Computational Physics
Exercise 3 : Random Numbers and Monte Carlo methods

MERCIER Wilfried - University of Bristol

March 25, 2017

Abstract

We want to use Monte Carlo techniques to solve different problems. This implies the use of random numbers with non-uniform distributions. Therefore we start by developping two algorithms to generate non-uniform random numbers. We look at the algebrical inversion and rejection techique algorithms and we compare their performances. Then we use those algorithms to study the distribution of gamma rays produced by unstable nuclei in a nuclear physics experiment.

1 Generating random numbers with non uniform distributions

1.1 Direct Sampling

For later purposes we want to use sequences of random numbers not necessarily following a uniform distribution (i.e. probabilities are not the same for each value). If our variable can only take a few finite values we can imagine that some probabilities will be non zero. However if we are dealing with a continuous variable $x \in [a, b]$ we can see that any probability will go to zero as there are in principle[i] an infinite amount of possible values x can take. In this case we need to consider the cumulative distribution function F[1]

$$F(x) = \int_a^x g(x)dx \tag{1.1}$$

Where $g(x) \geq 0$ is the probability distribution function. F represents the probability of getting a value between a and x and because of normalization we have $F(b) - F(a) = 1$. For $x \in [a, b]$ such that $F(x)$ is defined as in Equation 1.1, the fundamental theorem of sampling applies[2]

$$\int_0^1 e^{iFt} g_F(F)dF = \int_a^b e^{iF(x)t} g(x)dx \tag{1.2}$$

An by using the fact that $dF = g(x)dx$ we see that $g_F(F) = 1$ for any $g(x)$. In other words, for any PDF, the cummulative distribution function F will always be uniform and the integral is therefore giving us a link between a uniform distribution and a non-uniform one.
This is of great importance since we want to generate random numbers following non-uniform distributions $g(x)$ using uniform random generators.

1.2 Algebrical inversion

The first method to generate random numbers following a non-uniform distribution from a uniform one is to start again from Equation 1.1. As previously shown the values of F will follow a uniform distribution. Therefore we can associate F to the values which are going to be returned by our uniform generator. Since x follows the distribution $g(x)$ we are interested in we want to find an expression relating x to F.
This can be done algebrically if F is invertible. Indeed if $F(x)$ is an invertible analytical function we have:

$$x(F) = F^{-1} \tag{1.3}$$

The values of x will follow the distribution $g(x)$ since this expression was derived from Equation 1.1. Hence if $F(x)$ can be expressed analytically from the integral we can find an expression of x as a function of F such that it follows the distribution $g(x)$.

1.3 Rejection Technique

The idea of rejection technique[3] is to consider a new function $f(x)$ such that $\forall x \in [a, b], \quad f(x) > g(x)$ such as, for instance, the function with constant value g_{max}.
Instead of choosing only one uniform random number, we choose two of them:

- x in the range $[a, b]$
- y in the range $[0, 1]$

These two values can be seen as the coordinates (x, y) of a point. If we plot the PDF we know it will always lie in the rectangle defined by $[a, b] \times [0, 1]$.
We can choose to cut this rectangle in vertical bins. If we do so, each point (x, y) randomly choosen will lie in one of these bins. More importantly if we repeat the process

[i] This is true in the mathematical sense but technically wrong since computers will always be precise up to a certain digit

many times, the points will fill the space under and above the curve defined by $g(x)$ in each bin. This implies that in the limit of the size of the bin is going to 0 the proportion of points under the curve will be equal to the value of g in the bin

$$\frac{N_{under}}{N_{above} + N_{under}} = g(x) \tag{1.4}$$

Hence by only keeping the values of x such that $y < g(x)$, we are forming a sequence of numbers following the distribution $g(x)$. However this will only be true for a large sequence of numbers.

