Nick Yallop – ny16281

Exercise 3:

**Random Numbers in a Distribution**

Report:

***Problem 1***

***Random Variable Coordinate Transforms***

Given a (pseudo-)random number generator, such as the functions in the *random* module for python, one can transform the probability density function (pdf) from one distribution to another in a number of ways. As an example; using a uniform random variable to generate a variable with a pdf proportional to sin(x).  
The easiest way to find this transformation is to ensure that the cumulative distributions between any two intervals be the same, under a coordinate transform x->x’, i.e.

Returning to the example, P’(x’) would be a uniform distribution where P’(x’)=1/(x’max-x’0) and P(x) ∝ sin(x), so;

Where A is the normalisation constant, i.e. .

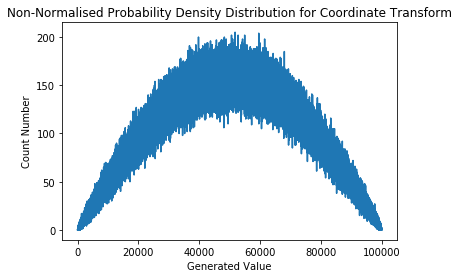
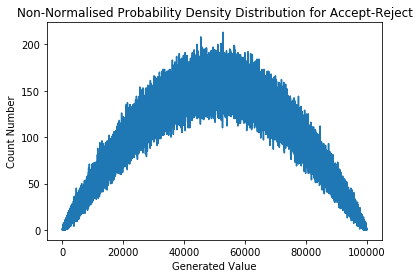
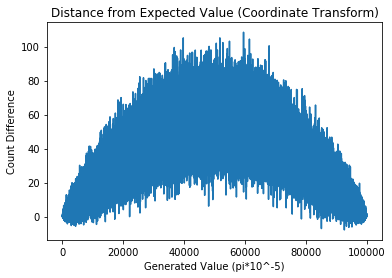
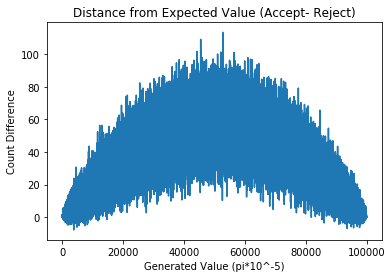
So, for a distribution proportional to sin(x) in the range [0 < x < ]; x0 = 0, xqen = x < and the above right-hand-side function becomes; , the inverse of which is; . The maxima is at f() = 1, so x’max = 1.

Thus, to transform from a uniform variable to the sin(x) distribution desired, one must call sin(x) = P(x) = f-1(f(P(x))) = f-1(Uniform(0,1)) = arcos(1- Uniform(0,1)).

However, the function that arises from this analysis is sometimes difficult/impossible to inverse explicitly so other methods are needed. One method that can be applied beyond the limits of the coordinate transform method is the *reject-accept­­* method. This method takes a uniformly randomly selected number from the domain of one’s desired variable (above it’s 0 < x < ), then generates a Uniform(0,1) value and if the resulting “dice roll” from 0 to 1 is less than the value of the pdf at x, which is P(x), the variable value is accepted. If the “dice roll” is greater than f(x) the method restarts, reselecting from the domain and re-rolling the dice until a result is accepted.

***Results***

To test these conclusions programmes were written that generated the P(x) = sin(x)/2 variable above, one using the coordinate transform, one using the reject-accept method. These variables were then generated and rounded to the nearest (), ten million times and graphed, as seen below:

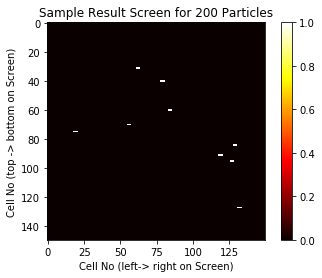
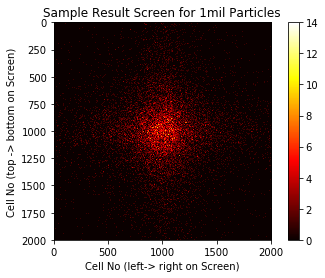
In each of these the greatest difference between the amount of times each number was generated was at most 110 (or slightly more than a 1/105 of the total events) away from the expected amount. Comparing the efficiency of these methods is difficult as the coordinate transform method requires a single, complex calculation (on the computer’s behalf) which is different depending on the desired distribution, while the reject-accept method requires many (and sometimes many, many) more simple calculations. However, the discrepancy between these two methods become negligible over small domains and the reject-accept time dominates over large domains. This is due to the values for the pdf being, on average, lower for random variables with larger domains so that the cpf still tends to 1 as the variable tends to its maxima. This results in more “rolls” on average, while the coordinate transform is a single calculation.

***Problem 2***

***Simulations***

Now that a couple of methods are derived, and their limits analysed, one can use these to construct the random distributions found/expected in physical problems. One such problem is the decay, in an isotropically random direction, of unstable nuclei being fired at a screen capable of producing a signal upon receiving a product of the particle’s radioactive decay, and what the screen would look like after a certain number of nuclei have been shot.

