In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:

```
N = 1000
distributions = [lambda x: np.random.binomial(50, x, N), lambda x: np.random.exp
onential(x, N),
                 lambda x: np.random.normal(x, 2.1, N)]
params = [np.random.uniform(0, 1) for i in range(len(distributions))]
samples = [distributions[i](params[i]) for i in range(len(distributions))]
names = ['binomial', 'exponential', 'normal']
```

Функции для определения нижней границы дисперсии по правилу Рао-Крамера.

In [3]:

```
def bound1(size):
    return params[0]*(1 - params[0])/size/50.0

def bound2(size):
    return params[1]**2/size

def bound3(size):
    return 2.1/size

bounds = [bound1, bound2, bound3]
```

Эффективные оценки

Функции оценок

In [8]:

```
def est1(sample):
    return np.mean(sample)/50.0

def est2(sample):
    return (len(sample)-1)/np.mean(sample)

def est3(sample):
    return np.mean(sample)

estimations = [est1, est2, est3]
```

Функия, которая будет рисовать графики:

In [9]:

```
def plot_disp(limits=None):
    for i in range(len(bootstrap_disp)):
        plt.plot(np.array(range(N)[2:]) + 1, bootstrap_disp[i], label='experimen
t')
        plt.plot(np.array(range(N)[2:]) + 1, [bounds[i](x) for x in np.array(ran
ge(N)[2:]) + 1], label='theoretical bound')
        if limits:
            plt.axis(limits[i])
        plt.title(names[i], fontsize=20)
        plt.xlabel('sample size', fontsize=20)
        plt.ylabel('dispersion', fontsize=20)
        plt.legend()
        plt.show()
```

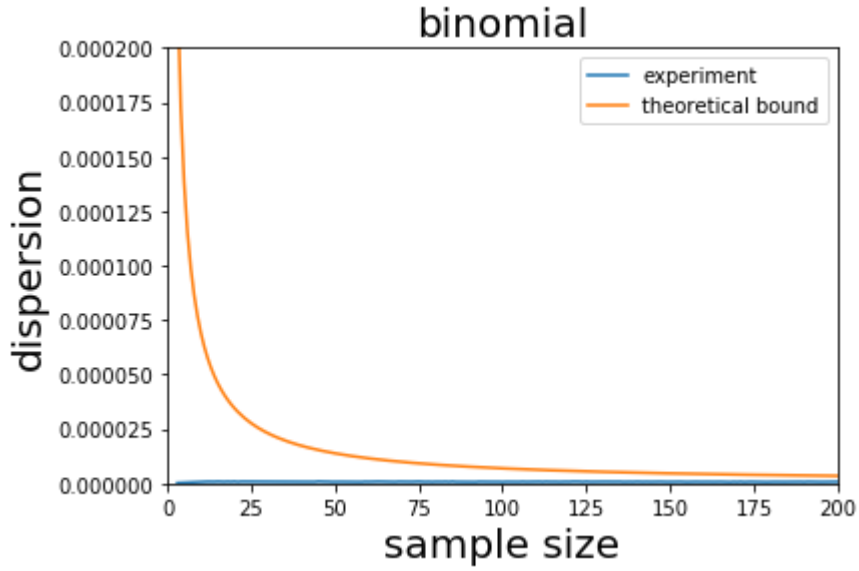
В этом куске и происходят все расчеты. Проходим по всем размерам выборок и всем распределениям, оцениваем параметр, создаем массив оценок по бутсртепным выборкам, считаем его среднеквадратичную дисперсию.

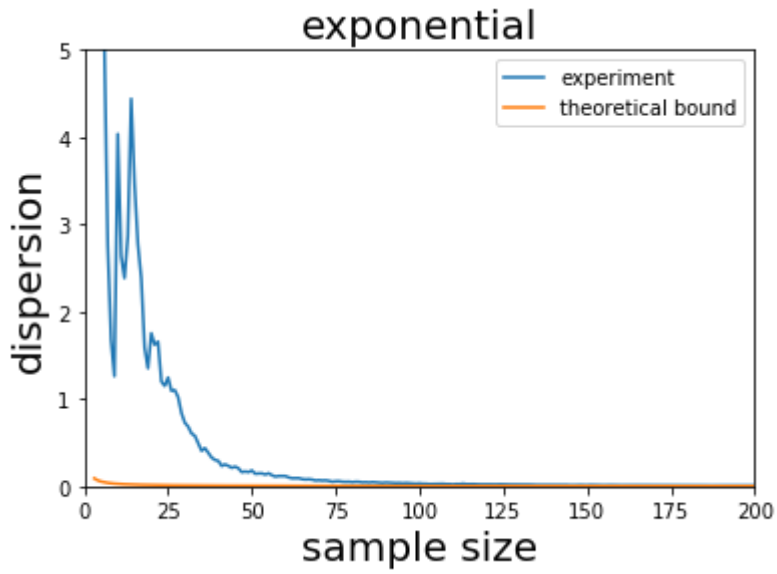
In [10]:

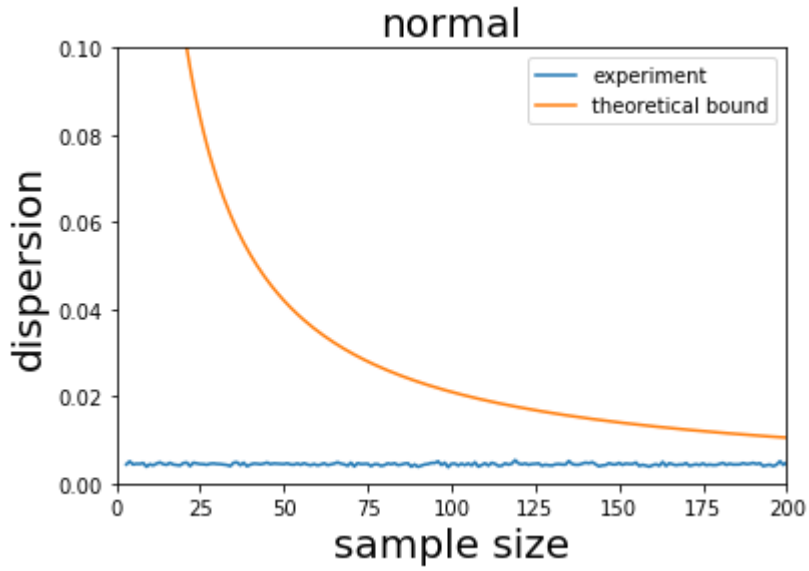
```
bootstrap_disp = [[] for i in range(len(distributions))]
for i in range(N)[2:]:
    for j in range(len(distributions)):
        estimation = estimations[j](samples[j][:i])
        bootstrap_est = np.array([estimations[j](distributions[j](estimation)) f
or k in range(500)])
        bootstrap_disp[j].append(np.var(bootstrap_est))
```

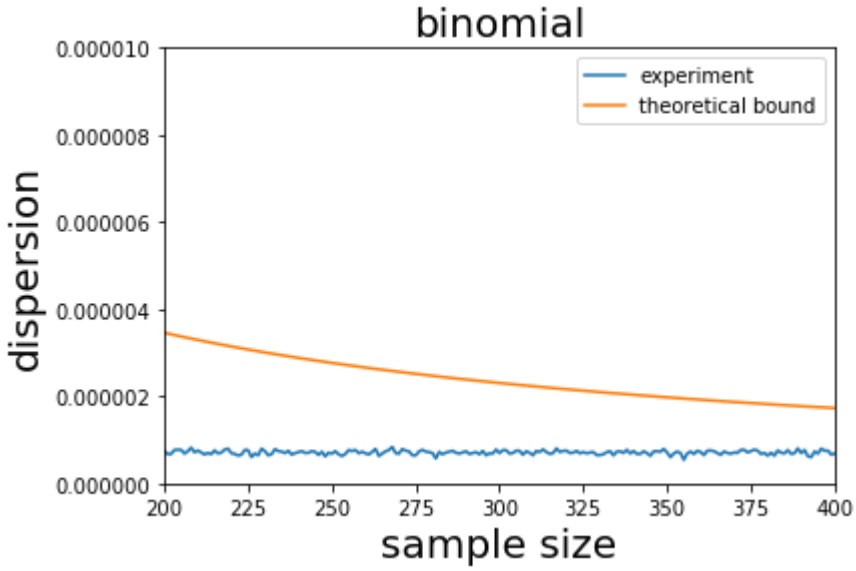
In [14]:

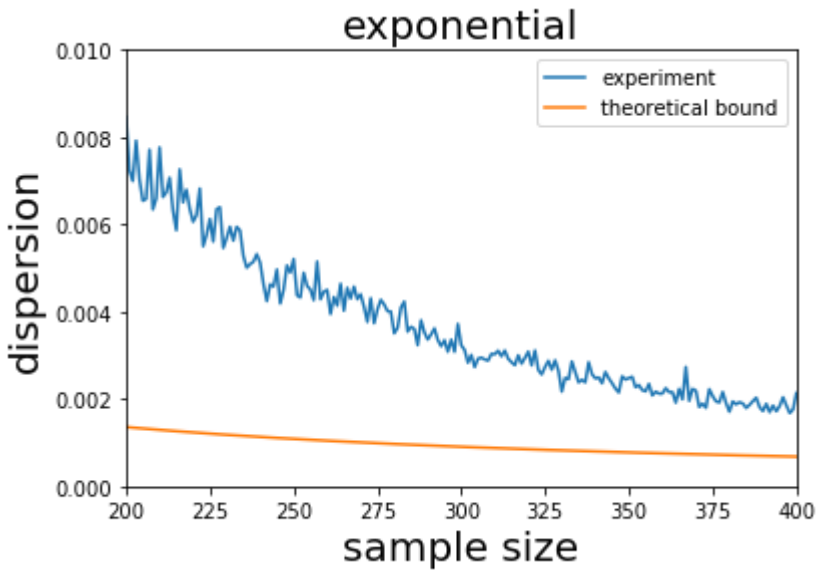
```
plot_disp([[0, 200, 0, 0.0002], [0, 200, 0, 5], [0, 200, 0, 0.1]])
plot_disp([[200, 400, 0, 0.00001], [200, 400, 0, 0.01], [200, 400, 0, 0.01]])
plot_disp([[400, 1000, 0, 0.0000025], [400, 1000, 0, 0.002], [400, 1000, 0,
0.01]])
```

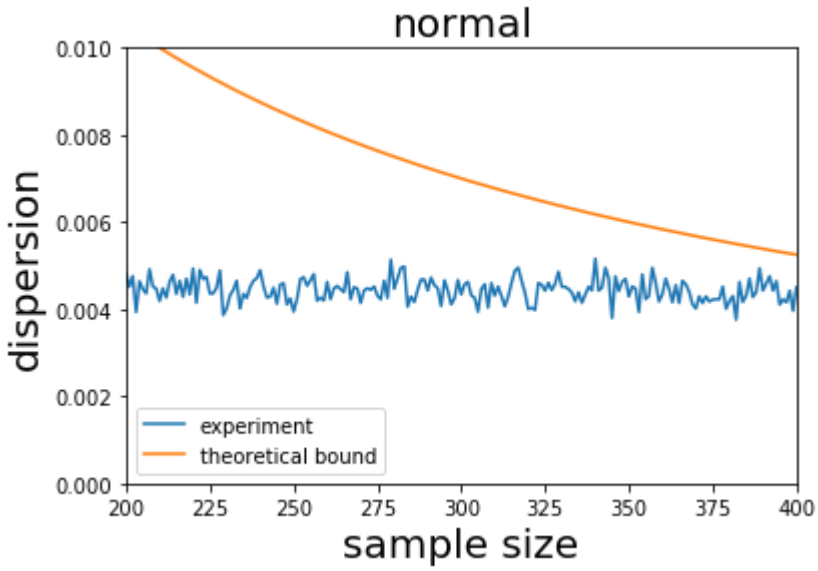

binomial
experiment
theoretical bound
dispersion
sample size
0.000200
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0.000000
0
25
50
75
100
125
150
175
200