1.4 Algebrical inversion for $\mathbf{g(x) = \sin(x)}$

We want to use the algebrical inversion and rejection techniques to generate random numbers following a probability distribution $\forall x \in [0, \pi], \quad g(x) = \sin(x)$ from a uniform generator. To do so, we will use the uniform random number generator from the GSL library[4].
The rejection technique can be used directly as described in section 1.3. However in order to use the algebrical inversion we need to perform some calculations first. From equation 1.1, using $g(x) = \sin(x)$ we have

$$F(x) = \int_0^x \sin(x)dx \tag{1.5}$$

Hence we see we have $x = \arccos(1 - F)$. Remarking that if F is our uniform random variable, $1 - F$ is also uniform, we see we can finally rewrite it as

$$x = \arccos G \tag{1.6}$$

Where $G = 1 - F$ is defined between -1 and +1 in order to retrieve our angles between 0 and π.

1.5 Numerical solutions and analysis

As shown in Figure 1, we can observe that the distribution in both cases seems to evolve towards a sine. The only thing we know from these histograms is that it seems that the distributions tend towards sine distrbutions as the number of throws[ii] increases. However these plots do no tell us anything about how close the distribution will get from a sine or how much it will fluctuate if we repeat the process many times.
A way to quantity the goodness of a distribution to a sine is to perform a χ^2[5] test between that distribution and its best fitted sine. The p-value will tell us the probability that the null hypothesis (i.e. the hypothesis that the distribution and the sine fit have no difference) might be rejected even if it is true. Hence low p-values will indicate good fits.
If we repeat this for many distributions where each of them has a different number of throws, we can see how the p-value will evolve as the number of throws increases. Moreover if for each number of throws we build more than one distribution, we can compute a mean and a deviation. Therefore we can quantify how close a distribution is from its best fitted sine and we can see how this evolves as the number of throws increases and how much the p-value fluctuates for different number of throws.
Thus, we choose to fit a sine using the curve fit routine from the SciPy library in Python[6] and we perform the χ^2 test using the chisquare routine from SciPy[7] as well. The result is shown in Figure 2 where the number of throws varies from 100 to 100,000. As can be seen the p-value fluctuates widely in both cases for throws fewer than 20,000. The p-value and its deviation go both quickly to 0 beyond this point. Even though the algebrical inversion method seems to converge more quickly than the rejection technique, both of them give distributions quite close to sine distributions past the 20,000 number of throws point. This is consistent with what we observe in Figure 1 where the shape of the distributions tends towards a sine as the number of throws increases. This is also consistent with the law of large numbers as the probabilistic effects are more visible for few throws and the statistical shape is only visible for many throws.
Finally it should be noted that the number of radom numbers following the given distribution is not the same in both cases. As can be seen the maximum value reached by the distribution in the algebrical inversion case is higher than the rejection technique. This is due to the fact that the algebrical inversion will give a new random number for each uniform random number generated when the rejection technique will only keep those with the correct y coodinate, reducing the total number of random numbers.

2 Distribution of gamma rays emitted by decaying nuclei

2.1 Description of the experiment

We consider a beam of unstable nuclei moving at constant velocity $v = 2000\text{m.s}^{-1}$ in the $\hat{\mathbf{k}}$ direction. If we have N_0 nuclei at $t = 0$s, the number of undecayed nuclei at time t is given by[8]