To keep the problem simple the mother nuclei are assumed to be travelling along identical paths at identical, constant, speed, v. This means that the distance they travel while alive is proportional to the time they survive. The pdf for decay chance at time t is P(t) = λexp[-λt], so the pdf for time of decay is P(x) = 1 - exp[-λt]. The cumulative distribution function (cdf) for this pdf is notoriously difficult to find the inverse for so the reject-accept method is the best method we’ve got. From here two uniform random variables can be used to generate a random direction, one Uniform(0,2), theta, and one Uniform(0,), phi. With the position and direction of the decay, trigonometry can be used to calculate if and where the daughter hits the screen. From here, representing the screen as a matrix, with each node representing a single sensor cell, allows one to construct a distribution of event density on the screen. An example distribution from a million mother particles on a 200mx200m screen, and 200 mother particles on a 15mx15m screen, can be seen below (v = 2e3ms-1, λ = (1/550)e-6 s-1, screen 20m away from particle creation point) (in both, over 60% of daughter particles missed):



The limitation of this method is that the screen *requires* dimensions as an input to the algorithm so some particles will almost always be missed as they fly past/away from the screen. However, this can be useful in simulating experiments one may go on to construct in reality – it simulates a real-life restriction. Similarly, the discrete nature of the screen matrix simulates the resolution of the real detector.

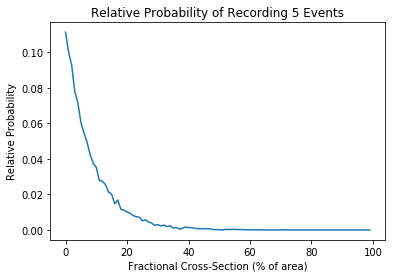
***Problem 3***

***Statistical Solutions***

The traditional way of representing the uncertainty, σ, on a value, x, is to print “x ± σ” to imply the true value lies inside the (x-σ , x+σ) interval with some confidence, nominally 68%. However, this representation expects the probability distribution inside the interval to take a form it very rarely does *and* to not include any non-physical values. A better way to approach this issue is to explore the area around the expected value using en-masse repeat simulations and, using the ability to reform pdf’s, explore the likelihood of a particular value producing the results seen, and then define the confidence interval as the range of values that make up a certain fraction (eg 0.68) of the area under the relative probability graph over that variable.

An example of this analysis is seen in finding the cross-section for a particle physics interaction. Given an integrated luminosity for the whole experiment, and a distribution for the number of background events that produce false positives, one can generate pdf’s of event number for different cross sections. Since the cross-section multiplied by the luminosity defines the probability density function of an event occurring at time t, integrating this value over the time of the whole experiment (taking the constant cross-section outside the integral) provides the distribution of event number probability. To simplify the problem we can assume the cross-section is distributed equally across the beam, so the “fractional cross-section” is more representative of the experiment (and doesn’t require beam dimensions to be defined). Fractional cross-section is unitless while integrated luminosity multiplied by fractional cross-section represents the event probability per unit area of beam. Modelling the luminosity as a sum of as many delta functions as there are total fired particles, each at random times inside the experiment time, results in the event number distribution being a poisson distribution with probability equal to the fractional cross-section.

To apply some numbers; for an integrated luminosity of 12/nb, a background event distribution of 5.7 ± 0.4 (in the classical, gaussian, sense), and a recorded number of events of 5, the limit for the 95% confidence interval needs to be found for the interaction cross-section. Thus, the relative probabilities of reading 5 events for a range of cross sections were found and graphed. Each value was then divided by the total number of “5” events over all simulations, resulting in a normalised probability distribution of cross-sections, given that data point. The integral from zero to some point, σ, was found such that the integral equalled 95%. To handle this a program was written that generated random numbers for the reaction events and background events from the given distributions, and these were added to find the total event number. Then, this experiment was simulated 50,000 times to generate the relative probabilities of “5 events”. These values were graphed against their corresponding cross-sections and the integral mentioned above was found. The resulting data can be seen below:



Using the above data, the 95% confidence interval for the fractional cross-section of the experiment in question is (0 , 0.25). Note, this integral is the first 2 decimal-place number for which the integral is greater than 95%. Thus, for a beam area of a nanobarn the limit on the 95% confidence interval for the cross-section is at 0.25nb.

To expand this model one *could* take beam dimensions into account. This would also expand the analysis of problem 2 as well. For example, one could assume that the particles are created inside the beam area with some distribution (most likely gaussian, mean at the beam centre) and travel with some diversion from the beam angle. From there one could construct new particle paths and decay points, then use the same trigonometry for problem 2. For problem 3, though, the relationship between beam area and interaction-cross section plays a large role in the event chances. For example, one could model the beam the same way as would improve problem 2, as well as modelling the target as an area the size of the cross-section then counting an interaction as the number of particles that contact that area. Thus, it’s possible for the beam to be smaller than the cross-section, leading to inaccurately many events, or for it to engulf the cross-section, leading to inaccurately few events. This was all solved by the introduction of fractional cross-section, uniformly distributed but this is highly unlikely in any physical experiment.