# Misure di Densità

Gruppo di lavoro A-1-3-2
Marco RICCI (m. 641084) m.ricci66@studenti.unipi.it
Martino BARBIERI (m. 640522) m.barbieri20@studenti.unipi.it

UniPi - Laboratorio di Fisica 1 - a.a. 2021-2022
Docente: Prof. Luca Baldini

## 1 Introduzione

Sappiamo che una quantità fissata di qualunque sostanza o materiale occupa un volume che varia soltanto se variano le condizioni in cui tale sostanza o materiale si trova (ad esempio se dovesse passare dallo stato solido allo stato liquido, o se cambiano la temperatura o la pressione). La massa $m$ per unità di volume $V$ è nota come densità:

$$\rho = \frac{m}{V}$$

### 1.1 Obiettivo

Il fine di questa esperienza è verificare se la densità di diversi oggetti composti dal medesimo materiale possa essere considerata costante, individuare tale valore e identificare le differenze di densità sulla base delle differenze dei materiali; infine confrontare le densità trovate con i valori aspettati.

### 1.2 Materiale utilizzato

1. Calibro ventesimale
2. Calibro cinquantesimale
3. Calibro Palmer (risoluzione 0.01 mm)
4. Bilancia di precisione (risoluzione 1 mg)
5. Alcuni solidi in alluminio, acciaio e ottone

## 2 Raccolta dati

### 2.1 Accorgimenti

#### 2.1.1 Calcolo del volume

I solidi potrebbero non essere regolari. Per il prisma misuriamo distanze di lati e spigoli opposti della base; per i parallelepipedi misuriamo i tre spigoli[1]; per i cilindri misuriamo altezza e diametri lungo due direzioni perpendicolari; per le sfere misuriamo tre diametri perpendicolari.
Eseguiamo ciascuna misura con tutti i tre calibri a nostra disposizione, in modo da valutare quanto la sensibilità degli strumenti influenza il risultato.
Segnaliamo che abbiamo misurato un errore di zero del calibro Palmer di 0.003 mm.

[1] a livello operativo misuriamo le distanze tra facce opposte

### 2.1.2 Misura della massa

Usiamo una bilancia molto precisa; essa potrebbe avere risposte diverse attorno a punti di equilibrio diversi; per assicurarci che questo non influisca sulle misure, effettuiamo le misure delle masse poggiando i solidi prima direttamente sul piatto della bilancia e successivamente in un contenitore, tarando la bilancia prima di effettuare le misure. La bilancia è molto sensibile: notiamo che solo camminare in prossimità dell'armadio che la sorregge oppure parlare altera la misura; cerchiamo di limitare queste interferenze. Tariamo la bilancia frequentemente ond'evitare "effetti memoria".

## 2.2 Misure

Nelle seguenti tabelle, accanto al nome di ciascuna colonna, è indicata la sensibilità dello strumento usato. Quando non diversamente specificato, adottiamo come incertezza tale valore diviso per $\sqrt{12}$, adottando l'ipotesi di probabilità uniforme.

### 2.2.1 Sferette d'acciaio

Per ciascuna sfera riportiamo il diametro medio misurato col calibro ventesimale $d_{20}$, col calibro cinquantesimale $d_{50}$ e col calibro Palmer $d_{\mathrm{P}}$, le masse misurate poggiando il solido direttamente sul piatto $m_{\mathrm{d}}$ e nel contenitore $m_{\mathrm{c}}$.

| Sfera | $d_{20}[\pm 0.005\,\mathrm{cm}]$ | $d_{50}[\pm 0.002\,\mathrm{cm}]$ | $d_{\mathrm{P}}[\pm 0.001\,\mathrm{cm}]$ | $m_{\mathrm{d}}[\pm 0.001\,\mathrm{g}]$ | $m_{\mathrm{c}}[\pm 0.001\,\mathrm{g}]$ |
|---|---|---|---|---|---|
| 1 | 1.430 | 1.430 | 1.430 | 11.879 | 11.880 |
| 2 | 1.430 | 1.430 | 1.430 | 11.899 | 11.902 |
| 3 | 1.667 | 1.666 | 1.667 | 18.907 | 18.909 |
| 4 | 1.825 | 1.818 | 1.827 | 24.835 | 24.837 |
| 5 | 2.223 | 2.220 | 2.225 | 44.828 | 44.828 |

### 2.2.2 Prisma a base esagonale di ottone

Riportiamo l'altezza $h$, le distanze tra lati opposti $d_{\mathrm{l}}$, le distanze tra angoli opposti $d_{\mathrm{v}}$ e le masse con la medesima indicizzazione già usata.

