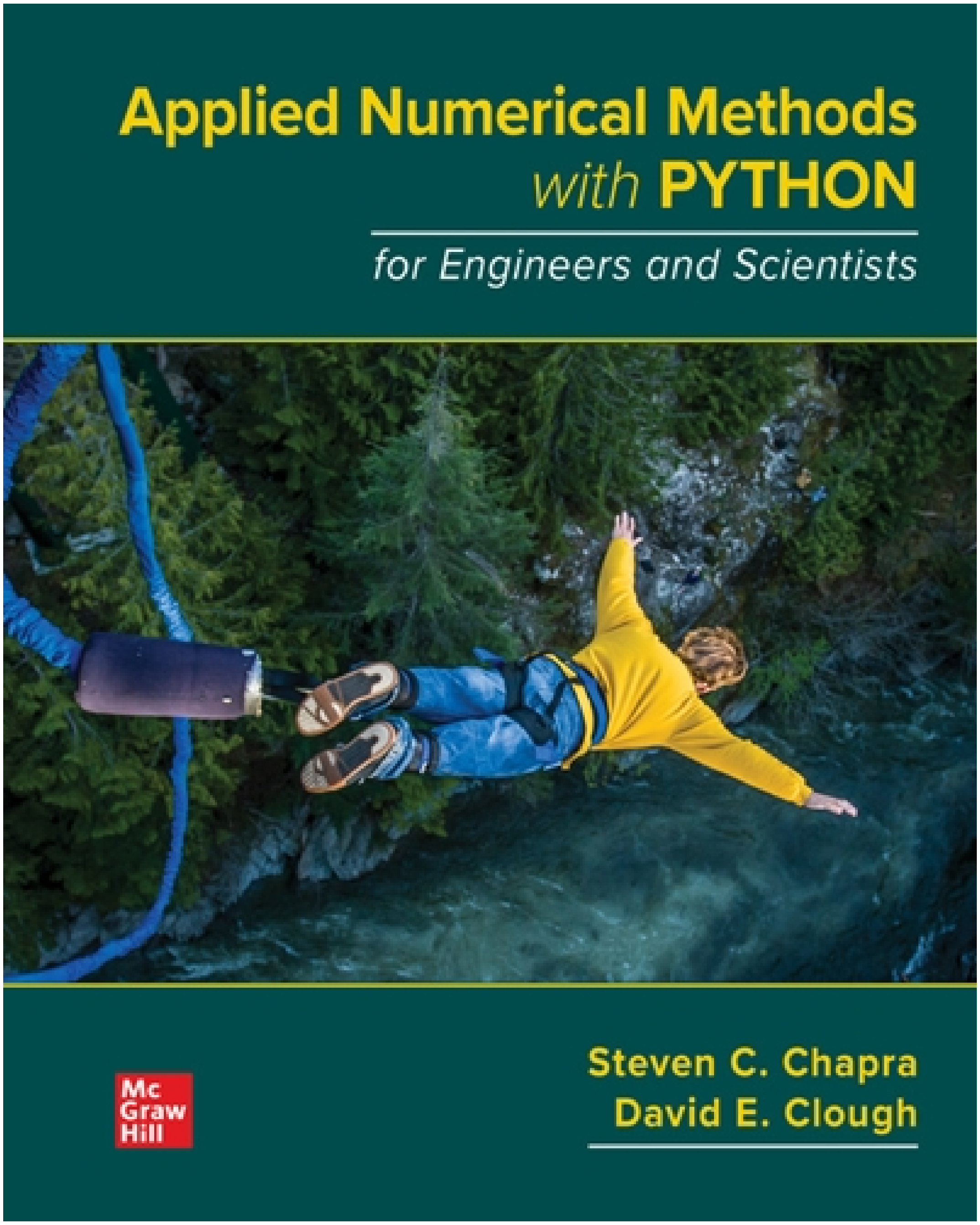
Solutions for Applied Numerical Methods with Python for Engineers and Scientists 1st Edition by Chapra

CLICK HERE TO ACCESS COMPLETE Solutions



Solutions

Chapter 2

2.1 Predict what will be displayed when the following Python commands are executed. Then try them. Explain how they work. Use help resources as needed.

```
import numpy as np
A = np.matrix(' 1,2,3 ; 2,4,6 ; 3,2,1 ')
print(A)

At = A.transpose()
print(At)

A1 = A[:,2]
print(A1)

nm = np.sum(np.diag(A))
print(nm)

Ad = np.delete(A,1,0)
print(Ad)
```

==

```
[[1 2 3]
 [2 4 6]
 [3 2 1]]
```

Transpose: columns and rows reversed

```
[[1 2 3]
 [2 4 2]
 [3 6 1]]
```

Extract third column of A (column index 2)

```
[[3]
 [6]
 [1]]
```

Sum of the diagonal elements of A

```
6
```

Delete the second row of A (row index 1)

```
[[1 2 3]
 [3 2 1]]
```

2.2 Create program statements in Python that compute a vector of y values based on the following formulas:

(a) $y = \dfrac{6t^3 - 3t - 4}{8\sin(5t)}$ $0.1 \le t \le 0.25$

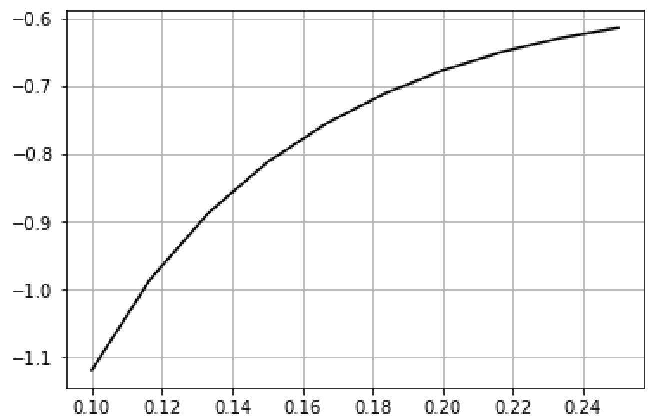
(b) $y = \dfrac{3t-2}{4t} - \dfrac{\pi}{2}t$ $1 \le t \le 5$

where t is a vector of 10 values in the range prescribed. Create a plot of y versus t for each of the above.

===

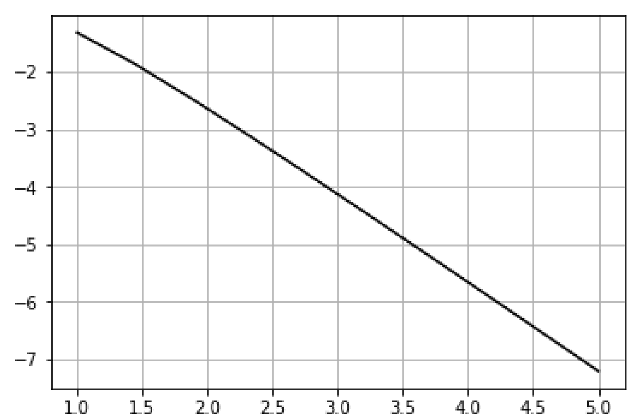
(a)

```
import numpy as np
import pylab
t = np.linspace(0.1,0.25,10)
y = (6*t**3-3*t-
4)/8/np.sin(5*t)
pylab.plot(t,y,c='k')
pylab.grid()
```



(b)

```
t = np.linspace(1,5,10)
y = (3*t-2)/4/t - math.pi/2*t
pylab.plot(t,y,c='k')
pylab.grid()
```



2.3 Write and test Python statements to compute and display a vector of *x* values using the following equation.

$$x = \frac{y(a+bz)^{1.8}}{z(1-y)}$$

where *y* and *z* are vectors of same length as *x*. Experiment with different values of *a* and *b* and different ranges of *y* and *z*. Comment on any issues you discover.

===

Case a) No issues.

```
import numpy as np
y = np.linspace(-0.5,0.5,10)
z = np.linspace(0.1,1,10)
a = 1 ; b = 1
x = y*(a+b*z**1.8)/z/(1-y)
print('y = ',y,'\nz = ',z,'\nx = ',x)
```

```
y =  [-0.5        -0.38888889 -0.27777778 -0.16666667 -0.05555556  0.05555556
  0.16666667  0.27777778  0.38888889  0.5       ]
z =  [0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1. ]
x =  [-3.38616311 -1.47726486 -0.80761114 -0.42577854 -0.13549206  0.13712979
  0.43606602  0.80250448  1.2919946   2.        ]
```

Case (b) Two issues.

```
import numpy as np
y = np.linspace(-0.5,0.5,3)
z = np.linspace(-1,1,3)
a = 1 ; b = 1
x = y*(a+b*z**1.8)/z/(1-y)
print('y = ',y,'\nz = ',z,'\nx = ',x)
```

```
y =  [-0.5  0.   0.5]
z =  [-1.  0.  1.]
x =  [nan nan  2.]
```

Cannot take –1 to the power 1.8.
z = 0 causes divide by zero

2.4 What will be displayed when the following Python statements are run?

(a)

```
import numpy as np
A = np.matrix([[1,2],[3,4],[5,6]])
print(A)
A2 = A[2,:].transpose()
print(A2)
```

(b)

```
y = np.array(np.arange(0,7,1.5)).transpose()
print(y)
```

(c)

```
a = 2 ; b = 8 ; c = 4
print(a/b*c)
print(a/b/c)
print(a/(b*c))
```

==

(a)

```
[[1 2]
 [3 4]
 [5 6]]
[[5]
 [6]]
```

3rd row transposed to a column vector

(b) `[0.  1.5 3.  4.5 6. ]` Note: Not a matrix, so transpose has no effect.

(c)

`1.0`	c is in the numerator
`0.0625`	c is in the denominator, read left-to-right
`0.0625`	b*c denominator set off by parentheses

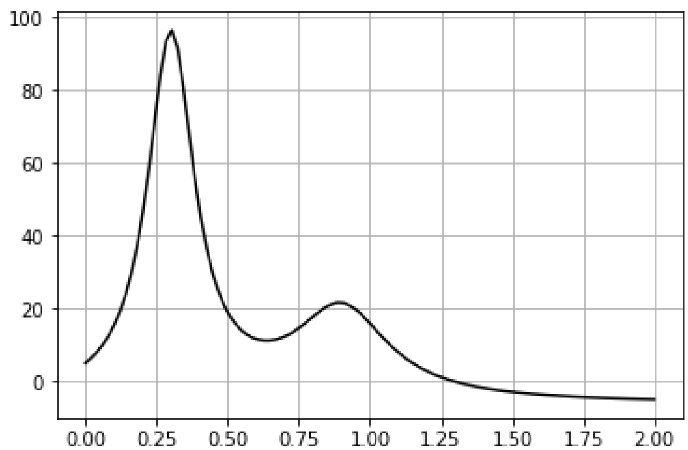
2.5 The "humps" function, $f(x)$, defines a curve that has two maxima (peaks) of unequal height over an interval $a \le x \le b$. An example is

$$f(x) = \frac{1}{(x-0.3)^2 + 0.01} + \frac{1}{(x-0.9)^2 + 0.04} - 6 \qquad 0 \le x \le 2$$

Write Python code to compute and plot $f(x)$ versus x for 100 values of x in the range specified.

===

```
import numpy as np
import pylab
x = np.linspace(0,2,100)
fx = 1/((x-0.3)**2+0.01)+1/((x-0.9)**2+0.04)-6
pylab.plot(x,fx,c='k')
pylab.grid()
```



2.6 Use the NumPy `linspace` function to create vectors equivalent to the following Python statements:

(a) `np.arange(4,35,6)`
(b) `np.arange(-4,2)`

==

(a)

```
import numpy as np
y1 = np.arange(4,35,6)
print('y1 = ',y1)
y2 = np.linspace(4,34,6)
print('y2 = ',y2)

y1 =  [ 4 10 16 22 28 34]
y2 =  [ 4. 10. 16. 22. 28. 34.]
```

Note that y1 contains integers and y2 floats.

(b)

```
x1 = np.arange(-4,2)
print('x1 = ',x1)
x2 = np.linspace(-4,1,6)
print('x2 = ',x2)

x1 =  [-4 -3 -2 -1  0  1]
x2 =  [-4. -3. -2. -1.  0.  1.]
```

2.7 Use NumPy's `arange` function to create vectors identical to the following created with the `linspace` function:

(a) `np.linspace(-2,1.5,8)`
(b) `np.linspace(8,4.5,8)`

===

(a)

```
import numpy as np
x1 = np.linspace(-2,1.5,8)
print('x1 = ',x1)
x2 = np.arange(-2,2,0.5)
print('x2 = ',x2)

x1 =  [-2.  -1.5 -1.  -0.5  0.   0.5  1.   1.5]
x2 =  [-2.  -1.5 -1.  -0.5  0.   0.5  1.   1.5]
```

(b)

```
y1 = np.linspace(8,4.5,8)
print('y1 = ',y1)
y2 = np.arange(8,4,-0.5)
print('y2 = ',y2)

y1 =  [8.  7.5 7.  6.5 6.  5.5 5.  4.5]
y2 =  [8.  7.5 7.  6.5 6.  5.5 5.  4.5]
```

2.8 The NumPy function, `linspace(a,b,n)`, generates an array of `n` equally spaced points between `a` and `b`. Describe the `arange` function that will generate the same result. Test your formulation for `a` = –3, `b` = 5, and `n` = 6.

==

That would be, assuming b > a,

`np.arange(a,b1,delta)`

where b1 = b+delta (or another small number)

and delta = (b-a)/(n-1)

Here is the example:

```
import numpy as np
a = -3
b = 5
n = 6
x1 = np.linspace(a,b,n)
print('x1 = ',x1)
delta = (b-a)/(n-1)
b1 = b + delta
x2 = np.arange(a,b1,delta)
print('x2 = ',x2)

x1 =  [-3.  -1.4  0.2  1.8  3.4  5. ]
x2 =  [-3.  -1.4  0.2  1.8  3.4  5. ]
```

2.9 The following Python statements create a matrix A:

```
import numpy as np
a1 = np.matrix(' 3 2 1 ')
a2 = np.matrix(np.arange(0,1.1,.5))
a3 = np.matrix(np.linspace(6,8,3))
A = np.vstack((a1,a2,a3))
```

(a) Write out the resulting matrix.
(b) Write two distinct Python commands to matrix multiply the second row by the third column of A. Confirm that these yield the same result, a scalar quantity.

==

(a)

```
import numpy as np
a1 = np.matrix(' 3 2 1 ')
a2 = np.matrix(np.arange(0,1.1,.5))
a3 = np.matrix(np.linspace(6,8,3))
A = np.vstack((a1,a2,a3))
print('A = ',A)

A =  [[3.   2.   1. ]
      [0.   0.5 1. ]
     [6.   7.   8. ]]
```

(b)

```
x1 = A[1,:]*A[:,2]
print('x1 = ',x1)
x2 = A[1,:].dot(A[:,2])
print('x2 = ',x2)

x1 =  [[8.5]]
x2 =  [[8.5]]
```

2.10 The following equation can be used to compute values of *y* as a function of *x*:

$$y = b\, e^{-ax} \sin(bx)\ \left(0.0012x^4 - 0.15x^3 + 0.075x^2 + 2.5x\right)$$

where *a* and *b* are parameters.

(a) Write Python code to compute a vector *y* for *a* = 2, *b* = 5, and *x* as a vector with values from 0 to π/2 in increments $\Delta x = \pi/40$.
(b) Compute the vector $z = y^2$ with element-by-element-wise operations.
(c) Combine x, y, and z into a 3-column matrix *W*. Display *W* using a "precision=3" format.
(d) Generate a plot of y and z versus x. Include a title, axis labels, and a legend. For the *y* versus *x* curve, format a solid blue line. For the *z* versus *x* curve, choose a magenta dashed line. Include grid lines.

===

(a)

```
import numpy as np
pi = np.pi
delta = pi/40
a = 2 ; b = 5
x = np.arange(0,pi/2+delta,delta)
print('x = ',x)
y = b*np.exp(-a*x)*np.sin(b*x)*(0.0012*x**4-x**3+0.075*x**2+2.5*x)
print('y = ',y)
```

```
x = [0.         0.07853982 0.15707963 0.23561945 0.31415927 0.39269908
     0.4712389  0.54977871 0.62831853 0.70685835 0.78539816 0.86393798
     0.9424778  1.02101761 1.09955743 1.17809725 1.25663706 1.33517688
     1.41371669 1.49225651 1.57079633]
y = [ 0.00000000e+00  3.21723310e-01  1.01737115e+00  1.70491585e+00
      2.10237284e+00  2.07300502e+00  1.62441901e+00  8.74433634e-01
      2.72455860e-16 -8.15379940e-01 -1.42402819e+00 -1.73966891e+00
     -1.74473436e+00 -1.48202390e+00 -1.03588248e+00 -5.08949209e-01
     -2.94162428e-16  4.13071433e-01  6.83000618e-01  7.95361488e-01
      7.64450734e-01]
```

(b)

```
z = y**2
print('z = ',z)
```

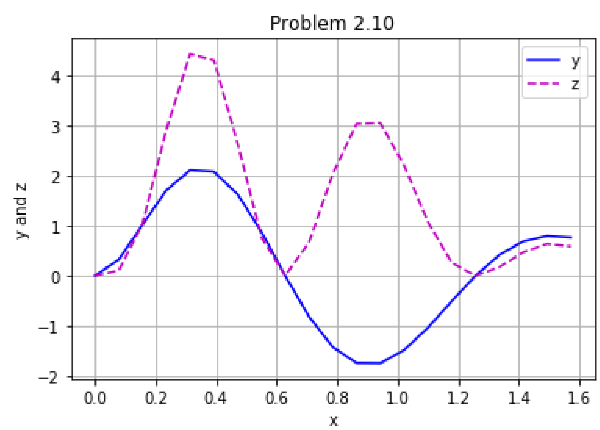
```
z =  [0.00000000e+00 1.03505888e-01 1.03504405e+00 2.90673806e+00
      4.41997157e+00 4.29734982e+00 2.63873711e+00 7.64634181e-01
      7.42321957e-32 6.64844446e-01 2.02785630e+00 3.02644790e+00
      3.04409797e+00 2.19639483e+00 1.07305251e+00 2.59029297e-01
      8.65315343e-32 1.70628009e-01 4.66489845e-01 6.32599896e-01
      5.84384925e-01]
```

(c)

```
xcol = np.matrix(x).transpose()
ycol = np.matrix(y).transpose()
zcol = np.matrix(z).transpose()
W = np.hstack((xcol,ycol,zcol))
np.set_printoptions(precision=3)
```

```
    print('W = ',W)

W = [[ 0.000e+00  0.000e+00  0.000e+00]
 [ 7.854e-02  3.217e-01  1.035e-01]
 [ 1.571e-01  1.017e+00  1.035e+00]
 [ 2.356e-01  1.705e+00  2.907e+00]
 [ 3.142e-01  2.102e+00  4.420e+00]
 [ 3.927e-01  2.073e+00  4.297e+00]
 [ 4.712e-01  1.624e+00  2.639e+00]
 [ 5.498e-01  8.744e-01  7.646e-01]
 [ 6.283e-01  2.725e-16  7.423e-32]
 [ 7.069e-01 -8.154e-01  6.648e-01]
 [ 7.854e-01 -1.424e+00  2.028e+00]
 [ 8.639e-01 -1.740e+00  3.026e+00]
 [ 9.425e-01 -1.745e+00  3.044e+00]
 [ 1.021e+00 -1.482e+00  2.196e+00]
 [ 1.100e+00 -1.036e+00  1.073e+00]
 [ 1.178e+00 -5.089e-01  2.590e-01]
 [ 1.257e+00 -2.942e-16  8.653e-32]
 [ 1.335e+00  4.131e-01  1.706e-01]
 [ 1.414e+00  6.830e-01  4.665e-01]
 [ 1.492e+00  7.954e-01  6.326e-01]
 [ 1.571e+00  7.645e-01  5.844e-01]]
```

(d)
```
pylab.plot(x,y,c='b',label='y')
pylab.plot(x,z,c='m',ls='--',label='z')
pylab.grid()
pylab.xlabel('x')
pylab.ylabel('y and z')
pylab.title('Problem 2.10')
pylab.legend(loc='upper right')
```



2.11 A simple electrical circuit consisting of a resistor, a capacitor, and an inductor (coil) is depicted in Fig. P2.11. When the switch is closed, the voltage potential across the capacitor as a function of time is given by

$$V(t) = V_0 \, e^{-\left(\frac{R}{2L}\right)t} \cos\left(\sqrt{\frac{1}{LC} - \left(\frac{R}{2L}\right)^2}\; t\right)$$

where t = time (s), V_0 = initial voltage (volts), R = resistance (ohms), L = inductance (henrys), C = capacitance (farads).[1] Use Python to generate a plot of this function from $t = 0$ to 0.8 s, given that $V_0 = 10$ volts, R = 60 ohms, $L = 9$ henrys, and $C = 50$ microfarads.

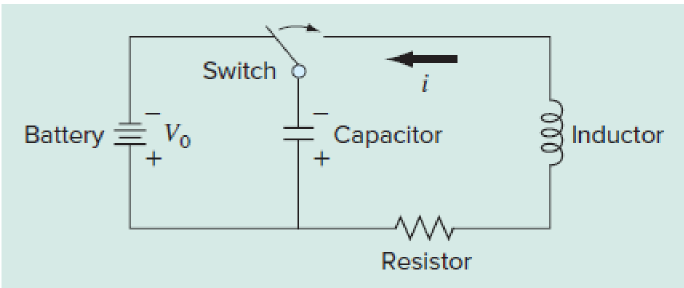
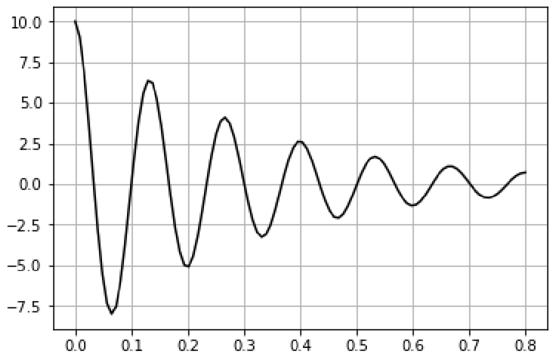


FIGURE P2.11

==

```
import numpy as np
import pylab
V0 = 10
L = 9
R = 60
C = 50e-6
t = np.linspace(0,0.8,100)
V = V0 * np.exp(-1/2*R/L*t) * np.cos(np.sqrt(1/L/C-(R/2/L)**2)*t)
pylab.plot(t,V,c='k')
pylab.grid()
```



[1]The SI-base units for the ohm is kg·m^2·s^{-3}·A^{-2} , the henry is kg·m^2·s^{-2}·A^{-2}, and the farad is A^2 s^4 kg^{-1} m^{-2}. Consequently, R/L has units 1/s and LC has units s^2. The symbol A represents current in amperes.