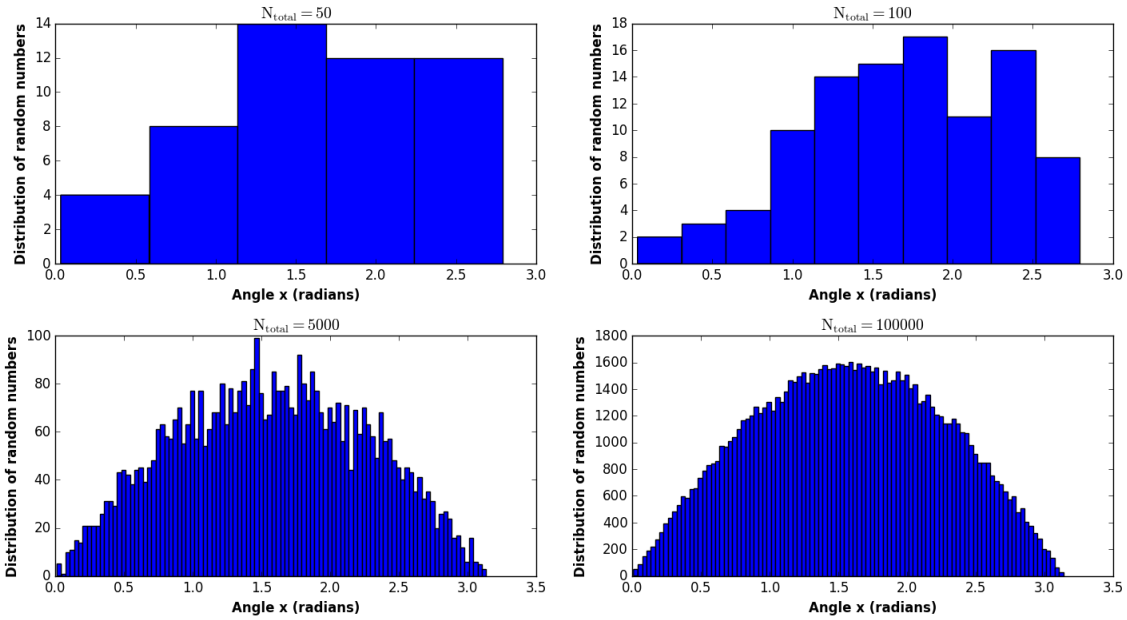
$$N(t) = N_0 e^{-t/\tau} \tag{2.1}$$

Where τ is the mean lifetime of the nuclei equal to 520μs in our case. When decaying, the nuclei emit gamma rays isotropically (i.e. no direction is preferred). A rectangular detector is placed perpendicular to the beam such that:

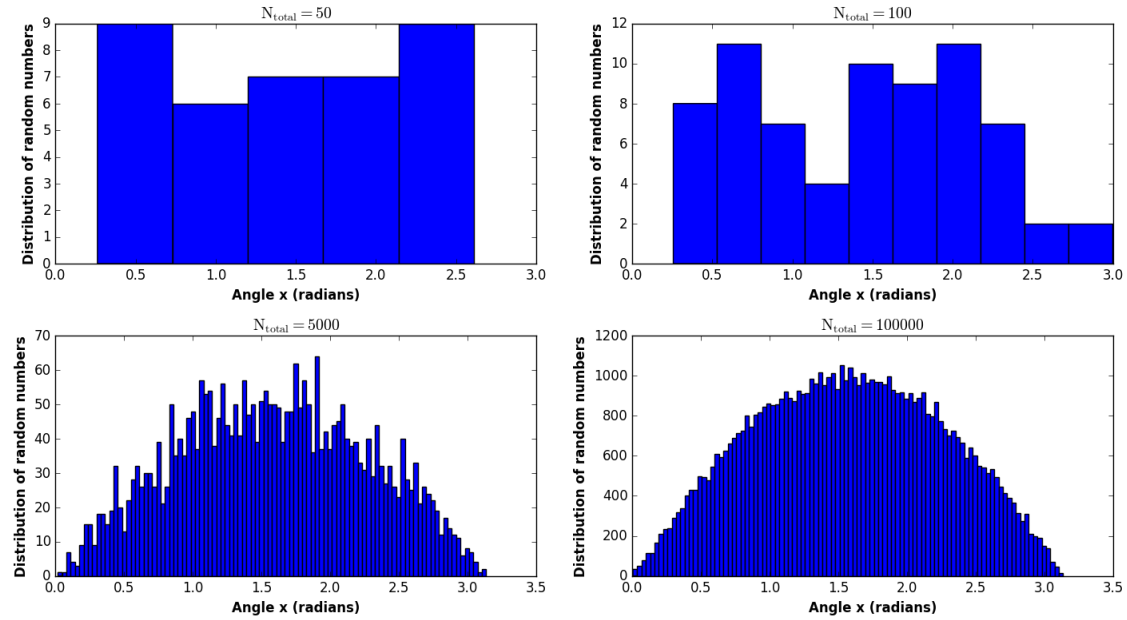
- the position of the beam in the x-y plane is $(0, 0)$
- at $t = 0$s, the distance between the beam and the detector is $d = 2$m
- $d \ll \Delta x, \Delta y \Rightarrow \Delta x, \Delta y \to \infty$ with $\Delta x, \Delta y$ the dimensions of the detector
- the resolution of the detector is 10cm in x and 30cm in y

We want to simulate the distribution of gamma rays detected using random numbers.

