Computational Physics – PH 354

Homework due on 24th Feb 2020

Exercise 1: Write a program to calculate and print the factorial of a number entered by the user. Write your program so that it calculates the factorial using *integer* variables, not floating-point ones. Use your program to calculate the factorial of 200.

Now modify your program to use floating-point variables instead and again calculate the factorial of 200. What do you find? Explain.

Exercise 2: Quadratic equations

a) Write a program that takes as input three numbers, a, b, and c, and prints out the two solutions to the quadratic equation $ax^2 + bx + c = 0$ using the standard formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

Use your program to compute the solutions of $0.001x^2 + 1000x + 0.001 = 0$.

b) There is another way to write the solutions to a quadratic equation. Multiplying top and bottom of the solution above by $-b \mp \sqrt{b^2 - 4ac}$, show that the solutions can also be written as

$$x = \frac{2c}{-b \mp \sqrt{b^2 - 4ac}}.$$

Add further lines to your program to print out these values in addition to the earlier ones and again use the program to solve $0.001x^2 + 1000x + 0.001 = 0$. What do you see? How do you explain it?

c) Using what you have learned, write a new program that calculates both roots of a quadratic equation accurately in all cases.

This is a good example of how computers don't always work the way you expect them to. If you simply apply the standard formula for the quadratic equation, the computer will sometimes get the wrong answer. In practice the method you have worked out here is the correct way to solve a quadratic equation on a computer, even though it's more complicated than the standard formula. If you were writing a program that involved solving many quadratic equations this method might be a good candidate for a user-defined function: you could put the details of the solution method inside a function to save yourself the trouble of going through it step by step every time you have a new equation to solve.

Exercise 3: Calculating derivatives

Suppose we have a function $f(x)$ and we want to calculate its derivative at a point x. We can do that with pencil and paper if we know the mathematical form of the function, or we can do it on the computer by making use of the definition of the derivative:

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \lim_{\delta \to 0} \frac{f(x+\delta) - f(x)}{\delta}.$$

On the computer we can't actually take the limit as δ goes to zero, but we can get a reasonable approximation just by making δ small.

a) Write a program that defines a function `f(x)` returning the value $x(x-1)$, then calculates the derivative of the function at the point $x = 1$ using the formula above with $\delta = 10^{-2}$. Calculate the true value of the same derivative analytically and compare with the answer your program gives. The two will not agree perfectly. Why not?

b) Repeat the calculation for $\delta = 10^{-4}$, 10^{-6}, 10^{-8}, 10^{-10}, 10^{-12}, and 10^{-14}. You should see that the accuracy of the calculation initially gets better as δ gets smaller, but then gets worse again. Why is this?

Exercise 4: Consider the equation $x = 1 - \mathrm{e}^{-cx}$, where c is a known parameter and x is unknown. This equation arises in a variety of situations, including the physics of contact processes, mathematical models of epidemics, and the theory of random graphs.

a) Write a program to solve this equation for x using the relaxation method (another name for fixed point iteration method) for the case $c = 2$. Calculate your solution to an accuracy of at least 10^{-6}.

b) Modify your program to calculate the solution for values of c from 0 to 3 in steps of 0.01 and make a plot of x as a function of c. You should see a clear transition from a regime in which $x = 0$ to a regime of nonzero x. This is another example of a phase transition. In physics this transition is known as the *percolation transition*; in epidemiology it is the *epidemic threshold*.

c) Write a program (or modify the previous one) to solve the same equation $x = 1 - \mathrm{e}^{-cx}$ for $c = 2$, again to an accuracy of 10^{-6}, but this time using fixed point iteration with acceleration. Have your program print out the answers it finds along with the number of iterations it took to find them.

Exercise 5: The biochemical process of *glycolysis*, the breakdown of glucose in the body to release energy, can be modeled by the equations

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -x + ay + x^2 y, \qquad \frac{\mathrm{d}y}{\mathrm{d}t} = b - ay - x^2 y.$$

Here x and y represent concentrations of two chemicals, ADP and F6P, and a and b are positive constants. One of the important features of nonlinear linear equations like these is their *stationary points*, meaning values of x and y at which the derivatives of both variables become zero simultaneously, so that the variables stop changing and become constant in time. Find the stationary points of these glycolysis equations

a) Demonstrate analytically that the solution of these equations is

$$x = b, \qquad y = \frac{b}{a + b^2}.$$

b) Show that the equations can be rearranged to read

$$x = y(a + x^2), \qquad y = \frac{b}{a + x^2}$$

and write a program to solve these for the stationary point using the relaxation method with $a = 1$ and $b = 2$. You should find that the method fails to converge to a solution in this case.

c) Find a different way to rearrange the equations such that when you apply the relaxation method again it now converges to a fixed point and gives a solution. Verify that the solution you get agrees with part (a).

Exercise 6: Wien's displacement constant

Planck's radiation law tells us that the intensity of radiation per unit area and per unit wavelength λ from a black body at temperature T is

$$I(\lambda) = \frac{2\pi hc^2\lambda^{-5}}{e^{hc/\lambda k_B T} - 1},$$

where h is Planck's constant, c is the speed of light, and k_B is Boltzmann's constant.

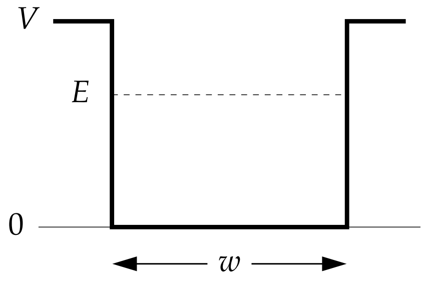
a) Write a program to calculate the wavelength λ at which the emitted radiation is strongest. Solve the resulting equation to an accuracy of $\epsilon = 10^{-6}$ using the bisection method, and hence find a value for the displacement constant. Note: The wavelength of maximum radiation obeys the *Wien displacement law*:

$$\lambda = \frac{b}{T},$$

where the so-called *Wien displacement constant* is b.

b) The displacement law is the basis for the method of *optical pyrometry*, a method for measuring the temperatures of objects by observing the color of the thermal radiation they emit. The method is commonly used to estimate the surface temperatures of astronomical bodies, such as the Sun. The wavelength peak in the Sun's emitted radiation falls at $\lambda = 502\,\text{nm}$. From the equations above and your value of the displacement constant, estimate the surface temperature of the Sun.

Exercise 7: Consider a square potential well of width w, with walls of height V:



Using Schrödinger's equation, setup the equations for the allowed energies E of a single quantum particle of mass m trapped in the well.

a) For an electron (mass 9.1094×10^{-31} kg) in a well with $V = 20\,\text{eV}$ and $w = 1\,\text{nm}$, write a Python program to calculate the values of the first six energy levels in electron volts to an accuracy of 0.001 eV using false position method.

Exercise 8: The roots of a polynomial

Consider the sixth-order polynomial

$$P(x) = 924x^6 - 2772x^5 + 3150x^4 - 1680x^3 + 420x^2 - 42x + 1.$$

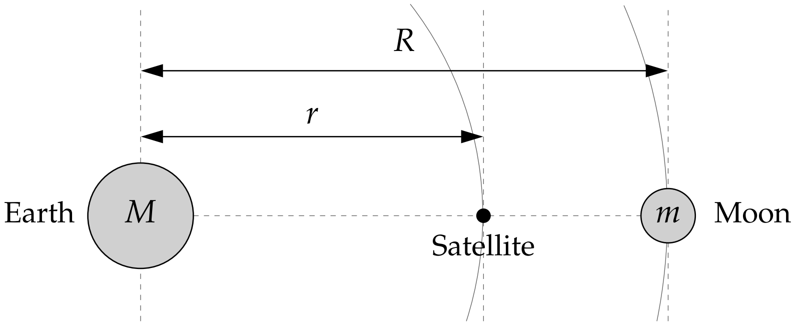
There is no general formula for the roots of a sixth-order polynomial, but one can find them easily enough using a computer.

a) Make a plot of $P(x)$ from $x = 0$ to $x = 1$ and by inspecting it find rough values for the six roots of the polynomial—the points at which the function is zero.

b) Write a Python program to solve for the positions of all six roots to at least ten decimal places of accuracy, using Newton's method.

Exercise 9: The Lagrange point

There is a magical point between the Earth and the Moon, called the L_1 Lagrange point, at which a satellite will orbit the Earth in perfect synchrony with the Moon, staying always in between the two. This works because the inward pull of the Earth and the outward pull of the Moon combine to create exactly the needed centripetal force that keeps the satellite in its orbit. Here's the setup:



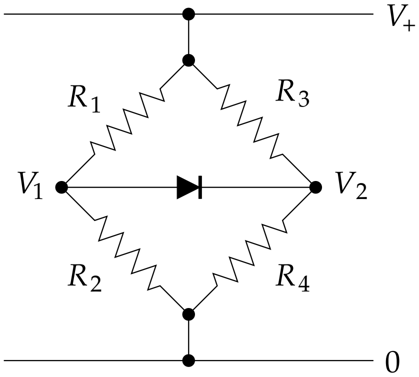
a) Assuming circular orbits, and assuming that the Earth is much more massive than either the Moon or the satellite, find the distance r from the center of the Earth to the L_1 point. Write a program that uses the secant method to solve for the distance r from the Earth to the L_1 point. Compute a solution accurate to at least four significant figures.