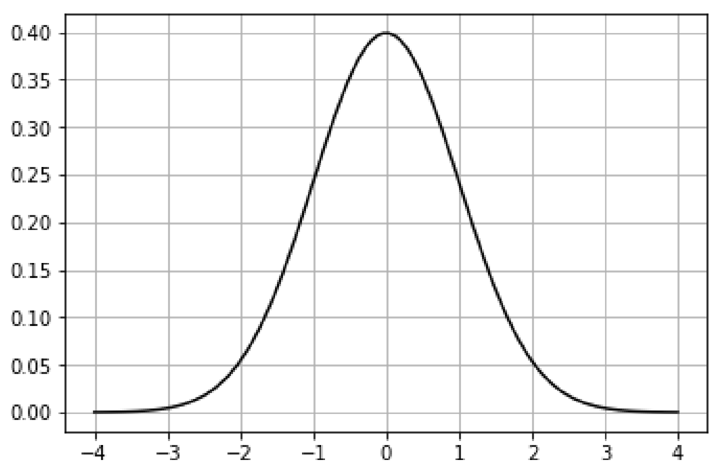
2.12 The standard normal probability density function is a bell-shaped curve that is described by

$$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}$$

Develop Python commands to generate a plot of this function from $z = -4$ to 4. Use enough points to provide a smooth curve plot. Label the ordinate as frequency and the abscissa as z. Compute and display the maximum value of $f(z)$. Where does it occur?

==

```
import numpy as np
import pylab
z = np.linspace(-4,4,100)
f = 1/np.sqrt(2*np.pi)*np.exp(-z**2/2)
pylab.plot(z,f,'k')
pylab.grid()
fmax = np.max(f)
print('f maximum = ',fmax)
```



`f maximum =  0.3986167793238104 6` (about 0.4)

2.13 If a force *F* (N) is applied to compress a spring, the spring's displacement *x* (m) can often be modeled simply by Hooke's Law:

$$F = k\,x$$

where *k* is the spring constant (N/m). The potential energy stored in the spring, *U* (J), can then be computed as

$$U = \frac{1}{2} k\,x^2$$

Five springs are tested and the following data compiled:

F, N	14	18	8	9	13
x, m	0.013	0.020	0.009	0.010	0.012

Use Python to store the *F* and *x* values in vectors. The compute vectors of the spring constants, *k*, and the potential energies, *U*. Use the `np.max` function to determine the maximum potential energy.

==

```
import numpy as np
x = np.array([0.013,0.02,0.009,0.010,0.012])
F = np.array([14,18,8,9,13])
k = F/x
U = 1/2*k*x**2
Umax = np.max(U)
print('x = ',x)
print('F = ',F)
print('k = ',k)
print('U = ',U)
print('maximum U = ',Umax)
```

```
x =  [0.013 0.02  0.009 0.01  0.012]
F =  [14 18  8  9 13]
k =  [1076.9231  900.      888.8889  900.     1083.3333]
U =  [0.091 0.18  0.036 0.045 0.078]
maximum U =  0.18000000000000002
```

2.14 The density of pure water (kg/m^3) at atmospheric pressure varies with temperature between the freezing point (0 °C) and the boiling point (100 °C). This relationship can be accurately described by a 4th-order polynomial.

$$\rho = 999.904 + 0.0508007T - 0.00748423T^2 + 4.11472\times10^{-5}\,T^3 - 1.2329\times10^{-7}\,T^4$$

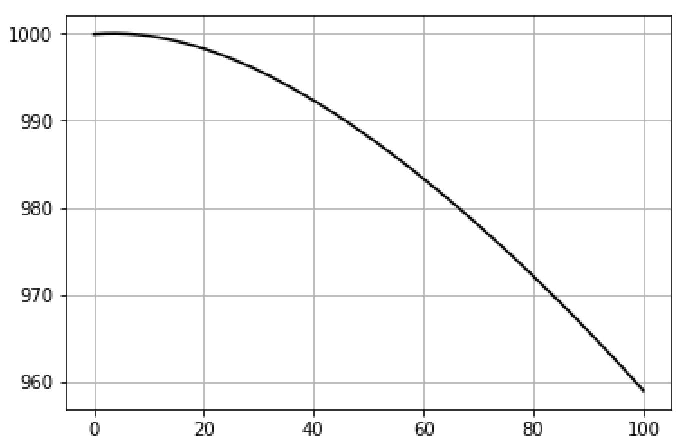
where ρ is the density in kg/m^3 and T is the temperature in °C. Use Python to generate a vector *TF* of Fahrenheit temperatures from 32 to 212 °F in 3.6 °F increments. Convert this vector to another vector *TC* of Celsius temperatures. Then, compute a vector of densities based on the formula above. Create a plot of ρ versus *TC*. Determine at what temperature the maximum density occurs. Recall that T[°C] = (T[°F] – 32)/1.8.

==

```
import numpy as np
import pylab
TF = np.arange(32,214,3.6)
TC = (TF-32)/1.8
rho = 999.904+0.0508007*TC-0.00748423*TC**2+4.11472e-5*TC**3-1.23329e-7*TC**4
rhomax = np.max(rho)
print('maximum density = ',rhomax)
pylab.plot(TC,rho,'k')
pylab.grid()
TCv = np.matrix(TC).transpose()
rhov = np.matrix(rho).transpose()
TrhoM = np.hstack((TCv,rhov))
np.set_printoptions(precision=4)
print(TrhoM)
```

```
maximum density =  999.9900569685759
[[  0.      999.904 ]
 [  2.      999.976 ]
 [  4.      999.9901]
 [  6.      999.9481]
 [  8.      999.852 ]
```

maximum density at approximately 4 degC



2.15 Manning's equation can be used to compute the velocity of water in a rectangular open channel:

$$U = \frac{\sqrt{S}}{n}\left(\frac{BH}{B+2H}\right)^{2/3}$$

where U = velocity (m/s), S = channel slope, n = roughness coefficient, B = channel width (m), and H = water depth (m). The following data are available for five channels:

n	*S*	*B*	*H*
0.035	0.0001	10	2.0
0.020	0.0002	8	1.0
0.015	0.0010	20	1.5
0.030	0.0007	24	3.0
0.022	0.0003	15	2.5

Create Python statements to store these values in a matrix *P* where each row represents one of the channels and each column represents one of the parameters. Add one or more statements to compute the velocities in a column vector *U* based on the values in the parameter matrix. Display the resulting vector.

==

```
import numpy as np
n = np.array((0.035,0.020,0.015,0.030,0.022))
S = np.array((0.0001,0.0002,0.0010,0.0007,0.0003))
B = np.array((10,8,20,24,15))
H = np.array((2.,1.,1.5,3.,2.5))
U = np.sqrt(S)/n*((B*H)/(B+2*H))**(2/3)
P = np.dstack(([n],[S],[B],[H]))
print('P = ',P)
np.set_printoptions(precision=3)
print('U = ',U)

P =  [[[3.5e-02 1.0e-04 1.0e+01 2.0e+00]
       [2.0e-02 2.0e-04 8.0e+00 1.0e+00]
       [1.5e-02 1.0e-03 2.0e+01 1.5e+00]
       [3.0e-02 7.0e-04 2.4e+01 3.0e+00]
       [2.2e-02 3.0e-04 1.5e+01 2.5e+00]]]
U =  [0.362 0.609 2.517 1.581 1.197]
```

2.16 It is accepted general practice in engineering and science to plot analytical mathematical equations using line segments and experimental data as markers. Occasionally, interconnecting lines are added to data plots for the purpose of pattern recognition. Here are a few data for concentration, c, versus time, t, for the photodegradation of aqueous bromine.

t, min	10	20	30	40	50	60
c, ppm	3.4	2.6	1.6	1.3	1.0	0.5

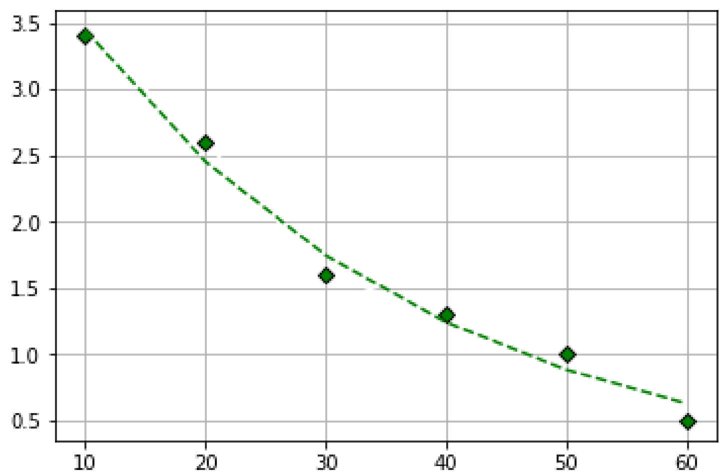
These data can be modeled by the following function:

$$c = 4.84e^{-0.034t}$$

Use Python to create a plot displaying both the data (using diamond-shaped, green-filled markers with black edges) and the function (using a green, dashed line). Plot the function for the range $0 \le t \le 70$ min. Include grid lines.

===

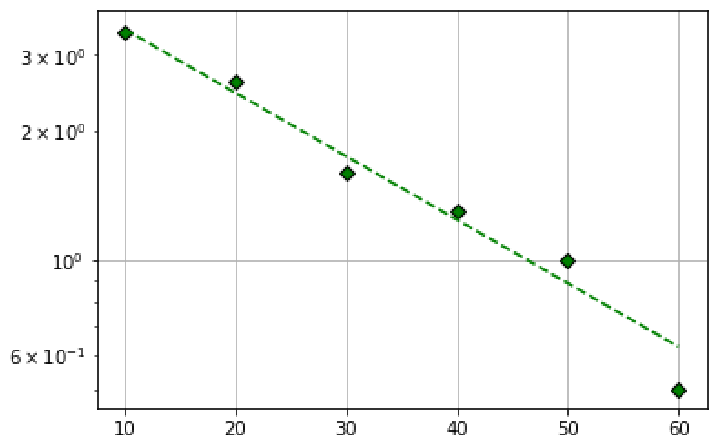
```
import numpy as np
import pylab
t = np.arange(10,70,10)
c = np.array([3.4,2.6,1.6,1.3,1.0,0.5])
c_calc = 4.84 * np.exp(-0.034*t)
pylab.plot(t,c,c='w',marker='D',mfc='g',mec='k')
pylab.plot(t,c_calc,c='g',ls='--')
pylab.grid()
```



2.17 The pylab interface to the Matplotlib module allows for semilog and log-log plots. Instead of using the `pylab.plot` function, you can use, for example, the `pylab.semilogy` function to apply a log scale to the ordinate. Modify your Python code from Prob. 2.16 to create a `semilogy` plot. Observe the resulting plot and explain its features.

==

```
import numpy as np
import pylab
t = np.arange(10,70,10)
c = np.array([3.4,2.6,1.6,1.3,1.0,0.5])
c_calc = 4.84 * np.exp(-0.034*t)
pylab.semilogy(t,c,c='w',marker='D',mfc='g',mec='k')
pylab.semilogy(t,c_calc,c='g',ls='--')
pylab.grid()
```



Log transformation linearized the model and the measurements.

2.18 Here are some wind tunnel data for force, *F*, versus velocity, *v*.

v, m/s	10	20	30	40	50	60	70	80
F, N	25	75	380	550	610	1220	830	1450

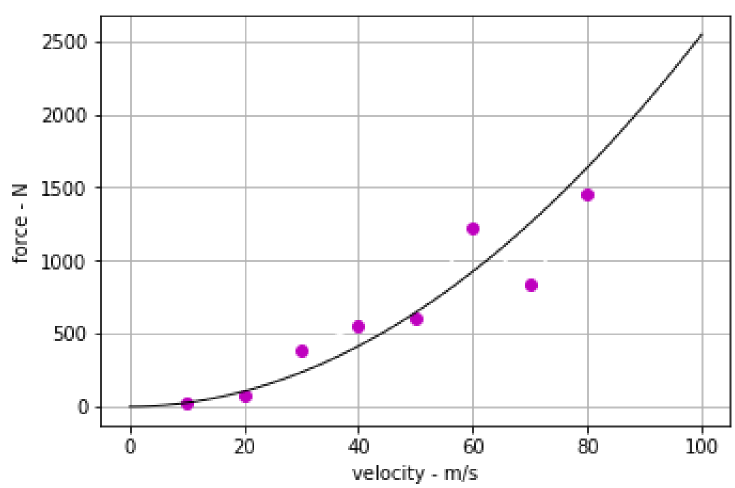
These data can be described by the following function:

$$F = 0.274\, v^{1.984}$$

Use Python to create a plot displaying both the data (using circular magenta markers) and the function (using a thin black dotted line). Plot the function over the range $0 \le v \le 100$ s, and label the axes. Include grid lines.

==

```
import numpy as np
import pylab
v = np.linspace(10,80,8)
F = np.array([25,75,380,550,610,1220,830,1450])
v_calc = np.linspace(0,100,50)
F_calc = 0.274*v_calc**1.984
pylab.plot(v,F,c='w',marker='o',mfc='m',mec='m')
pylab.plot(v_calc,F_calc,c='k',lw=1.)
pylab.grid()
pylab.xlabel('velocity - m/s')
pylab.ylabel('force - N')
```

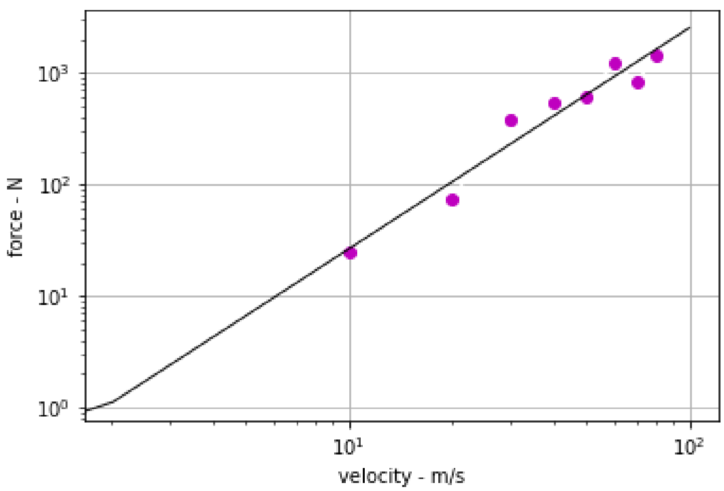


2.19 The `pylab.loglog` function is similar to the `pylab.plot` function except that logarithmic scales are used for both the *x* and *y* axes. Create a log-log plot of the data and function from Prob. 2.18. Comment on the resulting plot.

===

```
import numpy as np
```

```
import pylab
v = np.linspace(10,80,8)
F = np.array([25,75,380,550,610,1220,830,1450])
v_calc = np.linspace(0,100,50)
F_calc = 0.274*v_calc**1.984
pylab.loglog(v,F,c='w',marker='o',mfc='m',mec='m')
pylab.loglog(v_calc,F_calc,c='k',lw=1.)
pylab.grid()
pylab.xlabel('velocity - m/s')
pylab.ylabel('force - N')
```



Log-log plot linearizes the model and the data conform.

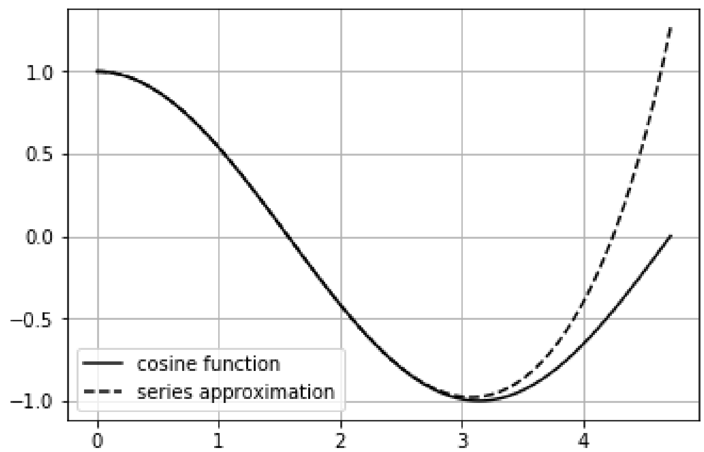
2.20 The Maclaurin series expansion for the cosine is

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \cdots$$

Use Python to create a plot of the cosine using the `np.cos` function (solid black line) along with a curve of the series expansion (black dashed line) up to and including the $x^8/8!$ term. Add a legend to your plot. Use the `math.factorial` function for the various $n!$ evaluations. Make the range of the function $0 \le x \le 3\pi/2$.

==

```
import numpy as np
import pylab
import math
x = np.linspace(0,3*np.pi/2,100)
cosx = np.cos(x)
seriesx = ( 1 - x**2/math.factorial(2) + x**4/math.factorial(4)
- x**6/math.factorial(6) + x**8/math.factorial(8) )
pylab.plot(x,cosx,c='k',label='cosine function')
pylab.plot(x,seriesx,c='k',ls='--',label='series approximation')
pylab.grid()
pylab.legend()
```



Series approximation departs from cosine function about x = 2.5.

2.21 You contact the jumpers used to generate the data in Table 2.1 and measure their frontal areas. The resulting values, which are ordered in the same sequence as the corresponding values in Table 2.1, are

A, m^2	0.455	0.402	0.452	0.486	0.531	0.475	0.487

(a) If the air density is $\rho = 1.223$ kg/m^3, use Python to compute values of the dimensionless drag coefficient, C_D.
(b) Determine the average, minimum, and maximum of the resulting C_D values.
(c) Create two plots. The first is A versus m, and the second is C_D versus m. Include descriptive axis labels and titles on the plots. Include grid lines.

===

Table 2.1 Data for the mass and associated terminal velocities of a number of jumpers

m, kg	83.6	60.2	72.1	91.1	92.9	65.3	80.9
v_t, m/s	53.4	48.5	50.9	55.7	54.0	47.7	51.1

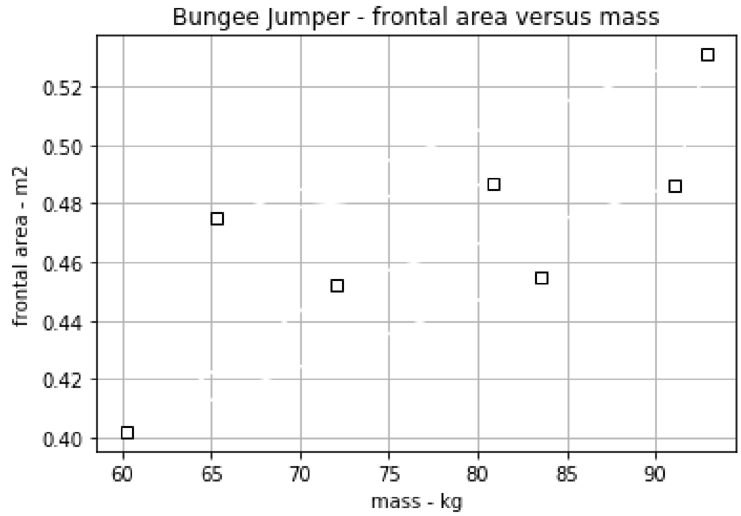
$$c_d = \frac{C_D \rho A}{2} \quad \Rightarrow \quad C_D = \frac{2c_d}{\rho A} \qquad\qquad c_d = \frac{m \cdot g}{v_t^2}$$

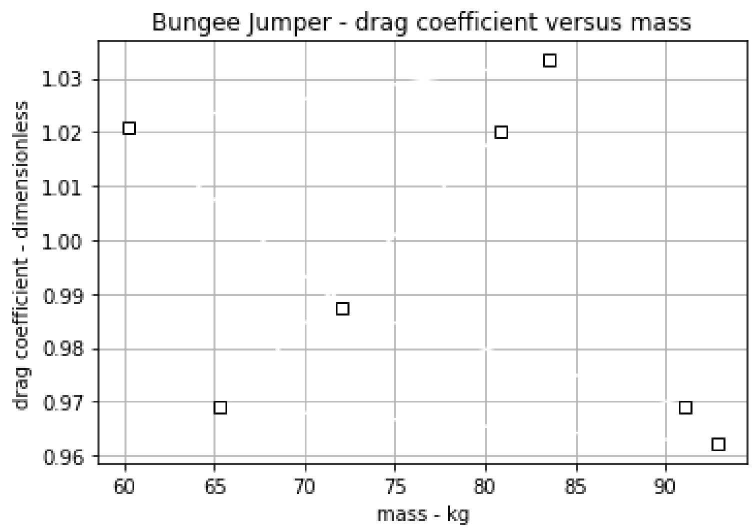
(a)
```
import numpy as np
import pylab
m = np.array([83.6,60.2,72.1,91.1,92.9,65.3,80.9])
vt = np.array([53.4,48.5,50.9,55.7,54.0,47.7,51.1])
A = np.array([0.455,0.402,0.452,0.486,0.531,0.475,0.487])
g = 9.807
rho = 1.223
cd = m*g/vt**2
C_D = 2*cd/rho/A
np.set_printoptions(precision = 4)
print('CD = ',C_D)

CD =  [1.0334 1.021  0.9874 0.969  0.9622 0.969  1.0203]
```

(b)
```
C_Davg = np.average(C_D)
C_Dmin = np.min(C_D)
C_Dmax = np.max(C_D)
print('average CD = ',C_Davg)
print('minimum CD = ',C_Dmin)
print('maximum CD = ',C_Dmax)

average CD =  0.9946062990344909
minimum CD =  0.9622178012623607
maximum CD =  1.0333610328572846
```

(c)
```
pylab.plot(m,A,c='w',marker='s',mfc='w',mec='k')
pylab.grid()
pylab.xlabel('mass - kg')
```

```
pylab.ylabel('frontal area - m2')
pylab.title('Bungee Jumper - frontal area versus mass')
pylab.figure()
pylab.plot(m,C_D,c='w',marker='s',mfc='w',mec='k')
pylab.grid()
pylab.xlabel('mass - kg')
pylab.ylabel('drag coefficient - dimensionless')
pylab.title('Bungee Jumper - drag coefficient versus mass')
```





2.22 The *Weibull probability distribution* is widely used to represent reliability in manufacturing processes. The density function for this distribution is given by

$$f(x) = \frac{\alpha}{\beta^{\alpha}} x^{\alpha-1} e^{-\left(\frac{x}{\beta}\right)^{\alpha}} \qquad x \geq 0$$

where *x* is a random variable, α is a shape parameter, and β is a scale parameter for the distribution. For a scale factor $\beta = 1$, the density becomes

$$f(x) = \alpha \cdot x^{\alpha-1} e^{-x^{\alpha}} \qquad x \geq 0$$

(a) Create a plot of *f*(*x*) for the range of $0 \leq x \leq 3$ with curves for $\alpha = 5$ and $\alpha = 2$. Use different colors for the curves and provide a legend. Include grid lines.
(b) Python's NumPy module has a sub-module `random` that provides a `weibull` function for generating random numbers according to the distribution with $\beta = 1$ and a given α value. You can generate *n* random numbers with the statements

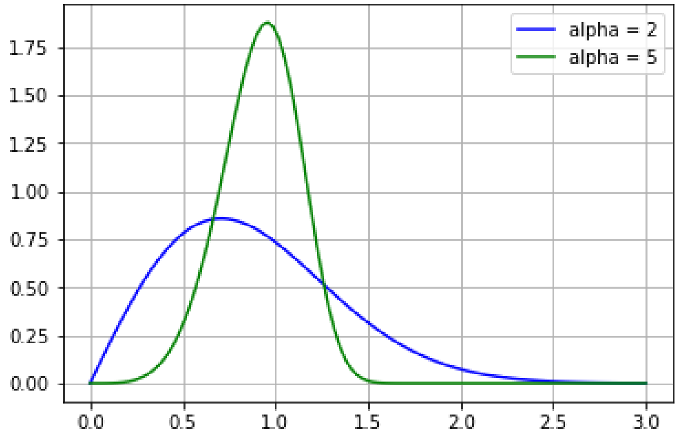
```
np.random.weibull(alpha,n)
```

Generate an array of 1000 random numbers for $\alpha = 5$. Plot a histogram chart for the results. Repeat the process for $\alpha = 2$. Comment on any differences you observe between the two histogram charts.