[ii] The term 'throw' will refer in the following to the process of generated random numbers



(a) Algebrical inversion



(b) Rejection technique

Figure 1: The two methods seem to gives similar results for the same number of throws. For few throws the distributions do not look like sine but as they increase they do seem to tend towards sine distributions. This intuition must be compared to the data plotted in Figure 2.

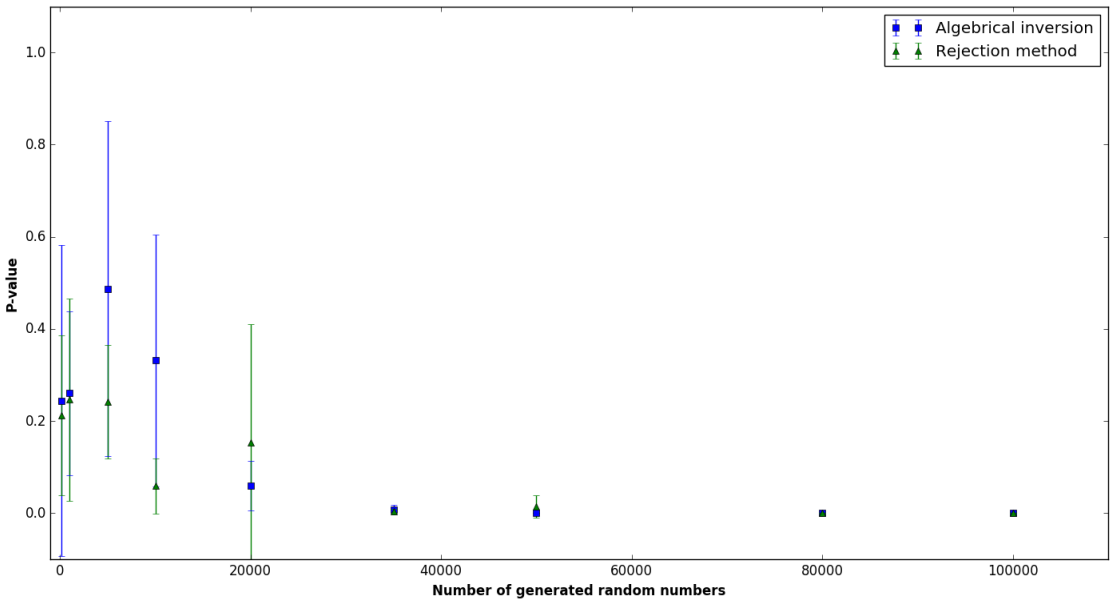


Figure 2: Change in the p-value from the χ^2 test between the distributions for throws of random numbers and their best fitted sine from least square fiiting. As expected the p-value varies widely for low frequencies (large deviation) indicating that there are not enough random numbers to form a proper $sin(x)$ distribution. As the number of throws increases both mean and deviation of the p-value deacrease towards 0.

2.2 Simulation of the experiment

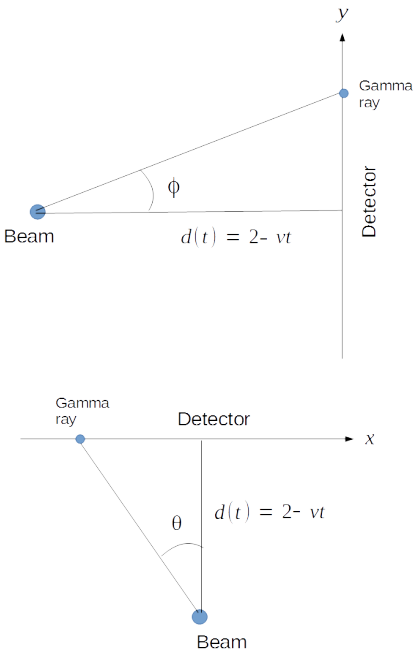


Figure 3: Parametrisation of the position of the detected gamma rays on the detector

