# Caduta di un grave

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

Sotto opportune ipotesi si può dimostrare che la legge oraria di un grave in caduta libera vicino alla superficie terrestre è

$$x(t) = x_0 + \frac{g}{\beta}t + \frac{\dot{x_0}\beta + g}{\beta^2}(1 - e^{-\beta t})$$

dove $x_0, \dot{x_0}$ sono la posizione e la velocità iniziale, $g$ è l'intensità del campo gravitazionale e $\beta$ è una costante che dipende dalla viscosità dell'aria $\mu$:

$$\beta = \frac{6\pi\mu r}{m}$$

dove $r, m$ sono rispettivamente il raggio e la massa della pallina. Ne lasciamo nell'Appendice A la dimostrazione.

### 1.1 Obiettivo

Il fine di questa esperienza è di verificare sperimentalmente se la legge oraria di caduta di un grave possa essere compatibile con la previsione teorica ottenuta tramite il calcolo differenziale. Quindi vogliamo comprendere in quali condizioni la coincidenza tra modello e realtà è maggiore ed infine ipotizzare le ragioni per cui vi sono delle discrepanze.

### 1.2 Materiale utilizzato

1. Varie sfere di diversa massa e diverso diametro (due sfere metalliche per cuscinetti per motorini, una pallina da subotto e una da ping-pong, una palla da biliardo, una pallina di gomma);
2. metro a nastro;
3. smartphone per riprendere la caduta;
4. bilancia elettronica;
5. calibro ventesimale.

# 2 Raccolta dati

In questa sezione descriveremo brevemente gli accorgimenti che abbiamo utilizzato per ridurre potenziali errori sistematici e raccogliere misure in una situazione il più possibile prossima a quella ideale, dove abbiamo ricavato analiticamente la legge oraria. Riporteremo poi gli esiti di qualche misura che abbiamo preso.

Sul sito `https://unilab.mbarbieri.it` rendiamo disponibile tutto il materiale raccolto ed elaborato. Lasciamo nell'Appendice B il codice Python utilizzato per l'analisi dei dati.

## 2.1 Accorgimenti

### 2.1.1 Gravi

Scegliamo gravi di diversa massa e diverso raggio per valutare se l'effetto di attrito viscoso è trascurabile. Utilizziamo solo oggetti di forma sferica perché su di essi è ben noto l'effetto che il fluido viscoso compie. Cerchiamo di non dare movimento rotatorio al corpo quando lo lasciamo cadere.

### 2.1.2 Riprese con lo smartphone

Utilizziamo un telefono (Samsung S20 FE 5G) in grado di registrare video in Slow Motion (240 fps) a un'alta risoluzione (1080p) con una discreta qualità. Per assicurarci di non riprendere la caduta del grave con un telefono inclinato, che distorcerebbe le distanze, verifichiamo tramite l'accelerometro del telefono che esso è verticale (a tal proposito utilizziamo l'applicazione Phyphox e ci assicuriamo che, in media, $g_z$ sia nulla). Fissiamo il telefono a una scala di alluminio. Per assicurarci che la frequenza effettiva del video sia quella attesa, registriamo un cronometro e contiamo il numero di frame presenti in 100 secondi. Assumendo praticamente esatto l'output di un timer di un tablet, risulta che 100 s di registrazione corrispondono a $(799.938 \pm 0.075)$ s di video a 30 fps, dove l'incertezza è stata calcolata con l'usuale propagazione delle incertezze e l'incertezza gli istanti iniziale e finale è data dal tempo in cui lo schermo del tablet cambia le cifre mostrate diviso $\sqrt{12}$, sotto l'ipotesi di probabilità uniforme.

### 2.1.3 Misurazione del diametro delle sfere

Utilizziamo un calibro ventesimale. Ci assicuriamo, misurando tre diametri ortogonali, che i gravi siano effettivamente delle sfere.

