# MLE_GrndTruth_against_various_distributions

September 16, 2018

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
```

```
In [2]: #Number of initialized data points
        N_points = 100000
        #Number of columns in histogram
        n_bins = 100
        Ground_truth = np.random.normal(1000,10,N_points)
```

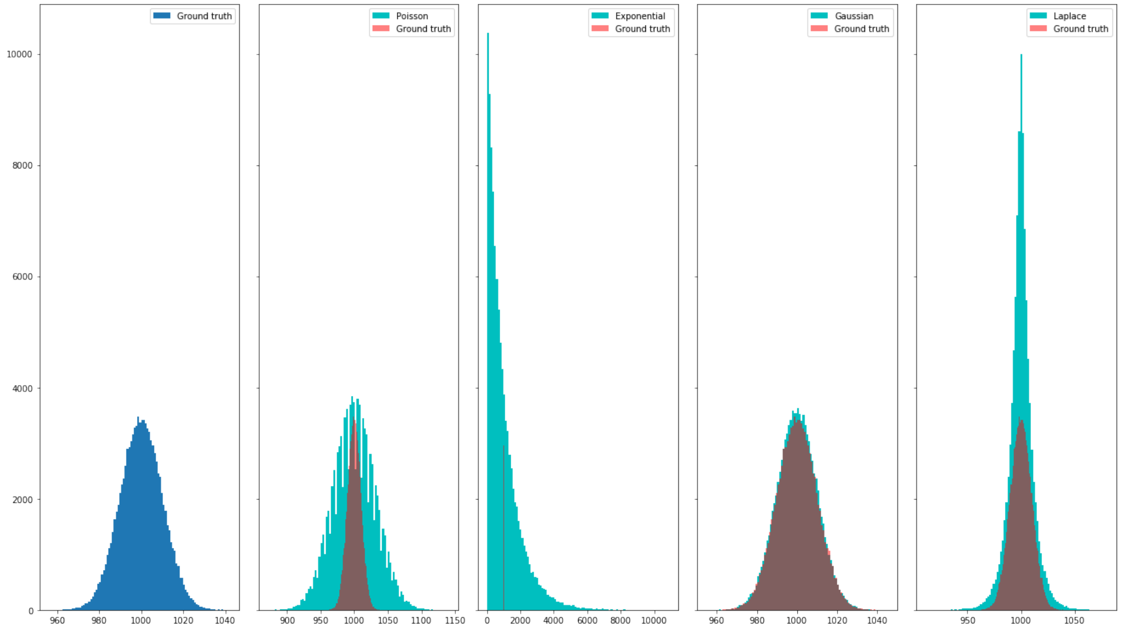
All plots in on place

```
In [3]: TH = np.mean(Ground_truth)
        Generated_binomial = np.random.binomial(TH,1,N_points)

        Lambda = np.mean(Ground_truth)
        Generated_poisson = np.random.poisson(Lambda,N_points)

        Beta = np.mean(Ground_truth)
        Generated_exponential = np.random.exponential(Beta,N_points)

        Mean = np.mean(Ground_truth)
        Std_dev = np.std(Ground_truth)
        Generated_Gaussian = np.random.normal(Mean,Std_dev,N_points)

        Mu = np.median(Ground_truth)
        Beta1 = np.mean(np.abs(Ground_truth-Mu))
        Generated_laplace = np.random.laplace(Mu,Beta1,N_points)
```

```
In [4]: fig, axs = plt.subplots(1, 5, sharey=True, tight_layout=True)
        axs[0].hist(Ground_truth, bins=n_bins,label = "Ground truth")
        axs[0].legend()
        # axs[1].hist(Generated_binomial, bins=n_bins)
        axs[1].hist(Generated_poisson, bins=n_bins, label = "Poisson",color = 'c')
        axs[1].hist(Ground_truth, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
        axs[1].legend()

        axs[2].hist(Generated_exponential, bins=n_bins,label = "Exponential",color='c')
        axs[2].hist(Ground_truth, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
```

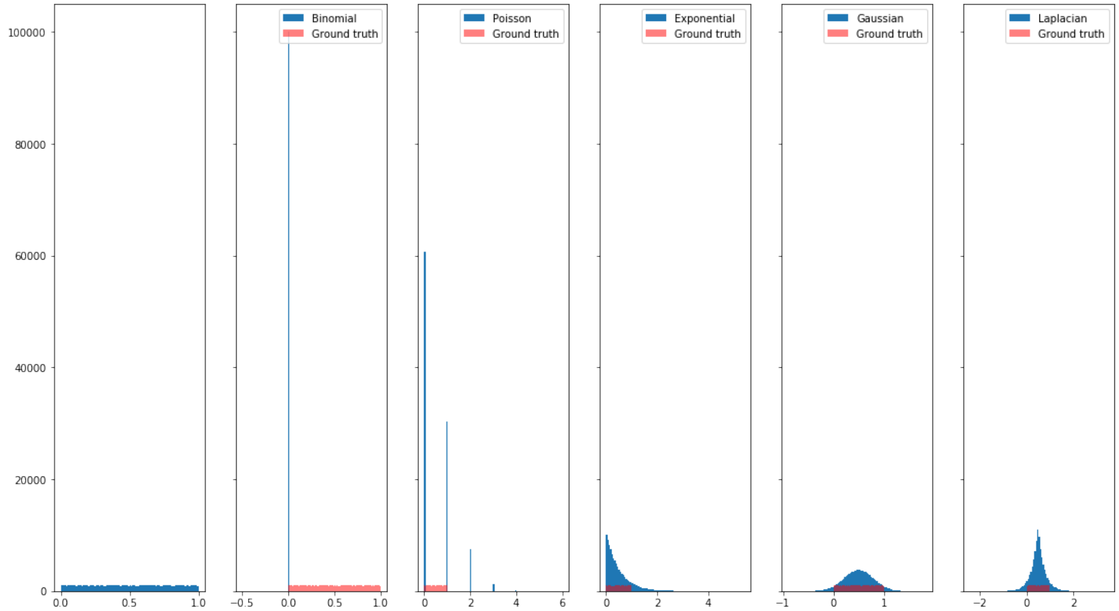
```
         axs[2].legend()

         axs[3].hist(Generated_Gaussian, bins=n_bins,label = "Gaussian",color='c')
         axs[3].hist(Ground_truth, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
         axs[3].legend()

         axs[4].hist(Generated_laplace, bins=n_bins,label = "Laplace",color='c')
         axs[4].hist(Ground_truth, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
         axs[4].legend()

         fig.set_size_inches(18.5, 10.5, forward=True)
         # plt.title("Histograms of Ground truth, Poisson, Exponential, Gaussian and Laplace di
         # plt.savefig("Distributions_Grndtruth_Gaussian.png")
```

```
/home/legion/.local/lib/python3.6/site-packages/matplotlib/figure.py:2299: UserWarning: This fi
  warnings.warn("This figure includes Axes that are not compatible "
```



Binomial distribution has a condition that the probability must be between 0 and 1 Hence choosing a different dataset which can cater these criterion Also, the binomial is a single trial binomial. For multi trial binomial, the math doesn't work out

```
In [5]: Ground_truth_1 = np.random.uniform(0,1,N_points)

        TH = np.mean(Ground_truth_1)
        Generated_binomial = np.random.binomial(TH,1,N_points)

        Lambda = np.mean(Ground_truth_1)
        Generated_poisson = np.random.poisson(Lambda,N_points)

        Beta = np.mean(Ground_truth_1)
        Generated_exponential = np.random.exponential(Beta,N_points)

        Mean = np.mean(Ground_truth_1)
        Std_dev = np.std(Ground_truth_1)
        Generated_Gaussian = np.random.normal(Mean,Std_dev,N_points)

        Mu = np.median(Ground_truth_1)
        Beta1 = np.abs(np.mean(np.abs(Ground_truth_1-Mu)))
        Generated_laplace = np.random.laplace(Mu,Beta1,N_points)

In [6]: fig, axs = plt.subplots(1, 6, sharey=True, tight_layout=False)
        axs[0].hist(Ground_truth_1, bins=n_bins,label="Ground truth")

        axs[1].hist(Generated_binomial, bins=n_bins,label="Binomial")
        axs[1].hist(Ground_truth_1, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
        axs[1].legend()

        axs[2].hist(Generated_poisson, bins=n_bins,label="Poisson")
        axs[2].hist(Ground_truth_1, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
        axs[2].legend()

        axs[3].hist(Generated_exponential, bins=n_bins,label="Exponential")
        axs[3].hist(Ground_truth_1, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
        axs[3].legend()

        axs[4].hist(Generated_Gaussian, bins=n_bins,label="Gaussian")
        axs[4].hist(Ground_truth_1, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
        axs[4].legend()

        axs[5].hist(Generated_laplace, bins=n_bins,label="Laplacian")
        axs[5].hist(Ground_truth_1, bins=n_bins,label = "Ground truth",color='r',alpha=0.5)
        axs[5].legend()

        fig.set_size_inches(18.5, 10.5, forward=True)
        # plt.title("Histograms of Ground truth, Binomial, Poisson, Exponential, Gaussian and
        # plt.savefig("Distributions_GrndTrth_Uniform.png")
```



One on one comparision

```
In [7]: print("Select distribution by number")
        print("1. Binomial")
        print("2. Poisson")
        print("3. Exponential")
        print("4. Gaussian")
        print("5. Laplacian")
        distri = int(input())
```

```
Select distribution by number
1. Binomial
2. Poisson
3. Exponential
4. Gaussian
5. Laplacian
3
```

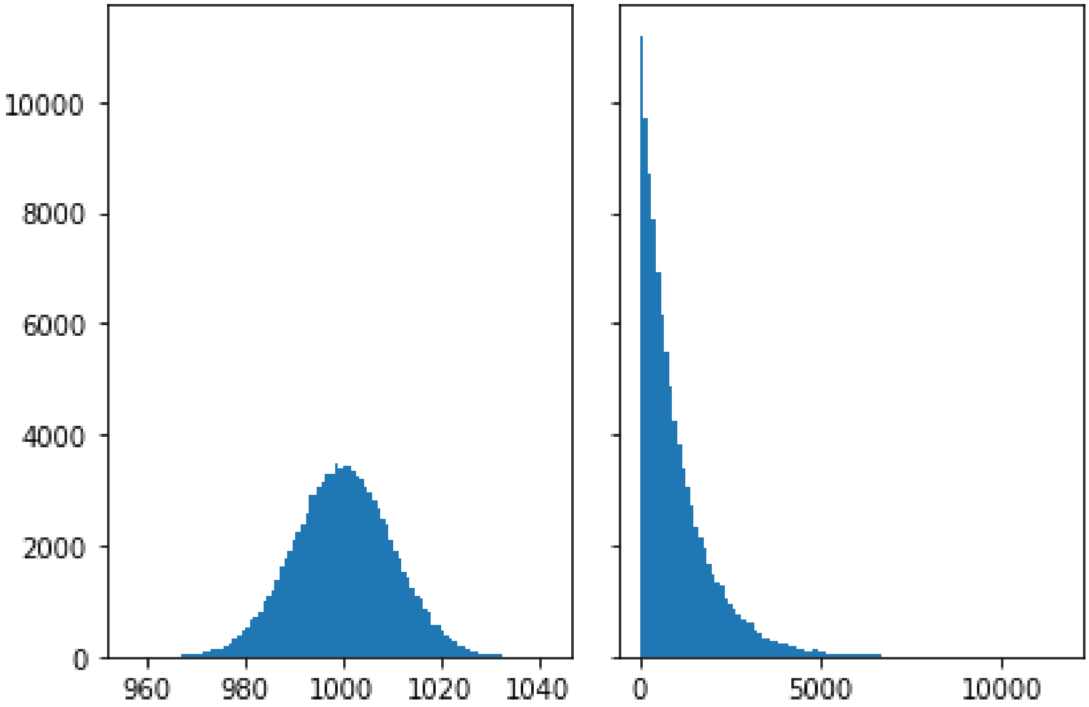
Here I've kept a plot with binomial distribution samples generated with different tries parameter N

