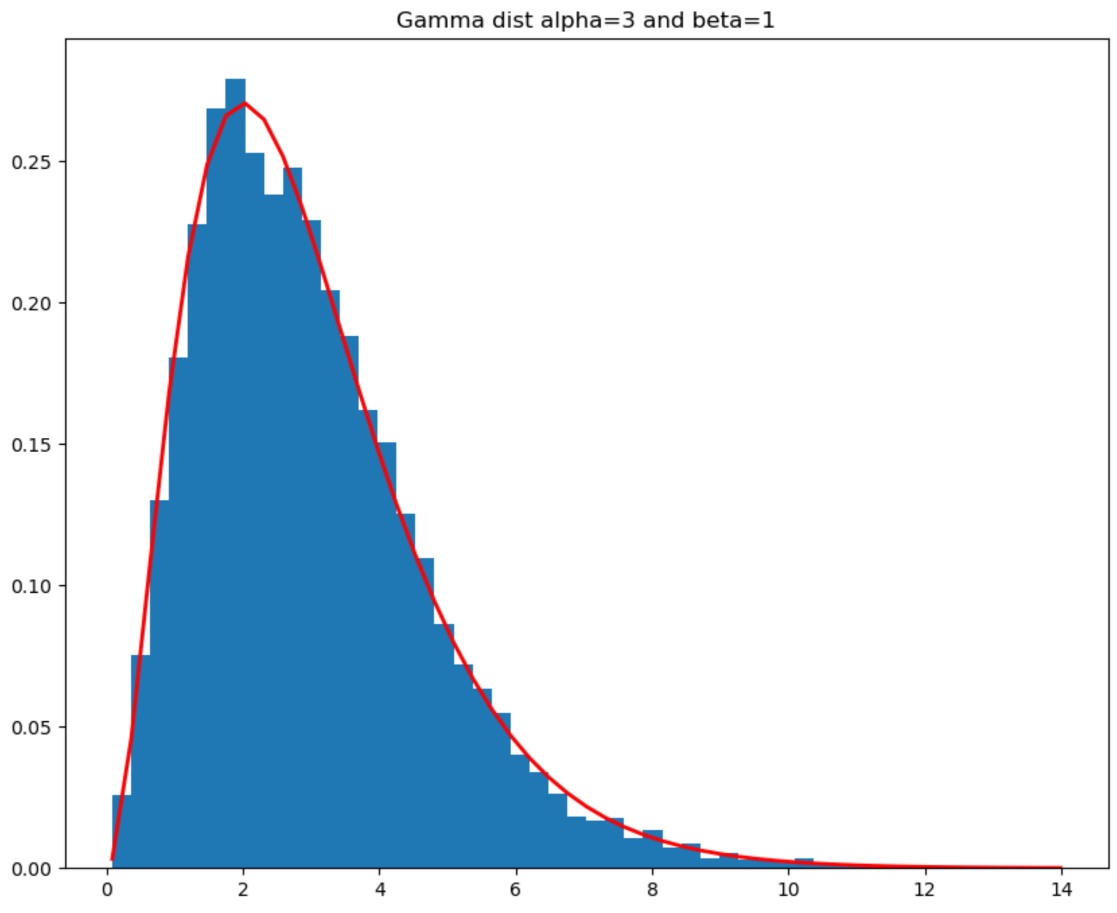
Assignment 4

January 12, 2023

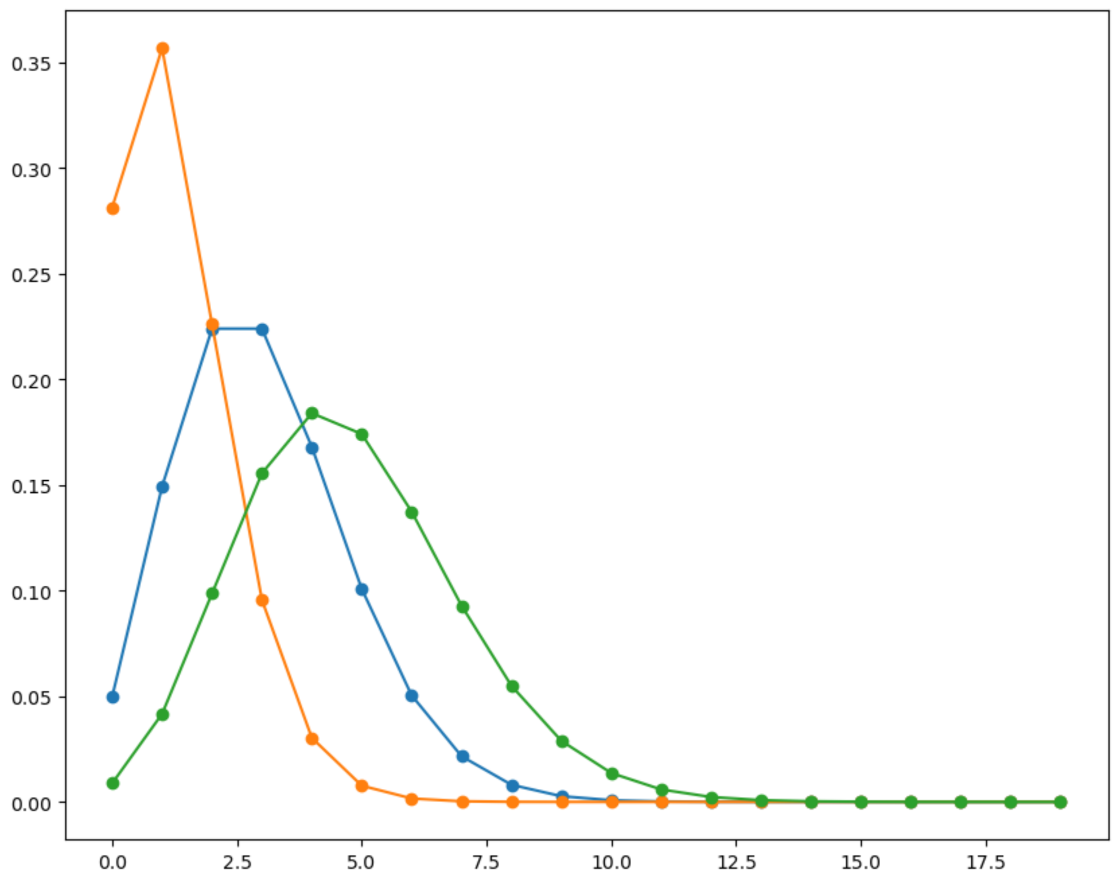
0.1 Question 9

[63]:
```
import numpy as np
```

[109]:
```
#(a)
shape = 3
beta = 1
scale=1/beta

gamma_sample = np.random.gamma(shape, scale, 10000)
import matplotlib.pyplot as plt

fig, ax = plt.subplots(figsize=(10,8))
# # plt.subplots_adjust(top = 0.99, bottom=0.01, hspace=0.5, wspace=0.2)
count, bins, ignored = ax.hist(gamma_sample, 50, density=True)
ax.set_title(f"Gamma dist alpha=3 and beta=1")

import scipy.special as sps
y = bins**(shape-1)*(np.exp(-bins/scale) /
                     (sps.gamma(shape)*scale**shape))
plt.plot(bins, y, linewidth=2, color='r')
plt.show()


fig.show()
```