Since the decay of a nucleus is a random process we can simulate it using random numbers. This can be done by randomly generating the time t when the decay happens for each nucleus. From Equation 2.1, we expect our random variable t to follow a distribution $g(t) = e^{-t/\tau}$ where t corresponds to the time of emission. Because the beam is moving at a speed $v = 2000\text{m.s}^{-1}$ and it is at a distance $d = 2\text{m}$ away from the screen at $t = 0\text{s}$, only the gamma rays emitted via nuclear decay between $t = 0\text{s}$ and $t = d/v$ will be detected.
Thus we can simulate the emission of gamma rays by generating uniform random numbers $t \in [0, d/v]$ and use the rejection technique to only keep those following the distribution $g(t)$ defined above.
We can then generate two uniform random numbers defining the position of the gamma rays on the detector. Because we can approximate the dimensions of the detector as infinite we could either generate random numbers between $+\infty$ and $-\infty$ or redefine the position in terms of two angles ϕ and θ such that

$$\tan\phi = \frac{y_\gamma}{d(t)}, \quad \tan\theta = \frac{x_\gamma}{d(t)} \tag{2.2}$$

This is represented in Figure 3. Since we know that the position of the decaying nucleus at any time t is given by $d = 2 - vt$, we can generate two uniform random numbers between $-\pi/2$ and $\pi/2$[iii] and convert them into x and y coordinates on the detector using Equation 2.2.

[iii] We do not need to consider gamma rays with at least one angle out of $[-\pi/2, \pi/2]$ since they will be moving backward and will never reach the detector

2.3 Discussion about error

Finally we need to take into account the fact that the detector has a certain resolution in x and y but also in the counting. Firstly, this implies that the position of the gamma ray cannot be know precisely and this will involve some error in the measurement
Any measurement of a variable n with some error σ can be represented by a Gaussian distribution[9]

$$g(n) = e^{-\frac{(n-n_0)^2}{2\sigma^2}} \tag{2.3}$$

In our case n_0 is the uniformly generated position in x and y and n is the corrected position after applying the error due to the accuracy of the detector σ where its values are given in subsection 2.1. In the general case, the corrected positions should be generated following this distribution with n varying between $-\infty$ and $+\infty$. But, because the detector is made of an array of small detectors with finite resolution σ_x and σ_y, we must not generate those corrected positions in $\mathbb{R}$. Instead they must be generated in $[n_0 - \sigma, n_0 + \sigma]$. Hence, by making the change of variable $n_{err} = n - n_0$ and by applying the error in x and y we get

$$\begin{aligned} \forall x_{err} \in [-0.05, 0.05], \quad g(x_{err}) = e^{-\frac{x_{err}^2}{0.02}} \\ \forall y_{err} \in [-0.15, 0.15], \quad g(y_{err}) = e^{-\frac{y_{err}^2}{0.18}} \end{aligned} \tag{2.4}$$

Where x_{err} and y_{err} are the errors to add up to the uniformly distributed positions x and y.
Secondly, there will be some error due to the counting. This type of error can be represented by Poisson noise[9]

$$\sigma_N = \sqrt{N} \tag{2.5}$$

With N the number of counts. Hence, before plotting the histogram representing the distribution of counts on the screen, we need to add up the Poisson noise for each bin using Equation 2.5.