### 2.1.4 Misurazione della massa delle sfere

Ci assicuriamo che la bilancia sia correttamente tarata e orizzontale.

## 2.2 Misure

| Sfera | Massa ( g $\pm$ 1 g) | Diametro (cm $\pm 5 \times 10^{-3}$ cm) |
|---|---|---|
| Per cuscinetti 1 | 16 | 1.595 |
| Per cuscinetti 2 | 12 | 1.440 |
| Biliardo | 166 | 5.700 |
| Gomma | 13 | 3.020 |
| Ping-pong | 3 | 4.025 |
| Subotto | 18 | 3.350 |

# 3 Analisi dati

## 3.1 Metodi di analisi

Per il rilevamento delle misure è stato utilizzato Tracker[1]. I dati sono stati esportati in file `.csv` e quindi elaborati con uno script in Python che lasciamo in Appendice B. Siccome l'incertezza sui tempi è molto minore di quella sulle posizioni (ipotesi richiesta dal metodo dei minimi quadrati), utilizziamo `scipy.optimize.curve_fit()` usando come ascisse l'istante di tempo associato a ciascun evento.

Lo script cerca una prima stima dei parametri in gioco per ciascun corpo (accelerazione $\vec{g}$ di gravità e condizioni iniziali) tramite un fit avente come modello un semplice moto parabolico; successivamente usa questi parametri come parametri iniziali per un ulteriore fit con un modello più raffinato che stima la forza di attrito viscoso[2]. Sui parametri ottimali del fit viene calcolato il *c.l.* dei punti con la legge oraria.

Viene quindi calcolata l'accelerazione di gravità che meglio approssima tutte quelle calcolate; siccome le varie stime di $g$ non sono tra loro compatibili, le incertezze stimate da `scipy.optimize.curve_fit()` non hanno senso fisico, quindi possono non essere considerate nella media.

Infine si confronta il valore del parametro $\beta$ stimato prima tramite la legge di Stokes e poi tramite il fit della legge oraria.

## 3.2 Incertezze

Tramite il metro a nastro calibriamo le distanze; siccome nei video non sempre è chiaramente percebibile ogni centimetro del metro, effettuiamo la calibrazione su 1 m. Ci aspettiamo che questo possa portare a un errore sistematico nella misura delle lunghezze nell'ordine di 1 mm su 1 m.

L'applicativo Tracker traccia la posizione di un oggetto in un video cercando un pattern in ciascun frame. Il pattern da noi utilizzato ha la dimensione di circa 0.5 cm. Ci aspettiamo dunque che le incertezze sulle singole misure delle posizioni siano comprese in un cerchio di quel diametro, ovvero siano nell'ordine di 0.3 cm. Ricavando il modello utilizzato abbiamo assunto che il campo di gravità sia uniforme e che il fluido mantenga viscosità costante.

La variazione durante l'esperimento del campo di gravità si può stimare rozzamente assumendo la terra omogenea e sferica e usando la Legge di Gravitazione Universale di Newton:

$$g = \frac{GM}{d^2} \quad \Delta g \simeq -2\frac{GM}{d^3}h = -2g\frac{h}{R_\mathrm{E}}$$

dove $h$ è la variazione di altezza dei gravi (nell'ordine di un paio di metri) e $R_\mathrm{E} \approx 6.4 \times 10^6$ m è il raggio medio terrestre. L'ipotesi nel nostro esperimento non è dunque verificata se l'incertezza relativa su $g$ è confrontabile o minore di $\frac{h}{R_\mathrm{E}} \approx 3.1 \times 10^{-7}$; ragionevolmente, a priori, inverosimile.