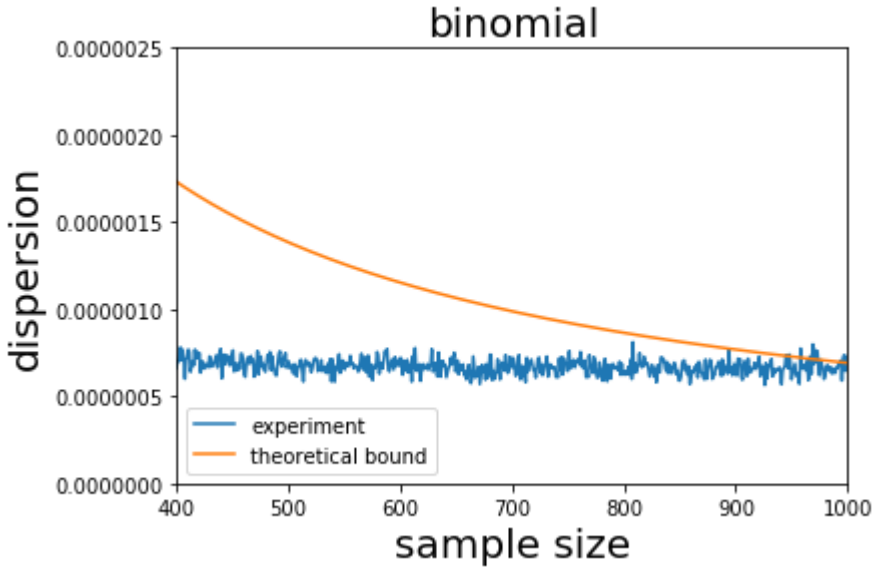

exponential
experiment
theoretical bound
dispersion
sample size
5
4
3
2
1
0
0
25
50
75
100
125
150
175
200

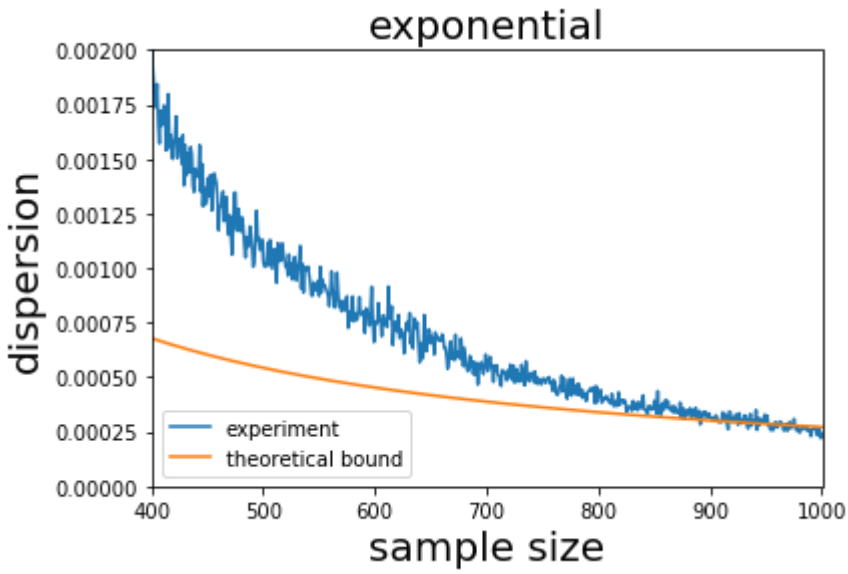

normal
experiment
theoretical bound
dispersion
sample size
0.10
0.08
0.06
0.04
0.02
0.00
0
25
50
75
100
125
150
175
200

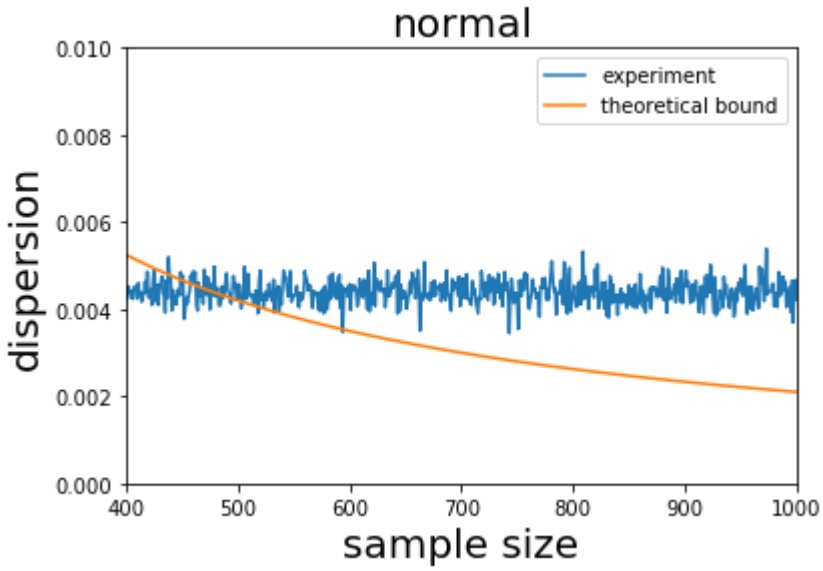

binomial
experiment
theoretical bound
dispersion
sample size
0.000010
0.000008
0.000006
0.000004
0.000002
0.000000
200
225
250
275
300
325
350
375
400


