Computational Physics 301 (2018-19)

Exercise 3: Random Numbers and Monte Carlo
Submission deadline: Friday 5th April 2019, 17:00

1 Introduction

This exercise addresses the use of "random" numbers in Monte Carlo techniques. These are often the fastest or most straightforward way of tackling complicated problems in computational analysis.

Monte Carlo techniques rely on the availability of reliable number generators. These are not truly random, but calculate a pseudo-random number based on their internal state. The state is changed with every call, so the next call will produce a different number. A good algorithm should produce a sequence of numbers that are close to random by any statistical test, and does not repeat until the state (typically an array of integers) returns to its starting value. Historically, some frequently used random number generators produced sequences that were rather short and easily predicted. Modern random number algorithms, such as *Mersenne Twister*, used within `numpy.random`, are better tested and avoid problem seen with earlier generators.

Note that the default initial state of a generator will be the same every time, resulting in the same random number sequence every time a program is run. If required, a different initial state can be made by providing a "seed".

2 Random Numbers in a Distribution

Generators normally provide numbers with a uniform distribution in a given range, so a common problem is to convert random numbers generated in a uniform distribution, $P(x)$ between x_0 and x_1 say, into a non-uniform one, $P'(x')$ between x'_0 and x'_1. This is a coordinate transformation from x to x' and assuming both distributions are normalised it is most easily achieved by requiring the cumulative distribution to the generated values to be the same, i.e. the required value x'_{req} for a generated value x_{gen} is given by

$$\int_{x'_0}^{x'_{req}} P'(x')dx' = \int_{x_0}^{x_{gen}} P(x)dx \tag{1}$$

If $P(x)$ is a uniform distribution between 0 and 1 the right-hand integral evaluates to be just x_{gen}, so defining

$$Q(x'_{req}) = \int_{x'_0}^{x'_{req}} P'(x')dx'$$

equation 1 becomes

$$x_{gen} = Q(x'_{req})$$

or

$$x'_{req} = Q^{-1}(x_{gen}) \tag{2}$$

If the definite integral Q is difficult to evaluate and then invert analytically, a "reject-accept" technique can be used instead. This involves first generating a random x' within the required range and then throwing a second uniform random number y between 0 and y_{max}. The upper bound y_{max} can be any value equal to or larger than the maximum possible value of P'. If $y < P'(x')$ then the value for x' is accepted, otherwise it is rejected. Conceptually this is the same as the standard Monte Carlo integration technique of defining a rectangle that completely encloses the desired function, throwing random 2D points within the rectangle and accepting the fraction that lie under the curve.

Task: Write two routines to generate random angles $0 < \theta < \pi$ in a distribution proportional to $\sin\theta$. The first routine should use an analytical formula (equation 2) to convert from a uniformly distributed variable, whilst the second should use the reject-accept method. You should verify that both methods give the required distribution, and compare their performance.

3 Simulations

A very common use of Monte Carlo is in simulating experimental data. In simulations, an entire experiment can be reproduced data point by data point, with random numbers being used to model unknowable or changing "nuisance" parameters such as the experimental resolution or quantum variations.

Physics Problem: In a nuclear physics experiment, a beam of unstable nuclei is injected travelling in the z direction at 2000 m/s. The nuclei decay with a mean lifetime of 550 μs, producing a gamma ray. The nuclei are not polarised so the gamma rays are emitted isotropically. A detector array is set up perpendicular to the beam, 2 m from the injection point. The resolution of the detector array is 10 cm in x and 30 cm in y.

Write a simulation to calculate the distribution of gamma rays detected in the experiment. You should use random numbers to model each nuclear decay, including the position of the decay itself and the direction of the gamma produced (using one of the routines you wrote for the Task above, or otherwise). You should also model the detector resolution by applying a "smearing" of the calculated hit positions by adding extra Gaussian-distributed terms with the widths given.

4 Statistical Solutions

Reliable random number generators and plentiful computing have allowed new approaches to statistical problems including the determination of experimental errors or confidence limits. For example consider the standard way of quoting an experimental result as $x \pm \sigma$, which is understood to mean that the true value lies within the range $x - \sigma$ to $x + \sigma$ with a certain probability (normally 68%). This is, however, based on various approximations which may not be true, especially in the case of limited statistics and/or non-physical regions (and is arguably not even a well defined quantity from a statistical point of view).

A more correct approach (with many additional advantages such as straightforward inclusion of correlations and nuisance parameters) is to generate ensembles of "pseudo-experiments" (often called Toy Monte Carlo) to generate distributions of experimental measurements for different model parameters. It is then straightforward to see which sets of model parameters cover the observed result, i.e. which ensembles include the measured value within their distributions at the specified probability.

Physics Problem: A collider experiment is looking for a hypothesised new particle X by counting the number of events observed to meet certain criteria. The number of X particles expected to be produced is $\mathcal{L}\sigma$ where the integrated luminosity $\mathcal{L} = 12/\text{nb}$ and σ is the (unknown) production cross section. The number of background events predicted to satisfy the criteria is 5.7 ± 0.4 where the error represents uncertainty in the theoretical modelling. If the total number of candidate events observed is 5, calculate the limit on the cross section σ that can be set at the 95% confidence level.

You will need to generate pseudo-experiments for a range of different cross sections. For each pseudo-experiment, throw random variables to model the Gaussian uncertainty on the background prediction and the Poisson variation in the background and signal production. Ensure that the statistics are sufficient to measure the experimental distribution for each cross section, and in particular the fraction of the distribution that is greater than the measured value (the confidence level).

How could additional uncertainties (e.g. luminosity) be included?

5 Guidance notes on report and code

A few guidance notes on report and code:

- Submit your code as a single text file (change .py to .txt) and include comments at the top that provide instructions for your demonstrator to run your code.
- The report should be concise, **with a maximum of 4 pages** including Tables and Figures as appropriate.
- A very brief introduction to each problem will be sufficient, you do not need to restate the problem in full.
- Give a brief description of the method you used to solve the problem and focus your discussion on the results you have obtained and their interpretation, including possible suggestions for future improvement.

Should you encounter any problems with submitting your work, please contact Dr. Brooke (james.brooke@bristol.ac.uk) or ask a demonstrator.