| | $x_{20}[\pm 0.005\,\mathrm{cm}]$ | $x_{50}[\pm 0.002\,\mathrm{cm}]$ | $x_{\mathrm{P}}[\pm 0.001\,\mathrm{cm}]$ |
|---|---|---|---|
| $h$ | 2.275 | 2.270 | 2.273 |
| $d_{\mathrm{l}1}$ | 0.995 | 0.988 | 0.994 |
| $d_{\mathrm{l}2}$ | 0.995 | 0.988 | 0.996 |
| $d_{\mathrm{l}3}$ | 0.995 | 0.988 | 0.996 |
| $d_{\mathrm{v}1}$ | 1.135 | 1.136 | 1.139 |
| $d_{\mathrm{v}2}$ | 1.137 | 1.137 | 1.138 |
| $d_{\mathrm{v}3}$ | 1.135 | 1.134 | 1.137 |

| $m_{\mathrm{d}}[\pm 0.001\,\mathrm{g}]$ | $m_{\mathrm{c}}[\pm 0.001\,\mathrm{g}]$ |
|---|---|
| 16.424 | 16.425 |

### 2.2.3 Parallelepipedi di ottone e alluminio

Riportiamo l'altezza $h$, i due lati della sezione $l_1$, $l_2$ e le masse con la medesima indicizzazione già usata.

| Ottone | $x_{20}[\pm 0.005\,\text{cm}]$ | $x_{50}[\pm 0.002\,\text{cm}]$ | $x_{\text{P}}[\pm 0.001\,\text{cm}]$ | $m_{\text{d}}[\pm 0.001\,\text{g}]$ | $m_{\text{c}}[\pm 0.001\,\text{g}]$ |
|---|---|---|---|---|---|
| $h$ | 4.150 | 4.148 | – | 34.737 | 34.738 |
| $l_1$ | 0.995 | 1.000 | 0.996 | | |
| $l_2$ | 0.995 | 1.000 | 0.998 | | |
| Alluminio 1 | $x_{20}[\pm 0.005\,\text{cm}]$ | $x_{50}[\pm 0.002\,\text{cm}]$ | $x_{\text{P}}[\pm 0.001\,\text{cm}]$ | $m_{\text{d}}[\pm 0.001\,\text{g}]$ | $m_{\text{c}}[\pm 0.001\,\text{g}]$ |
| $h$ | 0.815 | 0.810 | – | 8.015 | 8.016 |
| $l_1$ | 1.835 | 1.834 | 1.834 | | |
| $l_2$ | 2.010 | 2.010 | 2.008 | | |
| Alluminio 2 | $x_{20}[\pm 0.005\,\text{cm}]$ | $x_{50}[\pm 0.002\,\text{cm}]$ | $x_{\text{P}}[\pm 0.001\,\text{cm}]$ | $m_{\text{d}}[\pm 0.001\,\text{g}]$ | $m_{\text{c}}[\pm 0.001\,\text{g}]$ |
| $h$ | 1.785 | 1.780 | 1.784 | 4.835 | 4.835 |
| $l_1$ | 1.000 | 1.000 | 1.004 | | |
| $l_2$ | 1.005 | 1.004 | 1.004 | | |

#### 2.2.4 Cilindri di ottone e alluminio

Riportiamo l'altezza $h$, il diametro $d$ e le masse con la medesima indicizzazione già usata.

| Ottone 1 | $x_{20}[\pm 0.005\,\text{cm}]$ | $x_{50}[\pm 0.002\,\text{cm}]$ | $x_{\text{P}}[\pm 0.001\,\text{cm}]$ | $m_{\text{d}}[\pm 0.001\,\text{g}]$ | $m_{\text{c}}[\pm 0.001\,\text{g}]$ |
|---|---|---|---|---|---|
| $h$ | 1.595 | 1.590 | 1.595 | 10.470 | 10.471 |
| $d$ | 0.995 | 0.995 | 0.996 | | |
| Ottone 2 | $x_{20}[\pm 0.005\,\text{cm}]$ | $x_{50}[\pm 0.002\,\text{cm}]$ | $x_{\text{P}}[\pm 0.001\,\text{cm}]$ | $m_{\text{d}}[\pm 0.001\,\text{g}]$ | $m_{\text{c}}[\pm 0.001\,\text{g}]$ |
| $h$ | 3.740 | 3.740 | – | 24.561 | 24.562 |
| $d$ | 0.995 | 0.992 | 0.995 | | |
| Alluminio 1 | $x_{20}[\pm 0.005\,\text{cm}]$ | $x_{50}[\pm 0.002\,\text{cm}]$ | $x_{\text{P}}[\pm 0.001\,\text{cm}]$ | $m_{\text{d}}[\pm 0.001\,\text{g}]$ | $m_{\text{c}}[\pm 0.001\,\text{g}]$ |
| $h$ | 1.940 | 1.940 | 1.942 | 1.460 | 1.460 |
| $d$ | 0.595 | 0.590 | 0.595 | | |
| Alluminio 2 | $x_{20}[\pm 0.005\,\text{cm}]$ | $x_{50}[\pm 0.002\,\text{cm}]$ | $x_{\text{P}}[\pm 0.001\,\text{cm}]$ | $m_{\text{d}}[\pm 0.001\,\text{g}]$ | $m_{\text{c}}[\pm 0.001\,\text{g}]$ |
| $h$ | 1.945 | 1.944 | 1.946 | 5.900 | 5.900 |
| $d$ | 1.195 | 1.192 | 1.195 | | |

# 3 Analisi dati

## 3.1 Metodi di analisi e Calcolo delle incertezze

I dati sono stati elaborati con alcuni script in Python che lasciamo in Appendice A.