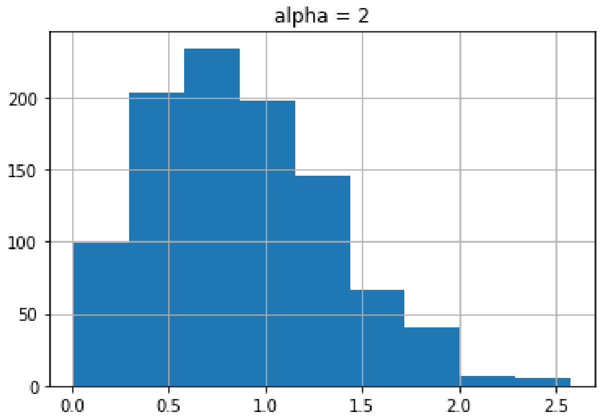
==

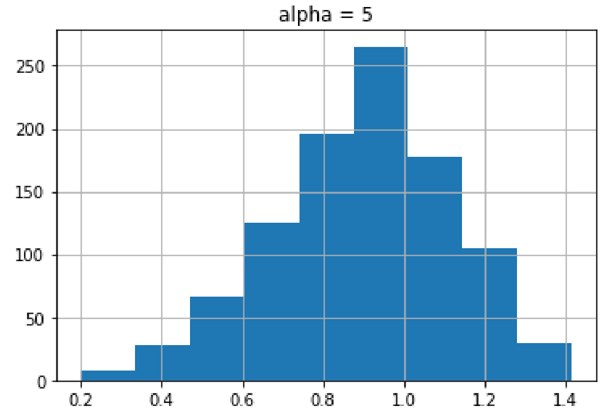
(a)

```
import numpy as np
import pylab
x = np.linspace(0,3,101)
alpha1 = 2
alpha2 = 5
f1 = alpha1 * x**(alpha1-1) * np.exp(-x**alpha1)
f2 = alpha2 * x**(alpha2-1) * np.exp(-x**alpha2)
pylab.plot(x,f1,c='b',label='alpha = 2')
pylab.plot(x,f2,c='g',label='alpha = 5')
pylab.grid()
pylab.legend(loc='upper right')
```



(b)
```
weib1 = np.random.weibull(alpha1,1000)
pylab.figure()
pylab.hist(weib1,9)
pylab.title('alpha = 2')
pylab.grid()
weib2 = np.random.weibull(alpha2,1000)
pylab.figure()
pylab.hist(weib2,9)
pylab.title('alpha = 5')
pylab.grid()
```





First chart has a tail to the right; whereas second chart has a slight tail to the left.

2.23 Predict what will be displayed after the following commands are entered in the Console:

(a) `np.arange(6)`
(b) `np.arange(6,1,-1)`
(c) `np.linspace(1,6,6)`
(d) `np.linspace(6,1,6)`
(e) `np.logspace(-2,2,5)`

After you have made your predictions, test the commands. Note any differences from your predictions and explain what you have learned.

==

(a) 0,1,2,3,4,5 `arange` does not include limit

```
import numpy as np
print(np.arange(6))
```

```
[0 1 2 3 4 5]
```

(b) 6,5,4,3,2 again, 1 is left out

```
print(np.arange(6,1,-1))
```

```
[6 5 4 3 2]
```

(c) 1,2,3,4,5,6 `linspace` <u>does</u> include the limit

```
print(np.linspace(1,6,6))
```

`[1. 2. 3. 4. 5. 6.]` notice that these are floating point

(d) 6,5,4,3,2,1

```
print(np.linspace(6,1,6))
```

```
[6. 5. 4. 3. 2. 1.]
```

(e) 0.01,0.1,1.0,10.0,100.0

```
print(np.logspace(-2,2,5))
```

```
[1.e-02 1.e-01 1.e+00 1.e+01 1.e+02]
```

2.24 The trajectory of an object can be modeled as

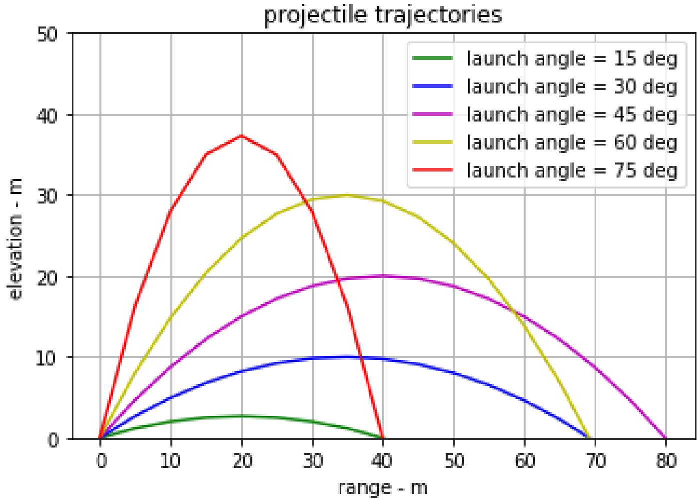
$$y = \tan(\theta_0)x - \frac{g}{2v_0^2\cos^2(\theta_0)}x^2 + y_0$$

where y = elevation (m), θ_0 = initial or launch angle (rad), x = horizontal distance or range (m), g = gravitational acceleration (9.81 m/s^2), v_0 = initial velocity (m/s), and y_0 = initial elevation (m). Program Python to find the trajectories for y_0 = 0 m and v_0 = 28 m/s for launch angles ranging from 15° to 75° in increments of 15°. Employ a range of horizontal distances from 0 to 80 m in increments of 5 m.

The results should be assembled in a matrix where the first dimension, the rows, correspond to the distances and the second dimension, the columns, correspond to the launch angles. Use this matrix to generate a plot of a family of curves of y versus x for the launch angles. Style each curve differently and include a legend describing the launch angles. As required, adjust the y-axis scale so the minimum is zero meters. Include grid lines.

==

```
import numpy as np
import pylab
x = np.arange(0,85,5)
theta0deg = np.arange(15,90,15)
theta0 = theta0deg/180*np.pi
y0 = 0
v0 = 28
g = 9.81
y = np.empty((17,5))
for i in range(5):
    y[:,i] = (np.tan(theta0[i])*x-g/(2*v0**2*np.cos(theta0[i])**2)*x**2+y0)
pylab.plot(x,y[:,0],c='g',label='launch angle = 15 deg')
pylab.plot(x,y[:,1],c='b',label='launch angle = 30 deg')
pylab.plot(x,y[:,2],c='m',label='launch angle = 45 deg')
pylab.plot(x,y[:,3],c='y',label='launch angle = 60 deg')
pylab.plot(x,y[:,4],c='r',label='launch angle = 75 deg')
pylab.ylim(0,50)
pylab.grid()
pylab.xlabel('range - m')
pylab.ylabel('elevation - m')
pylab.title('projectile trajectories')
pylab.legend(loc='upper right')
```


projectile trajectories
launch angle = 15 deg
launch angle = 30 deg
launch angle = 45 deg
launch angle = 60 deg
launch angle = 75 deg
elevation - m
range - m
0
10
20
30
40
50
0
10
20
30
40
50
60
70
80

2.25 The temperature dependence of the rate of a first-order chemical reaction is described by the *Arrhenius equation*:

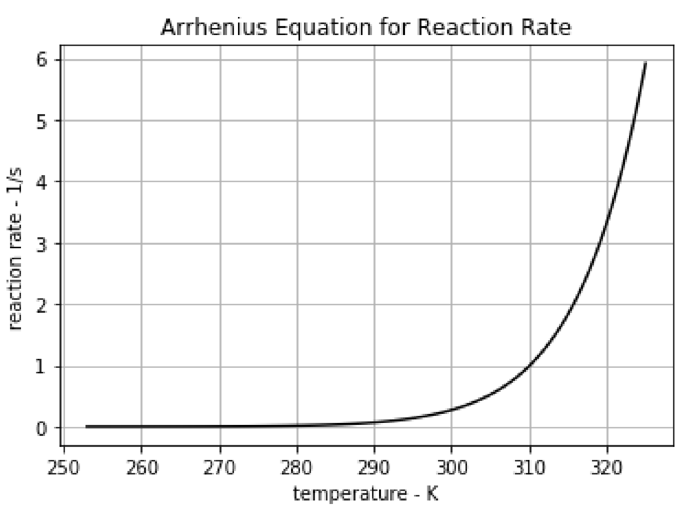
$$k = k_0 e^{-\frac{E}{RT}}$$

where k = reaction rate (1/s), k_0 = pre-exponential (or frequency) factor (1/s), E = activation energy (J/mol), R = universal gas law constant (8.314 J/(mol·K), and T = absolute temperature (K).

A reaction is modeled with $k_0 = 7\times10^{16}$ 1/s and $E = 1\times10^5$ J/mol. Write Python code to generate reaction rates for $253 \le T \le 325$ K. Create a plot for k versus T. Using the `pylab.semilogy` function, create a second plot of $\log_{10} k$ versus $1/T$. Comment on your results.

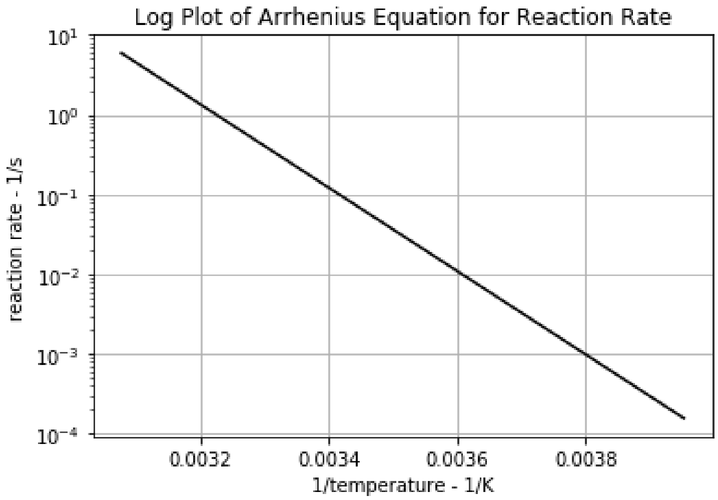
==

```
import numpy as np
import pylab
T = np.linspace(253,325,100)
R = 8.314
k0 = 7e16
E = 1e5
k = k0 * np.exp(-E/R/T)
pylab.plot(T,k,c='k')
pylab.grid()
pylab.xlabel('temperature - K')
pylab.ylabel('reaction rate - 1/s')
pylab.title('Arrhenius Equation for Reaction Rate')
```




```
pylab.figure()
pylab.semilogy(1/T,k,c='k')
pylab.grid()
pylab.xlabel('1/temperature - 1/K')
pylab.ylabel('reaction rate - 1/s')
```

```
pylab.title('Log Plot of Arrhenius Equation for Reaction Rate')
```



Log plot shows linear relationship with slope $-E/R$. Notice that the reaction rate varies over more than 4 orders of magnitude.

2.26 Figure P2.26*a* shows a uniform beam subjected to a linearly increasing load. The deflection of the beam, *y*(m), is depicted in Fig. 2.26*b* and can be computed from the equation

$$y = \frac{w_0}{120EIL}\left(-x^5 + 2L^2x^3 - L^4x\right)$$

where E = modulus of elasticity (N/m^2) and I = moment of inertia (m^4).

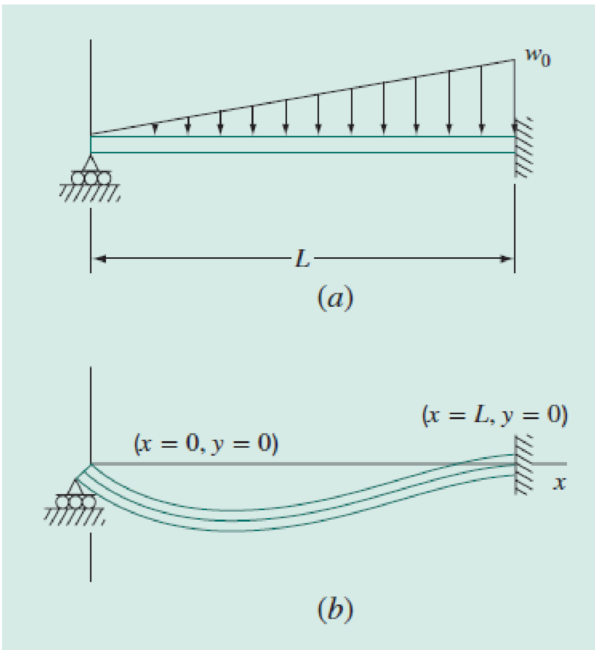


FIGURE P2.26

Employ this equation and calculus to generate Python pylab plots of the following quantities versus distance along the beam, *x*.

(a) deflection, *y*,

(b) slope, $\theta(x) = \frac{dy}{dx}$

(c) moment, $M(x) = EI\frac{d^2y}{dx^2}$

(d) shear, $V(x) = EI\frac{d^3y}{dx^3}$

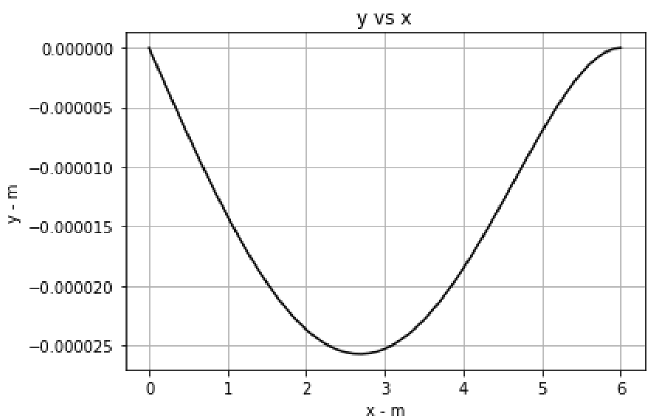
(e) loading, $w(x) = -EI\frac{d^4y}{dx^4}$

Use the following parameter settings for your computations: L = 600 cm, E = 50,000 kN/cm^2, I = 30,000 cm^2, w_0 = 2.5 kN/cm, and Δx = 10 cm. Handle units of measurement with care, converting to the SI standard. Include conversions in your code. Include grid lines, axis labels, and titles on your plots.

===

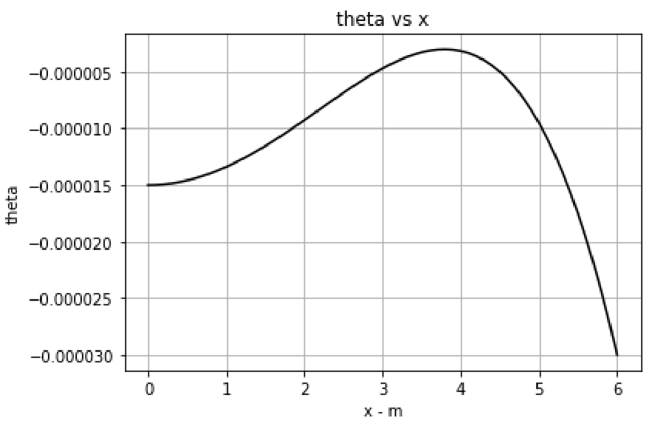
(a)

```
import numpy as np
import pylab
L = 600/100   # cm to m
E = 50000*1000*100*2  # kN/cm2 to N/m2
I = 30000/100**2  # cm2 to m2
w0 = 2.5*1000*100  # kN/cm to N/m
deltax = 10/100  #  cm to m
x = np.arange(0,L+deltax,deltax)
y = w0 / (120*E*I*L)*(-x**5+2*L**2*x**3-L**4*x)
pylab.plot(x,y,c='k')
pylab.grid()
pylab.xlabel('x - m')
pylab.ylabel('y - m')
pylab.title('y vs x')
```

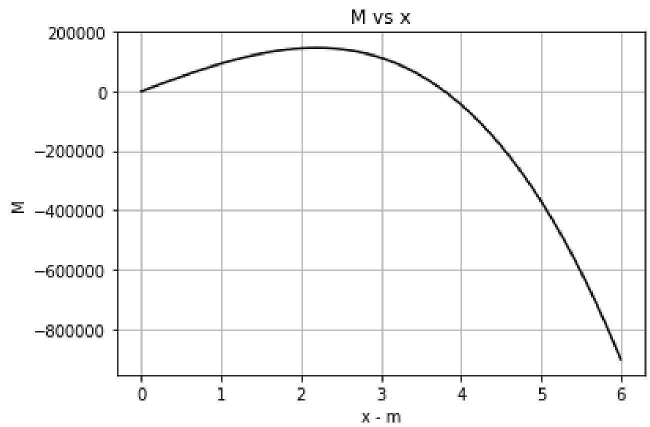


(b)

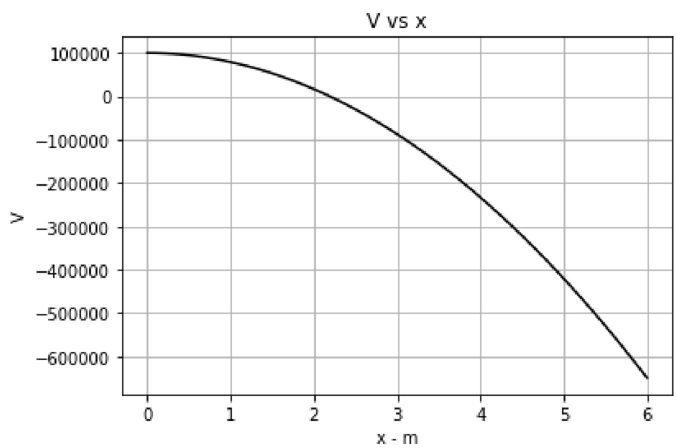
```
theta = w0 / (120*E*I*L)*(-5*x**4+4*L**2*x**2-L**4)
pylab.figure()
pylab.plot(x,theta,c='k')
pylab.grid()
pylab.xlabel('x - m')
pylab.ylabel('theta')
pylab.title('theta vs x')
```



(c)
```
M = w0 / (120*L) * (-20*x**3+8*L**2*x)
pylab.figure()
pylab.plot(x,M,c='k')
pylab.grid()
pylab.xlabel('x - m')
pylab.ylabel('M')
pylab.title('M vs x')
```

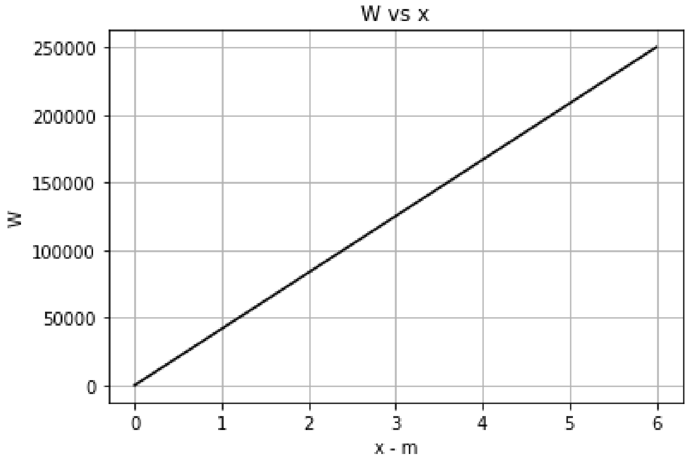


(d)
```
V = w0 / (120*L) * (-60*x**2+8*L**2)
pylab.figure()
pylab.plot(x,V,c='k')
pylab.grid()
pylab.xlabel('x - m')
pylab.ylabel('V')
pylab.title('V vs x')
```



(e)
```
W = - w0 / (120*L) * (-120*x)
pylab.figure()
pylab.plot(x,W,c='k')
pylab.grid()
pylab.xlabel('x - m')
pylab.ylabel('W')
```

```
pylab.title('W vs x')
```



2.27 The *butterfly curve* is given by the following parametric equations:

$$x = \sin(t)\cdot\left(e^{\cos(t)} - 2\cos(4t) - \sin^5\left(\frac{t}{12}\right)\right)$$
$$y = \cos(t)\cdot\left(e^{\cos(t)} - 2\cos(4t) - \sin^5\left(\frac{t}{12}\right)\right)$$

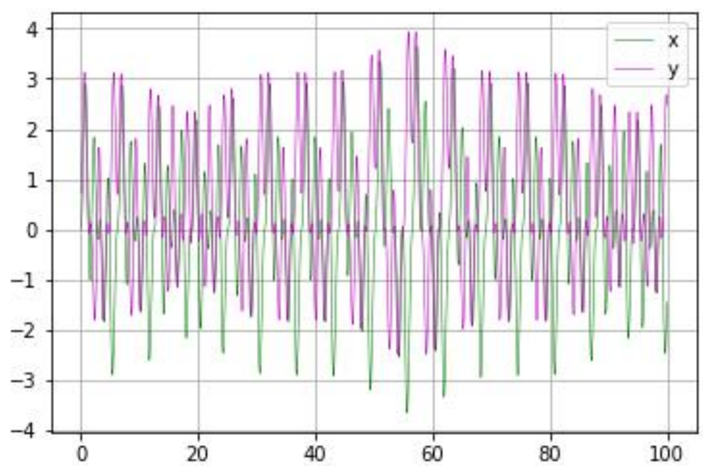
Generate values of x and y for $0 \le t \le 100$ in steps of $\Delta t = 1/16$. Use Python's Matplotlib pylab interface to construct the following plots:

(a) x and y versus t
(b) y versus x

For the plot in (a), use different black line styles and include a legend. Make the plot in (b) square, for example, make the x- and y-axes equal. Create axis titles and a plot title for both (a) and (b). Include grid lines on your plots.