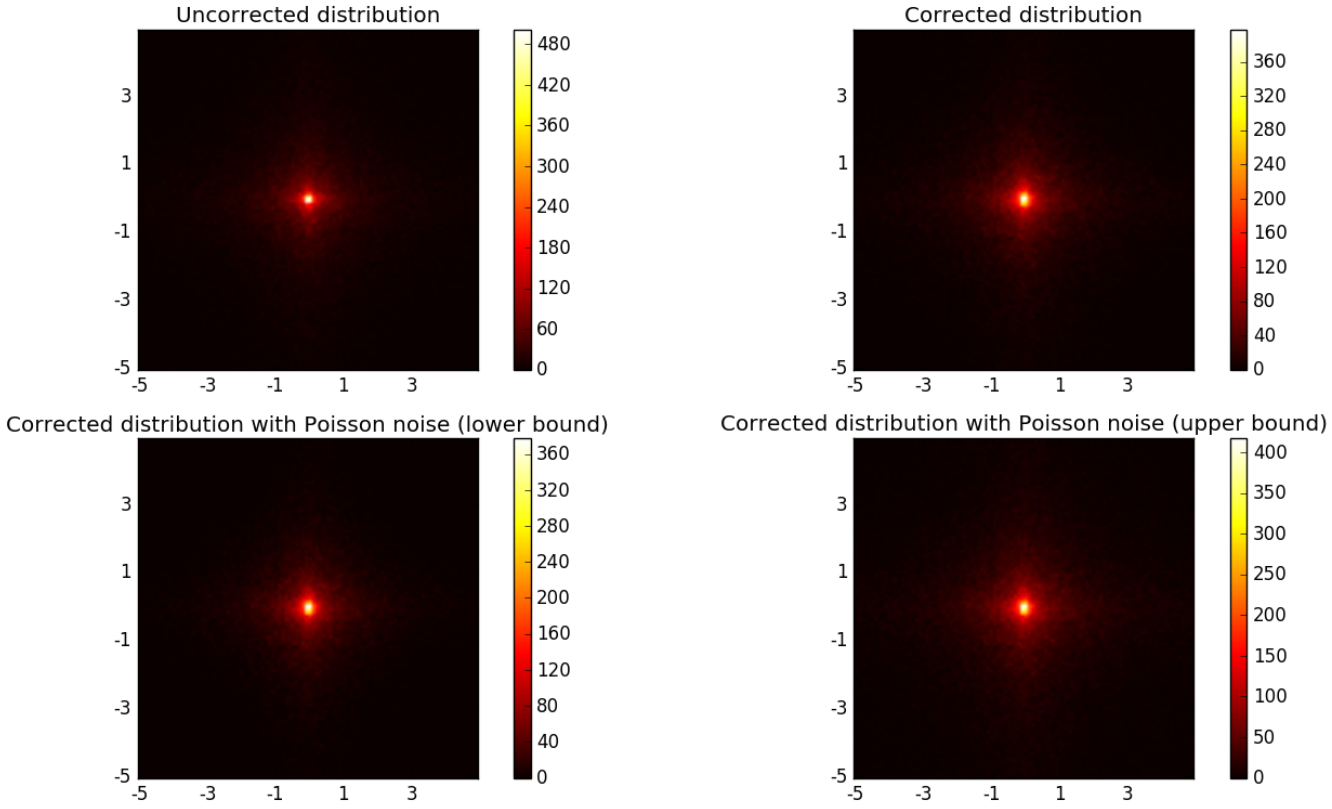


Figure 4: Histograms representing the distribution of detected gamma rays (position given in metres). The top left plot represents the position of the gamma rays on a detector with an infinite precision. The top right plot corresponds to the distribution with a random error due to the uncertainty in the measurement of the position of the gamma rays as described in subsection 2.2. The plots on the bottom show the lower and upper bounds when applying the Poisson noise described in Equation 2.5.

2.4 Analysis of the distribution

By looking at Figure 4, it can be seen that both the error from the resolution of the detector and Poisson noise do not widely affect the distribution. If we look at the two top histograms, we can see that the error that we induced because of the finite resolution of the dector spreads out the distribution in space. This is consistent with what we would expect since the positions are no longer defined with an infinite precision but with a certain broadening. Because Poisson noise is equal to the square root of the number of gamma rays detected in a certain pixel, the more gamma rays detected the greater the noise. By substracting or adding the noise to the corrected distribution we can define a lower and an upper bound. This is shown in the bottom plots of Figure 4, where the two histograms show us the extrema of the distribution. Since for each bin the number of gamma rays is offset by a factor $\pm\sqrt{N}$, only the bins with the largest number of rays will be greatly affected.