Per le sfere occorre calcolare valore medio e incertezza della densità $\rho = \bar{\rho} \pm \sigma_{\bar{\rho}}$. Per una singola misura:

$$\rho = \frac{m}{\frac{1}{6}\pi d^3} \qquad \sigma_\rho = \rho\sqrt{\left(\frac{\sigma_m}{m}\right)^2 + 9\left(\frac{\sigma_d}{d}\right)^2}$$

A priori, siccome il rapporto tra la risoluzione dello strumento e la misura è di un ordine di grandezza maggiore nel caso del diametro, mi aspetto che il contributo all'incertezza della massa sia trascurabile; uso quindi come ascissa la massa e come ordinata il volume (questo al fine di poter rispettare le ipotesi di utilizzo del metodo dei minimi quadrati). Tuttavia, siccome il calcolo è particolarmente semplice, per l'incertezza complessiva di $\rho$ utilizzo la formula non approssimata, considerando anche il contributo dell'incertezza della massa. Come incertezza per i diametri, uso la risoluzione strumentale divisa per la radice quadrata di 12 nell'ipotesi di probabilità uniforme. Calcoliamo la miglior stima della densità con `scipy.optimize.curve_fit()`, passando come modello la differenza tra il parametro $\rho_0$ (che ci darà la miglior stima di $\rho$) e la funzione $\rho - \rho_0$. Come incertezza adottiamo la deviazione standard della media:

$$\sigma_{\bar{\rho}} = \sqrt{\frac{\sum_{i=0}^{N}(\rho_i - \rho_0)^2}{N-1}}$$

La legge che ci aspettiamo tra $m$ e $d$ è una legge di potenza:

$$m = \frac{\pi}{6}\rho d^3$$

dunque, dividendo a destra e a sinistra per $m_0 = 1\,\mathrm{g}$, considerando $d_0 = 1\,\mathrm{cm}$ e estraendo il logaritmo troviamo una relazione lineare tra "il logaritmo della massa" e "il logaritmo del diametro".

$$\ln\frac{d}{d_0} = -\frac{1}{3}\ln\left(\frac{\pi}{6}\frac{\rho d_0^3}{m_0}\right) + \frac{1}{3}\ln\frac{m}{m_0}$$

Per quanto riguarda i solidi non sferici la situazione è più complicata: i bordi sono smussati, quindi il volume effettivo è minore di quello calcolato. Siccome le incertezze relative sulle masse sono trascurabili rispetto a quelle sui volumi, eseguiamo una regressione ai minimi quadrati ordinari tramite `scipy.optimize.curve_fit()`, usando come ascisse le masse e come ordinate i volumi. Ci aspettiamo che il volume "perso" a causa dei bordi sia dipendente dal solo materiale e sia proporzionale all'intero perimetro del solido. Nel caso del prisma a base esagonale possiamo farne una stima più precisa dalla differenza tra l'apotema e il valore che ci aspetteremmo per l'apotema misurando il raggio della circonferenza inscritta nella base: infatti ci possiamo aspettare che il raggio di curvatura del bordo del solido sullo spigolo sia di

$$r_\mathrm{c} \approx \frac{\frac{2\sqrt{3}}{3}d_\mathrm{l} - d_\mathrm{v}}{2}$$

Escludendo i termini in $r_\mathrm{c}^3$, per considerazioni di carattere geometrico, possiamo aspettarci che il volume perso sia:

- per il prisma:
$$\Delta V \simeq \left[\left(2\sqrt{3} - \pi\right)h + \sqrt{3}\,(4-\pi)\,d_\mathrm{l}\right] r_\mathrm{c}^2$$
- per il cilindro:
$$\Delta V \simeq \frac{1}{2}\pi(4-\pi)dr_\mathrm{c}^2$$
- per il parallelepipedo:
$$\Delta V \simeq (4-\pi)(h + l_1 + l_2)r_\mathrm{c}^2$$

Calcolo il volume (senza la correzione dovuta alla smussatura) con le seguenti formule:

- per il prisma:
$$V = \frac{\sqrt{3}}{2}d_\mathrm{l}^2 h \qquad \frac{\sigma_V}{V} = \sqrt{4\left(\frac{\sigma_{d_\mathrm{l}}}{d_\mathrm{l}}\right)^2 + \left(\frac{\sigma_h}{h}\right)^2}$$
- per il cilindro:
$$V = \frac{\pi}{4}d^2 h \qquad \frac{\sigma_V}{V} = \sqrt{4\left(\frac{\sigma_d}{d}\right)^2 + \left(\frac{\sigma_h}{h}\right)^2}$$
- per il parallelepipedo:
$$V = l_1 l_2 h \qquad \frac{\sigma_V}{V} = \sqrt{\left(\frac{\sigma_{l_1}}{l_1}\right)^2 + \left(\frac{\sigma_{l_2}}{l_2}\right)^2 + \left(\frac{\sigma_h}{h}\right)^2}$$

In alcune situazioni non è stato possibile, a causa della massima portata del calibro Palmer, misurare alcune grandezze con tale strumento. Tuttavia le misure effettuate con diversi calibri, per i solidi non sferici non sono compatibili, probabilmente a causa del fatto che la forma dei solidi è soggetta a irregolarità locali e alla risposta del materiale alla pressione dei calibri. Dunque le sensibilità degli strumenti non corrispondono alle incertezze. Adottiamo quindi come incertezze per le lunghezze le deviazione standard sulla media.

