Metodo dei minimi quadrati lineari

Supponiamo di avere un insieme di misure sperimentali che mettano in relazione la *grandezza misurata* con una *variabile indipendente* (di *controllo*) del sistema: per esempio, per mezzo di misure termochimiche, abbiamo misurato il calore specifico (a pressione costante, *Cp*) di una sostanza a temperature (*T*) diverse e variabili in un certo intervallo; in tal caso otteniamo infine un insieme di valori di *Cp* corrispondenti a un insieme di valori di *T*. Cerchiamo poi una relazione del tipo $Cp=f(T)$ che descriva o riassuma, in qualche modo, l'intero insieme di dati misurati, e che sia quindi in grado di fornire il valore più *corretto* possibile del calore specifico quando si inserisca nella *f* il valore di una data temperatura. Oppure ancora, potremmo aver misurato l'espansione termica α a temperature diverse e, anche qui, cercheremo una funzione $f(T)$ che ci restituisca α per ogni dato valore di *T*, con il minimo errore possibile.

la funzione $f(T)$ è quella che chiamiamo ***modello***. La scelta del *modello* può essere totalmente *empirica* (si cerca la funzione più efficace, dal punto di vista computazionale, per la riproduzione dei dati misurati, che sia anche *comoda* e semplice da utilizzarsi in vista di calcoli successivi), oppure può essere guidata da principi di tipo *teoretico*: per esempio, si sviluppa una teoria che predice che, stante certe condizioni, il *Cp* dipende dalla temperatura secondo una legge del tipo $a \cdot T^2$, dove *a* è una costante avente un preciso valore (*fissato* dalla teoria). Una legge simile non offre alcun *grado di libertà*, e tutto ciò che possiamo fare è *confrontare* il valore misurato di *Cp* a una certa *T* con quello *predetto* dal modello per la stessa *T*.

Ora, molto spesso accade che la teoria *esista*, ma non sia in grado di descrivere *accuratamente* la situazione *reale* perché i presupposti su cui quella si basa non sono esattamente soddisfatti dal sistema reale che ci interessa. È abbastanza frequente il caso per cui della teoria si mantenga la dipendenza di *massima* tra la variabile indipendente (di controllo) e la variabile dipendente (misurata). Nel nostro esempio, potremmo *trasformare* la costante *a* in un *parametro libero* da *ottimizzarsi* in modo da riprodurre al meglio le nostre misure. Il modello diventa perciò $f(a,T)= a \cdot T^2$ con *a* variabile da *ottimizzare* in base ai valori dei dati sperimentali. Volendo, potremmo anche *giocare* con la potenza del T^2, trasformando quel 2 in un numero da ottimizzarsi. Nel primo caso [$f(a;T)= a \cdot T^2$] la tecnica matematica da utilizzarsi per l'ottimizzazione prende il nome di ***minimi quadrati lineari*** (il modello è *lineare* nella variabile *a*) il secondo caso [$f(a,b;T)=a \cdot T^b$] rappresenta invece un problema *non lineare*. Consideriamo qui solo il caso lineare (che è anche molto *comune*).

Sia un insieme di *n misure* $\{y_1,\cdots, y_n\}$ in funzione di altrettanti valori della variabile indipendente x $\{x_1,\cdots, x_n\}$. Sia la funzione $y^c=ax^2$ il nostro modello (il simbolo y^c specifica il fatto che la variabile *y* ottenuta è *calcolata* in base al modello). Cerchiamo una funzione ε che esprima la discrepanza tra dati *y* misurati e dati y^c predetti dal modello. Una scelta naturale per ε è la ***somma dei quadrati degli scarti***:

$$\varepsilon = \sum_{i=1}^{n} (y_i - y_i^c)^2 = \sum_{i=1}^{n} (y_i - ax_i^2)^2 = \sum_{i=1}^{n} (y_i^2 + a^2 x_i^4 - 2ay_i x_i^2)$$

Notiamo esplicitamente che ε è funzione di a. Minimizziamo lo *scarto quadratico medio* $\varepsilon(a)$ rispetto al valore di a, e cioè *azzeriamo* la derivata della funzione $\varepsilon(a)$ rispetto ad a:

$$\frac{\partial \varepsilon(a)}{\partial a} = \sum_{i=1}^{n} (2ax_i^4 - 2y_i x_i^2) = 0 \;\rightarrow\; a = \frac{\sum_i y_i x_i^2}{\sum_i x_i^4}$$

