Lab10 Report

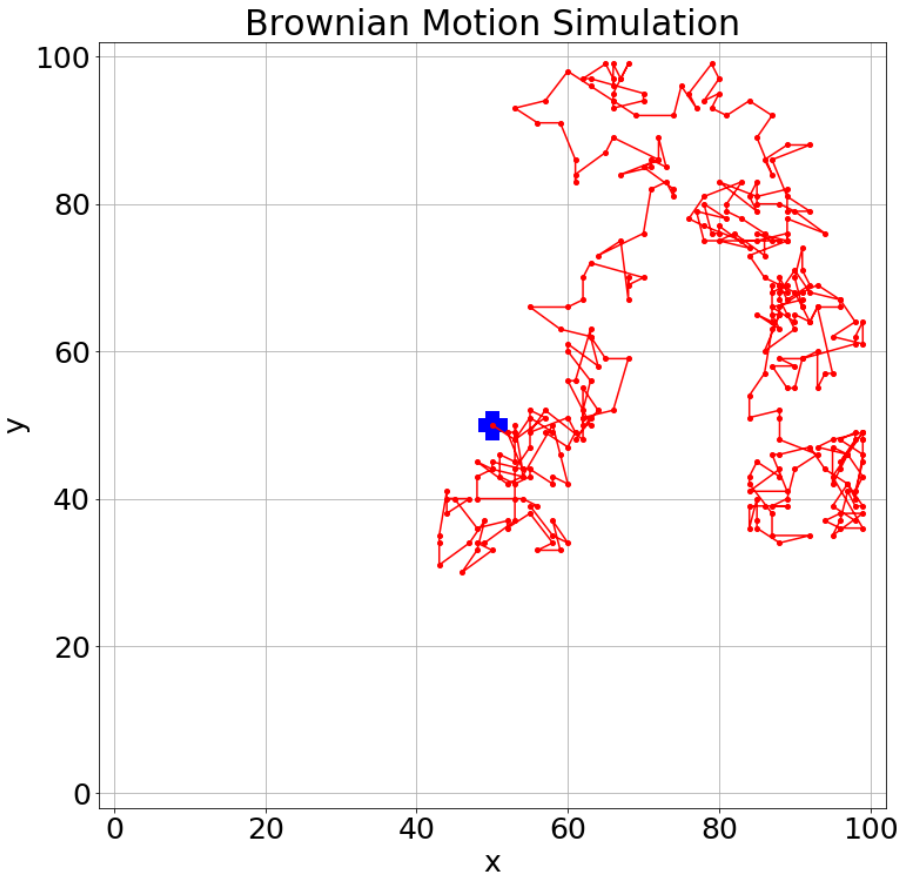
Zirui Wan (Question 1), Rundong Zhou (Question 2, 3)