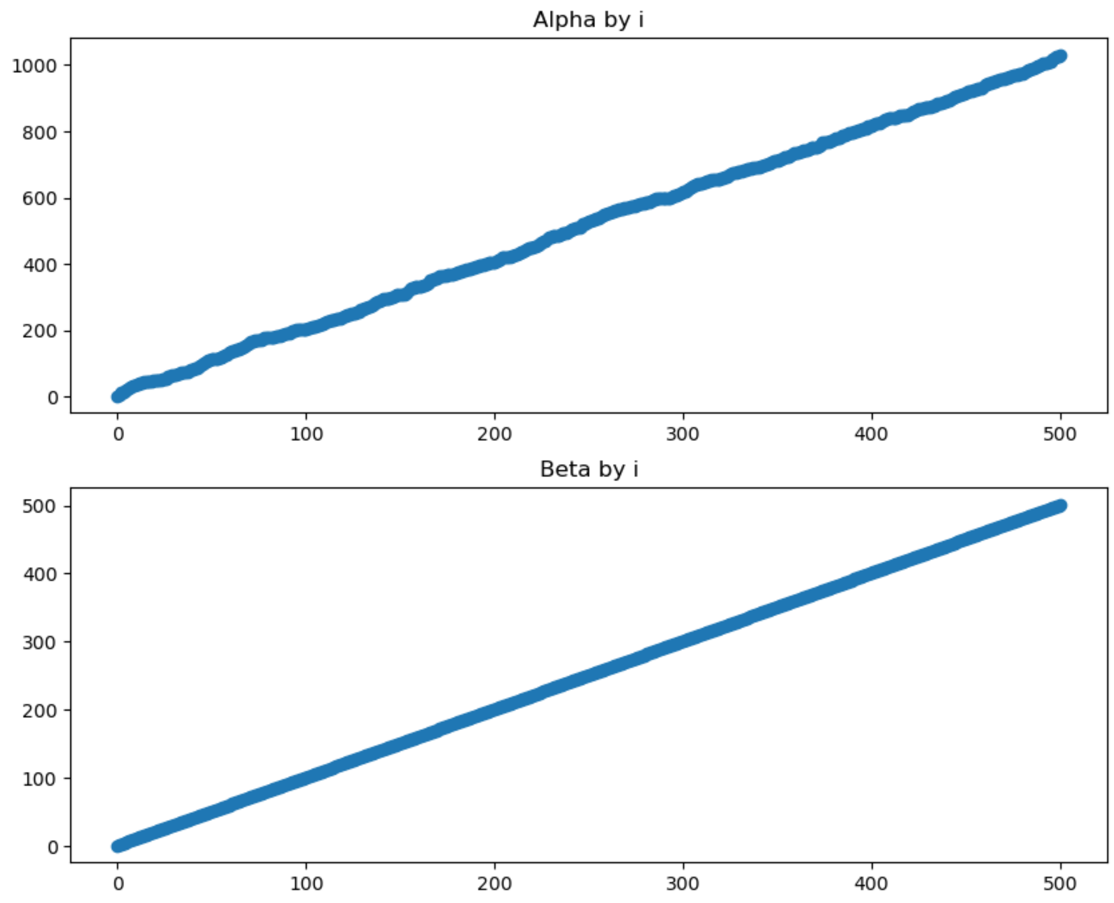


[110]:
```
#(b)
import matplotlib.pyplot as plt
import math

from scipy.stats import poisson

gamma_mean = shape / scale
gamma_std = math.sqrt(shape / (scale * scale))

means = [gamma_mean, gamma_mean - gamma_std, gamma_mean + gamma_std]

x = np.arange(0, 20)
fig, ax = plt.subplots(figsize=(10,8))

for i, m in enumerate(means):
    pmf = poisson.pmf(x, mu=m)
    ax.plot(x, pmf, marker='o')

fig.show()
```