exponential
experiment
theoretical bound
dispersion
sample size
0.010
0.008
0.006
0.004
0.002
0.000
200
225
250
275
300
325
350
375
400


normal
experiment
theoretical bound
dispersion
sample size
0.010
0.008
0.006
0.004
0.002
0.000
200
225
250
275
300
325
350
375
400







Другие оценки

In [15]:

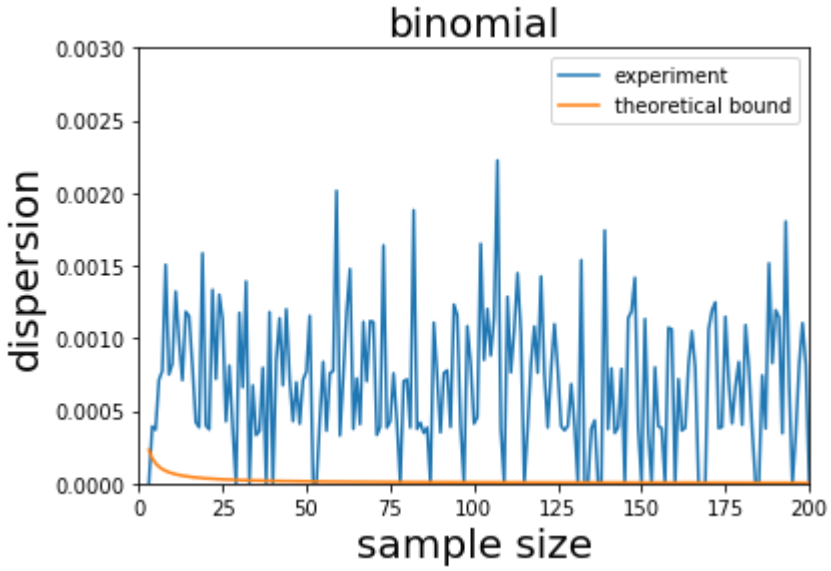
```
def est1(sample):
    return sample[-1]/50.0

def est2(sample):
    return (len(sample)-1)/np.mean(sample)

def est3(sample):
    return np.median(sample)

estimations = [est1, est2, est3]
```

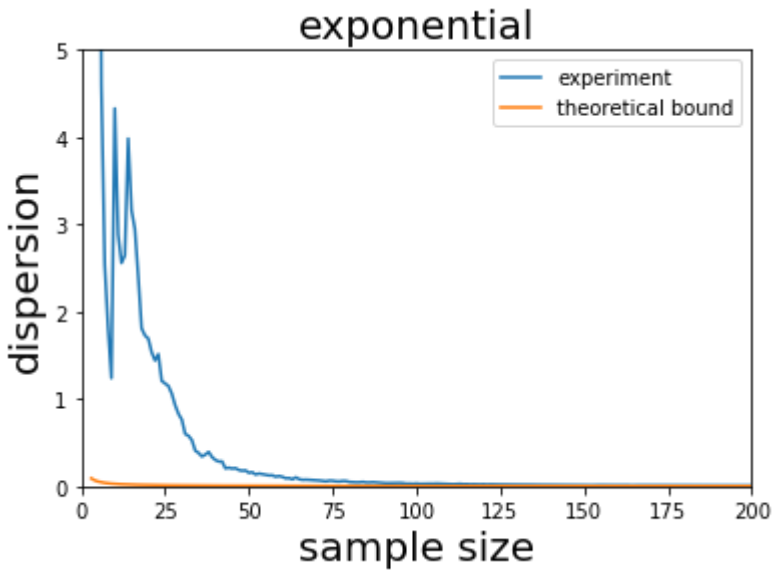
In [16]:

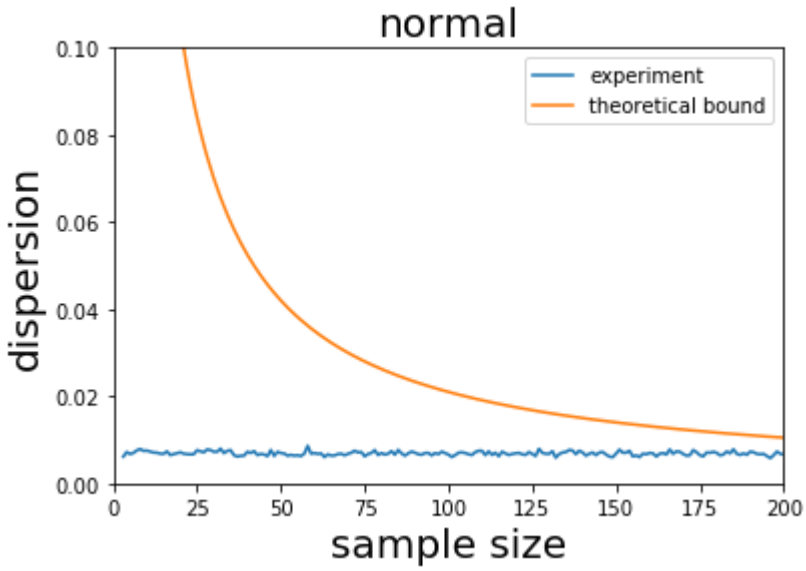
```
bootstrap_disp = [[] for i in range(len(distributions))]
for i in range(N)[2:]:
    for j in range(len(distributions)):
        estimation = estimations[j](samples[j][:i])
        bootstrap_est = np.array([estimations[j](distributions[j](estimation)) f
or k in range(500)])
        bootstrap_disp[j].append(np.var(bootstrap_est))
```

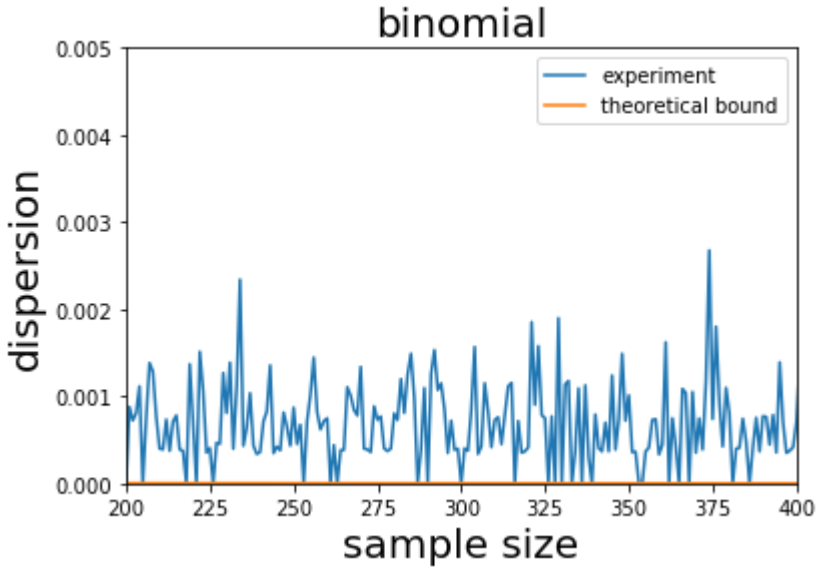
In [27]:

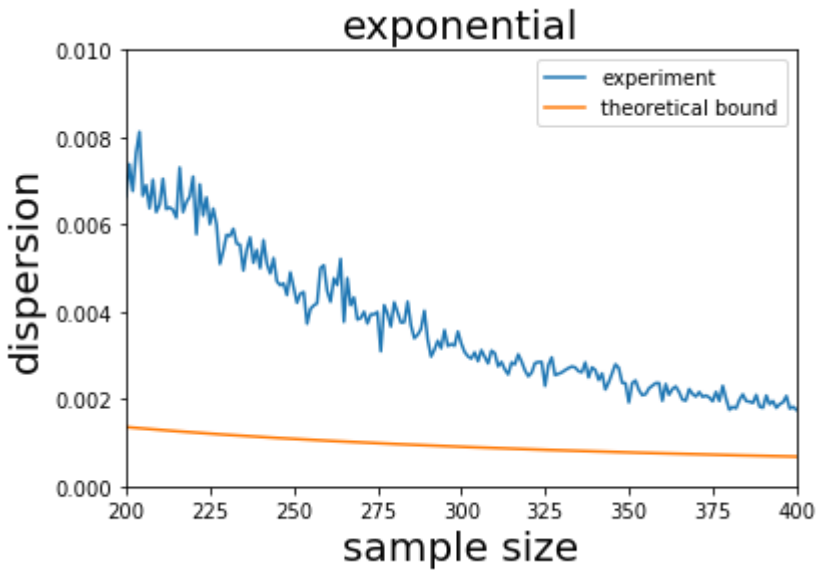
```
plot_disp([[0, 200, 0, 0.003], [0, 200, 0, 5], [0, 200, 0, 0.1]])
plot_disp([[200, 400, 0, 0.005], [200, 400, 0, 0.01], [200, 400, 0, 0.01]])
plot_disp([[400, 1000, 0, 0.004], [400, 1000, 0, 0.002], [400, 1000, 0, 0.01]])
```