November 29, 2020

1 Question 1

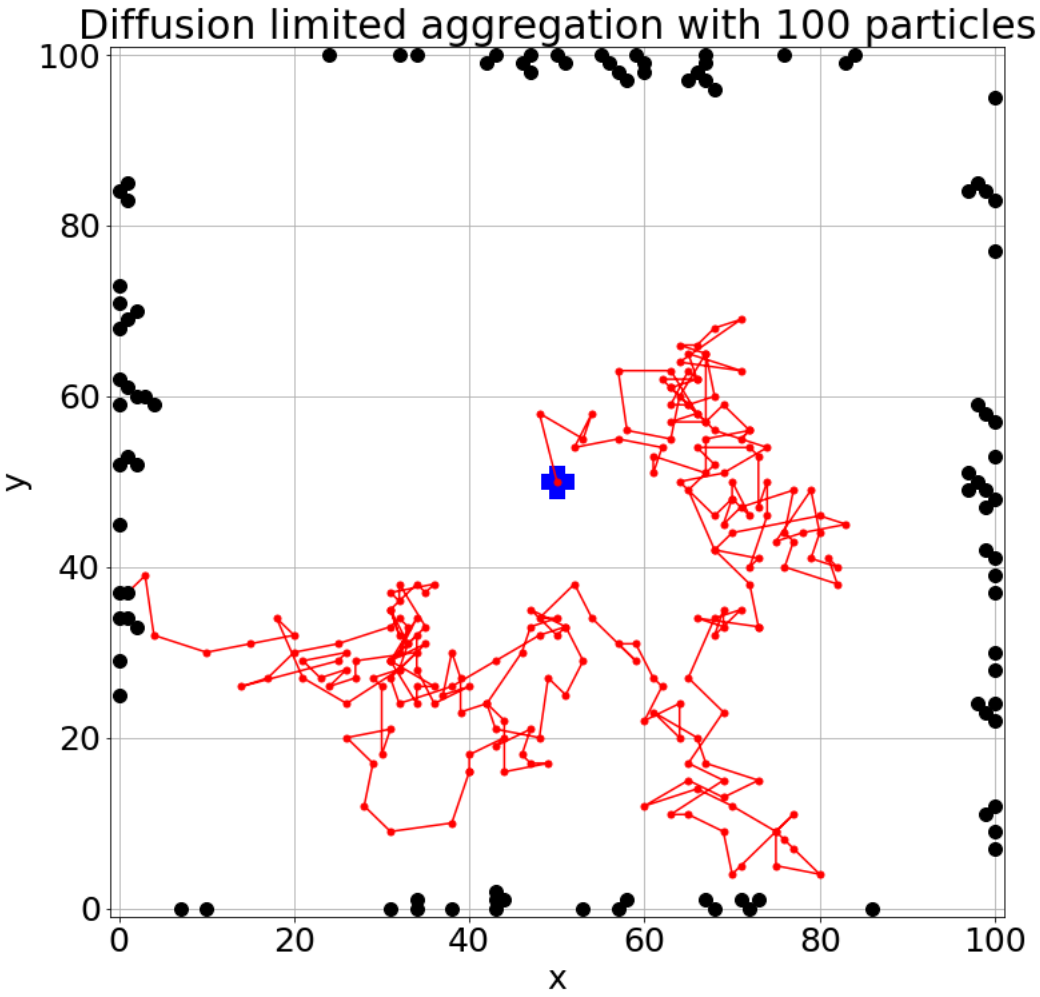
1.1 Q1 (a): Brownian motion

All codes submitted for this question have a fixed random seed for reproducibility.



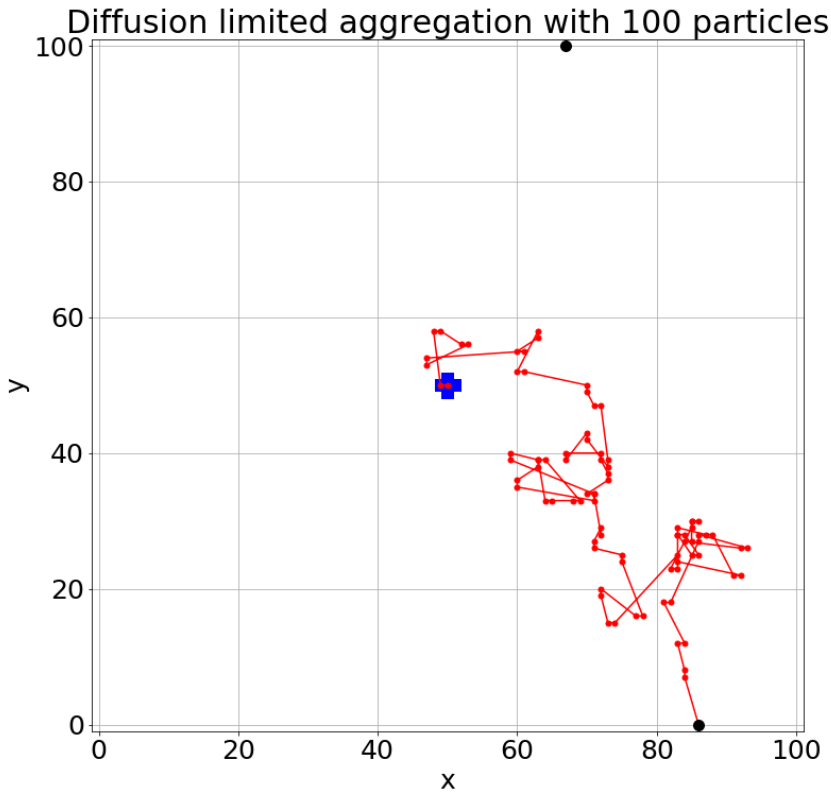
Using the code attached to this report the trajectory of the Brownian motion is simulated and plotted above. The code was based on the version provided by professor. I changed the "animation_interval" variable to be 15, so that the position of the particle will only be updated every 15 time steps. This is helpful for a cleaner visualization of the trajectory, otherwise there will be a lot of redundant steps. You can see that, when the particle is touching the boundary, it would not keep moving outsides.