## 3.2 Risultati dell'analisi

### 3.2.1 Sfere d'acciaio

Codice 1: Output dello script Python.

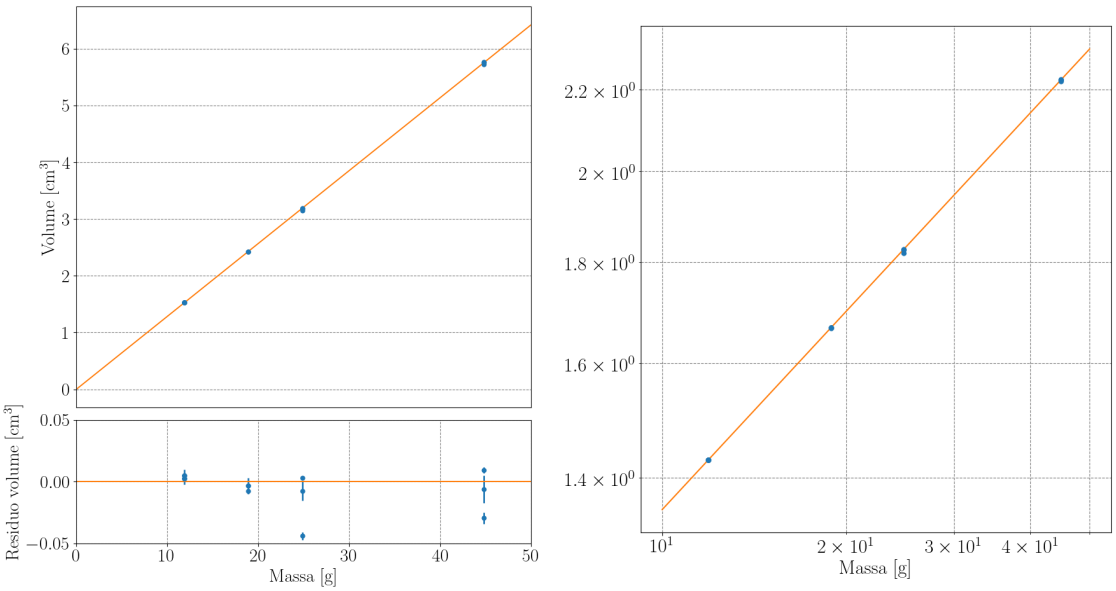
```
### Fit massa-volume ###
Parametri ottimali: [7.78498088]
Matrici di Covarianza: [[5.67705869e-05]]
Incertezza: [[0.00753463]]

### Fit massa-diametro ###
Parametri ottimali: [0.62690822 0.33287657]
Matrici di Covarianza: [[ 1.61736611e-06 -7.95783364e-07]
 [-7.95783364e-07  4.02183686e-07]]
Stime per le incertezze: [0.00127176 0.00063418]
Differenza tra 1/3 e index: -0.0004567584557430596
```

La densità trovata

$$\rho_{\text{acc}} = (7.785 \pm 0.008)\,\text{g cm}^{-3}$$

è compatibile con quella prevista per l'acciaio. Notiamo anche che l'indice della legge di potenza ($0.3329 \pm 0.0006$) e l'intercetta ($0.6269 \pm 0.0013$) sono compatibili con i risultati aspettati di $\frac{1}{3}$ e 0.6260. I dati in entrambi i grafici si dispongono attorno a rette.




### 3.2.2 Solidi d'alluminio

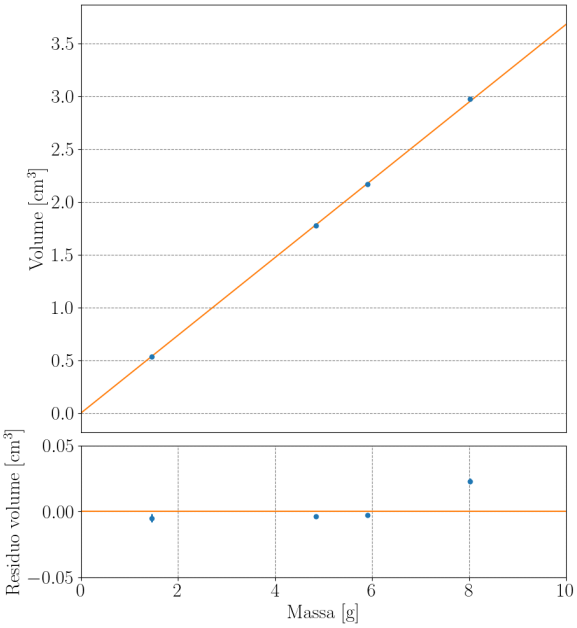
Codice 2: Output dello script Python.

```
### Fit massa-volume ###
Parametri ottimali: [2.71621882 0.00519642]
Matrici di Covarianza: [[3.56118447e-04 9.19179858e-05]
 [9.19179858e-05 2.74881749e-05]]
Incertezze: [0.0188711  0.00524292]
r +- sr = 0.072086 +- 0.036366
```

La densità trovata

$$\rho_{\text{all}} = (2.716 \pm 0.018)\,\text{g cm}^{-3}$$

è compatibile con quella prevista per l'alluminio. I dati in entrambi i grafici si dispongono attorno a rette. Il quadrato del raggio di smussatura dei bordi ($(0.0052 \pm 0.0052)$ cm) è compatibile con 0, indice del fatto che i solidi non sono perfettamente regolari.