Valutare quando la viscosità è costante è invece lavoro arduo. Approssimativamente, l'attrito viscoso ha due componenti: una causata da un flusso laminare, che dà un contributo lineare alla velocità, e uno causato da un flusso vorticoso, che genera effetti proporzionali al quadrato della velocità. Si può provare a stimare il rapporto tra i due effetti con il rapporto tra la velocità del grave e la velocità quadratica media delle particelle del fluido, che nel caso dell'aria in un'abitazione è di circa $v_\mathrm{qm} \approx 5.2 \times 10^2\ \mathrm{ms}^{-1}$. Siccome ci aspettiamo che almeno l'ordine di grandezza della velocità del grave, nella caduta da circa 2 m non venga alterato dall'attrito dell'aria,

[1] `https://physlets.org/tracker/`. Segnaliamo che sul nostro PC, la versione 6.0.5 non funziona. Abbiamo utilizzato la versione 5.1.5.

[2] Avere buoni parametri iniziali si è mostrato necessario per far andare a buon fine tutti i fit.

possiamo stimare la velocità massima raggiunta dal grave con le leggi del Moto Naturalmente Accelerato: $v_{\max} \simeq \sqrt{2gh} \approx 6\,\mathrm{ms}^{-1}$. L'effetto dovuto al flusso vorticoso del fluido è trascurabile solo se l'incertezza relativa di $g$ è molto più grande del prodotto tra l'effetto dell'attrito viscoso e il rapporto $\frac{v_{\mathrm{qm}}}{v_{\max}} \approx 1 \times 10^{-2}$.

La bilancia da noi utilizzata ha sensibilità 1 g. Il calibro ha sensibilità di un ventesimo di millimetro.