1.2 Q1 (b): Diffusion limited aggregation (DLA)



The Brownian motion code was modified to become a DLA simulator. The major difference is that when a particle reaches the boundary, it would be anchored. And any other particles touching anchored particles will be anchored as well. I wrote up 2 versions of the code. One version is able to interactively plot particles and trajectories as an animation. The final distribution of 100 particles and the trajectory of the last particle is shown above. You can see that the particles form an interesting scattering-like pattern on the boundary.

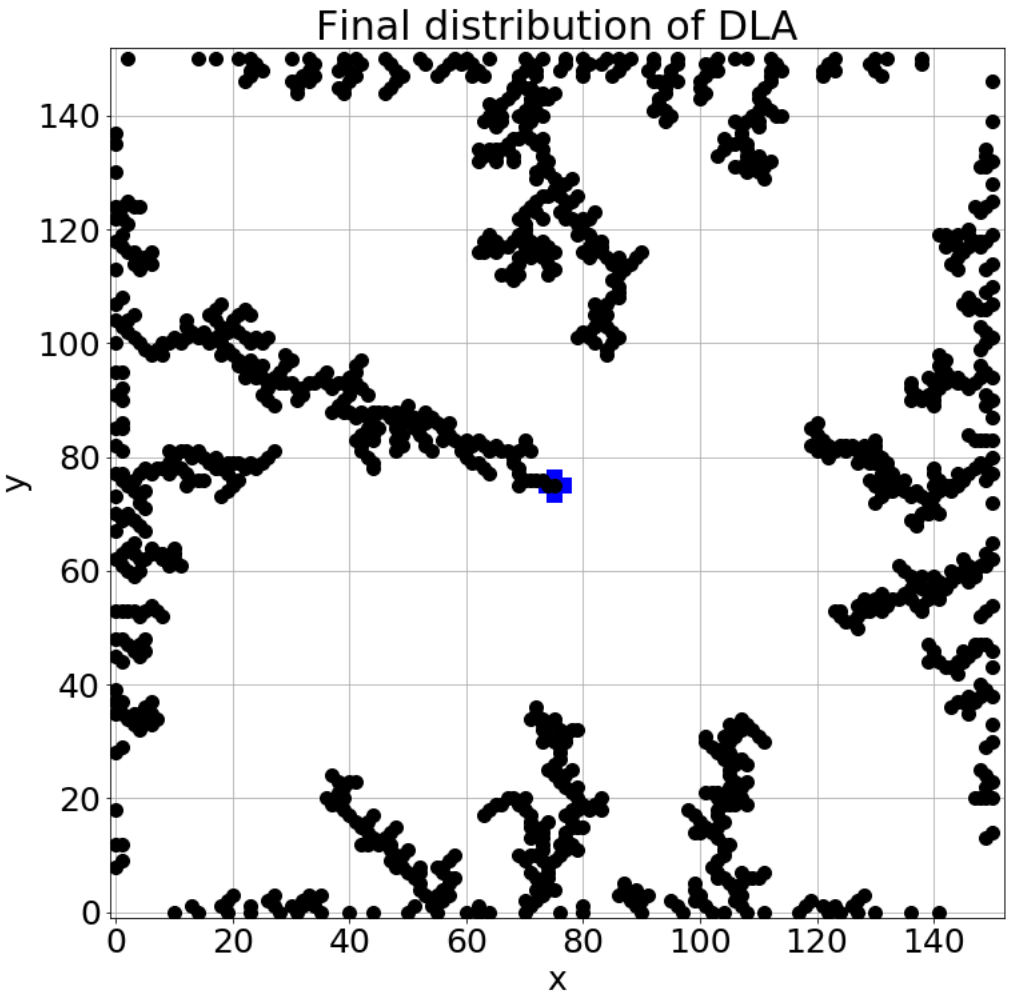
The other code generates the animation as a separate gif file. The gif animation is also included in this report.