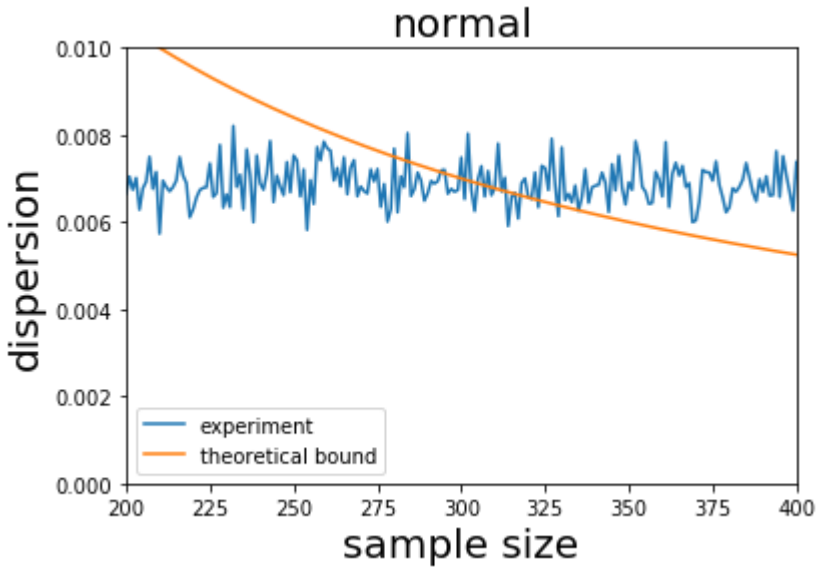

binomial
experiment
theoretical bound
dispersion
sample size
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
25
50
75
100
125
150
175
200

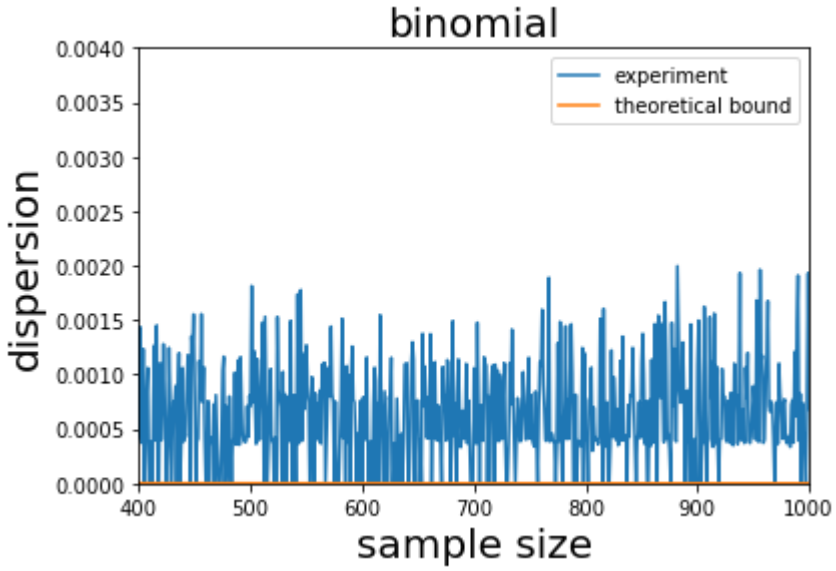

exponential
experiment
theoretical bound
dispersion
sample size
5
4
3
2
1
0
0
25
50
75
100
125
150
175
200

