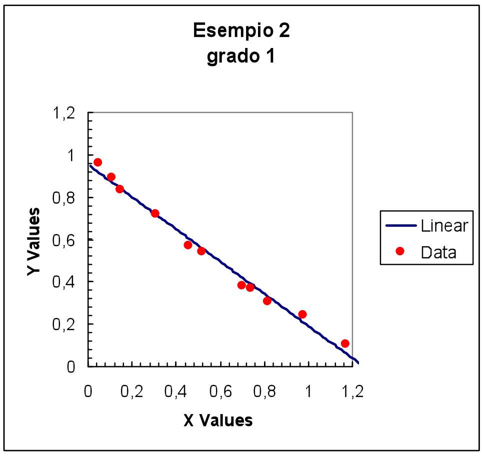
$$\begin{bmatrix} \sum_{i=1}^{m} x_i^4 & \sum_{i=1}^{m} x_i^3 & \sum_{i=1}^{m} x_i^2 \\ \sum_{i=1}^{m} x_i^3 & \sum_{i=1}^{m} x_i^2 & \sum_{i=1}^{m} x_i \\ \sum_{i=1}^{m} x_i^2 & \sum_{i=1}^{m} x_i & m \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \end{Bmatrix} = \begin{Bmatrix} \sum_{i=1}^{m} x_i^2 y_i \\ \sum_{i=1}^{m} x_i y_i \\ \sum_{i=1}^{m} y_i \end{Bmatrix}$$

# Esempio

Assegnati i punti

| X Value | Data |
|---|---|
| 0.05 | 0.956 |
| 0.11 | 0.89 |
| 0.15 | 0.832 |
| 0.31 | 0.717 |
| 0.46 | 0.571 |
| 0.52 | 0.539 |
| 0.7 | 0.378 |
| 0.74 | 0.37 |
| 0.82 | 0.306 |
| 0.98 | 0.242 |
| 1.17 | 0.104 |

Approssimante lineare:




# Esempio

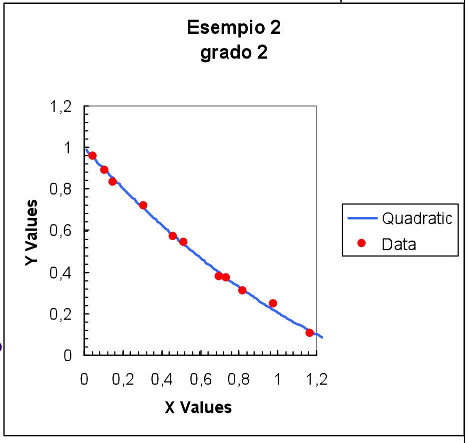
Approssimante quadratico:

a = 0.225,

b = -1.018 ,

c = 0.998

$$y = 0.225x^2 -1.018x + 0.998$$



## Approssimazione ai minimi quadrati

La tecnica può essere utilizzata per trovare approssimanti polinomiali di qualsiasi grado:

$$f_n(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

$$\begin{bmatrix} m & \sum_{i=1}^{m} x_i & \cdots & \sum_{i=1}^{m} x_i^n \\ \sum_{i=1}^{m} x_i & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \sum_{i=1}^{m} x_i^n & \cdots & \cdots & \sum_{i=1}^{m} x_i^{2n} \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{Bmatrix} = \begin{Bmatrix} \sum_{i=1}^{m} y_i \\ \sum_{i=1}^{m} x_i y_i \\ \vdots \\ \sum_{i=1}^{m} x_i^n y_i \end{Bmatrix}$$

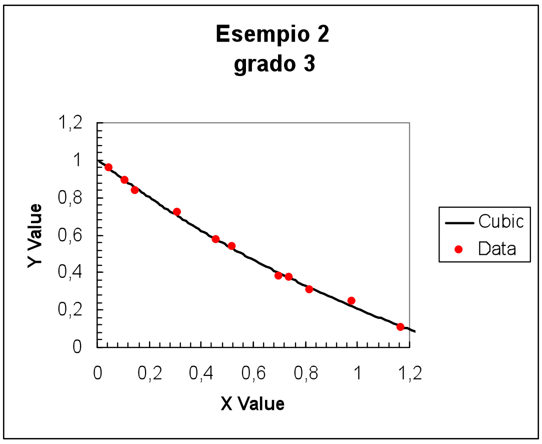
## Esempio *n*=3

$$y = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

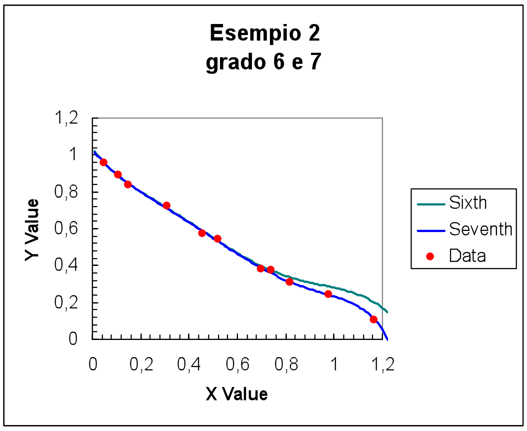
L'esempio 2 può essere approssimato con un **polinomio cubico** di coefficienti:

$a_0 = 1.004 \quad a_1 = -1.079$

$a_2 = 0.351 \quad a_3 = -0.069$



## Esempio $n$=6,7

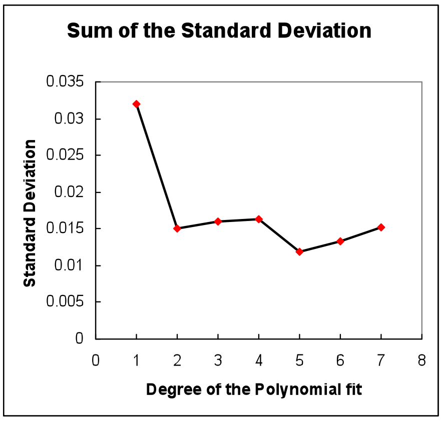



Calcolando i polinomi di più alto grado, per esempio grado 6 e 7, i risultati non sono molto promettenti.

## Esempio

La deviazione standard del polinomio approssimante mostra che il miglior *'fit'* si ha per n=5.

$$\sigma = \sqrt{\left(\frac{1}{m-n-1}\right)\sum_{k=1}^{m}\left(Y_k - y_k\right)^2}$$



# Approssimazione ai minimi quadrati: costo

L'approssimazione **lineare** richiede la risoluzione di un sistema di due equazioni, quella quadratica di un sistema di tre, … l'approssimazione di grado n richiede di risolvere un sistema lineare **(n+1) x (n+1)**

$$\boxed{f_n(x) = a_0 + a_1 x + \ldots + a_n x^n}$$

$$\begin{bmatrix} m & \sum_{i=1}^{m} x_i & \cdots & \sum_{i=1}^{m} x_i^n \\ \sum_{i=1}^{m} x_i & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \sum_{i=1}^{m} x_i^n & \cdots & \cdots & \sum_{i=1}^{m} x_i^{2n} \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{Bmatrix} = \begin{Bmatrix} \sum_{i=1}^{m} y_i \\ \sum_{i=1}^{m} x_i y_i \\ \vdots \\ \sum_{i=1}^{m} x_i^n y_i \end{Bmatrix}$$