.

In case that the animation in this report is playable (although it works with my Adobe reader DC), you can refer to the separate gif file submitted together with this report.

1.3 Q1 (b): Final distribution of DLA with large number of particles



The code was further modified, such that the DLA simulation could be completed when the centre point is anchored by a particle. The final distribution of particles after the DLA simulation finished is shown above. You can see that the final distribution shows an interesting pattern similar to tree branches. This pattern looks very similar to the picture from the copper aggregation experiment shown in the lab write-up. Quantitative analysis using this model could then

be conducted, if experimental data is provided. In case you are interested, the DLA simulation finished after 1179 particles are released from the centre of the grid.

2 Question 2, Volume of a 10-dimensional Hyper-sphere

After some googling, I found the general solution for the volume of a n-dimensional sphere (n is an integer) with radius R:

$$V_n = \frac{\pi^{\frac{n}{2}}}{\Gamma(\frac{n}{2}+1)} R^n$$

In our case, $n = 10$ and $R = 1$, so the analytic solution of the volume is given by:

$$V_{10} = \frac{\pi^5}{\Gamma(6)} = \frac{\pi^5}{120} \simeq 2.55$$

The integral we are about to calculate for the volume of a n-dimensional hyper-sphere with radius $R = 1$ is:

$$I = \int_{-1}^{1} ... \int_{-1}^{1} f(\vec{r})\, d\vec{r} \tag{1}$$

where

$$f(\vec{r}) = \begin{cases} 1 & \text{if } |\vec{r}| < 1 \\ 0 & \text{otherwise} \end{cases}$$

Please note that $\vec{r}$ is a vector with n components, and there are total n integral signs in Eqn.(1) corresponding to n dimensions.
From this, I am about to determine the analytic error for the Mean Value Monte-Carlo method for this integration. First, I need to calculate the expected value of the function $f(\vec{r})$. With a quick thought, we can realize that the expected value is just the chance for a random point to fall inside the hyper-sphere over the total region where we do the integral, which is a n-dimensional hyper-cube of length L.
So the expected value can be simply calculated by:

$$< f >= \frac{V}{L^n}$$

where V is the volume of the hyper-sphere. And, due to the nature of the function, we can tell:

$$< f^2 >=< f >= \frac{V}{L^n}$$

So we can obtain the variation of the function $f(\vec{r})$:

$$\text{var}\, f =< f^2 > - < f >^2 = \frac{V}{L^n} - \frac{V^2}{L^{2n}}$$

Let's look back to the textbook error equation:

$$\sigma = (b-a)^n \frac{\sqrt{\text{var } f}}{\sqrt{N}}$$

Please note the modified term $(b-a)^n$, since we are in n-dimension, and there are total n integral signs.
And we can substitute the result back to the equation:

$$\sigma = L^n \frac{\sqrt{\frac{V}{L^n} - \frac{V^2}{L^{2n}}}}{\sqrt{N}}$$

Please note, $b-a$ is just the length of the hyper-cube L.
With some simple algebra, we get:

$$\sigma = \sqrt{\frac{VL^n - V^2}{N}}$$

We plug in the parameters in our calculation, $V = \frac{\pi^5}{120}$, $L = 2$, $n = 10$, and $N = 1,000,000$:

$$\sigma \simeq 0.051$$

So I run the Mean Value method code for 30 times with N=1,000,000 to get a set of numerical results, and I calculate the average value and the standard deviation of the data set.

```
Standard deviation in V is: 0.04893648753067594
The average volume V of the hyper-sphere is: 2.5503744
```

We can conclude that our numerical and analytic result agree with each other very well!

3 Question 3, Importance sampling

3.1 Q3a, Evaluate the integral from textbook eqn. 10.34

Before I show you the histogram, here is a shot derivation of the drawing function:

$$\int_0^{x(z)} p(x')dx' = z$$

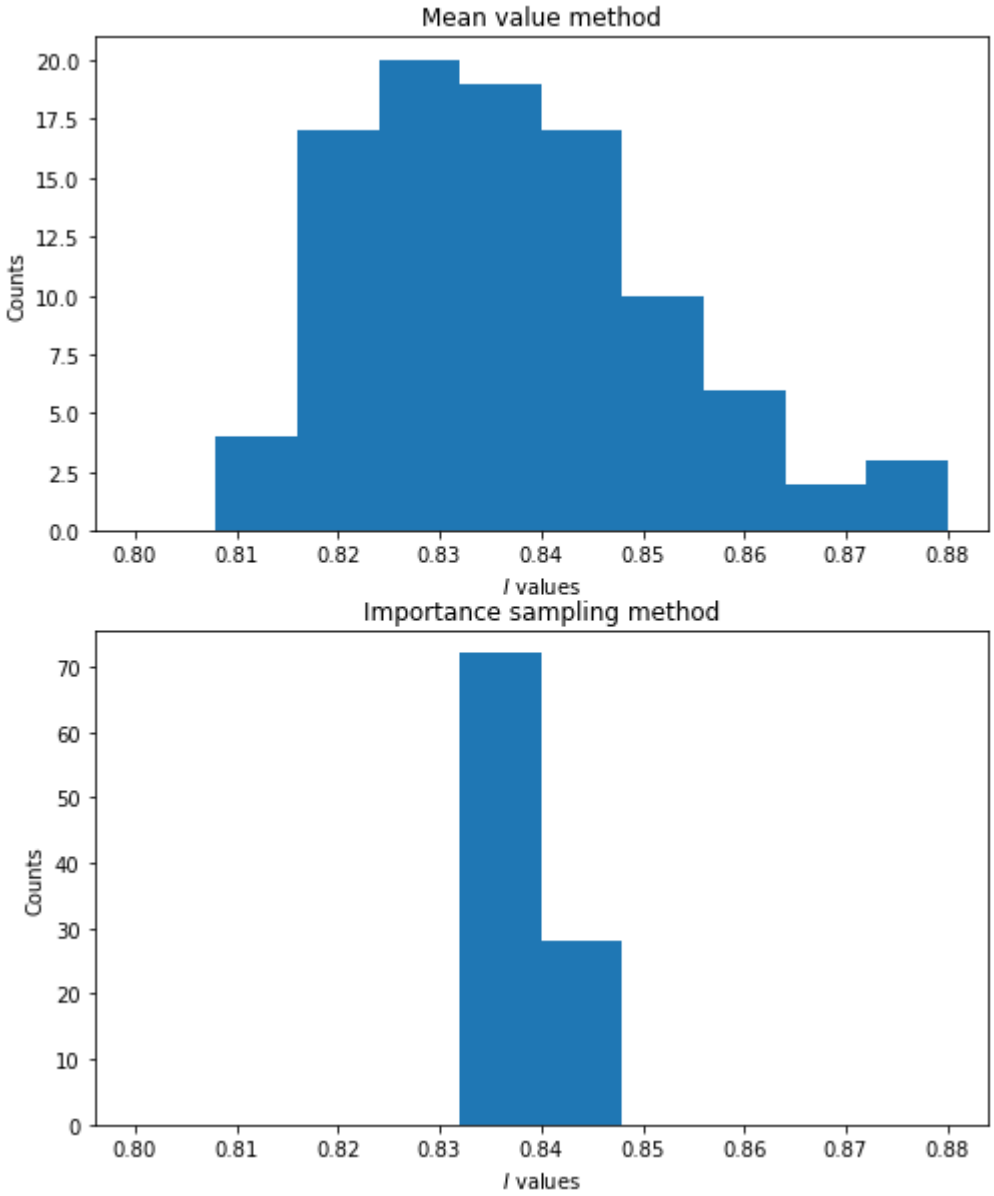
$$\frac{1}{2}\int_0^{x(z)} x'^{-\frac{1}{2}}dx' = z$$