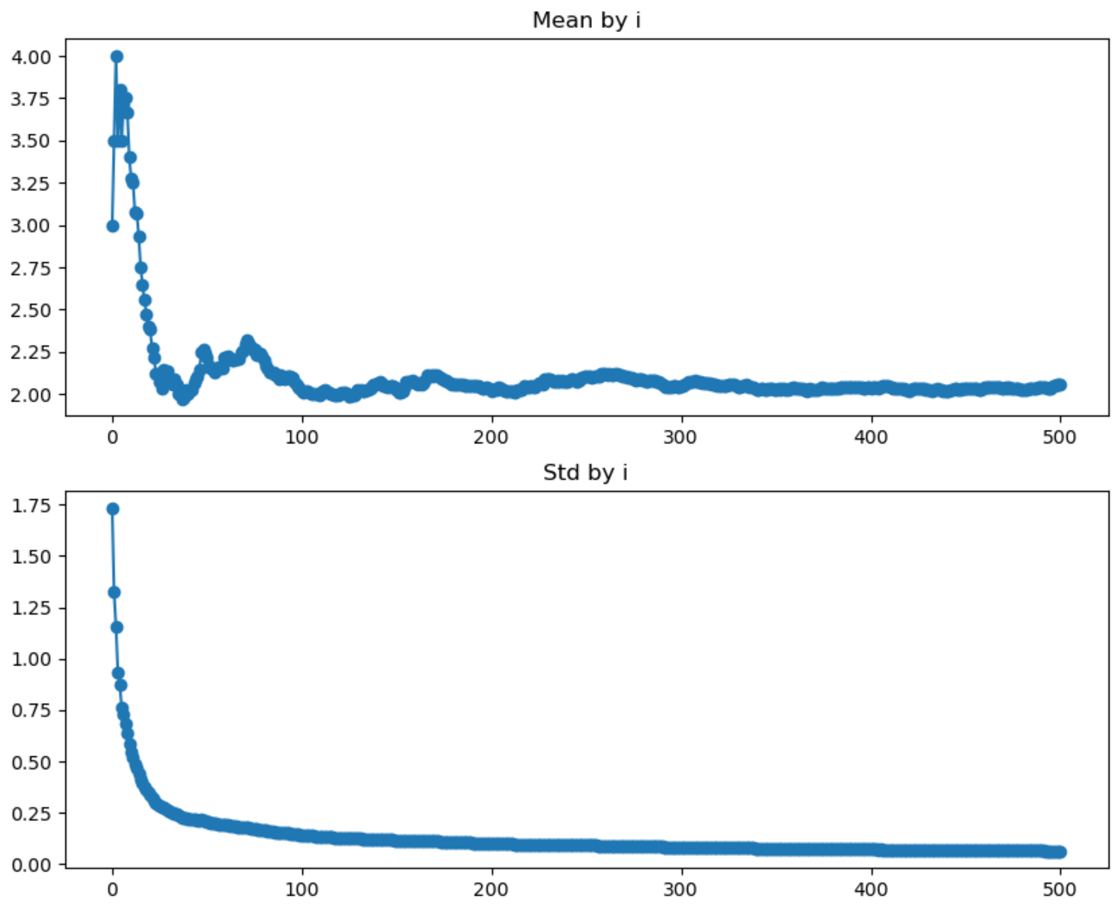


[111]:
```python
# (c)

sample = np.random.poisson(2, 500)

posterior_dist = [(shape, scale)]

for n in sample:
    a, b = posterior_dist[-1]
    posterior_dist.append((a + n, b + 1))

ass = np.array([a[0] for a in posterior_dist])
bss = np.array([a[1] for a in posterior_dist])
x = range(len(bss))
fig, ax = plt.subplots(2, figsize=(10,8))
ax[0].plot(x, ass, marker='o')
ax[0].set_title(f"Alpha by i")

ax[1].plot(x, bss, marker='o')
ax[1].set_title(f"Beta by i")
```

```
fig.show()
```



[127]:
```
# (d)
x = range(len(bss))
fig, ax = plt.subplots(2, figsize=(10,8))
ax[0].plot(x, ass/bss, marker='o')
ax[0].set_title(f"Mean by i")

ax[1].plot(x, np.sqrt(ass/(bss*bss)), marker='o')
ax[1].set_title(f"Std by i")

fig.show()
```