### 3.2.3 Solidi di ottone

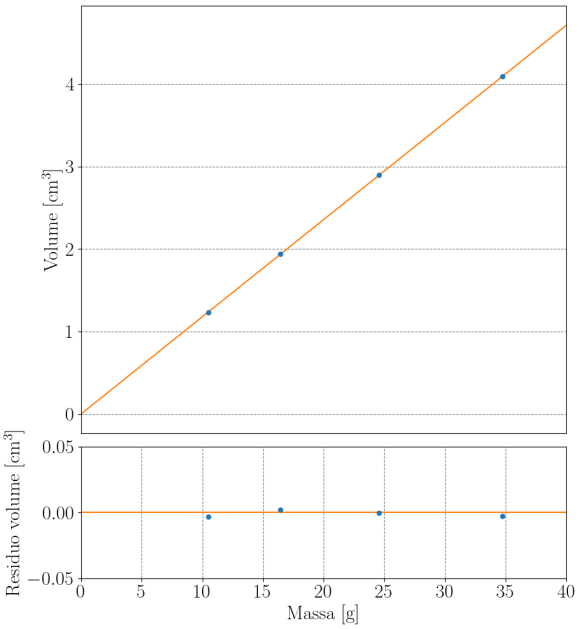
Codice 3: Output dello script Python.

```
### Fit massa-volume ###
Parametri ottimali: [8.48994481e+00 7.26516515e-03]
Matrici di Covarianza: [[7.11121659e-04 7.33359882e-05]
 [7.33359882e-05 8.11581093e-06]]
Incertezze: [0.02666686 0.00284883]
r +- sr = 0.085236 +- 0.016711
r +- sr = 0.006883 +- 0.000132
```

La densità trovata

$$\rho_{\text{all}} = (8.490 \pm 0.027)\,\text{g cm}^{-3}$$

è compatibile con quella prevista per l'ottone. I dati in entrambi i grafici si dispongono attorno a rette. Il quadrato del raggio di smussatura dei bordi ($(0.0052 \pm 0.0052)$ cm) è compatibile con 0; siccome la radice quadrata, usata per calcolare il primo valore di $r_c = 0.085\,\text{cm}$ da $r_c^2$, non è una funzione lipshitziana nell'intervallo contenente 0, l'incertezza associata a tale misura non può essere calcolata tramite la propagazione delle incertezze al primo ordine, quindi il valore dato in output dal programma è privo di significato. Le due stime di $r_c$ sono quindi compatibili. Non entrambe le misure sono compatibili con 0 ed è quindi ragionevole assumere che i bordi siano lievemente smussati (il decimo di millimetro è anche la stima che potevamo immaginarci). L'incertezza dovuta alla stima di $r_c$ ricavata dal fit deriva dall'irregolarità dei solidi; quella per misura diretta dalla natura intrinseca nella propagazione degli errori: stiamo infatti sottraendo quantità molto vicine tra loro.




# 4 Conclusioni

Abbiamo verificato per i solidi a nostra disposizione, nelle condizioni in cui è stata svolta la prova, il rapporto tra la massa e il volume dipende solo dal materiale.

Si manifesta un errore sistematico sulla misura del volume causato dalle smussature dei bordi e un errore casuale generato dalle irregolarità geometriche degli oggetti in esame; infatti l'incertezza relativa associata alle sferette d'acciaio per cuscinetti è minore di quelle per gli altri due materiali, a causa della maggior precisione nella realizzazione di tali oggetti.

Nel caso delle sferette d'acciaio abbiamo verificato che c'è una legge di potenza tra $d$ e $m$ e che l'indice di questa legge di potenza è compatibile con quello previsto dalla teoria.

Riportiamo le stime delle densità ottenute e quelle aspettate.

| | $\rho$[g cm$^{-1}$] misurato | $\sigma_\rho$[g cm$^{-1}$] misurato | $\rho$[g cm$^{-1}$] atteso |
|---|---|---|---|
| Acciaio | 7.785 | 0.008 | 7480-8000 |
| Alluminio | 2.716 | 0.018 | 2710 |
| Ottone | 8.490 | 0.027 | 8400-8700 |