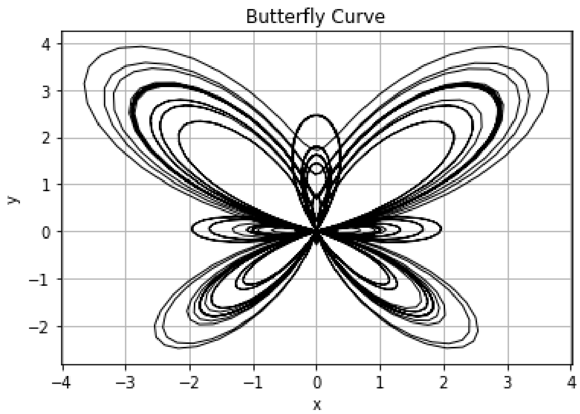
==

(a)
```
import numpy as np
import pylab
deltat = 1/16
t = np.arange(0,100+deltat,deltat)
x = np.sin(t)*(np.exp(np.cos(t))-2*np.cos(4*t)-np.sin(t/12)**5)
y = np.cos(t)*(np.exp(np.cos(t))-2*np.cos(4*t)-np.sin(t/12)**5)
pylab.plot(t,x,c='g',lw=0.5,label='x')
pylab.plot(t,y,c='m',lw=0.5,label='y')
pylab.grid()
pylab.legend()
```



(b)
```
pylab.figure()
pylab.plot(x,y,c='k',lw=1.)
pylab.grid()
pylab.xlabel('x')
```

```
pylab.ylabel('y')
pylab.title('Butterfly Curve')
```



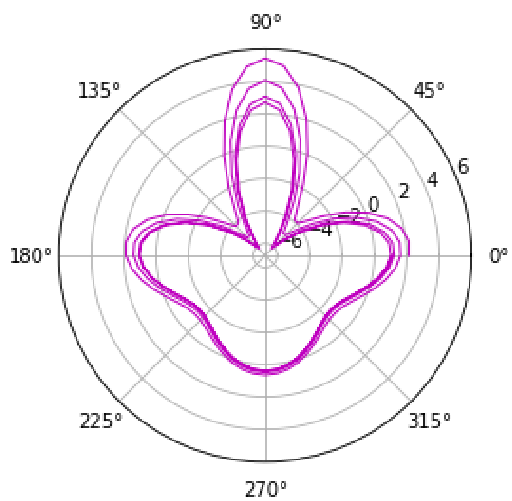
2.28 The butterfly curve from Prob. 2.27 can also be represented in polar coordinates as

$$r = e^{\sin(\theta)} - 2\cos(4\theta) - \sin^5\left(\frac{2\theta - \pi}{24}\right)$$

Using Python, generate values of *r* for $0 \le t \le 8\pi$ with $\Delta\theta = \pi/32$. Use the `pylab.polar` plotting function to generate the polar plot of the butterfly function with a dashed blue line. Include grid lines. Explore the polar plot command using the help facilities.

==

```
import numpy as np
import pylab
deltatheta = np.pi/32
theta = np.arange(0,8*np.pi+deltatheta,deltatheta)
r = np.exp(np.sin(theta))*(2*np.cos(4*theta)-np.sin((2*theta-np.pi)/24))
pylab.polar(theta,r,c='m',lw=1.)
```



Chapter 1

1.1 Use calculus to verify that Eq. (1.9) is a solution of Eq. (1.8) for the initial condition $v(0) = 0$.

==

You are given the following differential equation with the initial condition, $v(0) = 0$,

$$\frac{dv}{dt} = g - \frac{c_d}{m}v^2$$

Multiply both sides by m/c_d

$$\frac{m}{c_d}\frac{dv}{dt} = \frac{m}{c_d}g - v^2$$

Define $a = \sqrt{mg / c_d}$

$$\frac{m}{c_d}\frac{dv}{dt} = a^2 - v^2$$

Integrate by separation of variables,

$$\int \frac{dv}{a^2 - v^2} = \int \frac{c_d}{m}dt$$

A table of integrals can be consulted to find that

$$\int \frac{dx}{a^2 - x^2} = \frac{1}{a}\tanh^{-1}\frac{x}{a}$$

Therefore, the integration yields

$$\frac{1}{a}\tanh^{-1}\frac{v}{a} = \frac{c_d}{m}t + C$$

If $v = 0$ at $t = 0$, then because $\tanh^{-1}(0) = 0$, the constant of integration $C = 0$ and the solution is

$$\frac{1}{a}\tanh^{-1}\frac{v}{a} = \frac{c_d}{m}t$$

This result can then be rearranged to yield

$$v = \sqrt{\frac{gm}{c_d}}\tanh\left(\sqrt{\frac{gc_d}{m}}t\right)$$

1.2 Use calculus to solve Eq. (1.21) for the case where the initial velocity is **(a)** positive and **(b)** negative. **(c)** Based on your results for (**a**) and (**b**), perform the same computation as in Example 1.1 but with an initial velocity of −40 m/s. Compute values of the velocity from t = 0 to 12 s at intervals of 2 s. Note that for this case, the zero velocity occurs at t = 3.470239 s.

==

(a) For the case where the initial velocity is positive (downward), Eq. (1.21) is

$$\frac{dv}{dt} = g - \frac{c_d}{m} v^2$$

Multiply both sides by m/c_d

$$\frac{m}{c_d}\frac{dv}{dt} = \frac{m}{c_d} g - v^2$$

Define $a = \sqrt{mg / c_d}$

$$\frac{m}{c_d}\frac{dv}{dt} = a^2 - v^2$$

Integrate by separation of variables,

$$\int \frac{dv}{a^2 - v^2} = \int \frac{c_d}{m} dt$$

A table of integrals can be consulted to find that

$$\int \frac{dx}{a^2 - x^2} = \frac{1}{a}\tanh^{-1}\frac{x}{a}$$

Therefore, the integration yields

$$\frac{1}{a}\tanh^{-1}\frac{v}{a} = \frac{c_d}{m} t + C$$

If $v = +v_0$ at $t = 0$, then

$$C = \frac{1}{a}\tanh^{-1}\frac{v_0}{a}$$

Substitute back into the solution

$$\frac{1}{a}\tanh^{-1}\frac{v}{a} = \frac{c_d}{m} t + \frac{1}{a}\tanh^{-1}\frac{v_0}{a}$$

Multiply both sides by a, taking the hyperbolic tangent of each side and substituting a gives,

$$v = \sqrt{\frac{mg}{c_d}} \tanh\left(\sqrt{\frac{gc_d}{m}} t + \tanh^{-1}\sqrt{\frac{c_d}{mg}} v_0\right) \qquad (1)$$

(b) For the case where the initial velocity is negative (upward), Eq. (1.21) is

$$\frac{dv}{dt} = g + \frac{c_d}{m} v^2$$

Multiplying both sides of Eq. (1.8) by m/c_d and defining $a = \sqrt{mg/c_d}$ yields

$$\frac{m}{c_d}\frac{dv}{dt} = a^2 + v^2$$

Integrate by separation of variables,

$$\int \frac{dv}{a^2+v^2} = \int \frac{c_d}{m}dt$$

A table of integrals can be consulted to find that

$$\int \frac{dx}{a^2+x^2} = \frac{1}{a}\tan^{-1}\frac{x}{a}$$

Therefore, the integration yields

$$\frac{1}{a}\tan^{-1}\frac{v}{a} = \frac{c_d}{m}t + C$$

The initial condition, $v(0) = v_0$ gives

$$C = \frac{1}{a}\tan^{-1}\frac{v_0}{a}$$

Substituting this result back into the solution yields

$$\frac{1}{a}\tan^{-1}\frac{v}{a} = \frac{c_d}{m}t + \frac{1}{a}\tan^{-1}\frac{v_0}{a}$$

Multiplying both sides by a and taking the tangent gives

$$v = a\tan\left(a\frac{c_d}{m}t + \tan^{-1}\frac{v_0}{a}\right)$$

or substituting the values for a and simplifying gives

$$v = \sqrt{\frac{mg}{c_d}}\tan\left(\sqrt{\frac{gc_d}{m}}t + \tan^{-1}\sqrt{\frac{c_d}{mg}}v_0\right) \qquad (2)$$

(c) We use Eq. (2) until the velocity reaches zero. Inspection of Eq. (2) indicates that this occurs when the argument of the tangent is zero. That is, when

$$\sqrt{\frac{gc_d}{m}}t_{zero} + \tan^{-1}\sqrt{\frac{c_d}{mg}}v_0 = 0$$

The time of zero velocity can then be computed as

$$t_{zero} = -\sqrt{\frac{m}{gc_d}}\tan^{-1}\sqrt{\frac{c_d}{mg}}v_0$$

Thereafter, the velocities can then be computed with Eq. (1.9),

$$v = \sqrt{\frac{mg}{c_d}}\tanh\left(\sqrt{\frac{gc_d}{m}}(t - t_{zero})\right) \qquad (3)$$

Here are the results for the parameters from Example 1.2, with an initial velocity of –40 m/s.

$$t_{zero} = -\sqrt{\frac{68.1}{9.81(0.25)}}\tan^{-1}\left(\sqrt{\frac{0.25}{68.1(9.81)}}(-40)\right) = 3.470239 \text{ s}$$

Therefore, for t = 2, we can use Eq. (2) to compute

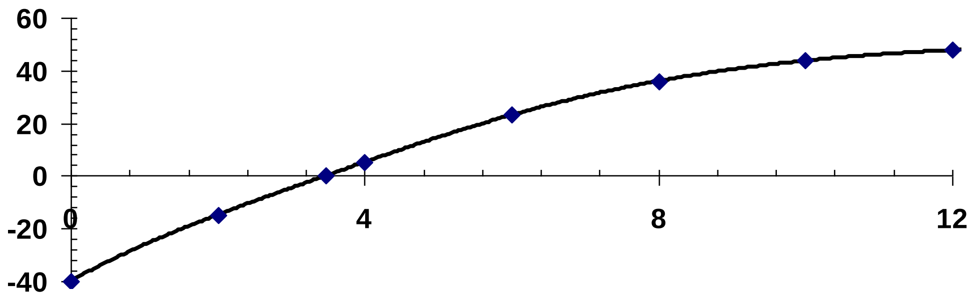
$$v = \sqrt{\frac{68.1(9.81)}{0.25}}\tan\left(\sqrt{\frac{9.81(0.25)}{68.1}}(2) + \tan^{-1}\sqrt{\frac{0.25}{68.1(9.81)}}(-40)\right) = -14.8093\frac{\text{m}}{\text{s}}$$

For t = 4, the jumper is now heading downward and Eq. (3) applies

$$v = \sqrt{\frac{68.1(9.81)}{0.25}}\tanh\left(\sqrt{\frac{9.81(0.25)}{68.1}}(4 - 3.470239)\right) = 5.17952\frac{\text{m}}{\text{s}}$$

The same equation is then used to compute the remaining values. The results for the entire calculation are summarized in the following table and plot:

t **(s)**	***v*** **(m/s)**
0	-40
2	-14.8093
3.470239	0
4	5.17952
6	23.07118
8	35.98203
10	43.69242
12	47.78758



1.3 The following information is available for a bank account:

Date	Deposits	Withdrawals	Balance
5/1			1512.33
	220.13	327.26	
6/1			
	216.80	378.61	
7/1			
	450.25	106.80	
8/1			
	127.31	350.61	
9/1			

Note that the money earns interest which is computed as

$$\text{interest} = iB_i$$

where i = the interest rate expressed as a fraction per month, and B_i the initial balance at the beginning of the month.

(a) Use the conservation of cash to compute the balance on 6/1, 7/1, 8/1, and 9/1 if the interest rate is 1% per month (i = 0.01/month). Show each step in the computation.

(b) Write a differential equation for the cash balance in the form

$$\frac{dB_i}{dt} = f[D(t), W(t), i]$$

where t = time (months), $D(t)$ = deposits as a function of time ($/month), $W(t)$ = withdrawals as a function of time ($/month). For this case, assume that interest is compounded continuously; that is, interest = iB.

(c) Use Euler's method with a time step of 0.5 month to simulate the balance. Assume that the deposits and withdrawals are applied uniformly over the month.

(d) Develop a plot of balance versus time for **(a)** and **(c)**.

==

(*a*) This is a transient computation. For the period ending June 1:

Balance = Previous Balance + Deposits – Withdrawals + Interest

Balance = 1512.33 + 220.13 – 327.26 + 0.01(1512.33) = 1420.32

The balances for the remainder of the periods can be computed in a similar fashion as tabulated below:

Date	Deposit	Withdrawal	Interest	Balance
1-May				$1,512.33
	$220.13	$327.26	$15.12	
1-Jun				$1,420.32
	$216.80	$378.61	$14.20	
1-Jul				$1,272.72
	$450.25	$106.80	$12.73	
1-Aug				$1,628.89
	$127.31	$350.61	$16.29	
1-Sep				**$1,421.88**

(*b*) $\frac{dB}{dt} = D(t) - W(t) + iB$

(*c*) for *t* = 0 to 0.5:

$$\frac{dB}{dt} = 220.13 - 327.26 + 0.01(1512.33) = -92.01$$

$$B(0.5) = 1512.33 - 92.01(0.5) = 1466.33$$

for *t* = 0.5 to 1:

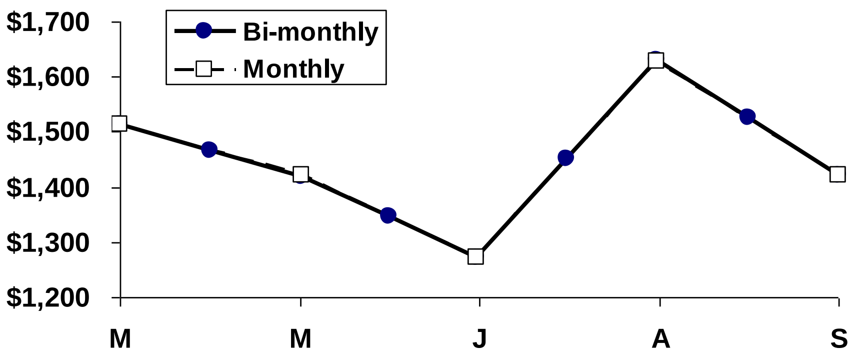
$$\frac{dB}{dt} = 220.13 - 327.260 + 0.01(1466.33) = -92.47$$

$$B(0.5) = 1466.33 - 92.47(0.5) = 1420.09$$

The balances for the remainder of the periods can be computed in a similar fashion as tabulated below:

Date	Deposit	Withdrawal	Interest	*dB*/*dt*	Balance
1-May	$220.13	$327.26	$15.12	-$92.01	$1,512.33
16-May	$220.13	$327.26	$14.66	-$92.47	$1,466.33
1-Jun	$216.80	$378.61	$14.20	-$147.61	$1,420.09
16-Jun	$216.80	$378.61	$13.46	-$148.35	$1,346.29
1-Jul	$450.25	$106.80	$12.72	$356.17	$1,272.12
16-Jul	$450.25	$106.80	$14.50	$357.95	$1,450.20
1-Aug	$127.31	$350.61	$16.29	-$207.01	$1,629.18
16-Aug	$127.31	$350.61	$15.26	-$208.04	$1,525.67
1-Sep					**$1,421.65**

(*d*) As in the plot below, the results of the two approaches are very close.



1.4 Repeat Example 1.2. Compute the velocity to t = 12 s, with a step size of **(a)** 1 and **(b)** 0.5 s. Can you make any statement regarding the errors of the calculation based on the results?

==

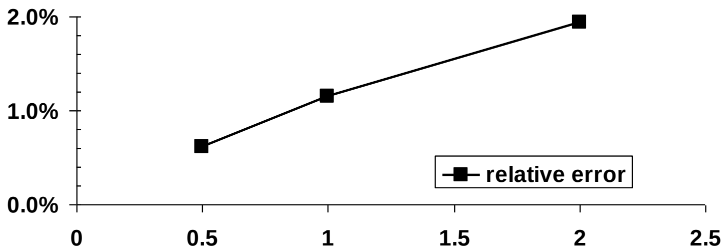
At t = 12 s, the analytical solution is 50.6175 (Example 1.1). The numerical results are:

step	v(12)	absolute relative error
2	51.6008	1.94%
1	51.2008	1.15%
0.5	50.9259	0.61%

where the relative error is calculated with

$$\text{absolute relative error} = \left|\frac{\text{analytical} - \text{numerical}}{\text{analytical}}\right| \times 100\%$$

The error versus step size can be plotted as



Thus, halving the step size approximately halves the error.

1.5 Rather than the nonlinear relationship of Eq. (1.7), you might choose to model the upward force on the bungee jumper as a linear relationship:

$$F_U = -c'v$$

where c' = a first-order drag coefficient (kg/s).

(a) Using calculus, obtain the closed-form solution for the case where the jumper is initially at rest ($v = 0$ at $t = 0$).
(b) Repeat the numerical calculation in Example 1.2 with the same initial condition and parameter values. Use a value of 11.5 kg/s for c'.
===

(a) The force balance is

$$\frac{dv}{dt} = g - \frac{c'}{m}v$$

Applying Laplace transforms,

$$sV - v(0) = \frac{g}{s} - \frac{c'}{m}V$$

Solve for

$$V = \frac{g}{s(s + c'/m)} + \frac{v(0)}{s + c'/m} \qquad (1)$$

The first term to the right of the equal sign can be evaluated by a partial fraction expansion,

$$\frac{g}{s(s + c'/m)} = \frac{A}{s} + \frac{B}{s + c'/m} \qquad (2)$$

$$\frac{g}{s(s + c'/m)} = \frac{A(s + c'/m) + Bs}{s(s + c'/m)}$$

Equating like terms in the numerators yields

$$A + B = 0$$

$$g = \frac{c'}{m}A$$

Therefore,

$$A = \frac{mg}{c'} \qquad B = -\frac{mg}{c'}$$

These results can be substituted into Eq. (2), and the result can be substituted back into Eq. (1) to give

$$V = \frac{mg/c'}{s} - \frac{mg/c'}{s + c'/m} + \frac{v(0)}{s + c'/m}$$

Applying inverse Laplace transforms yields

$$v = \frac{mg}{c'} - \frac{mg}{c'}e^{-(c'/m)t} + v(0)e^{-(c'/m)t}$$

or

$$v = v(0)e^{-(c'/m)t} + \frac{mg}{c'}\left(1 - e^{-(c'/m)t}\right)$$

where the first term to the right of the equal sign is the general solution and the second is the particular solution. For our case, *v*(0) = 0, so the final solution is

$$\boxed{v = \frac{mg}{c'}\left(1 - e^{-(c'/m)t}\right)}$$

Alternative solution: Another way to obtain solutions is to use separation of variables,

$$\int \frac{1}{g - \frac{c'}{m}v}\,dv = \int dt$$

The integrals can be evaluated as

$$-\frac{\ln\left(g - \frac{c'}{m}v\right)}{c'/m} = t + C$$

where *C* = a constant of integration, which can be evaluated by applying the initial condition

$$C = -\frac{\ln\left(g - \frac{c'}{m}v(0)\right)}{c'/m}$$

which can be substituted back into the solution

$$-\frac{\ln\left(g - \frac{c'}{m}v\right)}{c'/m} = t - \frac{\ln\left(g - \frac{c'}{m}v(0)\right)}{c'/m}$$

This result can be rearranged algebraically to solve for *v*,

$$v = v(0)e^{-(c'/m)t} + \frac{mg}{c'}\left(1 - e^{-(c'/m)t}\right)$$

where the first term to the right of the equal sign is the general solution and the second is the particular solution. For our case, *v*(0) = 0, so the final solution is

$$\boxed{v = \frac{mg}{c'}\left(1 - e^{-(c'/m)t}\right)}$$

(b) The numerical solution can be implemented as

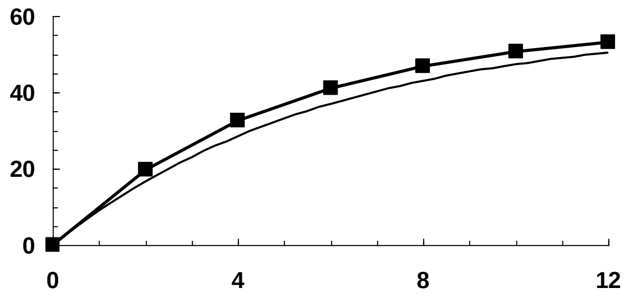
$$v(2) = 0 + \left[9.81 - \frac{11.5}{68.1}(0)\right]2 = 19.62$$

$$v(4) = 19.62 + \left[9.81 - \frac{11.5}{68.1}(19.62)\right]2 = 32.6136$$

The computation can be continued, and the results summarized and plotted as:

t	***v***	***dv/dt***
0	0	9.81
2	19.6200	6.4968
4	32.6136	4.3026
6	41.2187	2.8494
8	46.9176	1.8871
10	50.6917	1.2497

12	53.1911	0.8276
∞	58.0923	



Note that the analytical solution is included on the plot for comparison.

1.6 For the free-falling bungee jumper with linear drag (Prob. 1.5), assume a first jumper is 70 kg and has a drag coefficient of 12 kg/s. If a second jumper has a drag coefficient of 15 kg/s and a mass of 80 kg, how long will it take her to reach the same velocity jumper 1 reached in 9 s?

==

$$v(t) = \frac{gm}{c'}(1 - e^{-(c'/m)\,t})$$

jumper #1: $v(t) = \frac{9.81(70)}{12}(1 - e^{-(12/70)\,9}) = 44.99204\,\frac{\text{m}}{\text{s}}$

jumper #2: $44.99204 = \frac{9.81(80)}{15}(1 - e^{-(15/80)\,t})$

$$44.99204 = 52.32 - 52.32e^{-0.1875\,t}$$

$$0.14006 = e^{-0.1875\,t}$$

$$t = \frac{\ln 0.14006}{-0.1875} = 10.4836 \text{ s}$$

1.7 For the second-order drag model (Eq. 1.8), compute the velocity of a free-falling parachutist using Euler's method for the case where m = 80 kg and c_d = 0.25 kg/m. Perform the calculation from t = 0 to 20 s with a step size of 1 s. Use an initial condition that the parachutist has an upward velocity of 20 m/s at t = 0. At t = 10 s, assume that the chute is instantaneously deployed so that the drag coefficient jumps to 1.5 kg/m.

==

Note that the differential equation should be formulated as

$$\frac{dv}{dt} = g - \frac{c_d}{m} v|v|$$

This ensures that the sign of the drag is correct when the parachutist has a negative upward velocity. Before the chute opens ($t < 10$), Euler's method can be implemented as

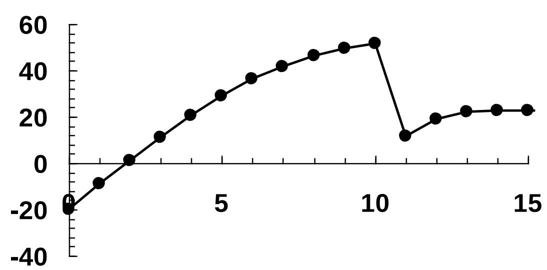
$$v(t+\Delta t) = v(t) + \left[9.81 - \frac{0.25}{80} v|v|\right]\Delta t$$

After the chute opens ($t \geq 10$), the drag coefficient is changed and the implementation becomes

$$v(t+\Delta t) = v(t) + \left[9.81 - \frac{1.5}{80} v|v|\right]\Delta t$$

Here is a summary of the results along with a plot:

Chute closed			Chute opened		
t	***v***	***dv/dt***	***t***	***v***	***dv/dt***
0	-20.0000	11.0600	10	51.5260	-39.9698
1	-8.9400	10.0598	11	11.5561	7.3060
2	1.1198	9.8061	12	18.8622	3.1391
3	10.9258	9.4370	13	22.0013	0.7340
4	20.3628	8.5142	14	22.7352	0.1183
5	28.8770	7.2041	15	22.8535	0.0172
6	36.0812	5.7417	16	22.8707	0.0025
7	41.8229	4.3439	17	22.8732	0.0003
8	46.1668	3.1495	18	22.8735	0.0000
9	49.3162	2.2097	19	22.8736	0.0000
			20	22.8736	0.0000



1.8 The amount of a uniformly distributed radioactive contaminant contained in a closed reactor is measured by its concentration c (becquerel/liter or Bq/L). The contaminant decreases at a decay rate proportional to its concentration; that is

Decay rate $= -kc$

where k is a constant with units of day^{-1}. Therefore, according to Eq. (1.14), a mass balance for the reactor can be written as

$$\frac{dc}{dt} = -kc$$

$$\begin{pmatrix}\text{change}\\ \text{in mass}\end{pmatrix} = \begin{pmatrix}\text{decrease}\\ \text{by decay}\end{pmatrix}$$

(a) Use Euler's method to solve this equation from $t = 0$ to 1 d with $k = 0.175$ d^{-1}. Employ a step size of $\Delta t = 0.1$ d. The concentration at $t = 0$ is 100 Bq/L.

(b) Plot the solution on a semi-log graph (i.e., ln c versus t) and determine the slope. Interpret your results.

==

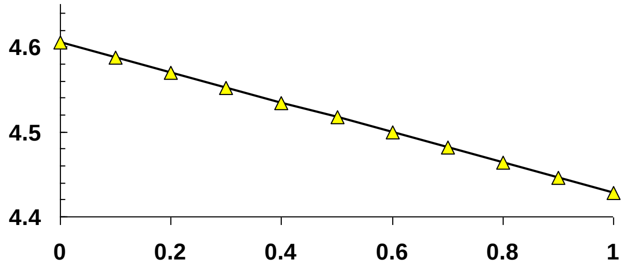
$$c(0.1) = 100 - 0.175(10)0.1 = 98.25 \text{ Bq/L}$$

$$c(0.2) = 98.25 - 0.175(98.25)0.1 = 96.5306 \text{ Bq/L}$$

The process can be continued to yield

t	*c*	*dc/dt*
0	100.0000	-17.5000
0.1	98.2500	-17.1938
0.2	96.5306	-16.8929
0.3	94.8413	-16.5972
0.4	93.1816	-16.3068
0.5	91.5509	-16.0214
0.6	89.9488	-15.7410
0.7	88.3747	-15.4656
0.8	86.8281	-15.1949
0.9	85.3086	-14.9290
1	83.8157	-14.6678

(b) The results when plotted on a semi-log plot yields a straight line



The slope of this line can be estimated as

$$\frac{\ln(83.8157)-\ln(100)}{1}=-0.17655$$

Thus, the slope is approximately equal to the negative of the decay rate. If we had used a smaller step size, the result would be more exact.

1.9 A storage tank (Fig. P1.9) contains a liquid at depth *y* where *y* = 0 when the tank is half full. Liquid is withdrawn at a constant flow rate *Q* to meet demands. The contents are resupplied at a sinusoidal rate 3*Q* $\sin^2(t)$. Equation (1.14) can be written for this system as

$$\frac{d(Ay)}{dt} = 3Q\sin^2(t) - Q$$

$$\begin{pmatrix}\text{change in}\\ \text{volume}\end{pmatrix} = (\text{inflow}) - (\text{outflow})$$

or, since the surface area *A* is constant

$$\frac{dy}{dt} = 3\frac{Q}{A}\sin^2(t) - \frac{Q}{A}$$

Use Euler's method to solve for the depth *y* from *t* = 0 to 10 d with a step size of 0.5 d. The parameter values are $A = 1250\ \text{m}^2$ and $Q = 450\ \text{m}^3/\text{d}$. Assume that the initial condition is *y* = 0.

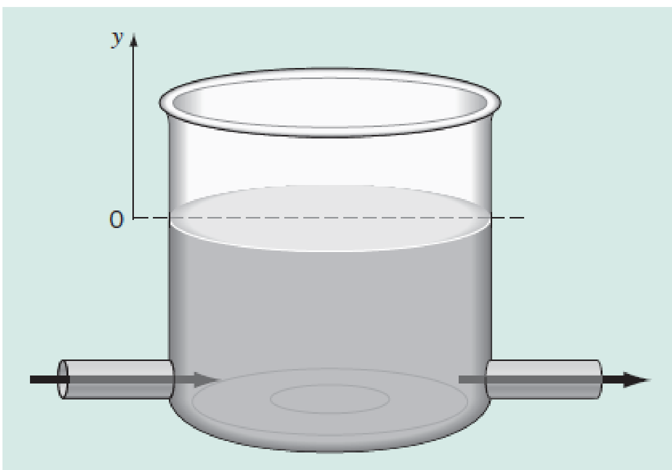


FIGURE P1.9

==

The first two steps yield

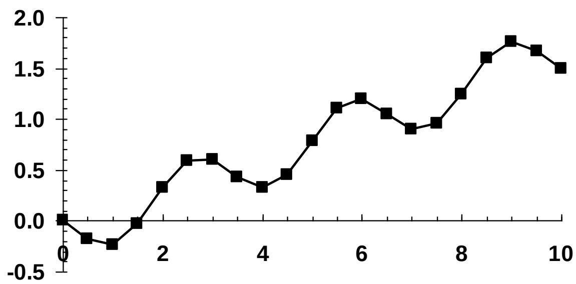
$$y(0.5) = 0 + \left[3\frac{450}{1250}\sin^2(0) - \frac{450}{1250}\right]0.5 = 0 + (-0.36)\ 0.5 = -0.18$$

$$y(1) = -0.18 + \left[3\frac{450}{1250}\sin^2(0.5) - \frac{450}{1250}\right]0.5 = -0.18 + (-0.11176)\ 0.5 = -0.23508$$

The process can be continued to give the following table and plot:

t	*y*	*dy/dt*	*t*	*y*	*dy/dt*
0	0.00000	-0.36000	5.5	1.10271	0.17761
0.5	-0.18000	-0.11176	6	1.19152	-0.27568
1	-0.23588	0.40472	6.5	1.05368	-0.31002
1.5	-0.03352	0.71460	7	0.89866	0.10616
2	0.32378	0.53297	7.5	0.95175	0.59023
2.5	0.59026	0.02682	8	1.24686	0.69714
3	0.60367	-0.33849	8.5	1.59543	0.32859

3.5	0.43443	-0.22711	9	1.75972	-0.17657
4	0.32087	0.25857	9.5	1.67144	-0.35390
4.5	0.45016	0.67201	10	1.49449	-0.04036
5	0.78616	0.63310			



1.10 For the same storage tank described in Prob. 1.9, suppose that the outflow is not constant but rather depends on the depth. For this case, the differential equation for depth can be written as

$$\frac{dy}{dt} = 3\frac{Q}{A}\sin^2(t) - \frac{\alpha(1+y)^{1.5}}{A}$$

Use Euler's method to solve for the depth y from t = 0 to 10 d with a step size of 0.5 d. The parameter values are A = 1250 m^2, Q = 450 m^3/d, and α = 150. Assume that the initial condition is y = 0.

==

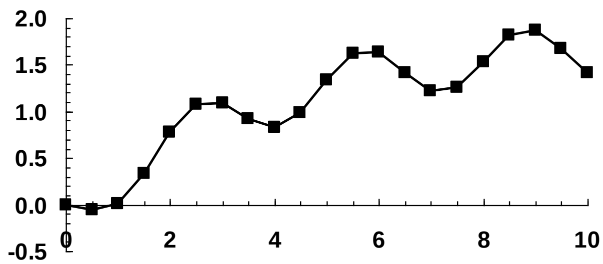
The first two steps yield

$$y(0.5) = 0 + \left[3\frac{450}{1250}\sin^2(0) - \frac{150(1+0)^{1.5}}{1250}\right]0.5 = 0 - 0.12(0.5) = \ -0.06$$

$$y(1) = -0.06 + \left[3\frac{450}{1250}\sin^2(0.5) - \frac{150(1-0.06)^{1.5}}{1250}\right]0.5 = -0.06 + 0.13887(0.5) = 0.00944$$

The process can be continued to give

t	***y***	***dy/dt***	***t***	***y***	***dy/dt***
0	0.00000	-0.12000	5.5	1.61981	0.02876
0.5	-0.06000	0.13887	6	1.63419	-0.42872
1	0.00944	0.64302	6.5	1.41983	-0.40173
1.5	0.33094	0.89034	7	1.21897	0.06951
2	0.77611	0.60892	7.5	1.25372	0.54423
2.5	1.08058	0.02669	8	1.52584	0.57542
3	1.09392	-0.34209	8.5	1.81355	0.12227
3.5	0.92288	-0.18708	9	1.87468	-0.40145
4	0.82934	0.32166	9.5	1.67396	-0.51860
4.5	0.99017	0.69510	10	1.41465	-0.13062
5	1.33772	0.56419			



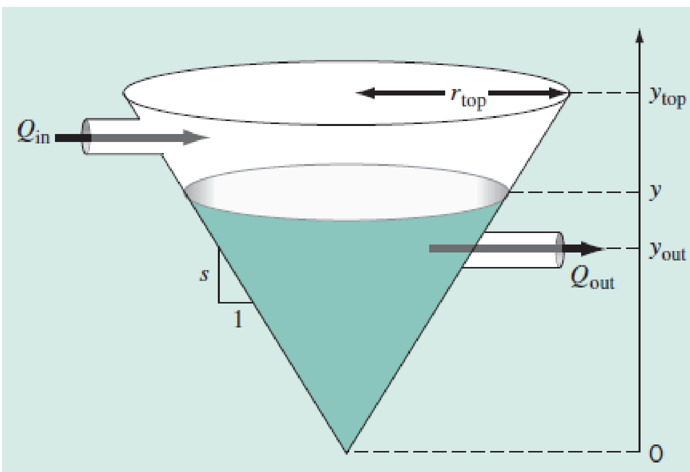
1.11 Apply the conservation of volume (see Prob. 1.9) to simulate the level of liquid in a conical storage tank (Fig. P1.11).

The liquid flows in at a sinusoidal rate of $Q_{in} = 3\sin^2(t)$ and flows out according to

$$Q_{out} = 3(y - y_{out})^{1.5} \qquad y > y_{out}$$
$$Q_{out} = 0 \qquad y \le y_{out}$$

where flow has units of m³/d and y = the elevation of the water surface above the bottom of the tank (m). Use Euler's method to solve for the depth y from $t = 0$ to 10 d with a step size of 0.5 d. The parameter values are $r_{top} = 2.5$ m, $y_{top} = 4$ m, and $y_{out} = 1$ m. Assume that the level is initially below the outlet pipe with $y(0) = 0.8$ m.

FIGURE P1.11



==

When the water level is above the outlet pipe, the volume balance can be written as

$$\frac{dV}{dt} = 3\sin^2(t) - 3(y - y_{out})^{1.5}$$

In order to solve this equation, we must relate the volume to the level. To do this, we recognize that the volume of a cone is given by $V = \pi r^2 y/3$. Defining the side slope as $s = y_{top}/r_{top}$, the radius can be related to the level ($r = y/s$) and the volume can be reexpressed as

$$V = \frac{\pi}{3s^2} y^3$$

which can be solved for

$$y = \sqrt[3]{\frac{3s^2 V}{\pi}} \qquad (1)$$

and substituted into the volume balance

$$\frac{dV}{dt} = 3\sin^2(t) - 3\left(\sqrt[3]{\frac{3s^2V}{\pi}} - y_{\text{out}}\right)^{1.5} \qquad (2)$$

For the case where the level is below the outlet pipe, outflow is zero and the volume balance simplifies to

$$\frac{dV}{dt} = 3\sin^2(t) \qquad (3)$$

These equations can then be used to solve the problem. Using the side slope of $s = 4/2.5 = 1.6$, the initial volume can be computed as

$$V(0) = \frac{\pi}{3(1.6)^2} 0.8^3 = 0.20944 \text{ m}^3$$

For the first step, $y < y_{\text{out}}$ and Eq. (3) gives

$$\frac{dV}{dt}(0) = 3\sin^2(0) = 0$$

and Euler's method yields

$$V(0.5) = V(0) + \frac{dV}{dt}(0)\Delta t = 0.20944 + 0(0.5) = 0.20944$$

For the second step, Eq. (3) still holds and

$$\frac{dV}{dt}(0.5) = 3\sin^2(0.5) = 0.689547$$

$$V(1) = V(0.5) + \frac{dV}{dt}(0.5)\Delta t = 0.20944 + 0.689547(0.5) = 0.554213$$

Equation (1) can then be used to compute the new level,

$$y = \sqrt[3]{\frac{3(1.6)^2(0.554213)}{\pi}} = 1.106529 \text{ m}$$

Because this level is now higher than the outlet pipe, Eq. (2) holds for the next step

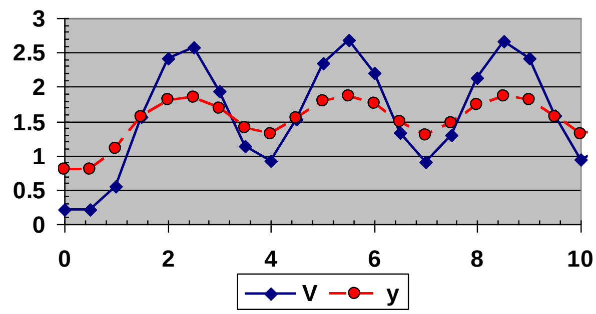
$$\frac{dV}{dt}(1) = 2.12422 - 3(1.106529 - 1)^{1.5} = 2.019912$$

$$V(1.5) = 0.554213 + 2.019912(0.5) = 1.564169$$

The remainder of the calculation is summarized in the following table and figure.

t	Q_{in}	V	y	Q_{out}	dV/dt
0	0	0.20944	0.8	0	0
0.5	0.689547	0.20944	0.8	0	0.689547
1	2.12422	0.554213	1.106529	0.104309	2.019912
1.5	2.984989	1.564169	1.563742	1.269817	1.715171
2	2.480465	2.421754	1.809036	2.183096	0.29737
2.5	1.074507	2.570439	1.845325	2.331615	-1.25711
3	0.059745	1.941885	1.680654	1.684654	-1.62491
3.5	0.369147	1.12943	1.40289	0.767186	-0.39804
4	1.71825	0.93041	1.31511	0.530657	1.187593

4.5	2.866695	1.524207	1.55031	1.224706	1.641989
5	2.758607	2.345202	1.78977	2.105581	0.653026
5.5	1.493361	2.671715	1.869249	2.431294	-0.93793
6	0.234219	2.202748	1.752772	1.95937	-1.72515
6.5	0.13883	1.340173	1.48522	1.013979	-0.87515
7	1.294894	0.902598	1.301873	0.497574	0.79732
7.5	2.639532	1.301258	1.470703	0.968817	1.670715
8	2.936489	2.136616	1.735052	1.890596	1.045893
8.5	1.912745	2.659563	1.866411	2.419396	-0.50665
9	0.509525	2.406237	1.805164	2.167442	-1.65792
9.5	0.016943	1.577279	1.568098	1.284566	-1.26762
10	0.887877	0.943467	1.321233	0.5462	0.341677



1.12 A group of 35 students attend a class in an insulated room which measures 11 by 8 by 3 m. Each student takes up about 0.075 m^3 and gives out about 80 W of heat (1 W = 1 J/s). Calculate the air temperature rise during the first 20 minutes of the class if the room is completely sealed and insulated. Assume the heat capacity C_v for air is 0.718 kJ/(kg K). Assume air is an ideal gas at 20 °C and 101.325 kPa. Note that the heat absorbed by the air Q is related to the mass of the air m the heat capacity, and the change in temperature by the following relationship:

$$Q = m\int_{T_1}^{T_2} C_v \, dT = mC_v(T_2 - T_1)$$

The mass of air can be obtained from the ideal gas law:

$$PV = \frac{m}{\text{Mwt}} RT$$

where P is the gas pressure, V is the volume of the gas, Mwt is the molecular weight of the gas (for air, 28.97 kg/kmol), and R is the ideal gas constant [8.314 kPa m^3/(kmol K)].

===

$$Q_{\text{students}} = 35 \text{ ind} \times 80 \frac{\text{J}}{\text{ind s}} \times 20 \text{ min} \times 60 \frac{\text{s}}{\text{min}} \times \frac{\text{kJ}}{1000 \text{ J}} = 3{,}360 \text{ kJ}$$

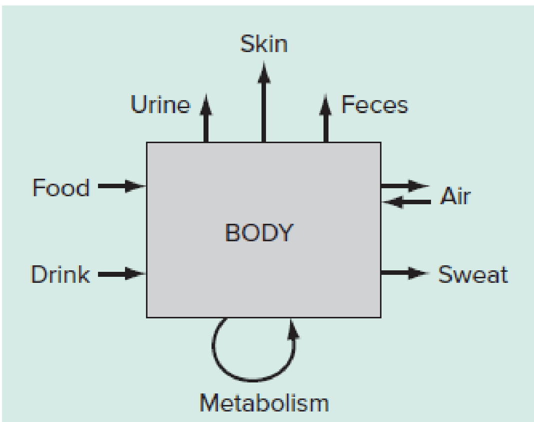
$$m = \frac{PV\text{Mwt}}{RT} = \frac{(101.325 \text{ kPa})(11\text{m} \times 8\text{m} \times 3\text{m} - 35 \times 0.075 \text{ m}^3)(28.97 \text{ kg/kmol})}{(8.314 \text{ kPa m}^3/(\text{kmol K})((20+273.15)\text{K})} = 314.796 \text{ kg}$$

$$\Delta T = \frac{Q_{\text{students}}}{mC_v} = \frac{3{,}360 \text{ kJ}}{(314.796 \text{ kg})(0.718 \text{ kJ/(kg K)})} = 14.86571 \text{ K}$$

Thus, the rise in temperature during the 20 minutes of the class is 14.86571 K. Therefore, the final temperature is (20 + 273.15) + 14.86571 = 308.01571 K**.**

1.13 Figure P1.13 depicts the various ways in which an average man gains and loses water in one day. One liter is ingested as food, and the body metabolically produces 0.3 liters. In breathing air, the exchange is 0.05 liters while inhaling, and 0.4 liters while exhaling over a one-day period. The body will also lose 0.3, 1.4, 0.2, and 0.35 liters through sweat, urine, feces, and through the skin, respectively. To maintain steady state, how much water must be drunk per day?