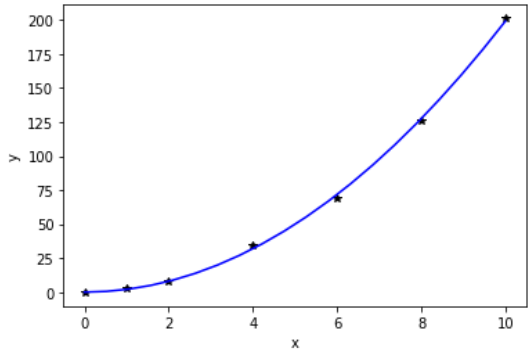
Il valore di a che abbiamo è quello che minimizza lo scarto quadratico medio tra i valori misurati e quelli *riprodotti* dal modello. Il metodo si estende rapidamente al caso di un insieme di parametri lineari, quali possono essere i coefficienti di un polinomio $y=a+bx+cx^2+dx^3+\ldots$; la funzione $\varepsilon(a,b,c,d,\ldots)$ si minimizza azzerando tutte le sue derivate parziali rispetto a tutti i parametri da cui dipende, e mettendo a sistema le equazioni risultanti.

Esistono algoritmi di ottimizzazione in tutti gli ambienti di programmazione che siano un minimo evoluti. L'esempio di programma che segue è scritto in python (il programma è caricato sul sito campusnet nella sezione esercitazioni).

```
# Best fit di una serie di dati y(x) con una funzione del tipo
# y=a*x^2

# import delle librerie di python
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.optimize import curve_fit

# dati di input
x=np.array([0, 1, 2, 4, 6, 8, 10])
y=np.array([-0.2, 2.4, 7.5, 34.4, 68.5, 125.8, 201.2 ])

# definizione della funzione y(x)=a*x^2
# myfunc accetta come input la serie di valori della
# variabile dipendente x, e il parametro "a" da ottimizzare
def myfunc(x,a):
    return a*x**2

# Ottimizzazione
# la funzione curve_fit usa la funzione myfunc, per determinare,
# per best fit (minimizzazione dello scarto quadratico medio),
# il valore del parametro "a", in base ai valori effettivi della
# variabile dipendente "y".
# I risultati di curve_fit sono passati alle variabili opt (valore di "a")
# e err (errore stimato su "a")
opt, err = curve_fit(myfunc, x, y)

print("\nParametro 'a' ottimizzato: %5.3f (%3.2e)\n" % (opt, err))

# Ai fini del plot, generazione di una lista di valori di x, e calcolo
# di y usando il modello y=a*x^2
x_list=np.linspace(min(x),max(x),20)
y_list=myfunc(x_list,opt)

# plot
plt.figure()
plt.plot(x,y,"k*")              # dati "reali"
plt.plot(x_list,y_list,"b-")    # dati calcolati dal modello
plt.xlabel("x")
```

```
plt.ylabel("y")
plt.show()

# Calcolo dei valori di y (yc) dal modello; calcolo della differenza
# y-yc per ogni valore di x
yc=myfunc(x,opt)
delta=y-yc

# Stampa di una tabella di valori x, y, yc e delta,
# usando le funzioni della libreria Pandas
serie=(x,y,yc,delta)
df=pd.DataFrame(serie, index=['x','y','y calc','delta'])
df=df.T
df2=df.round(3)
print("")
print(df2.to_string(index=False))
```

Il programma definisce due insiemi di variabili *x* (indipendente) e *y* (dipendente), ed esegue un fit del tipo ax^2. Il risultato (posto nella variabile *opt*) è *a*=1.993, con una *incertezza* su *a* (*err*) pari a $2.5 \cdot 10^{-4}$. Il grafico sottostante mostra i dati reali (*y*) come asterischi e la curva $yc=ax^2$ che li interpola (in blu).



x	y	y calc	delta
0.0	-0.2	0.000	-0.200
1.0	2.4	1.993	0.407
2.0	7.5	7.972	-0.472
4.0	34.4	31.888	2.512
6.0	68.5	71.747	-3.247
8.0	125.8	127.551	-1.751
10.0	201.2	199.298	1.902

Lo scarto quadratico medio: $\sqrt{\sum(y - yc)^2}$ vale 4.9.