PX1224: Computational Skills for Problem Solving
Assignment 2

1. Weather data [5 marks]

This problem makes use of historical weather data for Wales, obtained from the Met office website: **http://www.metoffice.gov.uk/climate/uk/datasets/#**

The downloaded data is available in the file wales_temp.txt. It has been modified slightly to remove the first year and the current year (as the data was incomplete for both of these).

a) *Read in the data*

*Read in the data file, **wales_temp.txt**, containing the mean temperature for Wales for the past 100 years as a two-dimensional array. Ensure that you skip header.* ***(not sure if it wants all 105 years of data or to cut off the first 5 years)***

Once you have read in the data into a two-dimensional array,

i. *extract a 1-dimensional array containing the years that the data was taken,*

ii. *extract a 1-dimensional array containing the annual average temperature and* ***[½]***

iii. *extract a 2-dimensional array containing the average monthly temperatures over the years.* ***[½]***

b) Investigate the data

Using the two dimensional array of average monthly temperatures,

i. Find the highest monthly average temperature

ii. Determine the number of months for which the average temperature was below zero? **[½]**

iii. Did the hottest October on record have a higher average temperature than the coolest August? What were the values? **[½]**

iv. Find the average temperature for each month and identify the warmest and coldest months **[½]** (**hint**: you can use the argument axis=0 in **mean()**, as described for **sum()** in the week 6 worksheet – no loops needed)

c) Temperature change over the years

Using the 1-dimensional arrays of years, and annual average temperature,

i. Determine the highest and lowest average yearly temperatures, and the years they occurred. **[½]**

ii. Make a histogram of the average yearly temperatures, using bins that are 0.5C wide. **[½] (not sure what 0.5C means – also modify fonts)**

iii. Repeat the above (plotted on the same histogram with the same bins) with only the data from 1990 onward. Does this show any evidence that the average temperature is increasing? **[½]**

iv. Make a plot of the average temperature vs. year. Fit a straight line through the data and calculate the slope, intercept and errors in both. Plot the best fit line on the graph. **[½] (error is zero for slope which seems too small – also modify fonts)**

v. What is the average annual increase/decrease in temperature and its uncertainty. Does this give evidence that the temperature is increasing? **[½] Please submit a Python code that performs the analysis described above.**

It should:

1. Read in the data and extract the requested arrays
2. Produce an output text file that contains the answers to the questions above, given to an appropriate number of significant figures.
3. Make and automatically save two figures: the histogram of annual temperatures and the plot of average temperature vs. year, with the best-fit line added.

2. Accuracy of numerical integration methods [5 marks]

In class during week 8, you performed the integral of the polynomial $p(x) = 3x^5 - 4x^4 - 7x^3 + x^2 + 2x + 10$ from -1.5 to 2.5 using a number of methods (rectangular integration, trapezium rule, Simpson's rule). Here, we would like to examine in a bit more detail the accuracy of the various methods. Each of the numerical methods will converge to the correct answer, but the rate of convergence will differ between methods. Each numerical technique is expected to converge to the correct answer at a rate proportional to $\mathbf{h}^n$, where **h** is the width of the intervals used in the integration. The value of **n** will differ for the different methods. This is discussed in the ***Introduction to Numerical Techniques*** included with this assignment. In this assignment, we will estimate the value of **n** for the different methods.

a) Calculate the left rectangular, trapezium and Simpson's integrals of the polynomial using **4 intervals** and calculate the **absolute** value of the error in the numerical integration.**[1] (Note: "error" means "difference from theoretical answer")**
b) Repeat a) using 8, 16 and 32 intervals.**[1]**
c) Make a plot showing the log of the error against the log of the number of intervals for each numerical method. [1]
d) For each method, fit a straight line to the data, and calculate the slope of the line. There's no need to calculate the errors.[½]
e) Use the answers from part d) to estimate the value of **n** for each method and compare it to the expected value.**[½]**
f) Plot the best fit lines on the graph and use them to estimate the number of points required to get the answer correct to 1 part in 10^{10} with each method. **[1] (Note: "correct to 1 part in 10^{10}" means difference is 10^{-10} or less)**

Hint: For an equation of the form, $y=Ax^n$, a graph of *log(x)* vs *log(y)* will have a gradient of ***n*** and an intercept of *log(A)*.

Please submit a Python code that performs the analysis described above.

It should:

1. Make and automatically save a plot of the error of the different numerical methods vs the number of steps, with the best fit lines added.
2. Produce an output text file that contains the answers to the questions above, given to an appropriate number of significant figures.