The values of the various parameters are:

$$\begin{aligned} G &= 6.674 \times 10^{-11}\,\mathrm{m^3kg^{-1}s^{-2}}, \\ M &= 5.974 \times 10^{24}\,\mathrm{kg}, \\ m &= 7.348 \times 10^{22}\,\mathrm{kg}, \\ R &= 3.844 \times 10^{8}\,\mathrm{m}, \\ \omega &= 2.662 \times 10^{-6}\,\mathrm{s^{-1}}. \end{aligned}$$

Exercise 10: Nonlinear circuits

Consider the following simple circuit, a variation on the classic Wheatstone bridge:



The resistors obey the normal Ohm law, but the diode obeys the diode equation:

$$I = I_0(\mathrm{e}^{V/V_T} - 1),$$

where V is the voltage across the diode and I_0 and V_T are constants.

a) Write a program to calculate the voltages V_1 and V_2 with the conditions

$$\begin{aligned} &V_+ = 5\,\mathrm{V}, \\ &R_1 = 1\,\mathrm{k\Omega}, \qquad R_2 = 4\,\mathrm{k\Omega}, \qquad R_3 = 3\,\mathrm{k\Omega}, \qquad R_4 = 2\,\mathrm{k\Omega}, \\ &I_0 = 3\,\mathrm{nA}, \qquad V_T = 0.05\,\mathrm{V}. \end{aligned}$$

You can use Newton's method to solve the equations.

b) The electronic engineer's rule of thumb for diodes is that the voltage across a (forward biased) diode is always about 0.6 volts. Confirm that your results agree with this rule.

Exercise 11:

Find the real and complex roots of the nonlinear equations discussed in class using Newton's method:

$$\begin{aligned} f(x,y) &= y^2(1-x) - x^3 = 0 \\ g(x,y) &= y^2 + x^2 - 1 = 0 \end{aligned}$$

Exercise 12: Gaussian elimination

Consider the matrix

$$B = 2 * \texttt{np.diag(np.full(200,1.0))} + \texttt{np.random.rand(200, 200)} / \sqrt{200}$$

This matrix (of size 200×200) has 2+$\frac{1}{\sqrt{200}} * \textit{random}$ on the diagonal and offdiagonal entries are $\frac{1}{\sqrt{200}} * \textit{random}$ where *random* is a random number between $(0,1)$. Construct a symmetric matrix $A = \frac{1}{2}(B + B^T)$. For

$$b = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Solve for x in $Ax = b$ using

a) LU decomposition (Gaussian elimination).

b) QR decomposition (Choose either modified Gram-Schmidt or Householder reflectors).

Exercise 13: Eigenvalues using in-built eig and QR

Consider the matrix:

$$\mathbf{A} = \begin{pmatrix} 1 & 4 & 8 & 4 \\ 4 & 2 & 3 & 7 \\ 8 & 3 & 6 & 9 \\ 4 & 7 & 9 & 2 \end{pmatrix}.$$

To find the eigenvalues of A, let $A_0 = A$. At the k-th step (starting with k = 0), we compute the QR decomposition $A_k = Q_k R_k$ where Q_k is an orthogonal matrix and R_k is an upper triangular matrix. We then form $A_{k+1} = R_k Q_k$. Note that

$$A_{k+1} = R_k Q_k = Q_k^{-1} Q_k R_k Q_k = Q_k^{-1} A_k Q_k = Q_k^{\mathsf{T}} A_k Q_k,$$

so all the A_k are similar and hence they have the same eigenvalues. This is not the most efficient algorithm to construct eigenvalues and not used widely but does give the correct result.

a) Write a complete program to calculate the eigenvalues and eigenvectors of the above real symmetric matrix using the QR algorithm. You can use the qr decomposition that you coded in the previous problem for this purpose. Continue the calculation until the magnitude of every off-diagonal element of the matrix is smaller than 10^{-6}. Test your program on the example matrix above.

b) Calculate the eigenvalues with built in eig function in numpy and compare the results with the above calculation.