Problems Elementary Mechanical Using Python

June 16th 2022

Chapter 2

First time

Problem 2.1 (Seconds) *For this problem we have to*

1. *Write a script that calculates the number of seconds, s, given the number of hours, h, according to the formula $s = 3600\ h$*

2. *Use the script to find the number of seconds in 1.5, 12 and 24 h*

Sol.

```
import numpy as np

def seconds(h):
    return 3600*h

if __name__ == '__main__':
    hours = np.array([1.5,12,24])
    for h in hours:
        print(f'{h} hours is equivalent {seconds(h)} seconds')
```

```
1.5 hours is equivalent 5400.0 seconds
12.0 hours is equivalent 43200.0 seconds
24.0 hours is equivalent 86400.0 seconds
```

Problem 2.2 (Spherical mass) *For this problem we have to*

1. *Write a script that calculates the mass of a sphere given its radius r and mass density ρ according to the formula $m = (4\pi/3)\rho r^3$.*

2. *Use the script to find the mass of a sphere of steel of radius $r = 1$ mm, $r = 1$ m and $r = 10$ m.*

Sol.

```
import numpy as np

def mass(rho,r):
    return (4*np.pi/3)*(rho)*(r**3)

def run():
```

```python
    rho = 8000
    print(f'The sphere of steel with density {rho} kg/m3')
    radius = np.array([1e-003,1,10])
    for r in radius:
        print(f'The sphere with {r} m of radius has {mass(rho,r)} kg')

if __name__=='__main__':
    run()
```

```
The sphere of steel with density 8000 kg/m3
The sphere with 0.001 m of radius has 3.351032163829113e-05 kg
The sphere with 1.0 m of radius has 33510.32163829113 kg
The sphere with 10.0 m of radius has 33510321.638291128 kg
```

Problem 2.3 (Angle) *For this place we have to*

1. *Write a function that for a point* (x, y) *returns the angle* θ *from the* x*-axis using the formula* $\theta = \arctan(y/x)$.
2. *Find the angles* θ *for the points* $(1, 1)$, $(-1, 1)$, $(-1, -1)$, $(1, -1)$.
3. *How would you change the function to return values of* θ *in the range* $[0, 2\pi]$*?*

Problem 2.4 (Unit vector) *For this problem we have to*

1. *Write a function that returns the two-dimensional unit vector,* (u_x, u_y)*, corresponding to an angle* θ *with the* x*-axis. You can use the formula* $(u_x, u_y) = (\cos\theta, \sin\theta)$*, where* θ *is given in radians.*
2. *Find the unit vectors for* $\theta = 0, \pi/6, \pi/3, \pi/2, 3\pi/2$.
3. *Rewrite the function to instead take the argument* θ *in degrees.*

Sol.

```python
import numpy as np

def unit_vector(angle):
    return np.cos(angle), np.sin(angle)

def run():
    angles = [0,np.pi/6,np.pi/3,np.pi/2,3*np.pi/2]
    for angle in angles:
        print(f'The unit vectors for {angle} radiants are {unit_vector(angle)}')

if __name__ == '__main__':
    run()
```

```
The unit vectors for 0 radiants are (1.0, 0.0)
The unit vectors for 0.5235987755982988 radiants are (0.86602540378, 0.49999999999)
The unit vectors for 1.0471975511965976 radiants are (0.50000000000, 0.86602540378)
The unit vectors for 1.5707963267948966 radiants are (6.123233995736766e-17, 1.0)
The unit vectors for 4.71238898038469 radiants are (-1.8369701987210297e-16, -1.0)
```

Problem 2.5 (Plotting the normal distribution) *The normal distribution, often called the Gaussian distribution, is given as:*

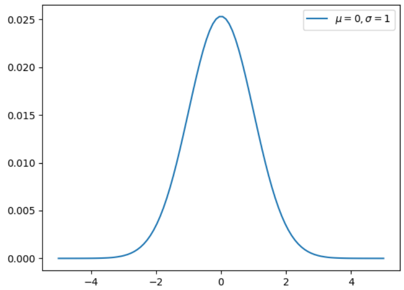
$$P(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma)^2} \tag{2.1}$$

Where mu is the average and sigma is the standard deviation.