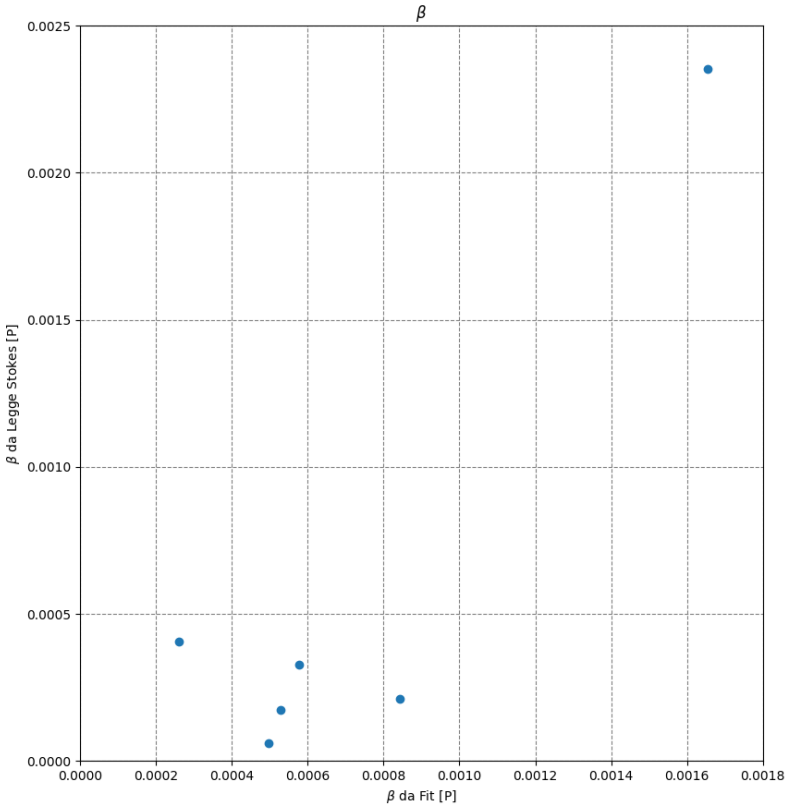
## 3.3 Risultati dell'analisi

Codice 1: Output dello script in Python.

```
### Sfera d'acciaio grande ###
g = 1060.573970 +- 2.397792
beta = 5.289384e-04 +- 2.180373e-03
c.l.: 86.718750%

### Sfera d'acciaio piccola ###
g = 1004.697859 +- 1.692684
beta = 8.434231e-04 +- 3.181383e-03
c.l.: 97.826087%

### Palla da Biliardo ###
g = 1038.890342 +- 4.020055
beta = 4.980826e-04 +- 3.600783e-03
c.l.: 36.974790%

### Pallina di gomma ###
g = 1020.578435 +- 2.001758
beta = 2.615096e-04 +- 5.396634e-04
c.l.: 80.141844%

### Pallina da Ping pong ###
g = 875.063312 +- 5.543450
beta = 1.654985e-03 +- 1.883207e-02
c.l.: 91.304348%

### Pallina da Subotto ###
g = 993.574866 +- 5.657712
beta = 5.766128e-04 +- 5.801780e-03
c.l.: 58.064516%

####################
#### Resoconto ####
####################
g = 998.896464 +- 26.627278
Correlazione tra beta:
[[1.         0.89606114]
 [0.89606114 1.        ]]
```

Le equazioni orarie seguono l'equazione prevista, come mostrano gli elevati *confidence levels*. I residui in tutti i casi si dispongono praticamente in modo uniforme attorno allo zero, indice del fatto che il modello adottato è adatto. Riportiamo per completezza il grafico relativo al moto di ciascuna sferetta.

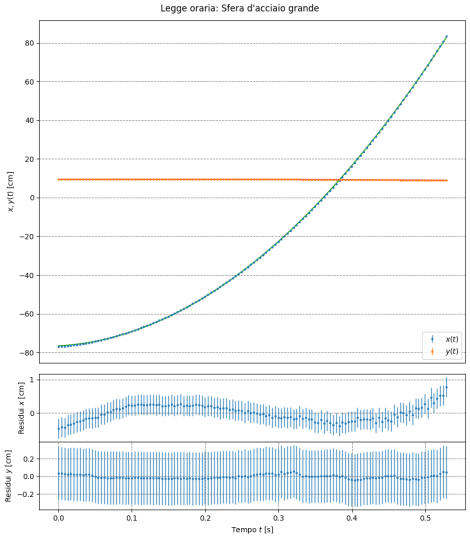
Tutti i valori dei coefficienti $\beta$ trovati sono compatibili con 0: questo perché la strumentazione non era adatta a misurare tale coefficiente e l'effetto dell'attrito viscoso è appena visibile. Tuttavia confrontando questi valori con quelli calcolati tramite la legge di Stokes, come fatto nel seguente grafico, emerge una forte correlazione (0.90) tra i valori calcolati nei due modi (per il calcolo tramite la legge di Stokes abbiamo usato come viscosità dell'aria $\mu = 18.6 \times 10^{-5}\,\mathrm{P}$); inoltre tali grandezze assumono il medesimo ordine di grandezza per ciascuna sferetta.