```
In [8]: if(distri == 1):
            Ground_truth_u = np.random.uniform(0,1,N_points)
            TH = np.mean(Ground_truth_u)
            Generated_bin_1 = np.random.binomial(1,TH,N_points)
            Generated_bin_5 = np.random.binomial(5,TH,N_points)
            Generated_bin_10 = np.random.binomial(10,TH,N_points)
            Generated_bin_100 = np.random.binomial(100,TH,N_points)
            Generated_bin_1000 = np.random.binomial(1000,TH,N_points)
            Generated_bin_10000 = np.random.binomial(10000,TH,N_points)
            Generated_bin_100000 = np.random.binomial(100000,TH,N_points)

            fig, axs = plt.subplots(1, 8, sharey=True, tight_layout=True)
            axs[0].hist(Ground_truth_u, bins=n_bins)
            axs[1].hist(Generated_bin_1, bins=n_bins)
            axs[2].hist(Generated_bin_5, bins=n_bins)
            axs[3].hist(Generated_bin_10, bins=n_bins)
            axs[4].hist(Generated_bin_100, bins=n_bins)
            axs[5].hist(Generated_bin_1000, bins=n_bins)
            axs[6].hist(Generated_bin_10000, bins=n_bins)
            axs[7].hist(Generated_bin_100000, bins=n_bins)
            fig.set_size_inches(18.5, 10.5, forward=True)
        #     plt.title("Binomial generated with varying tries (N)")
        #     plt.savefig("Distributions_Binomial_different_tries.png")
```

```
In [9]: if(distri == 2):
            Lambda = np.mean(Ground_truth)
            Generated = np.random.poisson(Lambda,N_points)
            fig, axs = plt.subplots(1, 2, sharey=True, tight_layout=True)
            axs[0].hist(Ground_truth, bins=n_bins)
            axs[1].hist(Generated, bins=n_bins)
```

```
In [10]: if(distri == 3):
             Beta = np.mean(Ground_truth)
             Generated = np.random.exponential(Beta,N_points)
             fig, axs = plt.subplots(1, 2, sharey=True, tight_layout=True)
             axs[0].hist(Ground_truth, bins=n_bins)
             axs[1].hist(Generated, bins=n_bins)
```

```
/home/legion/.local/lib/python3.6/site-packages/matplotlib/figure.py:2299: UserWarning: This fi
  warnings.warn("This figure includes Axes that are not compatible "
```



```
In [11]: if(distri == 4):
             Mean = np.mean(Ground_truth)
             Std_dev = np.std(Ground_truth)
             Generated = np.random.normal(Mean,Std_dev,N_points)
             fig, axs = plt.subplots(1, 2, sharey=True, tight_layout=True)
             axs[0].hist(Ground_truth, bins=n_bins)
             axs[1].hist(Generated, bins=n_bins)

In [12]: if(distri == 5):
             Mu = np.median(Ground_truth)
             Beta = np.mean(np.abs(Ground_truth-Mu))
             Generated = np.random.laplace(Mu,Beta,N_points)
             fig, axs = plt.subplots(1, 2, sharey=True, tight_layout=True)
             axs[0].hist(Ground_truth, bins=n_bins)
             axs[1].hist(Generated, bins=n_bins)

In [ ]:
```