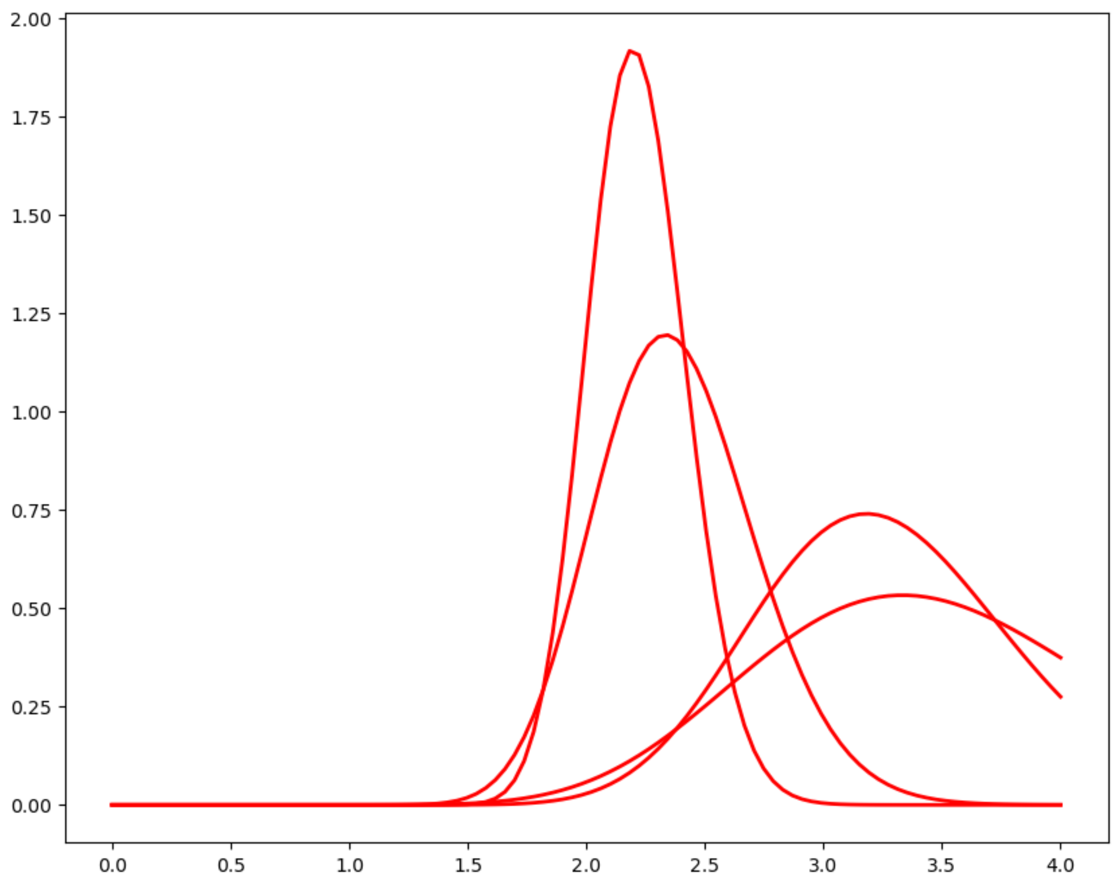


```
[121]: np.linspace(0, 5, 100)
```

```
[121]: array([0.        , 0.05050505, 0.1010101 , 0.15151515, 0.2020202 ,
              0.25252525, 0.3030303 , 0.35353535, 0.4040404 , 0.45454545,
              0.50505051, 0.55555556, 0.60606061, 0.65656566, 0.70707071,
              0.75757576, 0.80808081, 0.85858586, 0.90909091, 0.95959596,
              1.01010101, 1.06060606, 1.11111111, 1.16161616, 1.21212121,
              1.26262626, 1.31313131, 1.36363636, 1.41414141, 1.46464646,
              1.51515152, 1.56565657, 1.61616162, 1.66666667, 1.71717172,
              1.76767677, 1.81818182, 1.86868687, 1.91919192, 1.96969697,
              2.02020202, 2.07070707, 2.12121212, 2.17171717, 2.22222222,
              2.27272727, 2.32323232, 2.37373737, 2.42424242, 2.47474747,
              2.52525253, 2.57575758, 2.62626263, 2.67676768, 2.72727273,
              2.77777778, 2.82828283, 2.87878788, 2.92929293, 2.97979798,
              3.03030303, 3.08080808, 3.13131313, 3.18181818, 3.23232323,
              3.28282828, 3.33333333, 3.38383838, 3.43434343, 3.48484848,
              3.53535354, 3.58585859, 3.63636364, 3.68686869, 3.73737374,
              3.78787879, 3.83838384, 3.88888889, 3.93939394, 3.98989899,
              4.04040404, 4.09090909, 4.14141414, 4.19191919, 4.24242424,
              4.29292929, 4.34343434, 4.39393939, 4.44444444, 4.49494949,
              4.54545455, 4.5959596 , 4.64646465, 4.6969697 , 4.74747475,
              4.7979798 , 4.84848485, 4.8989899 , 4.94949495, 5.        ])
```