$$\frac{1}{2}\cdot 2\sqrt{x(z)} = z$$

So,

$$x(z) = z^2$$

And here I present the histograms for both methods:



From the histograms, we can tell that the Mean Value method has columns way more separated than the Importance Sampling method. I also calculated the standard deviation for both data sets:

```
Standard deviation for Mean value method is: 0.018644889802192175
Standard deviation for Importance sampling method is: 0.0012573680213327232
```

We can see that the standard deviation in the Importance Sampling method is

about 10 times smaller than the Mean Value method. We can conclude that the Importance Sampling method has huge advantage in evaluating this integral, which includes singularity.

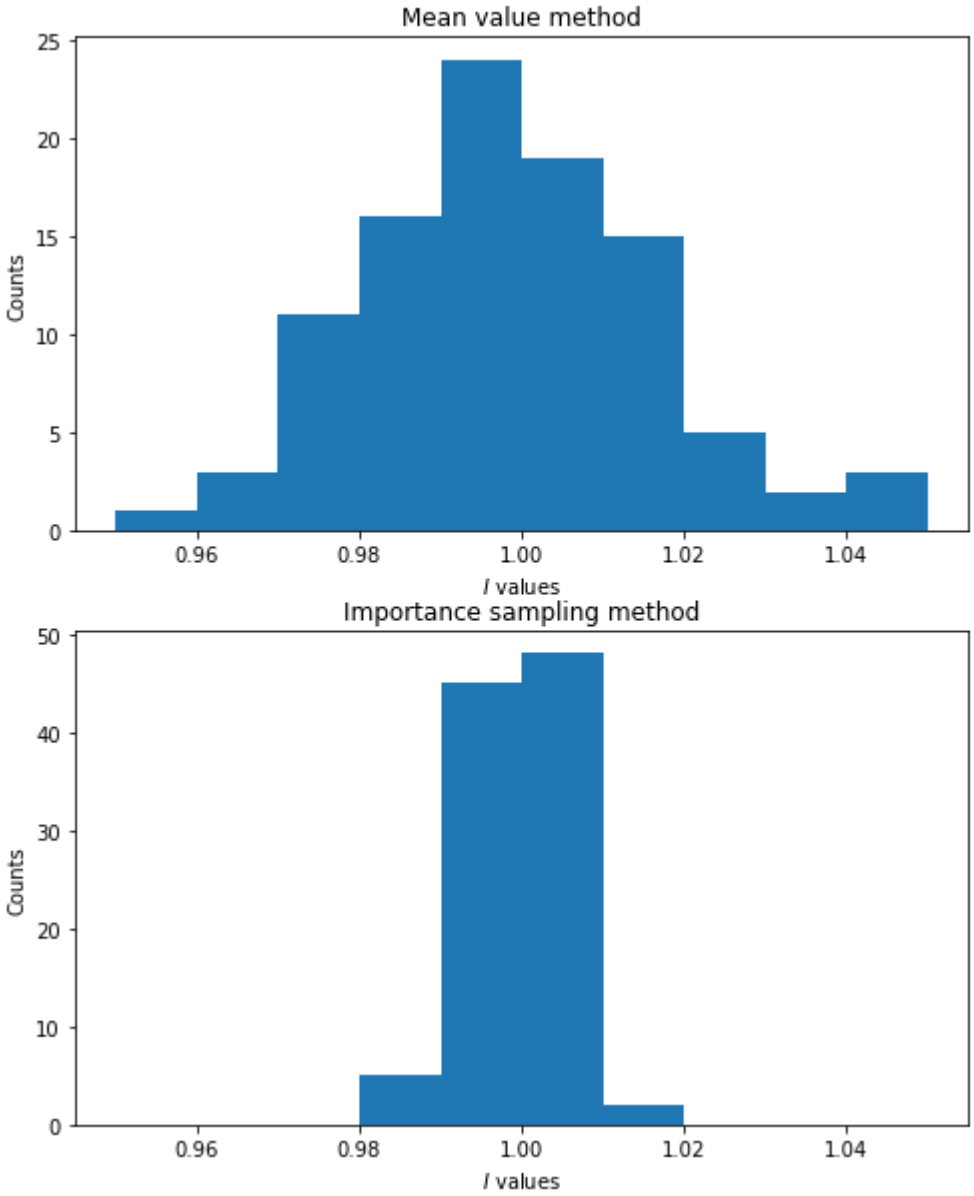
3.2 Q3b, Evaluate the rapidly varying integral but without singularity