# A Appendice: Codici Python utilizzati

```
import numpy as np
from scipy.optimize import curve_fit, minimize_scalar
from matplotlib import pyplot as plt
import math

plt.rcParams.update({'font.size': 22, 'font.family':'serif'})
plt.rcParams["text.usetex"] = True

# DATA
d = np.array([
    1.430, 1.430, 1.430,
    1.430, 1.430, 1.430,
    1.667, 1.666, 1.667,
    1.825, 1.818, 1.827,
    2.223, 2.220, 2.225
    ])
sd = np.array([
    0.005,0.002,0.001,
    0.005,0.002,0.001,
    0.005,0.002,0.001,
    0.005,0.002,0.001,
    0.005,0.002,0.001
])/(12.**0.5)
m = np.array([
    11.8795,11.8795,11.8795,
    11.9005,11.9005,11.9005,
    18.908,18.908,18.908,
    24.836,24.836,24.836,
    44.828,44.828,44.828
])
sm = np.array([
    0.001,0.001,0.001,
    0.003,0.003,0.003,
    0.002,0.002,0.002,
    0.002,0.002,0.002,
    0.001,0.001,0.001
])/(12.**0.5)

# FIT massa-volume
print(" ### Fit massa-volume ###")

rho = 6.*m/(math.pi*d**3)
srho = rho * ((sm/m)**2+9.*(sd/d)**2)**0.5

popt,pcov = curve_fit( lambda r,r0: r-r0, rho, np.zeros_like(rho), sigma = srho, bounds
    =(7.,9.) )

print("Parametri ottimali: %s" % popt)
print("Matrici di Covarianza: %s" % pcov)
print("Incertezza: %s" % pcov**0.5 )

# GRAFICO massa-volume
mar_size = 5
bar_l = 2

fig = plt.figure('Grafico volume-massa')

fig.add_axes((0.15, 0.32, 0.8, 0.65))
plt.errorbar(m,math.pi/6.*d**3,sm,math.pi/2.*d**2*sd, fmt='o', markersize=mar_size,
    linewidth=bar_l)
plt.plot((0,50.),(0,50./popt[0]))
plt.xlim(0.,50.)

plt.ylabel(r'Volume [cm$^3$]')
plt.gca().get_xaxis().set_visible(False)

plt.grid(which='both', ls='dashed', color='gray')

```

```
fig.add_axes((0.15, 0.1, 0.8, 0.2))

res = math.pi/6.*d**3 - 1/popt[0]*m
plt.errorbar(m,res, xerr=sm, yerr=3.*sd/d*math.pi/6.*d**3, fmt='o', markersize=mar_size
    , linewidth=bar_l)
plt.plot((0,50.),(0,0))
plt.grid(which='both', ls='dashed', color='gray')
plt.ylabel(r'Residuo volume [cm$^3$]')
plt.xlabel(r'Massa [g]')
plt.xlim(0.,50.)
plt.ylim(-0.05, 0.05)

plt.show()

# FIT massa-diametro
print("\n ### Fit massa-diametro ###")

popt,pcov = curve_fit( lambda m,norm,index: norm*m**index, m, d, sigma=sd )

print("Parametri ottimali: %s" % popt)
print("Matrici di Covarianza: %s" % pcov)
print("Stime per le incertezze: %s" % np.sqrt(pcov.diagonal()))
print("Differenza tra 1/3 e index: %s" % (popt[1]-1./3.) )

# GRAFICO massa-diametro
fig = plt.figure('Grafico diametro-massa')

plt.errorbar(m,d,sm,sd, fmt='o', markersize=mar_size, linewidth=bar_l)
plt.plot((10,50),(popt[0]*10**popt[1],popt[0]*50**popt[1]))

plt.xscale('log')
plt.yscale('log')

plt.ylabel(r'Diametro [cm]')
plt.xlabel(r'Massa [g]')

plt.grid(which='both', ls='dashed', color='gray')

plt.show()
```

