Consider:

A ball launched straight upwards in the presence of gravity

$$\frac{dp_y}{dt} = -mg$$

$$m\frac{dv}{dt} = -g$$

$$\frac{dv}{dt} = -g$$

We could solve analytically or w/ a Riemann Sum

Let's use Euler's method instead

```
import numpy as np
import matplotlib.pyplot as plt

G = 9.8

def dvdt():
    return -G

vi = 20
dt = 0.01
ti = 0
tf = 5
t = 0
v = vi
vlist =[v]
tlist = [t]
while t <= tf:
    vprime = dvdt()
    v = v + vprime * dt
    t+=dt
    tlist.append(t)
    vlist.append(v)
plt.plot(tlist,vlist)
plt.show()
```

```
import numpy as np
import matplotlib.pyplot as plt

G = 9.8

def dvdt():
    return -G

vi = 20
dt = 0.01
ti = 0
tf = 5
#Array containing all values of t
t = np.arange(ti,tf,dt)
#Empty array for velocity
v = np.zeros_like(t)
v[0] = vi
for i in range(1,t.size):
    vprime = dvdt()
    v[i] = v[i-1] + vprime * dt
plt.plot(t,v)
plt.show()
```

Question: What if I want $y(t)$, not just $v(t)$?

Solution:

$$v = \frac{dy}{dt}$$

If we know y at time t

$$y(t+\Delta t) \approx y(t) + v\Delta t$$

Procedure:

Start with v_i, y_i

$$v(t_i + \Delta t) \approx v_i + p'(t_i)\Delta t$$

use this v, $v(t_i+\Delta t)$, to estimate $y(t_i+\Delta t)$

$$y(t_i+\Delta t) \approx y_i + v(t_i+\Delta t)\Delta t$$

In general: $v_n = v(t_n) = v(t_i + n\Delta t)$

$y_n = y(t_n)$

$$v_{n+1} = v_n + v_n' \Delta t$$

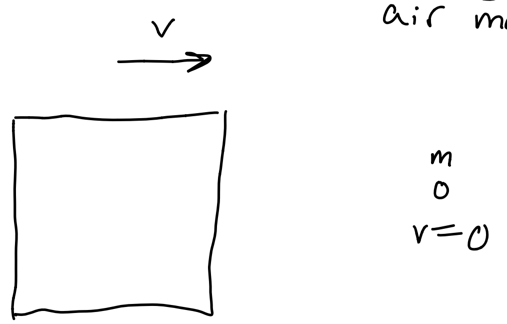
$$y_{n+1} = y_n + v_{n+1}\Delta t$$

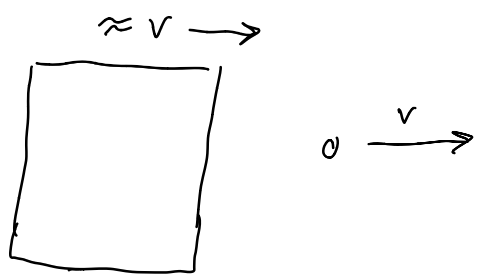
```
import numpy as np
import matplotlib.pyplot as plt

G = 9.8

def dvdt():
    return -G

vi = 20
yi = 10
dt = 0.01
ti = 0
tf = 5
t = np.arange(ti,tf,dt)
v = np.zeros_like(t)
y = np.zeros_like(t)
v[0] = vi
y[0] = yi
for i in range(1,t.size):
    vprime = dvdt()
    v[i] = v[i-1] + vprime * dt
    y[i] = y[i-1] + v[i] * dt

plt.plot(t,y)
plt.show()
```

Let's make it more interesting

Air resistance

Consider: A sliding block collides with an air molecule of mass m





Δp: $p_i = p_f + mv$

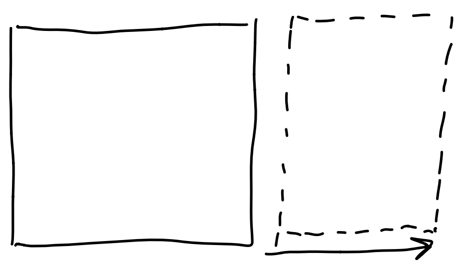
$\Delta p = -mv$

Each collision, the block loses an amount mv of momentum

m is very tiny, $\approx 5 \times 10^{-26}$ kg

But there are many collisions (lots of air molecules)

How many collisions in a time Δt?