3 Statistical study of a particle physics experiment

3.1 Introduction of the experiment

We consider a collider experiment whose goal is to look for the existence of a new particle by counting the number of events satisfying some selection criteria. The number of events measured is equal to 5 and can be due to two things[10]:

- The particle itself whose number of events is noted ν_s
- The background which corresponds to everything else but the particle and whose number of events is noted ν_b

Hence the total number of events is $\nu = \nu_s + \nu_b$. From the theory we already know that we should have $\nu_b = 5.8 \pm 0.4$ where the uncertainty corresponds to the deviation in the distribution of the background signal.
The number of signal events ν_s is directly related to the cross-section σ by

$$\nu_s = L\sigma \tag{3.1}$$

With $L = 10\text{fb}^{-1}$ the integrated luminosity. We want to use a statistical model of the experiment to quantity the upper limit on the cross-section given the expected background signal and the number of events which were measured.

3.2 Numerical study

Because each process is defined quantum mechanically there is a fixed probability to measure an event. If we assume we know the 'real' total number of events ν, because each event is independent of the others we can define the probability to detect N events using a Poisson distribution:

$$g(N,\nu) = \frac{(\nu)^N}{N!}e^{-\nu} \tag{3.2}$$

If we consider the case where there is no new particle (i.e. $\nu_s = 0$) the distribution becomes

$$g = \frac{(\nu_b)^N}{N!}e^{-\nu_b} \tag{3.3}$$

We can generate such a distribution because we already know the mean background number of events ν_b. However we cannot directly compute the values since the number of background events is defined with a certain uncertainty. Instead we need to add during the generation of random numbers following this law a variation of ν_b such that it follows a Gaussian distribution centred in 5.8 with deviation 0.4.
Before even computing this distribution, from the central limit theorem we know Poisson distributions behave as Gaussians for a large number of throws. Intituively, we can see that the probability to have more than 5 events when the mean is 5.8 and the deviation is $\sqrt{5.8}$[iv] will be quite high. This indicates that it is totally possible to get 5 events even if there are no new particles (i.e. the cross section is 0).
On the other hand if the cross section is non zero, the mean value in the Poisson distribution will increase and therefore we should expect the probability to increase as well. If, for instance, we consider an infinite cross-section the probability to get at least 5 events shoud be equal to 1.
However, we need to know how confident we are in our cross-section. In other words, if we define the ratio $\xi = \frac{n(N>5)}{n_{total}}$ with $n(N > k)$ the number of generated numbers from our Poisson distribution such that there are at least k events, at which ratio ξ must we stop to be confident enough in our result ? We decide to choose a confidence ratio of 95%
Hence, we choose to simulate the experiment by throwing random numbers such that they follow a Poisson distribution with mean ν. We vary the signal ν_s starting from $\nu_s = 0$ and for each value we generate the appropriate Poisson distribution. We also generate the background signal ν_b such that if follows the Gaussian distribution described above. Once the distribution is complete we compute the ratio of numbers with 5 or more events over the total number of generated numbers. We keep increasing the signal ν_s until the ratio reaches 95% and once it has reached this value we stop the algorithm and compute the upper bound on the cross-section.

[iv] For a Poisson distribution of a variable x, if we call $\bar{x}$ the mean, the deviation is given by $\sigma = \sqrt{\bar{x}}$

References

[1] William R. Gibbs, 1994, Computation in Modern Physics, 29

[2] William R. Gibbs, 1994, Computation in Modern Physics, 30

[3] William R. Gibbs, 1994, Computation in Modern Physics, 38-39

[4] GNU Scientific Library (GSL), Random Number Generation, https://www.gnu.org/software/gsl/manual/html_node/Random-Number-Generation.html

[5] John B. Kennedy, Adam N. Neville, 1986, Basic Statistical Methods for Engineers and Scientists, 282-283

[6] SciPy, Optimize, Curve Fit, https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html

[7] SciPy, Stats, Chisquare, https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.chisquare.html

[8] Kenneth S. Krane, 1988, Introductory Nuclear Physics, 164

[9] Kenneth S. Krane, 1988, Introductory Nuclear Physics, 219

[10] C.Grab, M.Donega, Institute for Particle Physics, ETH Zurich, http://ihp-lx.ethz.ch/Stamet/lectureNotes/PDFs/ch8.pdf