3. Newton's Law of cooling [5 marks]

Newton's law of cooling states that the rate of change of temperature of an object is proportional to the difference in temperature between the object and its surroundings. Mathematically, this can be written as

$$\frac{dT}{dt} = -k\left(T - T_s\right)$$

a) Write a code to use Euler's method to calculate the temperature of the object as a function of time. **[1]** (**Hint:** This is discussed in the ***Introduction to Numerical Techniques*** included with this assignment and can be implemented in a similar way as in Week 9.)

b) If you set $T_s=0,\ k=2$ and set the initial temperature to $T_0=1$, the equation becomes

$$\frac{dT}{dt} = -2T\,.$$

Evolve this from t=0 to 5 seconds with time steps of 0.01, 1/4 and 2 seconds. Make a plot showing these three evolutions.**[1]**

c) Suppose you are given a cup of hot tea, at temperature 90C and the room is at a temperature 20C. Take k=0.002s^{-1}. Use Euler's method to find out how long you have to wait until the tea has cooled sufficiently to be drinkable, say 60C?**[½]**

d) You plan to add 30ml of milk from the fridge (5C) to your 200ml of tea. When doing so, the temperature of the mixture will become $T_{mix} = \frac{T_{tea} V_{tea} + T_{milk} V_{milk}}{V_{tea} + V_{milk}}$

[Here, we have assumed that the specific heats of tea and milk are equal.]

If you add the milk immediately, how long will it take to reach 60C?**[½]**

e) Alternatively, you could let the tea cool and then add milk right before drinking. What temperature would it need to be before adding milk? How long would you have to wait before adding the milk?**[1]**

f) Make a plot showing the temperature against time for the three cases above: no milk, milk added immediately, milk added right before drinking.**[1] Please submit a Python code that performs the analysis described above.**

It should:

1. Produce an output text file that contains the answers to the questions above, given to an appropriate number of significant figures.
2. Make and automatically save two figures: the Euler evolution for part b and the plot of temperature vs time for the tea in part f.

4. Pendulum Motion [7½ marks]

In class you wrote a program to evolve the simple harmonic oscillator. Here we will solve for a pendulum, without assuming that the angle is small, and compare the results. Let's start with a simple pendulum that is neither damped nor driven. The equation of motion for a pendulum is:

$$\frac{d^2\vartheta}{dt^2}=\alpha=\frac{-g}{l}\sin\vartheta$$

a) Write an Euler-Cromer evolution code to calculate the angle ϑ and angular velocity ω of the pendulum at time t, given initial values ϑ_0 and ω_0. The total energy of the system is the kinetic energy plus gravitational potential energy. Write down an expression for the energy as a function of ϑ, ω, g, l and m. Add the energy calculation to your code. Use a time step in your code that keeps the energy constant to better than 1% accuracy.**[2]** **(Hint: this can be simulated in the same way as in Week 9. Instead of *x, v, a* we have ϑ, ω, and α)**

b) Often, the pendulum is approximated as a simple harmonic oscillator. To do this, we assume the angle is small, so that $\sin\vartheta \approx \vartheta$ and $\cos\vartheta \approx 1 - \vartheta^2/2$. Add to your existing program the code to evolve the pendulum using the small angle formula**[1]**

c) Take the mass of the pendulum to be $m=15kg$ with length $l=2.4m$, use $g=9.8m/s^2$. Take the initial angle initial angle of $\vartheta_0 = \pi/3$ radians and zero initial angular velocity, $\omega_0=0$. Evolve the system for 10 seconds, using both the pendulum and simple harmonic oscillator codes. Generate a figure with three subplots, the top showing angle ϑ vs time, the second angular velocity ω against time and the third energy vs time. **[1]**

d) Calculate the first time that the angular velocity is greater than zero, for both pendulum and harmonic oscillator. What fraction of a period does this correspond to? What is the period for the pendulum and the simple harmonic oscillator? How many oscillations before the approximation is wrong by 1 whole cycle? **[1]**

Damping and driving

We can add a damping term to the harmonic oscillator (a force that acts proportional to and opposite to the angular velocity). Also, we can add a driving term that pushes the pendulum with a maximum torque of $A \cdot l^2$ at an angular frequency ω_D. In this case, the equation of motion of the oscillator is

$$\frac{d^2\vartheta}{dt^2}=\frac{-g}{l}\sin\vartheta-\frac{C}{m}\frac{d\vartheta}{dt}+\frac{A}{m}\sin(\omega_D t)$$

e) Implement both the damping and driving terms in your code (there's no need to keep the harmonic oscillator approximation). **[1]**

f) Two children of mass $m= 15kg$ sit on swings with $l=2.4m$ and $C/m = 0.1s^{-2}$. One is pulled to $\vartheta_0= \pi/4$ rad and released from rest. The other starts with $\vartheta_0 = 0$, $\omega_0 = 0$ and is pushed with $A/m = 0.1^{s-1}$, $\omega_D=2\ rad/s$. Plot the motion of the two children (both position and velocity) for 60 seconds.**[1]**

g) Estimate by looking at the graph the graph:**[½]**

i. The time at which the second child starts to swing higher than the first.

ii. The maximum velocity (not angular velocity) of the second child.

Please submit a Python code that performs the analysis described above.

It should:

1. Produce an output text file that contains the answers to the questions above, given to an appropriate number of significant figures.
2. Make and automatically **save** two figures: a plot of the pendulum evolution vs time (showing angle, angular velocity and energy vs time) for part c; a plot of the angle and angular velocity vs time of the children on the swings for part f.