$\overline{V} = A v \Delta t$, A = Area of front of box

$\Delta x = v \Delta t$

In a time Δt, boxes moves through a volume of air $\overline{V} = A v \Delta t$

How many atoms does it hit during this time?

$$\# \text{ of atoms in volume } \overline{V} = \frac{\text{mass of } \overline{V}}{m} = \frac{\rho_{air} \overline{V}}{m}$$

$$\# \text{ of collisions} = \frac{\rho_{air} \overline{V}}{m}$$

$$\Delta p / \text{collision} = -mv$$

$$\Delta P_{tot} = -\frac{\rho_{air} \overline{V}}{m} m v = -\rho_{air} \overline{V} v = -\rho_{air} A v \Delta t \, v$$

$$\frac{\Delta p}{\Delta t} = -\rho_{air} A v^2$$

$$F_{air} = -\rho_{air} A v^2$$

$$F_{air} = -\frac{1}{2} C \rho A v^2$$

$$\frac{dp}{dt} = -\frac{1}{2} C \rho A v^2$$

C: unitless constant to account for shape

A: Cross sectional area (not total surface area)

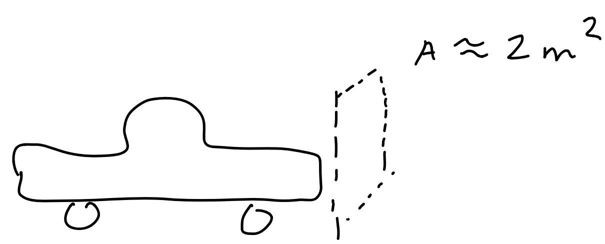
ρ: density $\left(\frac{kg}{m^3}\right)$ of material

Coasting car:

$C = 1$

$A = 2 m^2$

$\rho = 1.2 \frac{kg}{m^3}$



```
import numpy as np
import matplotlib.pyplot as plt

G = 9.8
rho = 1.2
A = 2
C = 1
mass = 1100

def dvdt(v,C,A,rho,mass):
    return -C * A * rho * np.abs(v) * v /mass

vi = 10
yi = 0
dt = 0.01
ti = 0
tf = 100
t = np.arange(ti,tf,dt)
v = np.zeros_like(t)
y = np.zeros_like(t)
v[0] = vi
y[0] = yi
for i in range(1,t.size):
    vprime = dvdt(v[i-1],C,A,rho,mass)
    v[i] = v[i-1] + vprime * dt
    y[i] = y[i-1] + v[i] * dt
```

```
import numpy as np
import matplotlib.pyplot as plt

G = 9.8
rho = 1.2
A = 2
C = 1
mass = 75

def dvdt(v,C,A,rho,mass):
    return -C * A * rho * v * np.abs(v) / mass  - G

vi = 0
yi = 10
dt = 0.01
ti = 0
tf = 5
t = np.arange(ti,tf,dt)
v = np.zeros_like(t)
y = np.zeros_like(t)
v[0] = vi
y[0] = yi
for i in range(1,t.size):
    vprime = dvdt(v[i-1],C,A,rho,mass)
    v[i] = v[i-1] + vprime * dt
    y[i] = y[i-1] + v[i] * dt
```

2D:

$$\frac{dV_x}{dt} = 0$$

$$\frac{dV_y}{dt} = -g$$

$$V_{x,n+1} = V_{x,n} + \frac{dV_{x,n}}{dt} \Delta t$$

$$V_{y,n+1} = V_{y,n} + \frac{dV_{y,n}}{dt} \Delta t$$

$$X_{n+1} = X_n + V_{x,n+1} \Delta t$$

$$y_{n+1} = y_n + V_{y,n+1} \Delta t$$

```
import numpy as np
import matplotlib.pyplot as plt

G = 9.8

def dvxdt():
    return 0
def dvydt():
    return -G

vi = 45
theta = 30
yi = 10
dt = 0.01
ti = 0
tf = 5
t = np.arange(ti,tf,dt)
vx = np.zeros_like(t)
vy = np.zeros_like(t)
x = np.zeros_like(t)
y = np.zeros_like(t)
vx[0] = vi * np.cos( np.deg2rad(theta) )
vy[0] = vi * np.sin( np.deg2rad(theta) )
y[0] = yi
for i in range(1,t.size):
    vxprime = dvxdt()
    vyprime = dvydt()
    vx[i] = vx[i-1] + vxprime * dt
    vy[i] = vy[i-1] + vyprime * dt
    x[i] = x[i-1] + vx[i] * dt
    y[i] = y[i-1] + vy[i] * dt
```

In 2D

$$\vec{F} = -\frac{1}{2} C A \rho v^2 \hat{v}$$

$$\hat{v} = \frac{1}{v} \langle v_x , v_y \rangle$$

$$F_x = -\frac{1}{2} C A \rho v^2 \frac{v_x}{v}$$

$$F_x = -\frac{1}{2} C A \rho v v_x$$

$$F_y = -\frac{1}{2} C A \rho v v_y$$

$$F_{x,net} = -\frac{1}{2} C A \rho v v_x$$

$$F_{y,net} = -mg - \frac{1}{2} C A \rho v v_y$$

$$\frac{dv_x}{dt} = -\frac{1}{2m} C A \rho v v_x$$

$$\frac{dv_y}{dt} = -g - \frac{1}{2m} C A \rho v v_x$$