FIGURE P1.13



===

$$\sum M_{in} - \sum M_{out} = 0$$

$$\text{Food} + \text{Drink} + \text{Air In} + \text{Metabolism} = \text{Urine} + \text{Skin} + \text{Feces} + \text{Air Out} + \text{Sweat}$$

$$\text{Drink} = \text{Urine} + \text{Skin} + \text{Feces} + \text{Air Out} + \text{Sweat} - \text{Food} - \text{Air In} - \text{Metabolism}$$

$$\text{Drink} = 1.4 + 0.35 + 0.2 + 0.4 + 0.3 - 1 - 0.05 - 0.3 = 1.3 \text{ L}$$

1.14 In our example of the free-falling bungee jumper, we assumed that the acceleration due to gravity was a constant value of 9.81 m/s^2. Although this is a decent approximation when we are examining falling objects near the surface of the earth, the gravitational force decreases as we move above sea level. A more general representation based on Newton's inverse square law of gravitational attraction can be written as

$$g(x) = g(0)\frac{R^2}{(R+x)^2}$$

where $g(x)$ = gravitational acceleration at altitude x (in m) measured upward from the earth's surface (m/s^2), $g(0)$ = gravitational acceleration at the earth's surface ($\cong$ 9.81 m/s^2), and R = the earth's radius ($\cong 6.37 \times 10^6$ m).

(a) In a fashion similar to the derivation of Eq. (1.8), use a force balance to derive a differential equation for velocity as a function of time that utilizes this more complete representation of gravitation. However, for this derivation, assume that upward velocity is positive.

(b) For the case where drag is negligible, use the chain rule to express the differential equation as a function of altitude rather than time. Recall that the chain rule is

$$\frac{dv}{dt} = \frac{dv}{dx}\frac{dx}{dt}$$

(c) Use calculus to obtain the closed form solution where $v = v_0$ at $x = 0$.

(d) Use Euler's method to obtain a numerical solution from $x = 0$ to 100,000 m using a step of 10,000 m where the initial velocity is 1500 m/s upward. Compare your result with the analytical solution.

==

(a) The force balance can be written as:

$$m\frac{dv}{dt} = -mg(0)\frac{R^2}{(R+x)^2} + c_d v|v|$$

Dividing by mass gives

$$\frac{dv}{dt} = -g(0)\frac{R^2}{(R+x)^2} + \frac{c_d}{m}v|v|$$

(b) Recognizing that $dx/dt = v$, the chain rule is

$$\frac{dv}{dt} = v\frac{dv}{dx}$$

Setting drag to zero and substituting this relationship into the force balance gives

$$\frac{dv}{dx} = -\frac{g(0)}{v}\frac{R^2}{(R+x)^2}$$

(c) Using separation of variables

$$v\,dv = -g(0)\frac{R^2}{(R+x)^2}dx$$

Integrating gives

$$\frac{v^2}{2} = g(0)\frac{R^2}{R+x} + C$$

Applying the initial condition yields

$$\frac{v_0^2}{2} = g(0)\frac{R^2}{R+0} + C$$

which can be solved for $C = v_0^2/2 - g(0)R$, which can be substituted back into the solution to give

$$\frac{v^2}{2} = g(0)\frac{R^2}{R+x} + \frac{v_0^2}{2} - g(0)R$$

or

$$v = \pm\sqrt{v_0^2 + 2g(0)\frac{R^2}{R+x} - 2g(0)R}$$

Note that the plus sign holds when the object is moving upwards and the minus sign holds when it is falling.

(d) Euler's method can be developed as

$$v(x_{i+1}) = v(x_i) + \left[-\frac{g(0)}{v(x_i)}\frac{R^2}{(R+x_i)^2}\right](x_{i+1} - x_i)$$

The first step can be computed as

$$v(10{,}000) = 1{,}500 + \left[-\frac{9.81}{1{,}500}\frac{(6.37\times10^6)^2}{(6.37\times10^6+0)^2}\right](10{,}000-0) = 1{,}500 + (-0.00654)10{,}000 = 1434.600$$

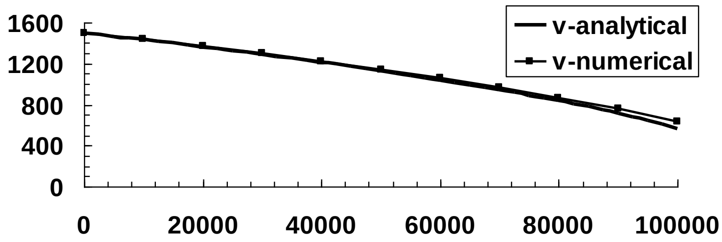
The remainder of the calculations can be implemented in a similar fashion as in the following table

x	*v*	*dv/dx*	*v*-analytical
0	1500.000	-0.00654	1500.000
10000	1434.600	-0.00682	1433.216
20000	1366.433	-0.00713	1363.388
30000	1295.089	-0.00750	1290.023
40000	1220.050	-0.00794	1212.476
50000	1140.644	-0.00847	1129.885
60000	1055.974	-0.00912	1041.050
70000	964.800	-0.00995	944.208
80000	865.319	-0.01106	836.581
90000	754.745	-0.01264	713.303
100000	628.364	-0.01513	564.203

For the analytical solution, the value at 10,000 m can be computed as

$$v = \sqrt{1{,}500^2 + 2(9.81)\frac{(6.37\times10^6)^2}{(6.37\times10^6+10{,}000)} - 2(9.81)(6.37\times10^6)} = 1433.216$$

The remainder of the analytical values can be implemented in a similar fashion as in the last column of the above table. The numerical and analytical solutions can be displayed graphically.


1600
1200
800
400
0
0
20000
40000
60000
80000
100000
v-analytical
v-numerical

1.15 Suppose that a spherical droplet of liquid evaporates at a rate that is proportional to its surface area.

$$\frac{dV}{dt} = -kA$$

where V = volume (mm^3), t = time (min), k = the evaporation rate (mm/min), and A = surface area (mm^2). Use Euler's method to compute the volume of the droplet from $t = 0$ to 10 min using a step size of 0.25 min. Assume that $k = 0.08$ mm/min and that the droplet initially has a radius of 2.5 mm. Assess the validity of your results by determining the radius of your final computed volume and verifying that it is consistent with the evaporation rate.

==

The volume of the droplet is related to the radius as

$$V = \frac{4\pi r^3}{3} \tag{1}$$

This equation can be solved for radius as

$$r = \sqrt[3]{\frac{3V}{4\pi}} \tag{2}$$

The surface area is

$$A = 4\pi r^2 \tag{3}$$

Equation (2) can be substituted into Eq. (3) to express area as a function of volume

$$A = 4\pi\left(\frac{3V}{4\pi}\right)^{2/3}$$

This result can then be substituted into the original differential equation,

$$\frac{dV}{dt} = -k4\pi\left(\frac{3V}{4\pi}\right)^{2/3} \tag{4}$$

The initial volume can be computed with Eq. (1),

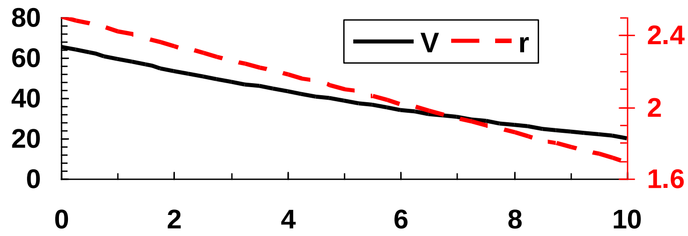
$$V = \frac{4\pi r^3}{3} = \frac{4\pi(2.5)^3}{3} = 65.44985 \text{ mm}^3$$

Euler's method can be used to integrate Eq. (4). Here are the beginning and last steps

t	*V*	*dV/dt*
0	65.44985	-6.28319
0.25	63.87905	-6.18225
0.5	62.33349	-6.08212
0.75	60.81296	-5.98281
1	59.31726	-5.8843
	•	
	•	
	•	
9	23.35079	-3.16064
9.25	22.56063	-3.08893
9.5	21.7884	-3.01804
9.75	21.03389	-2.94795

10	20.2969	-2.87868

A plot of the results is shown below. We have included the radius on this plot (dashed line and right scale):



Eq. (2) can be used to compute the final radius as

$$r = \sqrt[3]{\frac{3(20.2969)}{4\pi}} = 1.692182$$

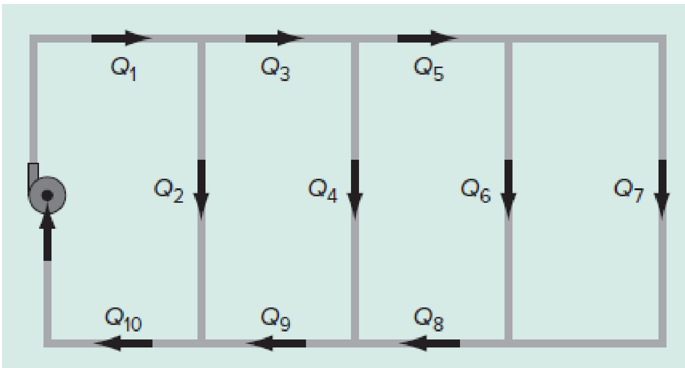
Therefore, the average evaporation rate can be computed as

$$k = \frac{(2.5 - 1.692182)\text{ mm}}{10\text{ min}} = 0.080782\frac{\text{mm}}{\text{min}}$$

which is approximately equal to the given evaporation rate of 0.08 mm/min.

1.16 A fluid is pumped into the network shown in Fig. P1.16. If $Q_2 = 0.7$, $Q_3 = 0.5$, $Q_7 = 0.1$, and $Q_8 = 0.3$ m^3/s, determine the other flows.

FIGURE P1.16

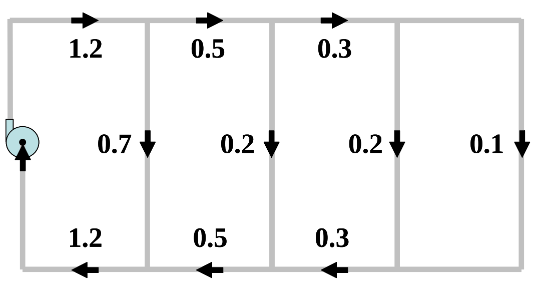


==

Continuity at the nodes can be used to determine the flows as follows:

$$Q_1 = Q_2 + Q_3 = 0.7 + 0.5 = 1.2 \text{ m}^3/\text{s}$$
$$Q_{10} = Q_1 = 1.2 \text{ m}^3/\text{s}$$
$$Q_9 = Q_{10} - Q_2 = 1.2 - 0.7 = 0.5 \text{ m}^3/\text{s}$$
$$Q_4 = Q_9 - Q_8 = 0.5 - 0.3 = 0.2 \text{ m}^3/\text{s}$$
$$Q_5 = Q_3 - Q_4 = 0.5 - 0.2 = 0.3 \text{ m}^3/\text{s}$$
$$Q_6 = Q_5 - Q_7 = 0.3 - 0.1 = 0.2 \text{ m}^3/\text{s}$$

Therefore, the final results are



1.17 *Newton's law of cooling* says that the temperature of a body changes at a rate proportional to the difference between its temperature and that of the surrounding medium (the ambient temperature),

$$\frac{dT}{dt} = -k(T - T_a)$$

where T = the temperature of the body (°C), t = time (min), k = the proportionality constant (per minute), and T_a = the ambient temperature (°C). Suppose that a cup of coffee originally has a temperature of 70 °C. Use Euler's method to compute the temperature from t = 0 to 20 min using a step size of 2 min if T_a = 20 °C and k = 0.019/min.

==

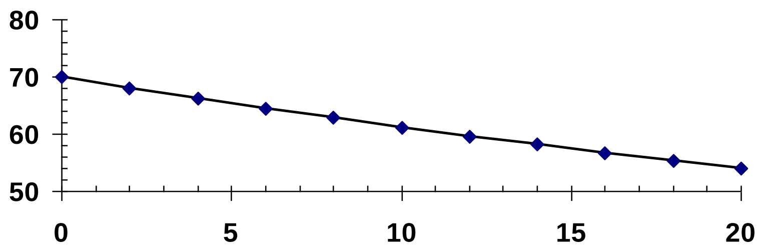
The first two steps can be computed as

$$T(1) = 70 + \left[-0.019(70 - 20)\right] 2 = 68 + (-0.95)2 = 68.1$$

$$T(2) = 68.1 + \left[-0.019(68.1 - 20)\right] 2 = 68.1 + (-0.9139)2 = 66.2722$$

The remaining results are displayed below along with a plot of the results.

t	***T***	***dT/dt***	***t***	***T***	***dT/dt***
0	70.00000	-0.95000	12.00000	59.62967	-0.75296
2	68.10000	-0.91390	14.00000	58.12374	-0.72435
4	66.27220	-0.87917	16.00000	56.67504	-0.69683
6	64.51386	-0.84576	18.00000	55.28139	-0.67035
8	62.82233	-0.81362	20.00000	53.94069	-0.64487
10	61.19508	-0.78271			



1.18 You are working as a crime scene investigator and must predict the temperature of a homicide victim over a 5-hr period. You know that the room where the victim was found was at 10 °C when the body was discovered.

(a) Use Newton's law of cooling (Prob. 1.17) and Euler's method to compute the victim's body temperature for the 5-hr period using values of $k = 0.12$/hr and $\Delta t = 0.5$ hr. Assume that the victim's body temperature at the time of death was 37 °C, and that the room temperature was at a constant value of 10 °C over the 5-hr period.

(b) Further investigation reveals that the room temperature had actually dropped linearly from 20 to 10 °C over the 5-hr period. Repeat the same calculation as in **(a)** but incorporate this new information.

(c) Compare the results from **(a)** and **(b)** by plotting them on the same graph.

==

(a) For the constant temperature case, Newton's law of cooling is written as

$$\frac{dT}{dt} = -0.12(T - 10)$$

The first two steps of Euler's methods are

$$T(0.5) = T(0) - \frac{dT}{dt}(0) \times \Delta t = 37 + 0.12(10 - 37)(0.5) = 37 - 3.2400 \times 0.50 = 35.3800$$

$$T(1) = 35.3800 + 0.12(10 - 35.3800)(0.5) = 35.3800 - 3.0456 \times 0.50 = 33.8572$$

The remaining calculations are summarized in the following table:

t	T_a	***T***	***dT/dt***
0:00	10	37.0000	-3.2400
0:30	10	35.3800	-3.0456
1:00	10	33.8572	-2.8629
1:30	10	32.4258	-2.6911
2:00	10	31.0802	-2.5296
2:30	10	29.8154	-2.3778
3:00	10	28.6265	-2.2352
3:30	10	27.5089	-2.1011
4:00	10	26.4584	-1.9750
4:30	10	25.4709	-1.8565
5:00	10	24.5426	-1.7451

(b) For this case, the room temperature can be represented as

$$T_a = 20 - 2t$$

where t = time (hrs). Newton's law of cooling is written as

$$\frac{dT}{dt} = -0.12(T - 20 + 2t)$$

The first two steps of Euler's methods are

$$T(0.5) = 37 + 0.12(20 - 37)(0.5) = 37 - 2.040 \times 0.50 = 35.9800$$

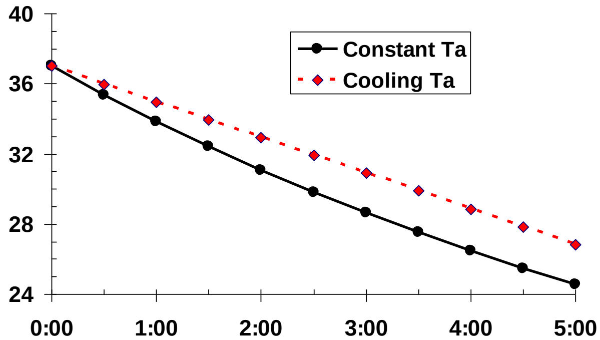
$$T(1) = 35.9800 + 0.12(19 - 35.9800)(0.5) = 35.9800 - 2.0376 \times 0.50 = 34.9612$$

The remaining calculations are summarized in the following table:

t	T_a	T	dT/dt
0:00	20	37.0000	-2.0400
0:30	19	35.9800	-2.0376
1:00	18	34.9612	-2.0353
1:30	17	33.9435	-2.0332
2:00	16	32.9269	-2.0312
2:30	15	31.9113	-2.0294
3:00	14	30.8966	-2.0276
3:30	13	29.8828	-2.0259
4:00	12	28.8699	-2.0244
4:30	11	27.8577	-2.0229
5:00	10	26.8462	-2.0215

Comparison with (a) indicates that the effect of the room air temperature has a significant effect on the expected temperature at the end of the 5-hr period (difference = 26.8462 – 24.5426 = 2.3036ºC).

(c) The solutions for (a) Constant T_a, and (b) Cooling T_a are plotted below:



1.19 The velocity is equal to the rate of change of distance, *x* (m):

$$\frac{dx}{dt} = v(t) \quad \text{(P1.19)}$$

Use Euler's method to numerically integrate Eqs. (P1.19) and (1.8) in order to determine both the velocity and distance fallen as a function of time for the first 10 seconds of freefall using the same parameters and conditions as in Example 1.2. Develop a plot of your results.

==

The two equations to be solved are

$$\frac{dv}{dt} = g - \frac{c_d}{m} v^2$$
$$\frac{dx}{dt} = v$$

Euler's method can be applied for the first step as

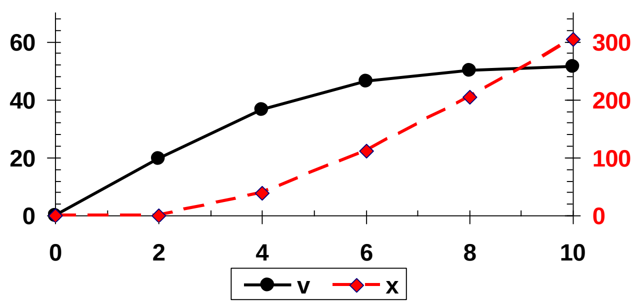
$$v(2) = v(0) + \frac{dv}{dt}(0)\Delta t = 0 + \left(9.81 - \frac{0.25}{68.1}(0)^2\right)(2) = 19.6200$$
$$x(2) = x(0) + \frac{dx}{dt}(0)\Delta t = 0 + 0(2) = 0$$

For the second step:

$$v(4) = v(2) + \frac{dv}{dt}(2)\Delta t = 19.6200 + \left(9.81 - \frac{0.25}{68.1}(19.6200)^2\right)(2) = 19.6200 + 8.3968(2) = 36.4137$$
$$x(4) = x(2) + \frac{dx}{dt}(2)\Delta t = 0 + 19.6200(2) = 39.2400$$

The remaining steps can be computed in a similar fashion as tabulated and plotted below:

t	*x*	*v*	*dx/dt*	*dv/dt*
0	0.0000	0.0000	0.0000	9.8100
2	0.0000	19.6200	19.6200	8.3968
4	39.2400	36.4137	36.4137	4.9423
6	112.0674	46.2983	46.2983	1.9409
8	204.6640	50.1802	50.1802	0.5661
10	305.0244	51.3123	51.3123	0.1442



1.20 In addition to the downward force of gravity (weight) and drag, an object falling through a fluid is also subject to a buoyancy force which is proportional to the displaced volume (*Archimedes' principle*). For example, for a sphere with diameter d (m), the sphere's volume is $V = \pi d^3/6$, and its projected area is $A = \pi d^2/4$. The buoyancy force can then be computed as $Fb = -\rho V g$. We neglected buoyancy in our derivation of Eq. (1.8) because it is relatively small for an object like a bungee jumper moving through air. However, for a denser fluid like water, it becomes more prominent.

(a) Derive a differential equation in the same fashion as Eq. (1.8) but include the buoyancy force and represent the drag force as described in Sec. 1.4.

(b) Rewrite the differential equation from **(a)** for the special case of a sphere.

(c) Use the equation developed in **(b)** to compute the terminal velocity (i.e., for the steady-state case). Use the following parameter values for a sphere falling through water: sphere diameter = 1 cm, sphere density = 2700 kg/m^3, water density = 1000 kg/m^3, and $C_d = 0.47$.

(d) Use Euler's method with a step size of $\Delta t = 0.03125$ s to numerically solve for the velocity from $t = 0$ to 0.25 s with an initial velocity of zero.

