Esercizio 1

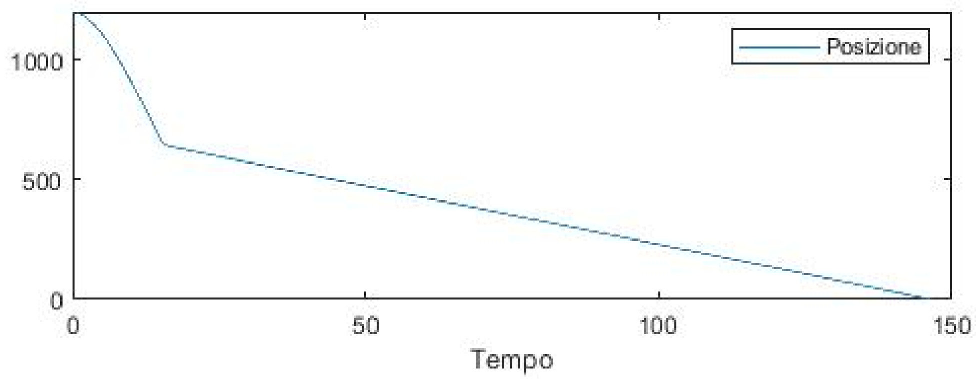
La seguente function restituisce il tempo che il paracadutista impega ad atterrare, risolvendo l'ovvia equazione del moto del paracadutista con il metodo di Runge-Kutta. Tale equazione è la seguente:

$$y''(x) = -\frac{k_{1/2}}{m} y'(x) - g$$

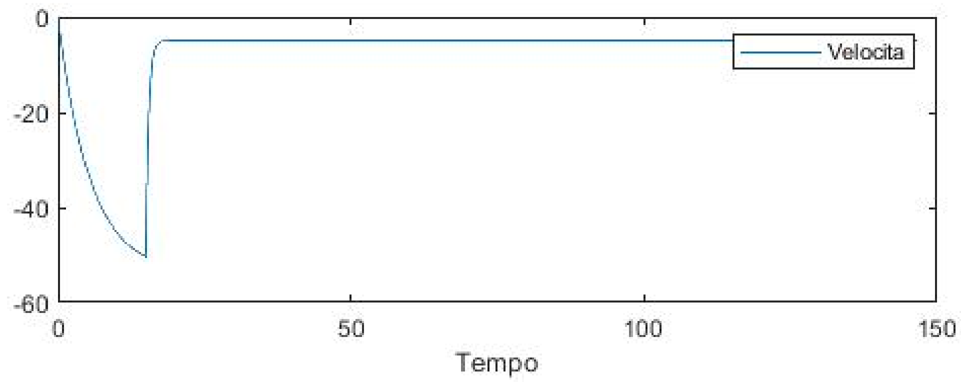
con k uguale al valore corrispondente alla fase in cui si trova il paracadutista (con o senza paracadute). Abbiamo cercato l'intervallo di integrazione opportuno per ottenere il punto di atterraggio, che si rivelerà essere poco sotto a 150,per cui abbiamo scelto 150 come estremo dell'intervallo di integrazione.

```matlab
function xmax=paracadutista(h)
k1=16.4;
k2=180;
g=9.81;
m=90;
xp=15;
alt=1200;

f1=@(x,y) [y(2); -(k1/m)*y(2)-g];
[x1,y1]=RK4(f1, [0, xp], [alt; 0], h);
n=length(x1);
f2=@(x,y) [y(2); -(k2/m)*y(2)-g];
[x2,y2]=RK4(f2, [xp, 150], [y1(1,n); y1(2,n)], h);

y=[y1,y2];
x=[x1,x2];
l=find(y(1,:)<0,1);
xmax=x(l);

subplot(2,1,1)
plot(x(1:l),y(1,1:l))
axis([0 xmax 0 1200])
legend('Posizione')
xlabel('Tempo')

subplot(2,1,2)
plot(x(1:l),y(2,1:l))
legend('Velocita')
xlabel('Tempo')
end
```

La function riporta come tempo di atterraggio $x_{max} = 146.700$. Facendo calcolare a matlab anche $v_{max} = y_2(l)$ la velocità al tempo di atterraggio otteniamo -4.9050, che rimane costante da quando è trascorso il transiente post apertura del paracadute in poi, come si nota dal grafico seguente.