Codice 4: Codice per l'analisi dei dati delle sfere d'acciaio.

```
import numpy as np
from scipy.optimize import curve_fit, minimize_scalar
from matplotlib import pyplot as plt
import math

plt.rcParams.update({'font.size': 22, 'font.family':'serif'})
plt.rcParams["text.usetex"] = True

# DATA
#parallelepipedi
p1_h = np.array([0.815,0.810])
p1_l = np.array([1.835,1.834,1.834])
p1_L = np.array([2.010,2.010,2.008])
p2_h = np.array([1.785,1.780,1.784])
p2_l = np.array([1.000,1.000,1.004])
p2_L = np.array([1.005,1.004,1.004])
#cilindri
c1_h = np.array([1.940,1.940,1.942])
c1_d = np.array([0.595,0.590,0.595])
c2_h = np.array([1.945,1.944,1.946])
c2_d = np.array([1.195,1.192,1.195])

# MEDIE e COEFF. SMUSSAMENTO
def avgstd(x):
    return x.std() / ( len(x)*(len(x)-1) )**0.5

p1_hm = p1_h.mean()
p1_hs = avgstd(p1_h)
```

```
29
30 p1_lm = p1_l.mean()
31 p1_ls = avgstd(p1_l)
32
33 p1_Lm = p1_L.mean()
34 p1_Ls = avgstd(p1_L)
35
36 p1_p = (4-math.pi)*(p1_hm+p1_lm+p1_Lm)
37 p1_v = p1_hm*p1_lm*p1_Lm
38 p1_vs = ( (p1_hs/p1_hm)**2 + (p1_ls/p1_lm)**2 + (p1_Ls/p1_Lm)**2 )**0.5
39
40
41 p2_hm = p2_h.mean()
42 p2_hs = avgstd(p2_h)
43
44 p2_lm = p2_l.mean()
45 p2_ls = avgstd(p2_l)
46
47 p2_Lm = p2_L.mean()
48 p2_Ls = avgstd(p2_L)
49
50 p2_p = (4-math.pi)*(p2_hm+p2_lm+p2_Lm)
51 p2_v = p2_hm*p2_lm*p2_Lm
52 p2_vs = ( (p2_hs/p2_hm)**2 + (p2_ls/p2_lm)**2 + (p2_Ls/p2_Lm)**2 )**0.5
53
54
55 c1_hm = c1_h.mean()
56 c1_hs = avgstd(c1_h)
57
58 c1_dm = c1_d.mean()
59 c1_ds = avgstd(c1_d)
60
61 c1_p = 0.5*math.pi*(4-math.pi)*c1_dm
62 c1_v = 0.25*math.pi*c1_dm**2*c1_hm
63 c1_vs = ( (c1_hs/c1_hm)**2 + 4*(c1_ds/c1_dm)**2 )**0.5
64
65
66 c2_hm = c2_h.mean()
67 c2_hs = avgstd(c2_h)
68
69 c2_dm = c2_d.mean()
70 c2_ds = avgstd(c2_d)
71
72 c2_p = 0.5*math.pi*(4-math.pi)*c2_dm
73 c2_v = 0.25*math.pi*c2_dm**2*c2_hm
74 c2_vs = ( (c2_hs/c2_hm)**2 + 4*(c2_ds/c2_dm)**2 )**0.5
75
76 # IMPACCHETTAMENTO
77 # x = (m, p), y = v
78 x = (
79         np.array( [ 8.0155, 4.835, 1.460, 5.900 ] ),
80         np.array( [ p1_p, p2_p, c1_p, c2_p ] )
81     )
82 y = np.array( [ p1_v, p2_v, c1_v, c2_v ] )
83 sy = np.array( [ p1_vs, p2_vs, c1_vs, c2_vs ] )
84
85 # FIT massa-volume
86 print(" ### Fit massa-volume ###")
87
88 popt,pcov = curve_fit( lambda x,rho,r2: 1/rho*x[0]+r2*x[1], x, y, sigma = sy )
89
90 print("Parametri ottimali: %s" % popt)
91 print("Matrici di Covarianza: %s" % pcov)
92 print("Incertezze: %s" % pcov.diagonal()**0.5)
93
94 print("r +- sr = %f +- %f" % (popt[1]**0.5,0.5*(pcov[1,1]/popt[1])**0.5))
95
96 # GRAFICO
97 mar_size = 5
98 bar_l = 2
```

```
99
100 fig = plt.figure('Grafico volume-massa')
101
102 fig.add_axes((0.15, 0.32, 0.8, 0.65))
103 plt.errorbar(x[0],y-popt[1]*x[1],0,sy, fmt='o', markersize=mar_size, linewidth=bar_l)
104 plt.plot((0,10.),(0,10./popt[0]))
105 plt.xlim(0.,10.)
106
107 plt.ylabel(r'Volume [cm$^3$]')
108 plt.gca().get_xaxis().set_visible(False)
109
110 plt.grid(which='both', ls='dashed', color='gray')
111
112 fig.add_axes((0.15, 0.1, 0.8, 0.2))
113
114 res = y-popt[1]*x[1] - 1/popt[0]*x[0]
115 plt.errorbar(x[0],res, yerr=sy, fmt='o', markersize=mar_size, linewidth=bar_l)
116 plt.plot((0,10.),(0,0))
117 plt.grid(which='both', ls='dashed', color='gray')
118 plt.ylabel(r'Residuo volume [cm$^3$]')
119 plt.xlabel(r'Massa [g]')
120 plt.xlim(0.,10.)
121 plt.ylim(-0.05, 0.05)
122
123 plt.show()
```