===

(a) The force balance with buoyancy can be written as

$$m\frac{dv}{dt} = mg - \frac{1}{2}\rho v|v| AC_d - \rho V g$$

Divide both sides by mass,

$$\frac{dv}{dt} = g\left(1 - \frac{\rho V}{m}\right) - \frac{\rho A C_d}{2m} v|v|$$

(b) For a sphere, the mass is related to the volume as in $m = \rho_s V$ where ρ_s = the sphere's density (kg/m^3). Substituting this relationship gives

$$\frac{dv}{dt} = g\left(1 - \frac{\rho}{\rho_s}\right) - \frac{\rho A C_d}{2\rho_s V} v|v|$$

The formulas for the volume and projected area can be substituted to give

$$\frac{dv}{dt} = g\left(1 - \frac{\rho}{\rho_s}\right) - \frac{3\rho C_d}{4\rho_s d} v|v|$$

(c) At steady state ($dv/dt = 0$),

$$g\left(\frac{\rho_s - \rho}{\rho_s}\right) = \frac{3\rho C_d}{4\rho_s d} v^2$$

which can be solved for the terminal velocity

$$v = \sqrt{\frac{4gd}{3C_d}\left(\frac{\rho_s - \rho}{\rho}\right)}$$

Substituting the values, the terminal velocity is found to be,

$$v = \sqrt{\frac{4 \times 9.8 \times 0.01}{3 \times 0.47}\left(\frac{2700 - 1000}{1000}\right)} = 0.68783 \ \frac{\text{m}}{\text{s}}$$

(d) Before implementing Euler's method, the parameters can be substituted into the differential equation to give

$$\frac{dv}{dt} = 9.81\left(1 - \frac{1000}{2700}\right) - \frac{3(1000)0.47}{4(2700)(0.01)}v^2 = 6.176667 - 13.055556v^2$$

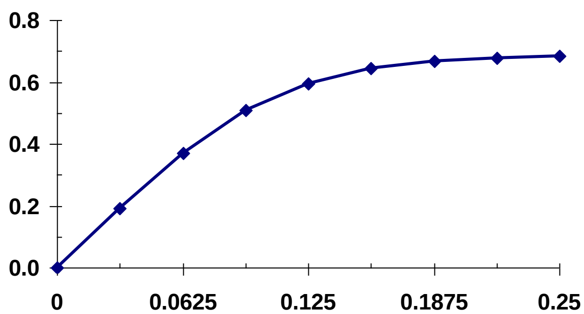
The first two steps for Euler's method are

$$v(0.03125) = 0 + (6.176667 - 13.055556(0)^2)0.03125 = 0.193021$$

$$v(0.0625) = 0.193021 + (6.176667 - 13.055556(0.193021)^2)0.03125 = 0.370841$$

The remaining steps can be computed in a similar fashion as tabulated and plotted below:

t	*v*	*dv/dt*
0	0.000000	6.176667
0.03125	0.193021	5.690255
0.0625	0.370841	4.381224
0.09375	0.507755	2.810753
0.125	0.595591	1.545494
0.15625	0.643887	0.763953
0.1875	0.667761	0.355136
0.21875	0.678859	0.160023
0.25	0.683860	0.071055



1.21 As noted in Sec. 1.4, a fundamental representation of the drag force, which assumes turbulent conditions (i.e., a high Reynolds number), can be formulated as

$$F_D = -\frac{1}{2}\rho A C_d v|v|$$

where F_d = the drag force (N), ρ = fluid density (kg/m^3), A = the frontal area of the object on a plane perpendicular to the direction of motion (m^2), υ = velocity (m/s), and C_d = a dimensionless drag coefficient.

(a) Write the pair of differential equations for velocity and position (see Prob. 1.19) to describe the vertical motion of a sphere with diameter, d (m), and a density of ρ_s (kg/m^3). The differential equation for velocity should be written as a function of the sphere's diameter.

(b) Use Euler's method with a step size of Δt = 2 s to compute the position and velocity of a sphere over the first 14 seconds. Employ the following parameters in your calculation: d = 120 cm, ρ = 1.3 kg/m^3, ρ_s = 2700 kg/m3, and C_d = 0.47. Assume that the sphere has the initial conditions: $x(0)$ = 100 m and $\upsilon(0)$ = −40 m/s.

(c) Develop a plot of your results (i.e., y and υ versus t) and use it to graphically estimate when the sphere would hit the ground.

(d) Compute the value for the bulk second-order drag coefficient, cd' (kg/m). Note that the bulk second-order drag coefficient is the term in the final differential equation for velocity that multiplies the term υ $|\upsilon|$.

==

(a) The force balance can be written as

$$m\frac{dv}{dt} = mg - \frac{1}{2}\rho v|v| A C_d$$

Dividing by mass gives

$$\frac{dv}{dt} = g - \frac{\rho A C_d}{2m} v|v| \qquad (1)$$

The mass of the sphere is $\rho_s V$ where V = volume (m^3). The projected area and volume of a sphere are $\pi d^2/4$ and $\pi d^3/6$, respectively. Substituting these relationships gives

$$\frac{dv}{dt} = g - \frac{3\rho C_d}{4d\rho_s} v|v|$$

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

(b) The first step for Euler's method is

$$\frac{dv}{dt} = 9.81 - \frac{3(1.3)0.47}{4(1.2)2700}(-40)|-40| = 10.0363$$

$$\frac{dx}{dt} = -40$$

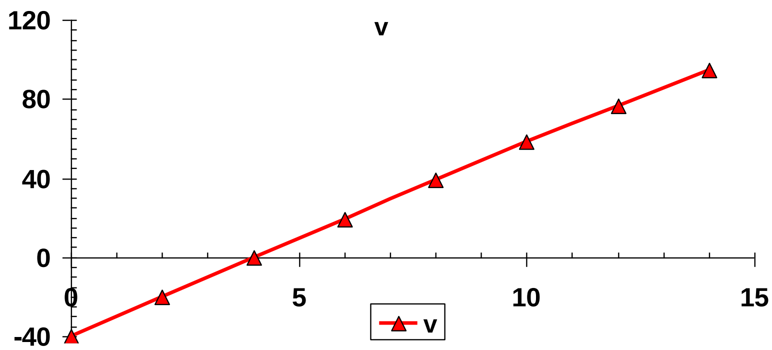
$$v = -40 + 10.0363(2) = -19.9274$$

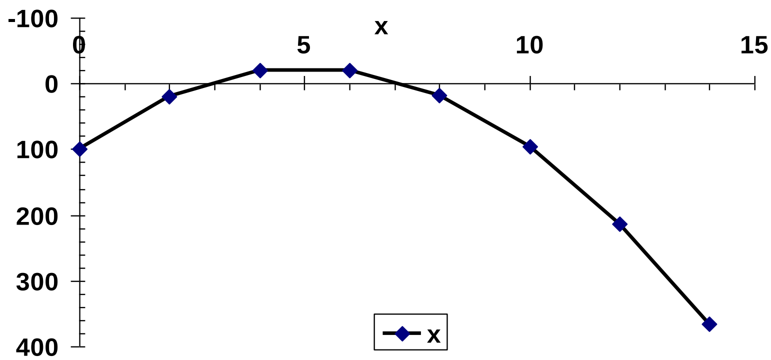
$$x = 100 - 40(2) = 20$$

The remaining steps are shown in the following table:

t	*x*	*v*	*dx/dt*	*dv/dt*
0	100.0000	-40.0000	-40.0000	10.0363
2	20.0000	-19.9274	-19.9274	9.8662
4	-19.8548	-0.1951	-0.1951	9.8100
6	-20.2450	19.4249	19.4249	9.7566
8	18.6049	38.9382	38.9382	9.5956
10	96.4813	58.1293	58.1293	9.3321
12	212.7399	76.7935	76.7935	8.9759
14	366.3269	94.7453	94.7453	8.5404

(c) The results can be graphed as (notice that we have reversed the axis for the distance, *x*, so that the negative elevations are upwards.





(d) Inspecting the differential equation for velocity (Eq. 1) indicates that the bulk drag coefficient is

$$c' = \frac{\rho A C_d}{2}$$

Therefore, for this case, because $A = \pi(1.2)^2/4 = 1.131\ \text{m}^2$, the bulk drag coefficient is

$$c' = \frac{1.3(1.131)0.47}{2} = 0.3455\frac{\text{kg}}{\text{m}}$$

1.22 As depicted in Fig. P1.22, a spherical particle settling through a quiescent fluid is subject to three forces: the downward force of gravity (F_G), and the upward forces of buoyancy (F_B) and drag (F_D). Both the gravity and buoyancy forces can be computed with Newton's second law with the latter equal to the weight of the displaced fluid. For laminar flow, the drag force can be computed with *Stoke's law*,

$$F_D = 3\pi\mu dv$$

where μ = the dynamic viscosity of the fluid (N s/m^2), d = the particle diameter (m), and v = the particle's settling velocity (m/s). The mass of the particle can be expressed as the product of the particle's volume and density, ρ_s (kg/m^3), and the mass of the displaced fluid can be computed as the product of the particle's volume and the fluid's density, ρ (kg/m^3). The volume of a sphere is $\pi d^3/6$. In addition, laminar flow corresponds to the case where the dimensionless Reynolds number, Re, is less than 1, where Re = $\rho dv/\mu$.

(a) Use a force balance for the particle to develop the differential equation for dv/dt as a function of d, ρ, ρ_s, and μ.

(b) At steady-state, use this equation to solve for the particle's terminal velocity.

(c) Employ the result of **(b)** to compute the particle's terminal velocity in m/s for a spherical silt particle settling in water: d = 10 µm, ρ = 1 g/cm3, ρ_s = 2.65 g/cm^3, and μ = 0.014 g/(cm·s).

(d) Check whether flow is laminar.

(e) Use Euler's method to compute the velocity from t = 0 to 2-15 s with Δt = 2-18 s given the initial condition: $v(0) = 0$.

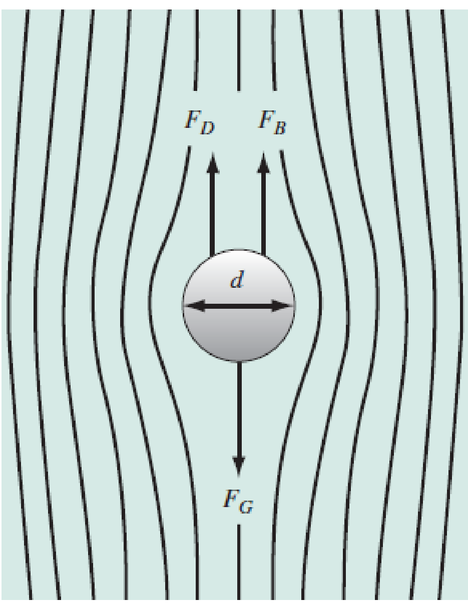


FIGURE P1.22

===

(a) A force balance on a sphere can be written as:

$$m\frac{dv}{dt} = F_{\text{gravity}} - F_{\text{buoyancy}} - F_{\text{drag}}$$

where

$F_{\text{gravity}} = mg$ $\qquad F_{\text{buoyancy}} = \rho V g$ $\qquad F_{\text{drag}} = 3\pi\mu d v$

Substituting the individual terms into the force balance yields

$$m\frac{dv}{dt} = mg - \rho V g - 3\pi\mu d v$$

Divide by m

$$\frac{dv}{dt} = g - \frac{\rho V g}{m} - \frac{3\pi\mu d v}{m}$$

Note that $m = \rho_s V$, so

$$\frac{dv}{dt} = g - \frac{\rho g}{\rho_s} - \frac{3\pi\mu d v}{\rho_s V}$$

The volume can be represented in terms of more fundamental quantities as $V = \pi d^3/6$. Substituting this relationship into the differential equation gives the final differential equation

$$\boxed{\frac{dv}{dt} = g\left(1 - \frac{\rho}{\rho_s}\right) - \frac{18\mu}{\rho_s d^2} v}$$

(b) At steady-state, the equation is

$$0 = g\left(1 - \frac{\rho}{\rho_s}\right) - \frac{18\mu}{\rho_s d^2} v$$

which can be solved for the terminal velocity

$$\boxed{v_\infty = \frac{g}{18}\frac{\rho_s - \rho}{\mu} d^2}$$

This equation is sometimes called *Stokes Settling Law*.

(c) Before computing the result, it is important to convert all the parameters into consistent units. For the present problem, the necessary conversions are

$$d = 10\ \mu\text{m} \times \frac{\text{m}}{10^6\ \mu\text{m}} = 10^{-5}\,\text{m} \qquad \rho = 1\frac{\text{g}}{\text{cm}^3} \times \frac{10^6\ \text{cm}^3}{\text{m}^3} \times \frac{\text{g}}{10^3\ \text{kg}} = 1000\frac{\text{kg}}{\text{m}^3}$$

$$\rho_s = 2.65\frac{\text{g}}{\text{cm}^3} \times \frac{10^6\ \text{cm}^3}{\text{m}^3} \times \frac{\text{g}}{10^3\ \text{kg}} = 2650\frac{\text{kg}}{\text{m}^3} \qquad \mu = 0.014\frac{\text{g}}{\text{cm s}} \times \frac{100\ \text{cm}}{\text{m}} \times \frac{\text{kg}}{1000\ \text{g}} = 0.0014\frac{\text{kg}}{\text{m s}}$$

The terminal velocity can then computed as

$$v_\infty = \frac{9.81}{18}\frac{2650 - 1000}{0.0014}(1\times10^{-5})^2 = 6.42321\times10^{-5}\,\frac{\text{m}}{\text{s}}$$

(d) The Reynolds number can be computed as

$$\text{Re} = \frac{\rho d v}{\mu} = \frac{1000(10^{-5})6.42321\times10^{-5}}{0.0014} = 0.0004588$$

This is far below 1, so the flow is very laminar.

(e) Before implementing Euler's method, the parameters can be substituted into the differential equation to give

$$\frac{dv}{dt} = 9.81\left(1 - \frac{1000}{2650}\right) - \frac{18(0.0014)}{2650(0.00001)^2}v = 6.108113 - 95{,}094v$$

The first two steps for Euler's method are

$$v(3.8147\times10^{-6}) = 0 + (6.108113 - 95{,}094(0))\times3.8147\times10^{-6} = 2.33006\times10^{-5}$$

$$v(7.6294\times10^{-6}) = 2.33006\times10^{-5} + (6.108113 - 95{,}094(2.33006\times10^{-5}))\times3.8147\times10^{-6} = 3.81488\times10^{-5}$$

The remaining steps can be computed in a similar fashion as tabulated and plotted below:

t	***v***	***dv/dt***	***t***	***v***	***dv/dt***
0	0	6.108113	2.29×10^{-5}	5.99×10^{-5}	0.409017
3.81×10^{-6}	2.33×10^{-5}	3.892358	2.67×10^{-5}	6.15×10^{-5}	0.260643
7.63×10^{-6}	3.81×10^{-5}	2.480381	3.05×10^{-5}	6.25×10^{-5}	0.166093
1.14×10^{-5}	4.76×10^{-5}	1.580608	3.43×10^{-5}	6.31×10^{-5}	0.105842
1.53×10^{-5}	5.36×10^{-5}	1.007233	3.81×10^{-5}	6.35×10^{-5}	0.067447
1.91×10^{-5}	5.75×10^{-5}	0.641853			

1.23 As depicted in Fig. P1.23, the downward deflection, y (m), of a cantilever beam with a uniform load, w = 10,000 kg/m, can be computed as

$$y = \frac{w}{24EI}\left(x^4 - 4Lx^3 + 6L^2x^2\right)$$

where x = distance (m), E = the modulus of elasticity = 2×10^{11} Pa, I = moment of inertia = 3.25×10^{-4} m4, and L = length = 4 m. This equation can be differentiated to yield the slope of the downward deflection as a function of x

$$\frac{dy}{dx} = \frac{w}{24EI}\left(4x^3 - 12Lx^2 + 12L^2x\right)$$

If $y = 0$ at $x = 0$, use this equation with Euler's method ($\Delta x = 0.125$ m) to compute the deflection from x = 0 to L. Develop a plot of your results along with the analytical solution computed with the first equation.

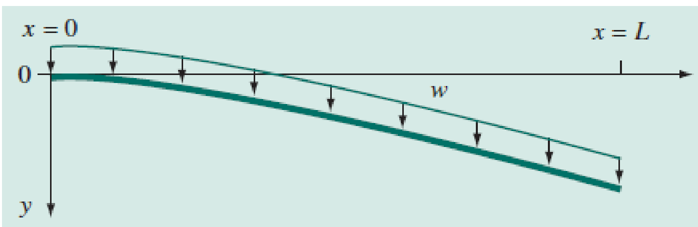


FIGURE P1.23
A cantilever beam.

==

Substituting the parameters into the differential equation gives

$$\frac{dy}{dx} = \frac{10000}{24(2\times10^{11})0.000325}\left(4x^3 - 12(4)x^2 + 12(4)^2x\right)$$
$$= 2.5641\times10^{-5}\left(x^3 - 12x^2 + 48x\right)$$

The first step of Euler's method is

$$\frac{dy}{dx} = 2.5641\times10^{-5}\left((0)^3 - 12(0)^2 + 48(0)\right) = 0$$
$$y(0.125) = 0 + 0(0.125) = 0$$

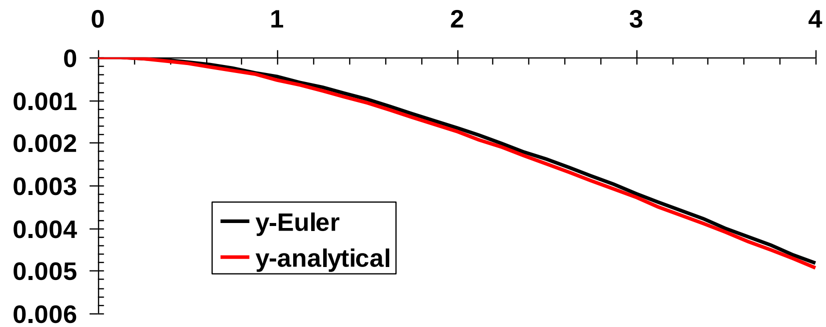
The second step is

$$\frac{dy}{dx} = 2.5641\times10^{-5}\left((0.125)^3 - 12(0.125)^2 + 48(0.125)\right) = 0.000149$$
$$y(0.25) = 0 + 0.000149(0.125) = 1.86361\times10^{-5}$$

The remainder of the calculations along with the analytical solution are summarized in the following table and plot. Note that the results of the numerical and analytical solutions are close.

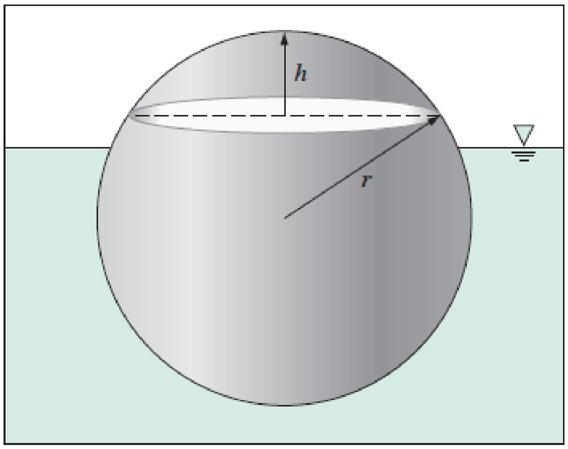
x	*y*-Euler	*dy/dx*	*y*-analytical	*x*	*y*-Euler	*dy/dx*	*y*-analytical
0	0	0	0	2.125	0.001832	0.001472	0.001925
0.125	0	0.000149	9.42E-06	2.25	0.002016	0.001504	0.002111

0.25	1.86E-05	0.000289	3.69E-05	2.375	0.002204	0.001531	0.002301
0.375	5.47E-05	0.00042	8.13E-05	2.5	0.002395	0.001554	0.002494
0.5	0.000107	0.000542	0.000141	2.625	0.00259	0.001574	0.00269
0.625	0.000175	0.000655	0.000216	2.75	0.002787	0.001591	0.002887
0.75	0.000257	0.000761	0.000305	2.875	0.002985	0.001605	0.003087
0.875	0.000352	0.000859	0.000406	3	0.003186	0.001615	0.003288
1	0.000459	0.000949	0.000519	3.125	0.003388	0.001624	0.003491
1.125	0.000578	0.001032	0.000643	3.25	0.003591	0.00163	0.003694
1.25	0.000707	0.001108	0.000777	3.375	0.003795	0.001635	0.003898
1.375	0.000845	0.001177	0.00092	3.5	0.003999	0.001638	0.004103
1.5	0.000992	0.00124	0.001071	3.625	0.004204	0.00164	0.004308
1.625	0.001147	0.001298	0.00123	3.75	0.004409	0.001641	0.004513
1.75	0.00131	0.001349	0.001395	3.875	0.004614	0.001641	0.004718
1.875	0.001478	0.001395	0.001567	4	0.004819	0.001641	0.004923
2	0.001653	0.001436	0.001744				



1.24 Use *Archimedes' principle* to develop a steady-state force balance for a spherical ball of ice floating in seawater (Fig. P1.24). The force balance should be expressed as a third-order polynomial (cubic) in terms of height of the cap above the water line (h), and the seawater's density (ρ_f), the ball's density (ρ_s) and radius (r).

FIGURE P1.24



==

[Note that students can easily get the underlying equations for this problem off the web]. The volume of a sphere can be calculated as

$$V_s = \frac{4}{3}\pi r^3$$

The portion of the sphere above water (the "cap") can be computed as

$$V_a = \frac{\pi h^2}{3}(3r - h)$$

Therefore, the volume below water is

$$V_s = \frac{4}{3}\pi r^3 - \frac{\pi h^2}{3}(3r - h)$$

Thus, the steady-state force balance can be written as

$$\rho_s g \frac{4}{3}\pi r^3 - \rho_f g\left[\frac{4}{3}\pi r^3 - \frac{\pi h^2}{3}(3r - h)\right] = 0$$

Cancelling common terms gives

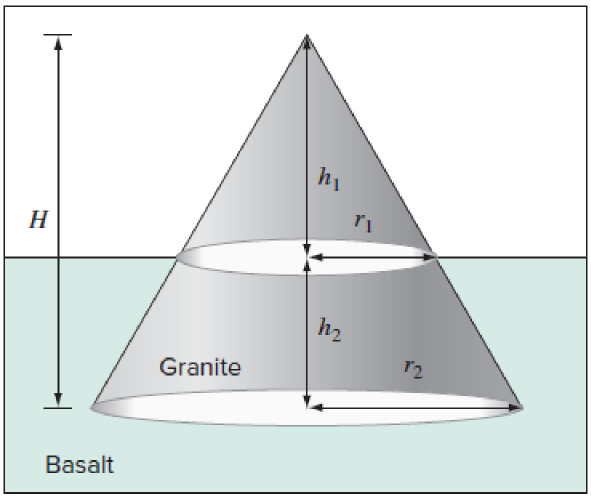
$$\rho_s \frac{4}{3} r^3 - \rho_f\left[\frac{4}{3} r^3 - \frac{h^2}{3}(3r - h)\right] = 0$$

Collecting terms yields

$$\frac{\rho_f}{3}h^3 - r\rho_f h^2 - (\rho_s - \rho_f)\frac{4}{3}r^3 = 0$$

1.25 Beyond fluids, *Archimedes' principle* has proven useful in geology when applied to solids on the earth's crust. Figure P1.25 depicts one such case where a lighter conical granite mountain "floats on" a denser basalt layer at the earth's surface. Note that the part of the cone below the surface is formally referred to as a *frustum*. Develop a steady-state force balance for this case in terms of the following parameters: basalt's density (ρ_b), granite's density (ρ_g), the cone's bottom radius (r), and the height above (h_1) and below (h_2) the earth's surface.