Posizione
1000
500
0
0
50
100
150
Tempo



Velocita
0
-20
-40
-60
0
50
100
150
Tempo

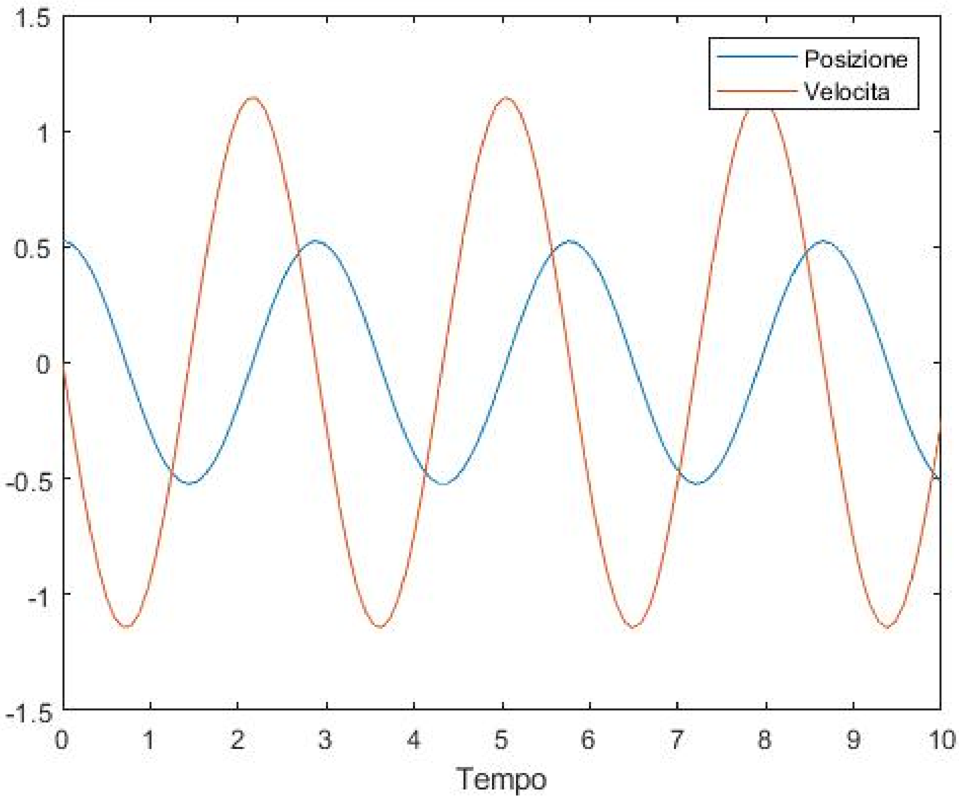
Esercizio 2

L'equazione che dobbiamo risolvere mediante `ode45` è la seguente:

$$y'' = \frac{g}{l} sin(y)$$

e ne grafichiamo posizione e velocità

```
y0=pi/6;
y1=0;
l=2;
f=@(t,y) [y(2);-9.81/l*sin(y(1))];
[t,sol]=ode45(f,[0 10],[y0;y1]);
plot(t,sol(:,1),t,sol(:,2));
legend('Posizione','Velocita');
xlabel('Tempo')
```