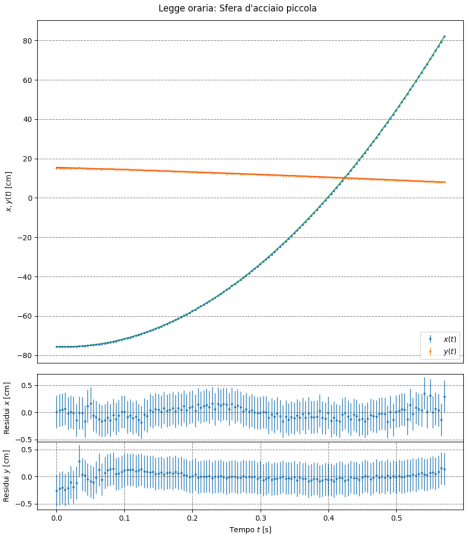


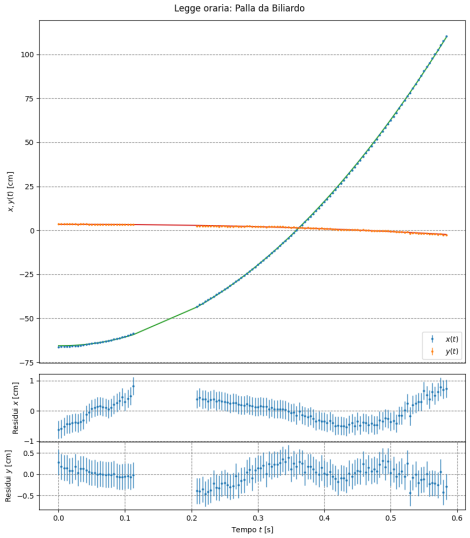

Il valore di $g$ trovato

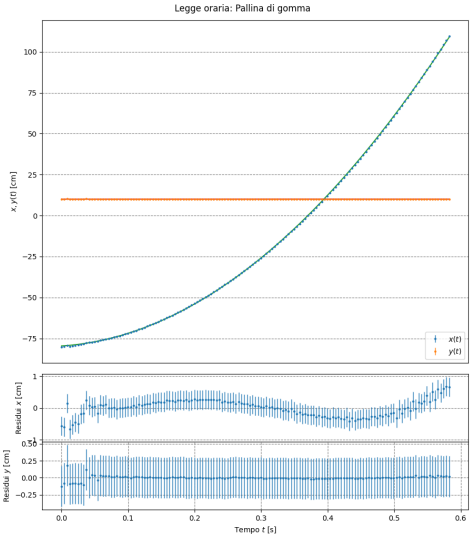
$$g = (999 \pm 27)\,\mathrm{Gal}$$

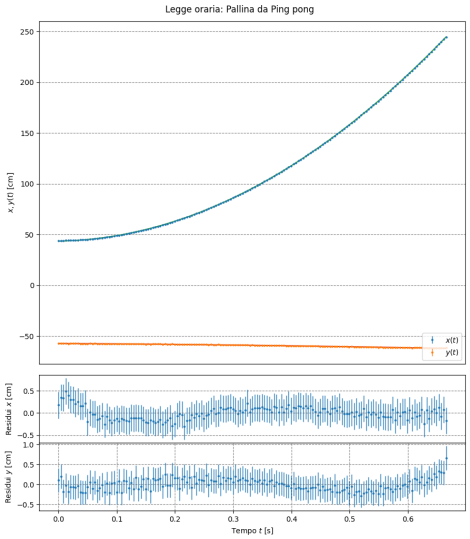
è compatibile col valore atteso (981 Gal).
Le ipotesi adottate nel modello sussistono: infatti la massima accelerazione causata dalla forza d'attrito viscoso è nell'ordine di grandezza di $\beta v_{\max} \approx 1.5\,\mathrm{Gal}$; possiamo stimare quindi la componente non lineare della forza d'attrito compiere un'accelerazione nell'ordine di 0.015 Gal, del tutto trascurabile.

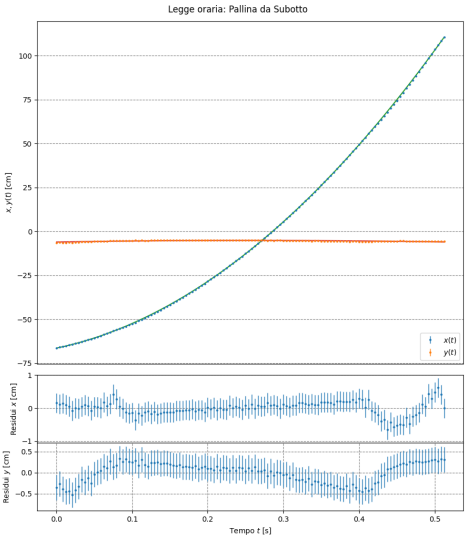

Legge oraria: Sfera d'acciaio grande
x, y(t) [cm]
x(t)
y(t)
Residui x [cm]
Residui y [cm]
Tempo t [s]