Since the existence of the `numpy.random.normal()` function, I no longer need to derive the drawing function by myself.
However, there is something I wish to mention with this drawing process. Since we are using this python module, there is possibility that we draw a number out of our integral range 0 to 10. Although the chance is extremely low for a normal distribution with mean of 5 and standard deviation of 1.
Actually the probability is around 10^{-6}. But with N=10,000 and n=100, total 10^6 draws, there is about 70% chance that we draw a number out of our range every time we run the program. (Sorry I don't remember anything from my statistic course, this is something called Poisson process or Gamma distribution or sth. else... which I don't remember, but 70% is a fair rate)
So to eliminate such cases, I add these lines to the code to make sure all the numbers drawn by `numpy.random.normal()` is in desired range:

```
x = random.normal(5, 1)
if x < 0:
    x = 0
elif x > 10:
    x = 10
```

Well, then I realize this process is not quite necessary. Since the function evaluated at values outside the integral range is extremely small, and there is no singularity outside the range. A draw out of the range will not affect the calculation at all. ?
Or maybe it is necessary, since $w(x)$ has the $e^{-(x-5)^2}$ term and it sit on the denominator. While the nominator $f(x)$ has magnitude of $e^{|x-5|}$. A draw extremely far out from the range will lead to huge result. (i.e. for $x = -2$, $\frac{f(x)}{w(x)} = 91029.46$). Which is still handle-able for python. And the possibility for such draws which are 7 stds away from the mean is extremely low.
I will keep this thought process in the report just for fun.

The following is the histograms I obtain for both methods:



I plot the histogram over 0.95 to 1.05 with total 10 brackets.
We can tell that Importance sampling method still has a big advantage.
We can now look into the standard deviations:

```
Standard deviation for Mean value method is: 0.019203390386868156
Standard deviation for Importance sampling method is: 0.006133432846390354
```

This time, the standard deviation for Importance method is about 3 times

smaller than the Mean value method. Still big advantage, not not as huge as when we evaluate the function with singularity.
So we can conclude that the Importance Sampling method has more advantage at evaluating integrals with singularity.