Esercizio 3

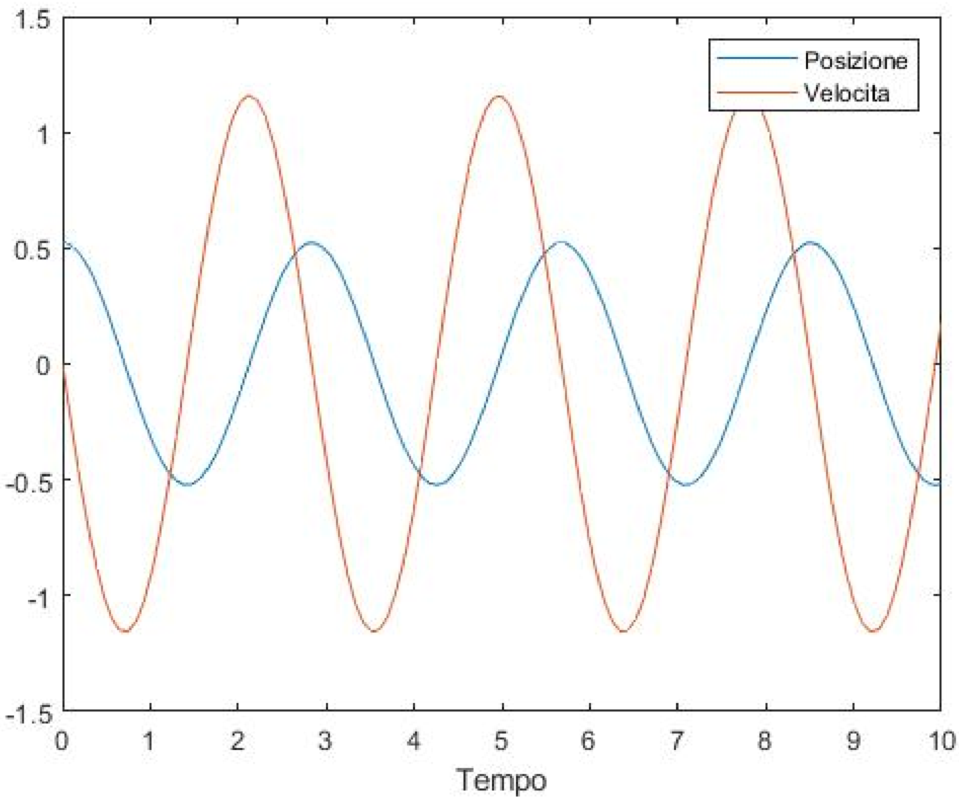
L'equazione che dobbiamo risolvere mediante `ode45` è la seguente:

$$y'' = \frac{g}{l}y$$

e ne grafichiamo posizione e velocità

```
y0=pi/6;
y1=0;
l=2;
f=@(t,y) [y(2);-9.81/l*y(1)];
[t,sol]=ode45(f,[0 10],[y0;y1]);
plot(t,sol(:,1),t,sol(:,2));
legend('Posizione','Velocita');
xlabel('Tempo')
```

Non notiamo differenze significativa (almeno ad occhio nudo) tra questo modello e quello precedente.



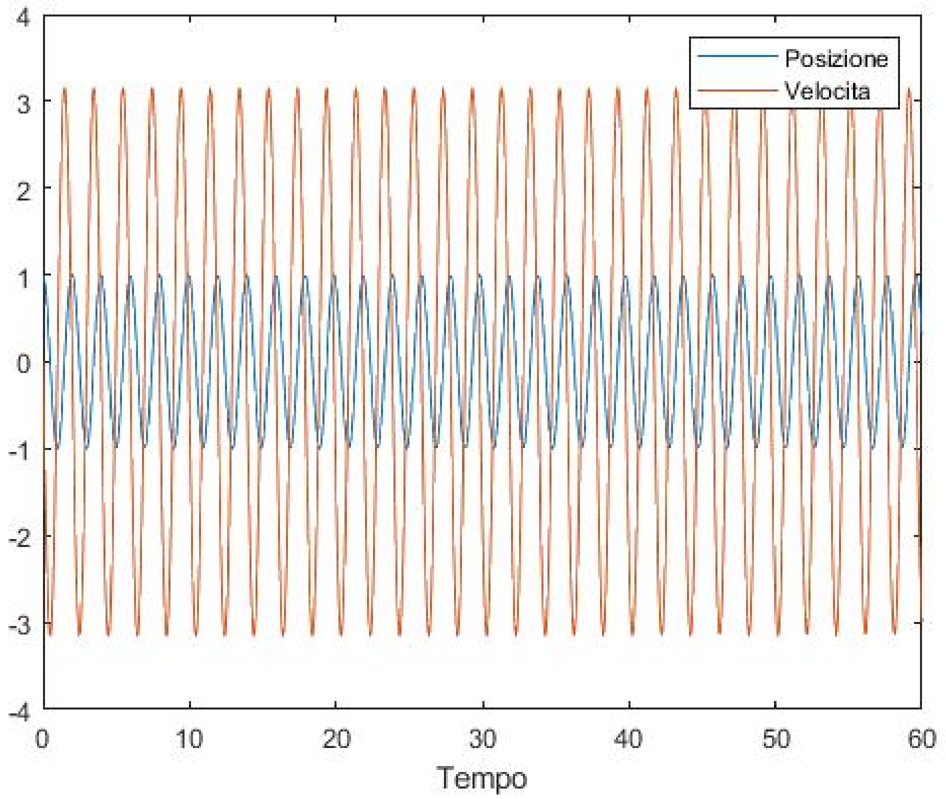
Esercizio 4

Per prima cosa scriviamo una function che, assegnati massa, costante elastica, costante di smorzamento, forza applicata, intervallo di tempo e valori iniziali plotti posizione e velocità dell'oscillatore armonico con i dati in questione:

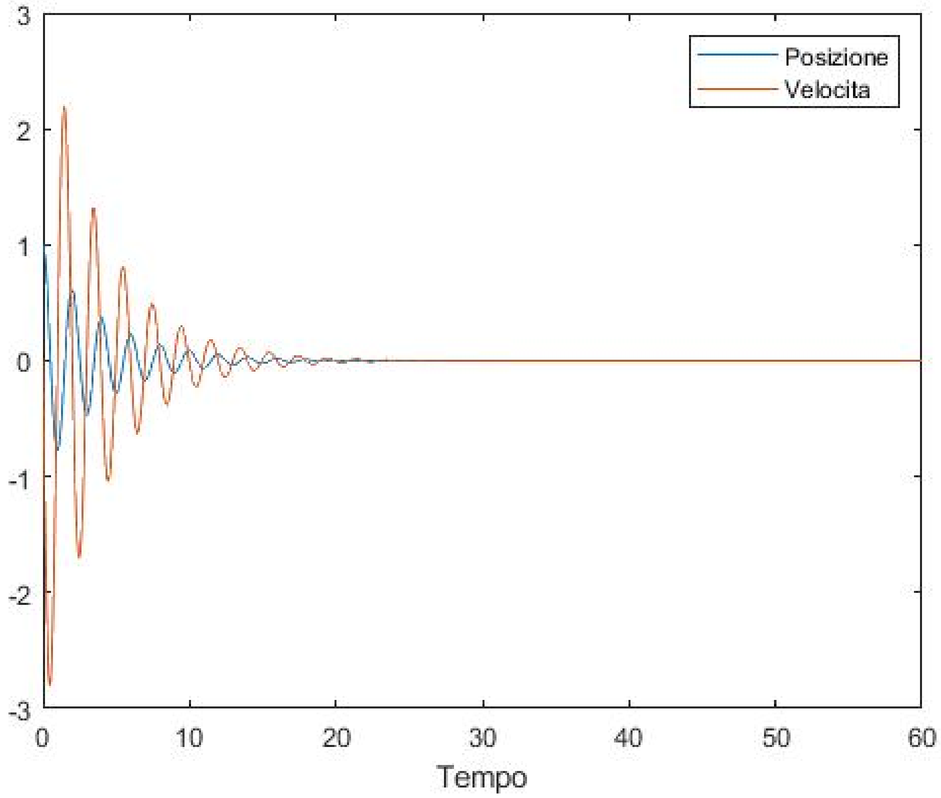
```
function []=oscillatore(m,h,k,F,intervallo,y0,y1)
f=@(t,y) [y(2);-h/m*y(1)-k/m*y(2)+F];
[t,sol]=ode45(f,intervallo,[y0;y1]);
plot(t,sol(:,1),t,sol(:,2));
legend('Posizione','Velocita');
xlabel('Tempo')
end
```

Scriviamo uno script che consenta di scegliere il caso test con un ciclo `switch`. Abbiamo usato `menu` perchè sembrava di implementazione enormemente più compatta, anche se probabilmente la differenza è minima.