Legge oraria: Sfera d'acciaio piccola
x, y(t) [cm]
x(t)
y(t)
Residui x [cm]
Residui y [cm]
Tempo t [s]


Legge oraria: Palla da Biliardo
x, y(t) [cm]
x(t)
y(t)
Residui x [cm]
Residui y [cm]
Tempo t [s]


Legge oraria: Pallina di gomma
x, y(t) [cm]
x(t)
y(t)
Residui x [cm]
Residui y [cm]
Tempo t [s]


Legge oraria: Pallina da Ping pong
x, y(t) [cm]
x(t)
y(t)
Residui x [cm]
Residui y [cm]
Tempo t [s]


Legge oraria: Pallina da Subotto
x, y(t) [cm]
x(t)
y(t)
Residui x [cm]
Residui y [cm]
Tempo t [s]

# 4 Conclusioni

Abbiamo verificato che diversi corpi cadono spinti da una forza gravitazionale proporzionale alla loro massa inerziale. Abbiamo calcolato che l'accelerazione di gravità $g$ in prossimità della superficie terrestre, a Pisa, vale

$$g = (999 \pm 27)\,\mathrm{Gal}$$

Abbiamo notato che corpi con poca massa e grande dimensione vengono maggiormente frenati nel loro moto dalla forza di attrito viscoso dell'aria e abbiamo verificato che nelle usuali condizioni tale forza è linearmente proporzionale alla velocità, a meno di un coefficiente che dipende anche dalle proprietà dell'aria stessa. Tale coefficiente è nell'ordine di $2 \times 10^{-4}\,\mathrm{P}$ (compatibilmente con quanto affermato dalla Legge di Stokes).

# A Appendice: Legge oraria di un grave in un fluido newtoniano.

Si consideri un grave sferico di massa $m$ e raggio $r$ in un campo di gravità uniforme e costante $\vec{g}$ che si muove in un fluido newtoniano, anch'esso uniforme e fermo. Sia $x(t)$ la posizione del centro della sfera in funzione del tempo. Sul grave agiscono quindi le sole forze peso $m\vec{g}$ e viscoso, che si può stimare tramite la formula di Stokes: $-6\pi\mu r\dot{\vec{x}}$.

Si prenda un set di coordinate $Oxyz$ rispetto a una base ortonormale. Per la seconda legge di Newton:

$$m\ddot{\vec{x}} = m\vec{g} - 6\pi\mu r\dot{\vec{x}}$$

$$\begin{cases} \ddot{x} = -g_x - \frac{6\pi\mu r}{m}\dot{x} \\ \ddot{y} = -g_y - \frac{6\pi\mu r}{m}\dot{y} \\ \ddot{z} = -g_z - \frac{6\pi\mu r}{m}\dot{z} \end{cases}$$

Le equazioni possono essere risolte prima integrando $LHS$ e $RHS$ e poi tramite il metodo di separazione delle variabili. Le costanti di integrazione si ricavano dalle condizioni iniziali. Per brevità definiremo $\beta = \frac{6\pi\mu r}{m}$.

$$\begin{cases} \dot{x} = \dot{x}_0 - g_x t - \beta x \\ \dot{y} = \dot{y}_0 - g_y t - \beta y \\ \dot{z} = \dot{z}_0 - g_z t - \beta z \end{cases} \quad \begin{cases} x(t) = x_0 + \frac{g_x}{\beta}t + \frac{\dot{x}_0\beta + g_x}{\beta^2}(1 - e^{-\beta t}) \\ y(t) = y_0 + \frac{g_y}{\beta}t + \frac{\dot{y}_0\beta + g_y}{\beta^2}(1 - e^{-\beta t}) \\ z(t) = z_0 + \frac{g_z}{\beta}t + \frac{\dot{z}_0\beta + g_z}{\beta^2}(1 - e^{-\beta t}) \end{cases}$$