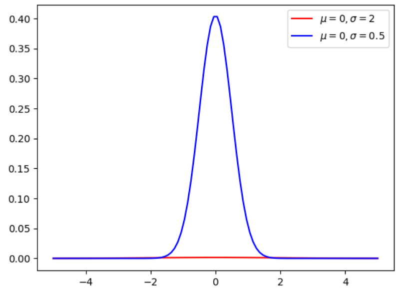
1. *Make a function* $normal(x, mu, sigma)$ *that returns the normal distribution value* $P(x, \mu, \sigma)$ *as given by the formula 2.1.*
2. *Use this function to plot the normal distribution for* $-5 < x < 5$ *for* $\mu = 0$ *and* $\sigma = 1$.
3. *Plot the normal distribution for* $-5 < x < 5$ *for* $\mu = 0$ *and* $\sigma = 2$ *and* $\sigma = 0.5$ *in the same plot.*
4. *Plot the normal distribution* $-5 < x < 5$ *for* $\sigma = 1$ *and* $\mu = 0, 1, 2$ *in three subplots above each other.*

Sol.

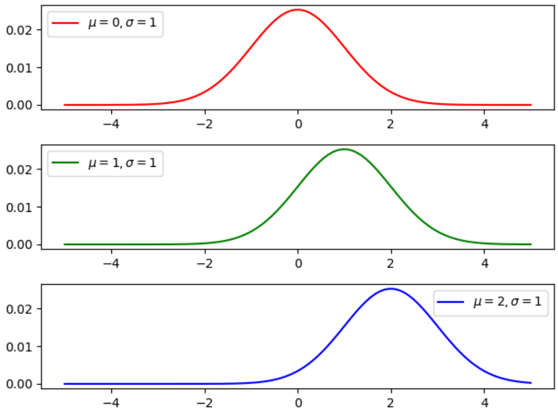
```
import numpy as np
import matplotlib.pyplot as plt


def normal(x,mu,sigma):
    return (1/np.square(2*np.pi*(sigma**2)))*np.exp(-((x-mu)**2)/(2*(sigma**2)))

def run_b(x):
    fig, axes =plt.subplots()
    axes.plot(x,normal(x,0,1), label='$\mu=0,\sigma=1$')
    axes.legend()
    fig.savefig('Document/img/chapter2/2-5/2_5_plot_a.png')

def run_c(x):
    fig, axes = plt.subplots(1,1)
    axes.plot(x, normal(x,0,2), 'r', label='$\mu=0,\sigma=2$')
    axes.plot(x, normal(x,0,0.5), 'b', label='$\mu=0,\sigma=0.5$')
    axes.legend()
    fig.savefig('Document/img/chapter2/2-5/2_5_plot_b.png')

def run_d(x):
    fig, (ax1,ax2,ax3) = plt.subplots(3,1)
    ax1.plot(x, normal(x,0,1), 'r', label='$\mu=0,\sigma=1$')
    ax1.legend(loc='upper left')
    ax2.plot(x, normal(x,1,1), 'g', label='$\mu=1,\sigma=1$')
    ax2.legend(loc='upper left')
    ax3.plot(x, normal(x,2,1), 'b', label='$\mu=2,\sigma=1$')
    ax3.legend()
    plt.tight_layout()
    fig.savefig('Document/img/chapter2/2-5/2_5_plot_c.png')

if __name__=='__main__':
    x = np.linspace(-5,5,100)
    run_b(x)
    run_c(x)
    run_d(x)
```



(a) For $-5 < x < 5,\ \mu = 0$ and $\sigma = 1$



(b) For $-5 < x < 5,\ \mu = 0$ and $\sigma = 2, 0.5$



(c) For $-5 < x < 5,\ \mu = 0, 1, 2$ and $\sigma = 1$

Figure 2.1: Solutions of the problem 2.5