PHSX 815/615, CHEM 914 HW2

Due by Thur. Feb. 3 by 11 AM

Please complete by making the files available in your shared github repository CompPhysics-YourName in a new folder called HW2. Except for Task 1, which is a MWE, for using LaTeX, the other questions/problems should be addressed in a pdf file for your own LaTeX based report, together with appropriately documented programs. The programs can be in the programming language of your choice (C++, Fortran, python), but not (MATLAB, Mathematica).

Task 1
Goal: Get started with LaTeX. If you have not used it before, the easiest is likely to be to use Overleaf – you will need an account.

You should use the `MyLaTeXDocument.zip` file from the LaTeXTemplate github repository, `git clone https://github.com/grahamwwilson/LaTeXTemplate.git` that is accessible from the class web page. I demonstrated this in class on Thursday both from the command line, but also from Overleaf. Please remake the pdf file, changing the author to yourself, adding the date, and putting the updated document, `MyLateXDocument-YourName.pdf` in the HW2 folder.

Task 2 Report the operating system, and python, python2, and python3 versions that you have at hand. You could use the `pythoncheck.sh` code if appropriate.

Problem 1
Goal: Use Monte Carlo integration for a 1-d definite integral. Use the hit/miss rejection method with uniformly distributed random numbers to evaluate the following 1-d integral with a relative precision of at least 0.01% on the integral. Make sure you use a precise value of π.

$$I = \int_0^{2\pi} x^2 \cos^4 x \, \mathrm{d}x$$

Problem 2
Goal: Check Gaussian random number generation. Plots.
a) Generate 1,000,000 approximate standardized Gaussian random numbers (with zero mean and unit variance) using the approximate algorithm of summing 12 uniform random numbers with

$$x = \sum_{i=1}^{12} u_i, \text{where } u_i \sim \mathrm{Un}(-0.5, 0.5)$$

b) Use an exact algorithm for the above (eg. the `random.gauss` implementation in python) or if you need to roll your own, the Box-Muller method is a possibility, and compare b) with a). c) What is different? d) Why does a) work reasonably well?
Make sure you pay attention to the graphical presentation of the distributions.

Problem 3
Goal: learn more about probability distributions
a) Find some probability distribution Wikipedia that is new to you, and hopefully of interest, and write down its probability density function and a brief summary of what it is useful for.
b) The logistic distribution is similar to the normal distribution but with "heavier tails" (positive excess kurtosis). The standardized pdf (with $\mu = 0$, and $s = 1$) is

$$f(x) = \frac{e^{-x}}{(1 + e^{-x})^2} ,$$

and the standardized cdf is

$$F(x) = \frac{1}{2}[1 + \tanh(\frac{x}{2})] .$$

By generating standardized logistic random numbers, z, from uniform random numbers (on (0,1)), using the transformation method, you should be able to produce logistic random numbers, y, with mean μ, and scale parameter, s, using

$$y = \mu + sz$$

c) Plot these logistic random numbers and compare with the Gaussian random numbers from problem 2.

Problem 4
Electoral College Simulator.
Goals: use of Gaussian random numbers for simulations, useful plots/tables of results.
In this exercise we will use random numbers to simulate potential alternative outcomes of the 2020 presidential election. See the table in 2020 election. We'll use the actual vote percentages of the two major party candidates, and for every electoral college vote where the margin was less than 10%, we will sample from a normal distribution centered on the actual vote percentage, but with an rms of 1%.

So for example for Arizona (AZ), we sample the Biden vote percentage from a Gaussian centered on 49.36 with an rms of 1.0. If the random number says 47.36% (-2%) for Biden take this as implying 51.06% (+2%) for Trump. So 100% anti-correlated (neglect small third-party effects). We are simulating some of the underlying variability and chance associated with such votes - think of it maybe as the distribution of possible alternative weather scenarios on election day. The attached file, `EC-Results.txt` has a list with state names/electoral college votes that sums up to 538. The electoral college votes beyond the 10% margin are lumped into "BLUE" and "RED" aggregated states which will effectively have zero probability of influencing your estimates. You should histogram the probability distribution of the total number of electoral college votes for Biden that would have been obtained if these assumptions were correct.

How often is he still the winner (at least 270 EC votes)? What if the rms uncertainty was 0.2%? What if the rms uncertainty was 2%? What if the rms uncertainty was 0.01%? I am expecting Monte Carlo based estimates that should also have a statistical uncertainty reported. I assume you will do at least 10,000 repetitions of the experiments.

Additional studies could look for example at the effect of a particular bias, or a different assumption on the underlying distribution (for example a logistic distribution instead of a Gaussian). What if there is some systematic bias in the reported Biden percentage where 0.1% of his votes should have been allocated to Trump in all states?