# B Appendice: Codici Python utilizzati

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

mar_size = 2
bar_l    = 1

# DEFINIZIONI LEGGE ORARIA PER FIT
# funzione
def s(T, gx, gy, x0, y0, dx0, dy0, beta):
    t, x, y = T
    Dx = x0 + gx/beta*t + ( dx0/beta - gx/beta**2 )\
        * ( 1 - np.exp(-t*beta) ) - x
    Dy = y0 + gy/beta*t + ( dy0/beta - gy/beta**2 )\
        * ( 1 - np.exp(-t*beta) ) - y
    return (Dx**2+Dy**2)**0.5

def s0x(t, g, x0, v0):
    return x0 + v0*t + 0.5*g*t**2

def sx(t,g,x0,dx0,beta):
    return x0 + g/beta*t + ( dx0/beta - g/beta**2 )\
        * ( 1 - np.exp(-t*beta) )

# IMPORTA I DATI
T      = [0, 0, 0, 0, 0, 0]
```

```
gg     = [0, 0, 0, 0, 0, 0]
sgg    = [0, 0, 0, 0, 0, 0]
bbeta  = [0, 0, 0, 0, 0, 0]
sbbeta = [0, 0, 0, 0, 0, 0]

T[0] = np.loadtxt(  "acciaiosx.csv", delimiter=';',
        skiprows=2, unpack=True  )  # cuscinetto grande
T[1] = np.loadtxt(  "acciaiodx.csv", delimiter=';',
        skiprows=2, unpack=True  )  # cuscinetto piccolo
T[2] = np.loadtxt(  "biliardo.csv",  delimiter=';',
        skiprows=2, unpack=True  )  # biliardo
T[3] = np.loadtxt(  "gomma.csv",     delimiter=';',
        skiprows=2, unpack=True  )  # gomma
T[4] = np.loadtxt(  "ping_pong.csv", delimiter=';',
        skiprows=2, unpack=True  )  # ping pong
T[5] = np.loadtxt(  "subotto.csv",   delimiter=';',
        skiprows=2, unpack=True  )  # subotto

for i in T:
    i[0] = np.array(i[0])-i[0][0]
    # uso cgs
    i[0] /= 8.
    i[1] *= 100.
    i[2] *= 100.

desc = [
        "Sfera d'acciaio grande",
        "Sfera d'acciaio piccola",
        "Palla da Biliardo",
        "Pallina di gomma",
        "Pallina da Ping pong",
        "Pallina da Subotto"
       ]

# CALCOLA IL FIT
for i in range(6):
    print(" ### %s ###" % desc[i])
    # prima stima valori di best fit
    poptx, pcovx = curve_fit(s0x,T[i][0],T[i][1])
    popty, pcovy = curve_fit(s0x,T[i][0],T[i][2])

    gx,  x0,  vx0  = poptx
    sgx, sx0, svx0 = (pcovx.diagonal())**0.5
    gy,  y0,  vy0  = popty
    sgy, sy0, svy0 = (pcovy.diagonal())**0.5

    # migliramento precedente stima
    popt, pcov = curve_fit(
        s,T[i],np.zeros_like(T[i][0]),
        sigma=0.3*np.ones_like(T[i][0]),
        p0=[
            gx, gy, x0, y0, vx0, vy0, 1e-3
           ],
        bounds=(
           [
               gx-5*sgx, gy-5*sgy,
               x0-5*sx0, y0-5*sy0,
               vx0-5*svx0, vy0-5*svy0,
               1e-4
           ],
           [
               gx+5*sgx, gy+5*sgy,
```