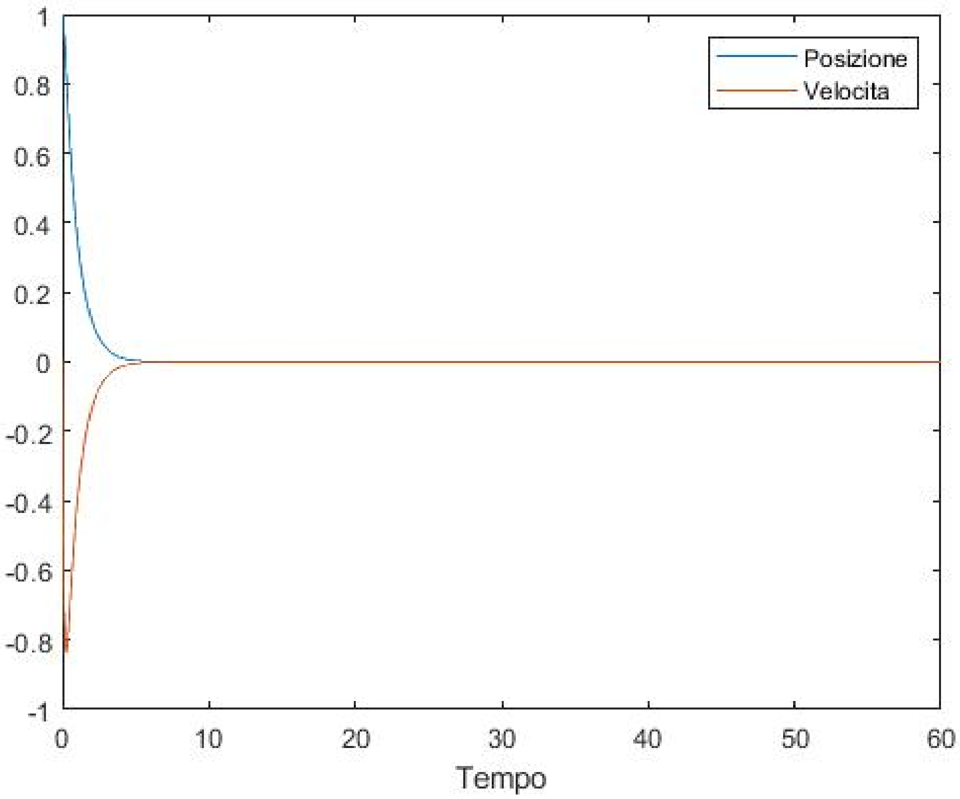
```
choice = menu('Scegli un tipo di oscillatore:','Libero non smorzato','Libero
    sottosmorzato','Libero sovrasmorzato','Forzato smorzato');
switch choice
    case 1
        oscillatore(1,10,0,0,[0 60],1,0);
    case 2
        oscillatore(1,10,0.5,0,[0 60],1,0);
    case 3
        oscillatore(1,10,10,0,[0 60],1,0);
    case 4
        oscillatore(2,15,0.75,25,[0 60],2,0);
end
```



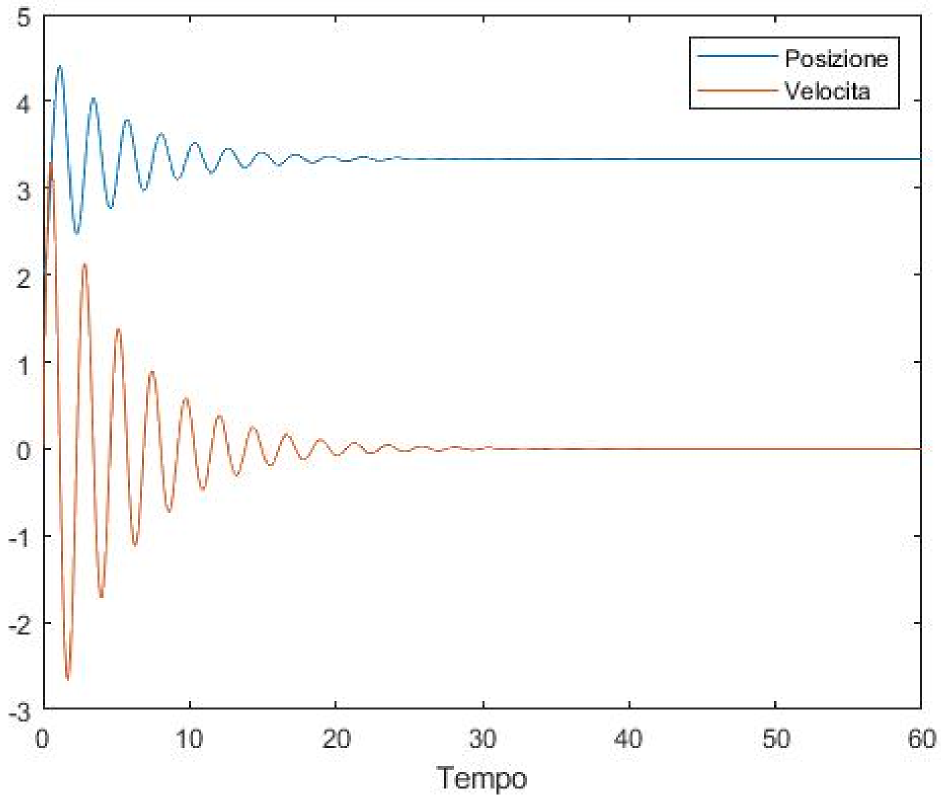
Libero non smorzato



Libero sottosmorzato



Libero sovrasmorzato



Forzato smorzato