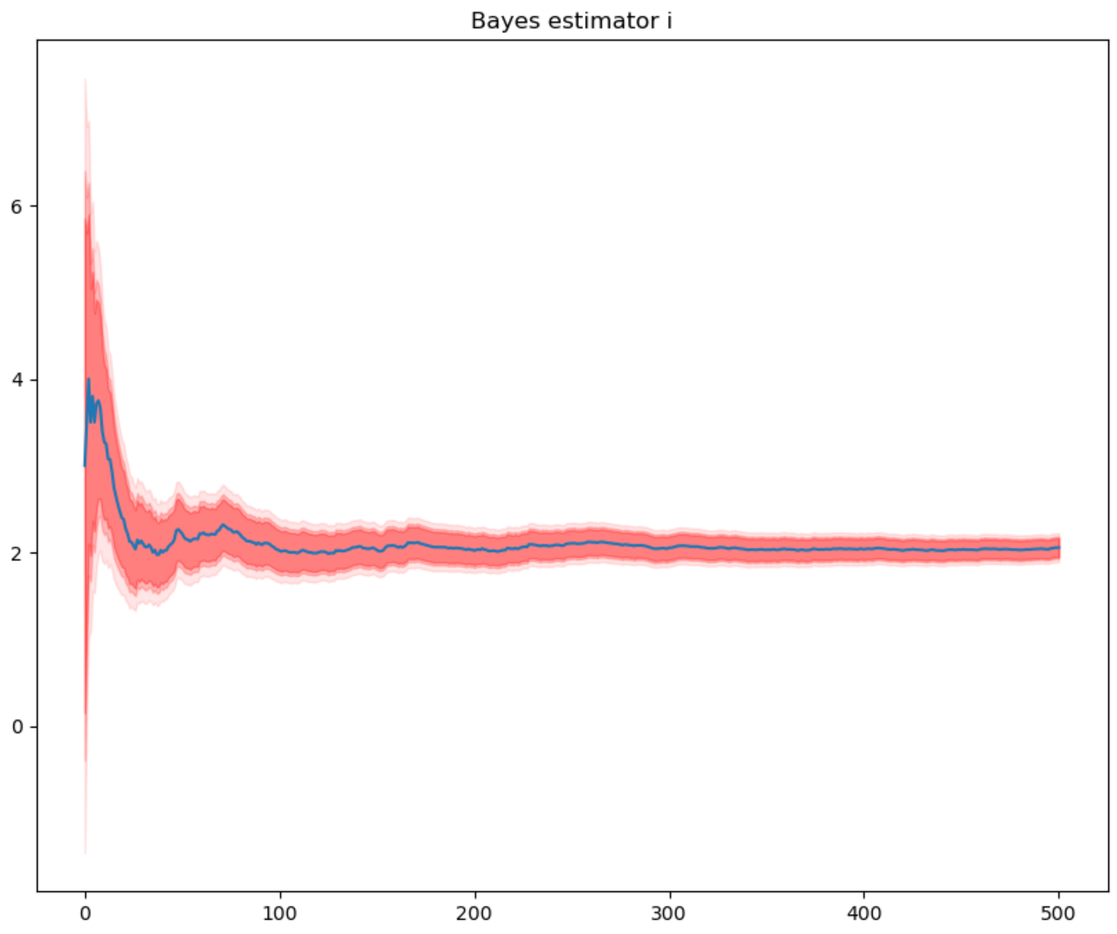
```
[125]: #(e)
       fig, ax = plt.subplots(figsize=(10,8))
       x = np.linspace(0, 4, 100)
       for i in [5, 10, 20, 50, 100, 500]:
           shape = ass[i]
           beta = bss[i]
           scale=1/beta

           y = x**(shape-1)*(np.exp(-x/scale) /
                                 (sps.gamma(shape)*scale**shape))
           ax.plot(x, y, linewidth=2, color='r')

       fig.show()
```

```
/tmp/ipykernel_1077/88553906.py:10: RuntimeWarning: invalid value encountered in
scalar multiply
  (sps.gamma(shape)*scale**shape))
/tmp/ipykernel_1077/88553906.py:9: RuntimeWarning: overflow encountered in power
  y = x**(shape-1)*(np.exp(-x/scale) /
```



[134]:
```
conf_intervals = [1.645, 1.960, 2.576]
```

[138]:
```
# (f)
x = range(len(bss))
fig, ax = plt.subplots(figsize=(10,8))

e_mean = ass/bss
e_std = np.sqrt(ass)/bss
ax.plot(x, e_mean)

ax.set_title(f"Bayes estimator i")
ax.fill_between(x, (e_mean-e_std*conf_intervals[0]),␣
 ↪(e_mean+e_std*conf_intervals[0]), color='r', alpha=.3)
ax.fill_between(x, (e_mean-e_std*conf_intervals[1]),␣
 ↪(e_mean+e_std*conf_intervals[1]), color='r', alpha=.2)
ax.fill_between(x, (e_mean-e_std*conf_intervals[2]),␣
 ↪(e_mean+e_std*conf_intervals[2]), color='r', alpha=.1)

fig.show()
```



(g) We know that the MLE estimator for poisson is the sample mean