Codice 5: Codice per l'analisi dei dati dei solidi in alluminio.

```
1  import numpy as np
2  from scipy.optimize import curve_fit, minimize_scalar
3  from matplotlib import pyplot as plt
4  import math
5
6  plt.rcParams.update({'font.size': 22, 'font.family':'serif'})
7  plt.rcParams["text.usetex"] = True
8
9  # DATA
10 #parallelepipedi
11 p1_h = np.array([4.150,4.148])
12 p1_l = np.array([0.995,1.000,0.996])
13 p1_L = np.array([0.995,1.000,0.998])
14
15 #prismi
16 p2_h = np.array([2.275,2.270,2.273])
17 # 988 o 998?
18 p2_l = np.array([0.995,0.998,0.994,
19                  0.995,0.998,0.996,
20                  0.995,0.998,0.996])
21 p2_dv = np.array([1.135,1.136,1.139,
22                  1.137,1.137,1.138,
23                  1.135,1.134,1.137])
24
25 #cilindri
26 c1_h = np.array([1.595,1.590,1.595])
27 c1_d = np.array([0.995,0.995,0.996])
28 c2_h = np.array([3.740,3.740])
29 c2_d = np.array([0.995,0.992,0.995])
30
31 # MEDIE e COEFF. SMUSSAMENTO
32 def avgstd(x):
33     return x.std() / ( len(x)*(len(x)-1) )**0.5
34
35 p1_hm = p1_h.mean()
36 p1_hs = avgstd(p1_h)
37
38 p1_lm = p1_l.mean()
39 p1_ls = avgstd(p1_l)
40
41 p1_Lm = p1_L.mean()
42 p1_Ls = avgstd(p1_L)
```

```
43
44 p1_p = (4-math.pi)*(p1_hm+p1_lm+p1_Lm)
45 p1_v = p1_hm*p1_lm*p1_Lm
46 p1_vs = ( (p1_hs/p1_hm)**2 + (p1_ls/p1_lm)**2 + (p1_Ls/p1_Lm)**2 )**0.5
47
48
49 p2_hm = p2_h.mean()
50 p2_hs = avgstd(p2_h)
51
52 p2_lm = p2_l.mean()
53 p2_ls = avgstd(p2_l)
54
55 p2_dvm = p2_dv.mean()
56 p2_dvs = avgstd(p2_dv)
57
58 p2_p = (2*3.**0.5-math.pi)*p2_hm+3.**0.5*(4-math.pi)*p2_lm
59 p2_v = 0.5*3.**0.5*p2_lm**2*p2_hm
60 p2_vs = ( 4*(p2_ls/p2_lm)**2 + (p2_hs/p2_hm)**2 )**0.5
61
62
63 c1_hm = c1_h.mean()
64 c1_hs = avgstd(c1_h)
65
66 c1_dm = c1_d.mean()
67 c1_ds = avgstd(c1_d)
68
69 c1_p = 0.5*math.pi*(4-math.pi)*c1_dm
70 c1_v = 0.25*math.pi*c1_dm**2*c1_hm
71 c1_vs = ( (c1_hs/c1_hm)**2 + 4*(c1_ds/c1_dm)**2 )**0.5
72
73
74 c2_hm = c2_h.mean()
75 c2_hs = avgstd(c2_h)
76
77 c2_dm = c2_d.mean()
78 c2_ds = avgstd(c2_d)
79
80 c2_p = 0.5*math.pi*(4-math.pi)*c2_dm
81 c2_v = 0.25*math.pi*c2_dm**2*c2_hm
82 c2_vs = ( (c2_hs/c2_hm)**2 + 4*(c2_ds/c2_dm)**2 )**0.5
83
84
85 # IMPACCHETTAMENTO
86 # x = (m, p), y = v
87 x = (
88         np.array( [ 34.7375, 16.4245, 10.4705, 24.5615 ] ),
89         np.array( [ p1_p, p2_p, c1_p, c2_p ] )
90     )
91 y = np.array( [ p1_v, p2_v, c1_v, c2_v ] )
92 sy = np.array( [ p1_vs, p2_vs, c1_vs, c2_vs ] )
93
94 # FIT massa-volume
95 print(" ### Fit massa-volume ###")
96
97 popt,pcov = curve_fit( lambda x,rho,r2: 1/rho*x[0]+r2*x[1], x, y, sigma = sy )
98
99 print("Parametri ottimali: %s" % popt)
100 print("Matrici di Covarianza: %s" % pcov)
101 print("Incertezze: %s" % pcov.diagonal()**0.5)
102
103 print("r +- sr = %f +- %f" % (popt[1]**0.5,0.5*(pcov[1,1]/popt[1])**0.5))
104 print("r +- sr = %f +- %f" % (0.5*(2./3.**0.5*p2_lm-p2_dvm),(1/3.*p2_ls**2+0.25*p2_dvs
    **2)**0.5))
105
106 # GRAFICO
107 mar_size = 5
108 bar_l = 2
109
110 fig = plt.figure('Grafico volume-massa')
111
```

```
fig.add_axes((0.15, 0.32, 0.8, 0.65))
plt.errorbar(x[0],y-popt[1]*x[1],0,sy, fmt='o', markersize=mar_size, linewidth=bar_l)
plt.plot((0,40.),(0,40./popt[0]))
plt.xlim(0.,40.)

plt.ylabel(r'Volume [cm$^3$]')
plt.gca().get_xaxis().set_visible(False)

plt.grid(which='both', ls='dashed', color='gray')

fig.add_axes((0.15, 0.1, 0.8, 0.2))

res = y-popt[1]*x[1] - 1/popt[0]*x[0]
plt.errorbar(x[0],res, yerr=sy, fmt='o', markersize=mar_size, linewidth=bar_l)
plt.plot((0,40.),(0,0))
plt.grid(which='both', ls='dashed', color='gray')
plt.ylabel(r'Residuo volume [cm$^3$]')
plt.xlabel(r'Massa [g]')
plt.xlim(0.,40.)
plt.ylim(-0.05, 0.05)

plt.show()
```

Codice 6: Codice per l'analisi dei dati dei solidi in ottone.