```
                x0+5*sx0, y0+5*sy0,
                vx0+5*svx0, vy0+5*svy0,
                2e-2
            ]
        )
    )

    g  = (popt[0]**2+popt[1]**2)**0.5
    sg = ( (popt[0]/g)**2*pcov[0][0]\
            + (popt[1]/g)**2*pcov[1][1] )**0.5

    beta  = popt[6]
    sbeta = pcov[6][6]**0.5

    print("g = %f +- %f\nbeta = %e +- %e" %
            (g,sg,beta,sbeta)
         )

    xt = sx(T[i][0],popt[0],popt[2],popt[4],popt[6])
    yt = sx(T[i][0],popt[1],popt[3],popt[5],popt[6])

    # CALCOLO DELL'INTERVALLO DI CONFIDENZA
    d = ((T[i][1]-xt)**2+(T[i][2]-yt)**2)**0.5
    print("c.l.: %f%%\n" %
            (100.*np.count_nonzero( d < .3 )/len(d)) )

    gg[i]     = g
    sgg[i]    = sg
    bbeta[i]  = beta
    sbbeta[i] = sbeta

    fig = plt.figure()
    fig.suptitle('Legge oraria: '+desc[i])

    fig.add_axes((0.1, 0.32, 0.85, 0.63))
    plt.errorbar(T[i][0],T[i][1],yerr=0.3,fmt='o',
            markersize=mar_size,linewidth=bar_l,label=r'$x(t)$')
    plt.errorbar(T[i][0],T[i][2],yerr=0.3,fmt='o',
            markersize=mar_size,linewidth=bar_l,label=r'$y(t)$')
    plt.plot(T[i][0],xt,T[i][0],yt)
    plt.ylabel(r'$x,y(t)$ [cm]')
    plt.gca().get_xaxis().set_visible(False)
    plt.grid(which='both', ls='dashed', color='gray')
    plt.legend(loc='lower right')

    fig.add_axes((0.1, 0.175, 0.85, 0.125))
    plt.errorbar(T[i][0],T[i][1]-xt,yerr=0.3,
            fmt='o',markersize=mar_size,linewidth=bar_l)
    plt.grid(which='both', ls='dashed', color='gray')
    plt.ylabel(r'Residui $x$ [cm]')
    plt.gca().get_xaxis().set_visible(False)

    fig.add_axes((0.1, 0.05, 0.85, 0.125))
    plt.errorbar(T[i][0],T[i][2]-yt,yerr=0.3,
            fmt='o',markersize=mar_size,linewidth=bar_l)
    plt.grid(which='both', ls='dashed', color='gray')
    plt.ylabel(r'Residui $y$ [cm]')
    plt.xlabel(r'Tempo $t$ [s]')

    plt.show()

# MEDIA ACCELERAZIONI g OTTENUTE
```

```
153 popt, pcov = curve_fit(lambda x,g:x-g,gg,np.zeros_like(gg))
154
155 print("""\
156  ####################
157  #### Resoconto ####
158  ####################\
159       """)
160 print( "g = %f +- %f" % (popt[0],pcov[0][0]**0.5) )
161
162 # COVARIANZA TRA beta ASPETTATO E PREVISTO
163 # diametri e masse sfere
164 d = np.array([ 1.595, 1.440, 5.700, 3.020, 4.025, 3.350 ])
165 m = np.array([ 16,    12,    166,   13,    3,     18    ])
166 beta_stokes = 3.*math.pi*d/m*18.6e-5
167
168 print("Correlazione tra beta:\n%s\n"
169         % (np.corrcoef(bbeta,beta_stokes)))
170
171 plt.title(r'$\beta$')
172 plt.errorbar(bbeta,beta_stokes,fmt='o')
173 plt.ylabel(r'$\beta$ da Legge Stokes [P]')
174 plt.xlabel(r'$\beta$ da Fit [P]')
175 plt.ylim([0,0.0025])
176 plt.xlim([0,0.0018])
177 plt.grid(which='both', ls='dashed', color='gray')
178 plt.show()
```

Codice 2: Esegue la regressione in funzione dei valori in alcuni file `.csv`