FIGURE P1.25



==

[Note that students can easily get the underlying equations for this problem off the web]. The total volume of a right circular cone can be calculated as

$$V_t = \frac{1}{3}\pi r_2^2 H$$

The volume of the frustum below the earth's surface can be computed as

$$V_b = \frac{\pi(H - h_1)}{3}\left(r_1^2 + r_2^2 + r_1 r_2\right)$$

Archimedes' principle says that, at steady state, the downward force of the whole cone must be balanced by the upward buoyancy force of the below ground frustum,

$$\frac{1}{3}\pi r_2^2 H g \rho_g = \frac{\pi(H - h_1)}{3}\left(r_1^2 + r_2^2 + r_1 r_2\right) g \rho_b \qquad (1)$$

Before proceeding we have too many unknowns: r_1 and h_1. So before solving, we must eliminate r_1 by recognizing that using similar triangles ($r_1/h_1 = r_2/H$)

$$r_1 = \frac{r_2}{H} h_1$$

which can be substituted into Eq. (1) (and cancelling the g's)

$$\frac{1}{3}\pi r_2^2 H\rho_g = \frac{\pi(H-h_1)}{3}\left(\left(\frac{r_2}{H}h_1\right)^2 + r_2^2 + \frac{r_2^2}{H}h_1\right)\rho_b$$

Therefore, the equation now has only 1 unknown: h_1, and the steady-state force balance can be written as

$$\frac{1}{3}\pi r_2^2 H\rho_g - \frac{\pi(H-h_1)}{3}\left(\left(\frac{r_2}{H}h_1\right)^2 + r_2^2 + \frac{r_2^2}{H}h_1\right)\rho_b = 0$$

Cancelling common terms gives

$$\frac{1}{3}r_2^2 H\rho_g - \frac{(H-h_1)}{3}\left(\left(\frac{r_2}{H}h_1\right)^2 + r_2^2 + \frac{r_2^2}{H}h_1\right)\rho_b = 0$$

and collecting terms yields

$$\frac{1}{3}r_2^2\left(H\rho_g - \frac{(H-h_1)}{3}\rho_b\right) - \frac{(H-h_1)}{3}\left(\frac{r_2^2}{H}h_1\right)\left(\frac{h_1}{H}+1\right)\rho_b = 0$$

1.26 As depicted in Fig. P1.26, an *RLC circuit* consists of three elements: a resistor (R), an inductor (L), and a capacitor (C). The flow of current across each element induces a voltage drop. Kirchhoff's second voltage law states that the algebraic sum of these voltage drops around a closed circuit is zero,

$$iR + L\frac{di}{dt} + \frac{q}{C} = 0$$

where i = current, R = resistance, L = inductance, t = time, q = charge, and C = capacitance. In addition, the current is related to charge as in

$$\frac{dq}{dt} = i$$

(a) If the initial values are $i(0) = 0$ and $q(0) = 1$ C, use Euler's method to solve this pair of differential equations from $t = 0$ to 0.1 s using a step size of $\Delta t = 0.01$ s. Employ the following parameters for your calculation: $R = 200\ \Omega$, $L = 5$ H, and $C = 10^{-4}$ F.

(b) Develop a plot of i and q versus t.

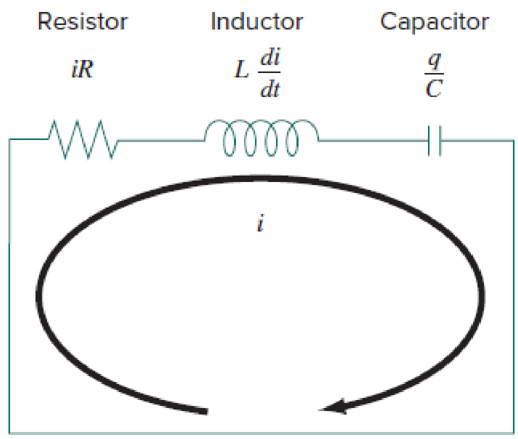


FIGURE P1.26

==

(a) The pair of differential equations to be solved are

$$\frac{di}{dt} = -\frac{R}{L}i - \frac{q}{LC}$$

$$\frac{dq}{dt} = i$$

At $t = 0$,

$$\frac{di}{dt} = -\frac{200}{5}i - \frac{1}{5(10^{-4})} = -40i - 2000 = -2000$$

$$\frac{dq}{dt} = i = 0$$

$i = 0 - 2000(0.01) = -20$

$q = 1 + 0(0.01) = 1$

At $t = 0.01$

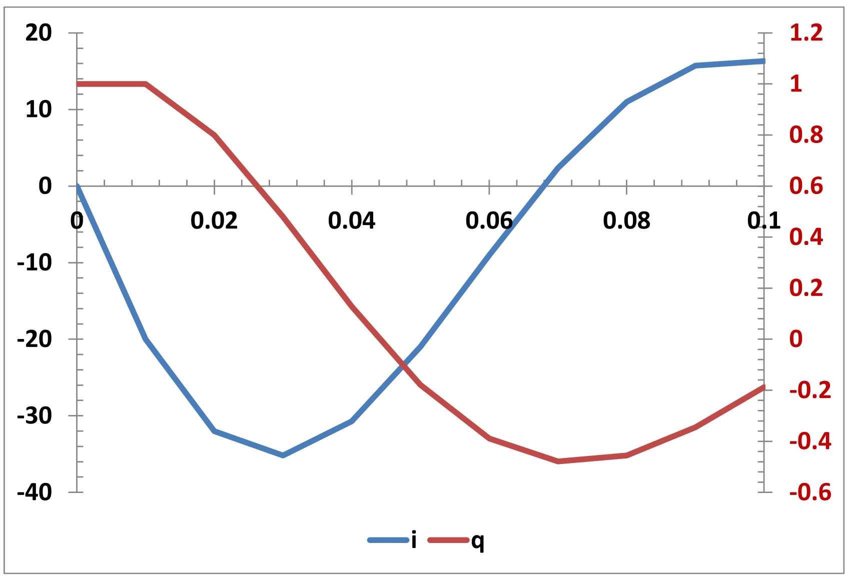
$i = -20 + (-40(-20) - 2000)0.01 = -20 - 1200(0.01) = -32$

$q = 1 - 20(0.01) = 0.8$

The calculation can be continued to give

t	*i*	*q*	*di/dt*	*dq/dt*
0	0	1	-2000	0
0.01	-20	1	-1200	-20
0.02	-32	0.8	-320	-32
0.03	-35.2	0.48	448	-35.2
0.04	-30.72	0.128	972.8	-30.72
0.05	-20.992	-0.1792	1198.08	-20.992
0.06	-9.0112	-0.38912	1138.688	-9.0112
0.07	2.37568	-0.47923	863.4328	2.37568
0.08	11.01001	-0.45547	470.5396	11.01005
0.09	15.71553	-0.34537	62.1236	15.71541
0.1	16.33665	-0.18822	-277.026	16.33665

(b)



1.27 Suppose that a parachutist with linear drag (m = 70 kg, c = 12.5 kg/s) jumps from an airplane flying at an altitude of 200 m with a horizontal velocity of 180 m/s relative to the ground.

(a) Write a system of four differential equations for x, y, $\upsilon x = dx/dt$ and $\upsilon y = dy/dt$.
(b) If the initial horizontal position is defined as $x = 0$, use Euler's methods with $\Delta t = 1$ s to compute the jumper's position over the first 10 seconds.
(c) Develop plots of y versus t and y versus x. Use the plot to graphically estimate when and where the jumper would hit the ground if the chute failed to open.
==

(a)

$$\frac{dv_x}{dt} = -\frac{c}{m}v_x \qquad \frac{dv_y}{dt} = g - \frac{c}{m}v_y \qquad \frac{dx}{dt} = v_x \qquad \frac{dy}{dt} = v_y$$

(b) Substituting the parameters

$$\frac{dv_x}{dt} = -0.178571 v_x \qquad \frac{dv_y}{dt} = 9.81 - 0.178571 v_y$$

$$\frac{dx}{dt} = v_x \qquad \frac{dy}{dt} = v_y$$

First step:

$$\frac{dv_x}{dt} = -0.178571(180) = -32.1429 \qquad \frac{dv_y}{dt} = 9.81 - 0.178571(0) = 9.81$$

$$\frac{dx}{dt} = 180 \qquad \frac{dy}{dt} = 0$$

$v_x = 180 - 32.1429(1) = 147.8571$

$v_y = 0 + 9.81(1) = 9.81$

$$x = 0 + 180(1) = 180$$

$$y = 0 + 0(1) = 0$$

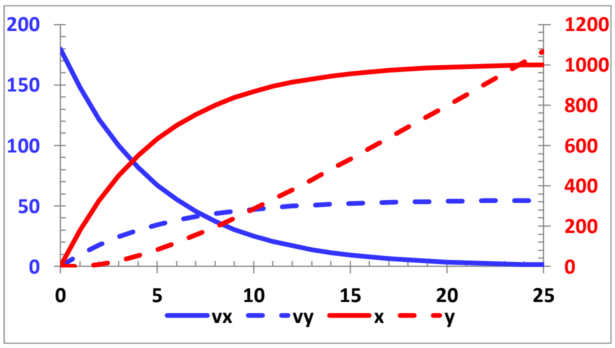
The calculation can be continued to give

t	v_x	v_y	x	y	dv_x/dt	dv_y/dt	dx/dt	dy/dt
0	180	0	0	0	-32.1429	9.81	180	0
1	147.8571	9.81	180	0	-26.4031	8.058214	147.8571	9.81
2	121.4541	17.86821	327.8571	9.81	-21.6882	6.619247	121.4541	17.86821
3	99.76585	24.48746	449.3112	27.67821	-17.8153	5.437239	99.76585	24.48746
4	81.95052	29.9247	549.0771	52.16568	-14.634	4.466303	81.95052	29.9247
5	67.3165	34.391	631.0276	82.09038	-12.0208	3.668749	67.3165	34.391
6	55.2957	38.05975	698.3441	116.4814	-9.87423	3.013615	55.2957	38.05975
7	45.42147	41.07337	753.6398	154.5411	-8.11098	2.47547	45.42147	41.07337
8	37.31049	43.54884	799.0613	195.6145	-6.66259	2.033422	37.31049	43.54884
9	30.6479	45.58226	836.3718	239.1633	-5.47284	1.670311	30.6479	45.58226
10	25.17506	47.25257	867.0197	284.7456	-4.49555	1.372041	25.17506	47.25257
11	20.67952	48.62461	892.1947	331.9982	-3.69277	1.127034	20.67952	48.62461
12	16.98674	49.75165	912.8742	380.6228	-3.03335	0.925778	16.98674	49.75165

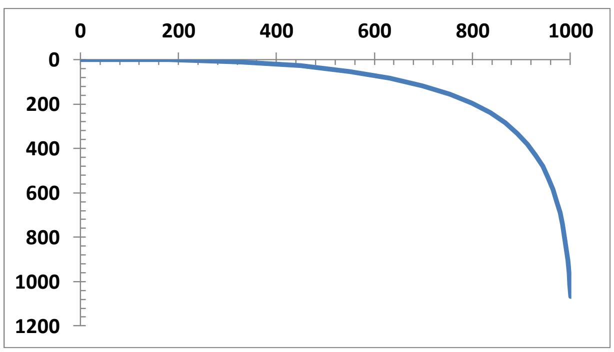
13	13.9534	50.67742	929.861	430.3744	-2.49168	0.76046	13.9534	50.67742
14	11.46172	51.43788	943.8144	481.0519	-2.04674	0.624664	11.46172	51.43788
15	9.414984	52.06255	955.2761	532.4897	-1.68125	0.513117	9.414984	52.06255
16	7.733737	52.57566	964.6911	584.5523	-1.38102	0.421489	7.733737	52.57566
17	6.352712	52.99715	972.4248	637.1279	-1.13441	0.346223	6.352712	52.99715
18	5.218299	53.34338	978.7775	690.1251	-0.93184	0.284397	5.218299	53.34338
19	4.28646	53.62777	983.9958	743.4685	-0.76544	0.233612	4.28646	53.62777
20	3.521021	53.86138	988.2823	797.0962	-0.62875	0.191896	3.521021	53.86138
21	2.892267	54.05328	991.8033	850.9576	-0.51648	0.157629	2.892267	54.05328
22	2.375791	54.21091	994.6956	905.0109	-0.42425	0.129481	2.375791	54.21091
23	1.951542	54.34039	997.0714	959.2218	-0.34849	0.106359	1.951542	54.34039
24	1.603053	54.44675	999.0229	1013.562	-0.28626	0.087366	1.603053	54.44675
25	1.316793	54.53411	1000.626	1068.009	-0.23514	0.071765	1.316793	54.53411

(c)

Plot of the four variables versus time



Plot of *y* versus *x*



Inspecting these figures and the numerical results indicates that the individual would hit the ground at a little over 24 seconds if the chute did not open.

1.28 Figure P1.28 shows the forces exerted on a hot air balloon system. Formulate the drag force as

$$F_D = \frac{1}{2}\rho_a v^2 A C_d$$

where ρ_a = air density (kg/m^3), υ = velocity (m/s), A = projected frontal area (m^2), and C_d = the dimensionless drag coefficient ($\cong$ 0.47 for a sphere). Note also that the total mass of the balloon consists of two components:

$$m = mG + mP$$

where mG = the mass of the gas inside the expanded balloon (kg), and mP = the mass of the payload (basket, passengers, and the unexpanded balloon = 265 kg). Assume that the ideal gas law holds ($P = \rho RT$), that the balloon is a perfect sphere with a diameter of 17.3 m, and that the heated air inside the envelope is at roughly the same pressure as the outside air.

Other necessary parameters are:

Normal atmospheric pressure, P = 101,300 Pa

The gas constant for dry air, R = 287 Joules/(kg K)

The air inside the balloon is heated to an average temperature, T = 100 ºC

The normal (ambient) air density, ρ = 1.2 kg/m^3.

(a) Use a force balance to develop the differential equation for $d\upsilon/dt$ as a function of the model's fundamental parameters.
(b) At steady-state, calculate the particle's terminal velocity.
(c) Use Euler's method and Excel to compute the velocity from t = 0 to 60 s with Δt = 2 s given the previous parameters along with the initial condition: $\upsilon(0)$ = 0. Develop a plot of your results.

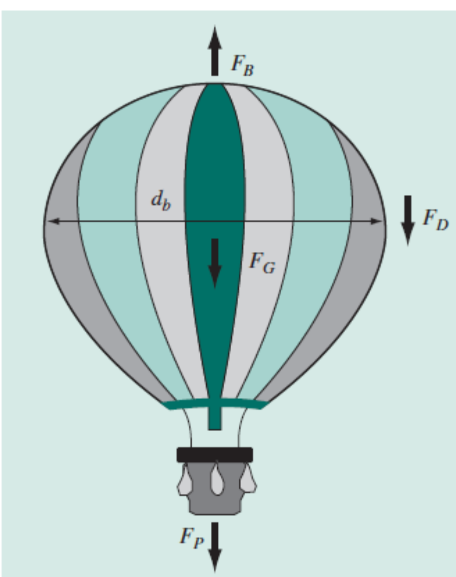


FIGURE P1.28
Forces on a hot air balloon: F_B = buoyancy, F_G = weight of gas, F_P = weight of payload (including the balloon envelope), and F_D = drag. Note that the direction of the drag is downward when the balloon is rising.

==

(a) $m = m_P + m_G = m_P + \frac{\pi d_b^3}{6}\frac{P}{RT_a}$

$$m\frac{dv}{dt} = F_B - F_G - F_P - F_D = V_b \rho_a g - V_b \rho_g g - m_p g - \frac{1}{2}\rho_a A_b C_d v^2 \quad (1)$$

$$m\frac{dv}{dt} = F_B - F_G - F_P - F_D = \frac{\pi d_b^3}{6}\rho_a g - \frac{\pi d_b^3}{6}\frac{P}{RT_a} g - m_p g - \frac{1}{2}\rho_a \frac{\pi d_b^2}{4} C_d v^2$$

(Note: Only the balloon's volume is used to calculate the buoyant force since it is much larger than the payload volume.)

$$\frac{dv}{dt} = \frac{\pi d_b^3}{6m}\rho_a g - \frac{\pi d_b^3}{6m}\frac{P}{RT_a} g - \frac{m_p}{m} g - \frac{1}{8}\rho_a \frac{\pi d_b^2}{m} C_d v^2$$

$$\frac{dv}{dt} = \frac{\pi d_b^3}{6\left(m_P + \frac{\pi d_b^3}{6}\frac{P}{RT_a}\right)}\rho_a g - \frac{\pi d_b^3}{6\left(m_P + \frac{\pi d_b^3}{6}\frac{P}{RT_a}\right)}\frac{P}{RT_a} g - \frac{m_p}{m} g - \frac{1}{8}\rho_a \frac{\pi d_b^2}{m} C_d v^2$$

$$\frac{dv}{dt} = \left[\frac{\pi d_b^3}{6\left(m_P + \frac{\pi d_b^3}{6}\frac{P}{RT_a}\right)}\rho_a - \frac{\pi d_b^3}{6\left(m_P + \frac{\pi d_b^3}{6}\frac{P}{RT_a}\right)}\frac{P}{RT_a} - \frac{m_p}{m}\right] g - \frac{1}{8}\rho_a \frac{\pi d_b^2}{m} C_d v^2$$

$$\frac{dv}{dt} = \left[\frac{\pi d_b^3}{6\left(m_P + \frac{\pi d_b^3}{6}\frac{P}{RT_a}\right)}\left[\rho_a - \frac{P}{RT_a}\right] - \frac{m_p}{m}\right] g - \frac{1}{8}\rho_a \frac{\pi d_b^2}{m_P + \frac{\pi d_b^3}{6}\frac{P}{RT_a}} C_d v^2$$

(b) Using Eq. (1) at steady state

$$0 = F_B - F_G - F_P - F_D = V_b \rho_a g - V_b \rho_g g - m_p g - \frac{1}{2}\rho_a A_b C_d v^2$$

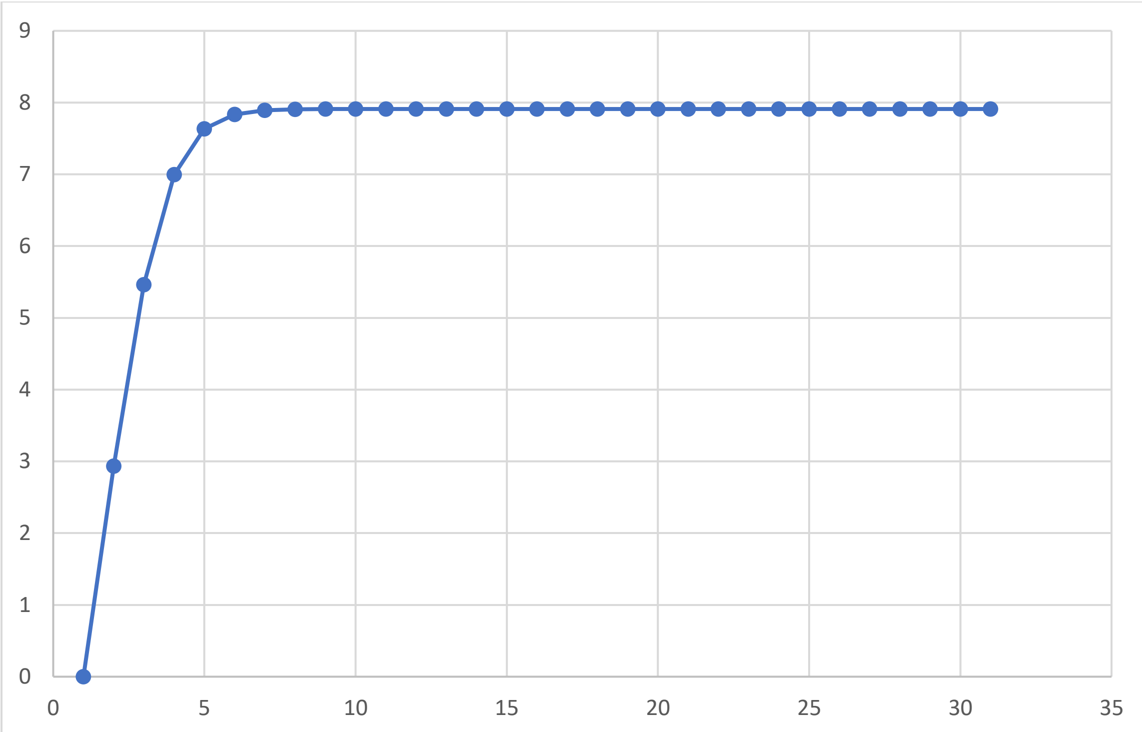
$$v = \sqrt{\frac{V_b \rho_a g - V_b \rho_g g - mg}{\frac{1}{2}\rho_a A_b C_d}} = \sqrt{\frac{2711(1.2)9.81 - 2711(0.9459)9.81 - 265(9.81)}{\frac{1}{2}(1.2)235(0.47)}} = \sqrt{\frac{4158.117}{66.27}} = 7.92118\,\frac{\text{m}}{\text{s}}$$

(c)

Cd	0.47			P	101300	Pa		Volume	2711
M_p	265	kg		R	287	J/kg.K		Area	235
d	17.3	m		T	373	K			
g	9.81			row_a	1.2	kg/m^3			
mg	2564.3349	N		row_g	0.9459	kg/m^3			
vterm	**7.92118**	m/s							
			Total mass	2829.3349					
FB	31914.44		FG	25166.61					
Fp	2599.65								

t	*v*	*dv/dt*
0	0	1.466133
2	2.932265	1.264689
4	5.461643	0.767267
6	6.996178	0.319383
8	7.634944	0.100422
10	7.835789	0.027624
12	7.891038	0.007268
14	7.905573	0.001888
16	7.909349	0.000489
18	7.910327	0.000127
20	7.91058	0.000033
22	7.910646	0.000008
24	7.910663	0.000002
26	7.910667	0.000001
28	7.910668	0
30	7.910669	0
.	.	.
.	.	.
.	.	.
56	7.910669	0
58	7.910669	0
60	7.910669	0

The values can be plotted as


9
8
7
6
5
4
3
2
1
0
0
5
10
15
20
25
30
35