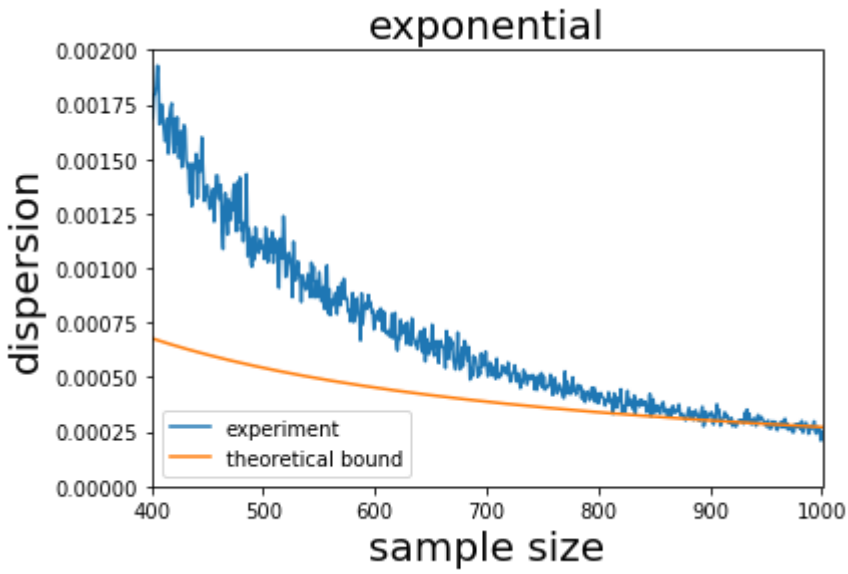

normal
experiment
theoretical bound
dispersion
sample size
0.10
0.08
0.06
0.04
0.02
0.00
0
25
50
75
100
125
150
175
200

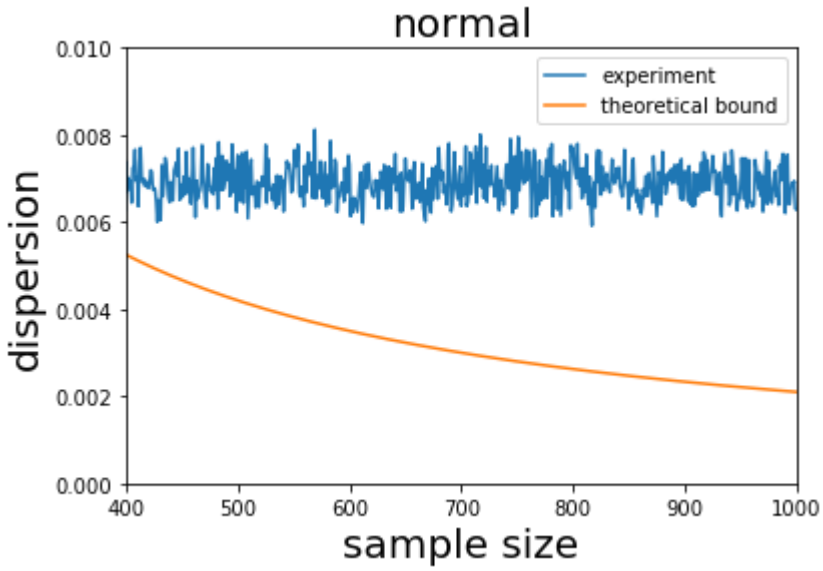

binomial
experiment
theoretical bound
dispersion
sample size
0.005
0.004
0.003
0.002
0.001
0.000
200
225
250
275
300
325
350
375
400


exponential
experiment
theoretical bound
dispersion
sample size
0.010
0.008
0.006
0.004
0.002
0.000
200
225
250
275
300
325
350
375
400


normal
experiment
theoretical bound
dispersion
sample size
0.010
0.008
0.006
0.004
0.002
0.000
200
225
250
275
300
325
350
375
400


binomial
experiment
theoretical bound
dispersion
sample size
0.0040
0.0035
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
400
500
600
700
800
900
1000


exponential
experiment
theoretical bound
dispersion
sample size
0.00200
0.00175
0.00150
0.00125
0.00100
0.00075
0.00050
0.00025
0.00000
400
500
600
700
800
900
1000


normal
experiment
theoretical bound
dispersion
sample size
0.010
0.008
0.006
0.004
0.002
0.000
400
500
600
700
800
900
1000

In [ ]: