Bootstrap demo for seminar 2

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

For bootstrap

In [2]:
```
from numpy.random import choice
```

Example 1:

Here is an example that was one of the first used to illustrate the bootstrap by Bradley Efron, the inventor of the bootstrap. The data are LSAT scores (for entrance to law school) and GPA.

In [3]:
```
LSATs = np.array([576, 635, 558, 578, 666, 580, 555, 661, 651, 605, 653, 575, 545, 572, 594])
GPAs = np.array([3.39, 3.30, 2.81, 3.03, 3.44, 3.07, 3.00, 3.43, 3.36, 3.13, 3.12, 2.74, 2.76, 2.88, 3.96])

data = np.stack([LSATs, GPAs]).T
```

Each data point is of the form $X_i = (Y_i, Z_i)$ where $Y_i = \mathrm{LSAT}_i$ and $Z_i = \mathrm{GPA}_i$. The law school is interested in the **correlation.**

The plug-in estimate is the sample correlation:

$$\hat{\theta} = \frac{\sum_i \left(Y_i - \bar{Y}\right)\left(Z_i - \bar{Z}\right)}{\sqrt{\sum_i \left(Y_i - \bar{Y}\right)^2 \sum_i \left(Z_i - \bar{Z}\right)^2}}$$

Let's compute it:

In [53]:
```
len(LSATs)
```

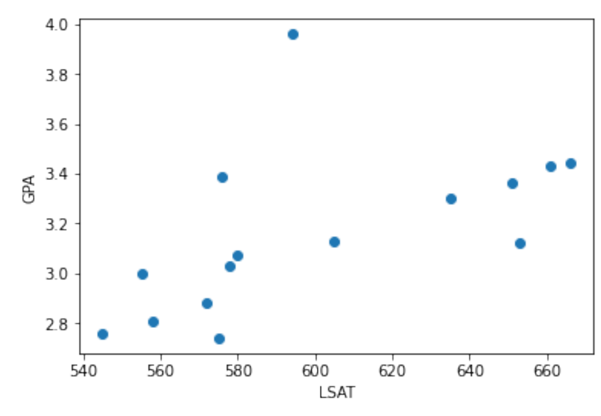
Out[53]:
```
15
```

In [4]:
```
scipy.stats.pearsonr(GPAs,LSATs)
```

Out[4]:
```
(0.5459189161795885, 0.03527161512732669)
```

In [5]:
```
plt.scatter(LSATs,GPAs)
plt.xlabel('LSAT')
plt.ylabel('GPA')
```

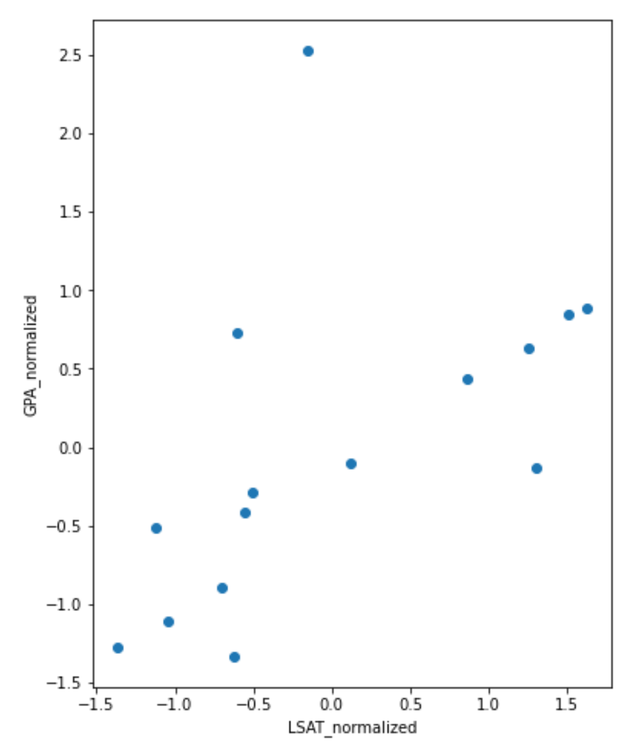
Out[5]:
```
Text(0, 0.5, 'GPA')
```



Let us **standardize** the data to plot it in equal aspect ratio (this does not change the correlation value!)

In [6]:
```
data_normalized = (data - data.mean(axis=0))/data.std(axis=0)

fig = plt.figure(figsize=(8, 8))
ax = fig.add_subplot(111)
ax.scatter(data_normalized[:,0],data_normalized[:,1])
ax.set_aspect('equal')
ax.set_xlabel('LSAT_normalized')
ax.set_ylabel('GPA_normalized')
```

Out[6]:
```
Text(0, 0.5, 'GPA_normalized')
```

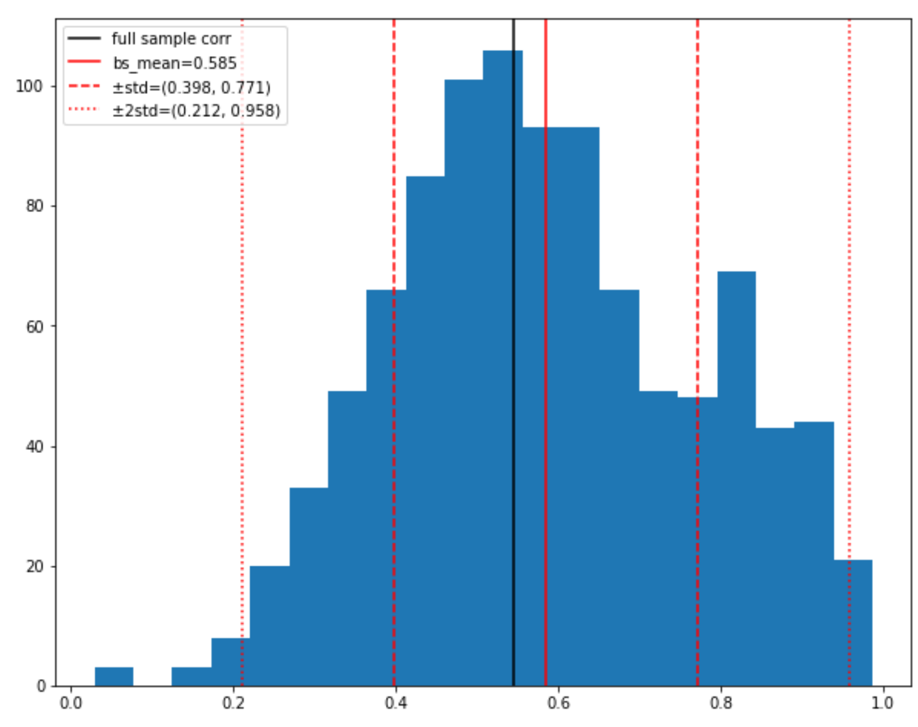


Now let's do bootstrap!

In [7]:
```
thetas = []

for i in range(1000):
    bs_sample = data[choice(range(15),15),:]
    thetas.append(scipy.stats.pearsonr(bs_sample[:,0],bs_sample[:,1])[0])
```

In [8]:
```
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111)

ax.hist(thetas,bins=20)

ax.axvline(scipy.stats.pearsonr(GPAs,LSATs)[0],color='black',label='full sample corr')

bs_mean = np.mean(thetas)
ax.axvline(bs_mean,color='red',label='bs_mean='+str(round(bs_mean,3)))

bs_std = np.std(thetas)

# ± std
pm_std = (round(bs_mean-bs_std,3),round(bs_mean+bs_std,3))
ax.axvline(pm_std[0],color='red',linestyle='--',label='±std='+str(pm_std))
ax.axvline(pm_std[1],color='red',linestyle='--')

# ± 2stds
pm_2std = (round(bs_mean-2*bs_std,3),round(bs_mean+2*bs_std,3))
ax.axvline(bs_mean+2*bs_std,color='red',linestyle=':',label='±2std='+str(pm_2std))
ax.axvline(bs_mean-2*bs_std,color='red',linestyle=':')

ax.legend(loc='upper left')
```

Out[8]:
```
<matplotlib.legend.Legend at 0x7f8ac9098f40>
```



Example 2

Let $X_1, \ldots, X_n \sim \mathrm{Uniform}(0, \theta)$. Let $\hat{\theta} = X_{\max} = \max(X_1, \ldots, X_n)$. Generate a dataset of size 50 with $\theta = 1$.

1. Find the distribution of $\hat{\theta}$. Compare the true distribution of $\hat{\theta}$ to the histograms from the bootstrap.
2. This is a case where the bootstrap does poorly. Why?

Comment:

The true distribution (CDF) of $\hat{\theta}$ is

$$F(x) = \begin{cases} 0, & \text{if } x \le 0 \\ \left(\frac{x}{\theta}\right)^n, & \text{if } 0 < x \le \theta \\ 1, & \text{if } x > \theta \end{cases}$$

So, in our case, $F(x) = x^{50}$ from 0 to 1.

Thus the PDF is $p(x) = F'(x) = 50x^{49}$ from 0 to 1.

Let's gain some intuition:

$p(0.8) =$

In [9]:
```
50*(0.8**49)
```

Out[9]:
```
0.0008920298079412274
```

$p(0.9) =$

In [10]:
```
50*(0.9**49)
```

Out[10]:
```
0.28632084485111775
```

$p(0.95) =$

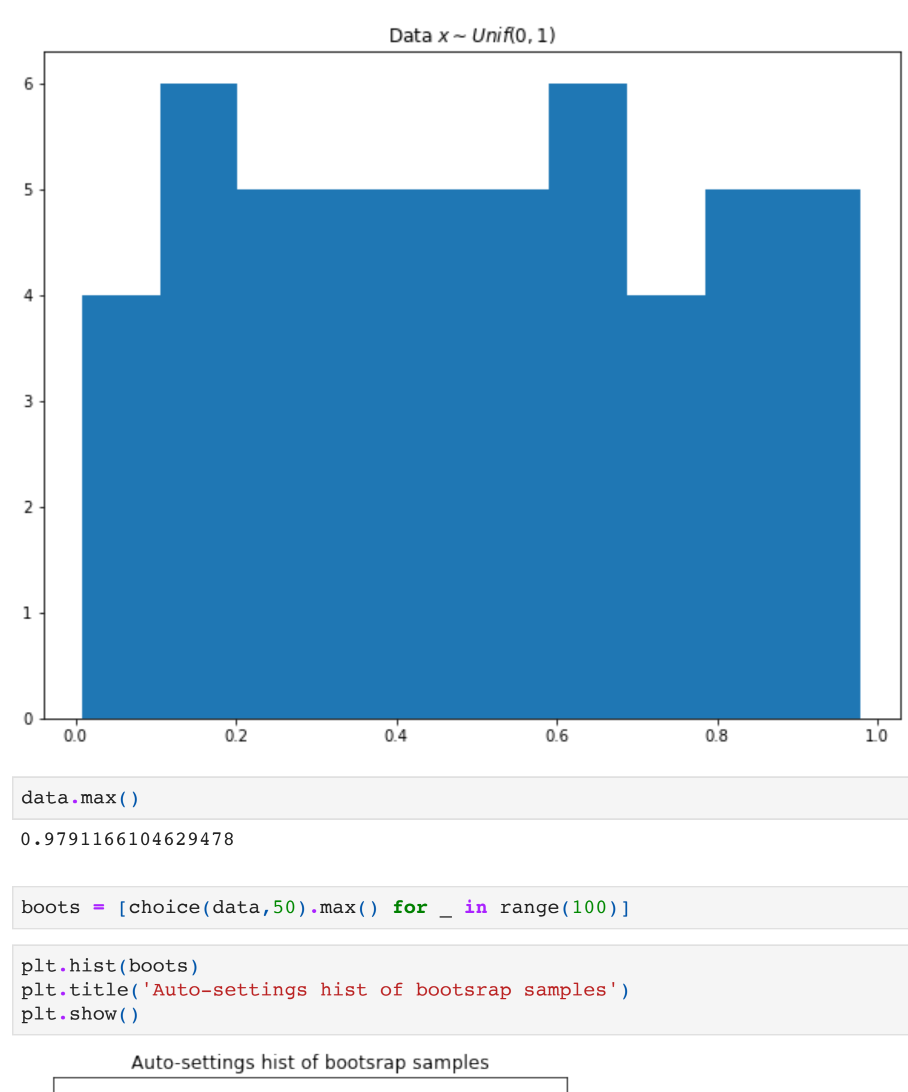
In [11]:
```
50*(0.95**49)
```

Out[11]:
```
4.04973554087964
```

In [35]:
```
data = np.random.uniform(low=0,high=1,size=50)
```

In [36]:
```
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111)
ax.hist(data)
ax.set_title('Data $x\sim Unif(0,1)$')
```

Out[36]:
```
Text(0.5, 1.0, 'Data $x\\sim Unif(0,1)$')
```

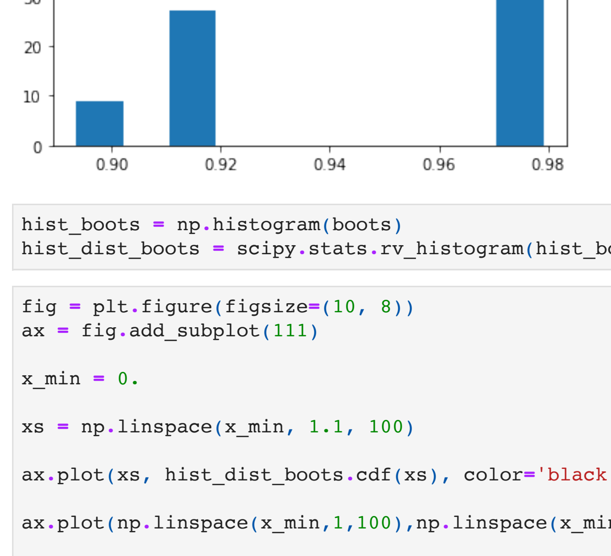


In [37]:
```
data.max()
```

Out[37]:
```
0.9791166104629478
```

In [38]:
```
boots = [choice(data,50).max() for _ in range(100)]
```

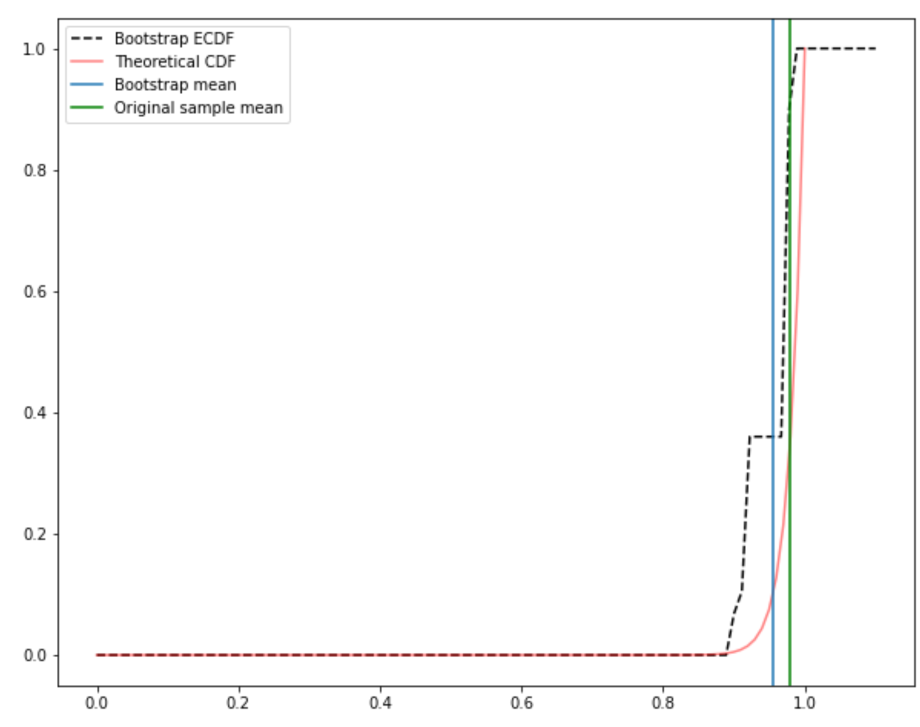
In [39]:
```
plt.hist(boots)
plt.title('Auto-settings hist of bootsrap samples')
plt.show()
```



In [40]:
```
hist_boots = np.histogram(boots)
hist_dist_boots = scipy.stats.rv_histogram(hist_boots)
```

In [56]:
```
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111)

x_min = 0.

xs = np.linspace(x_min, 1.1, 100)

ax.plot(xs, hist_dist_boots.cdf(xs), color='black', linestyle='--',label='Bootstrap ECDF')

ax.plot(np.linspace(x_min,1,100),np.linspace(x_min,1,100)**50,color='red',alpha=0.5,label='Theoretical CDF')

ax.axvline(np.mean(boots),label='Bootstrap mean')

ax.axvline(np.max(data),color='green',label='Original sample mean')

ax.legend()
```

Out[56]:
```
<matplotlib.legend.Legend at 0x7f8ad9950d00>
```



In [42]:
```
max(boots)
```

Out[42]:
```
0.9791166104629478
```

Why does bootstrap do not-so-well here?

$$P(\hat{\theta}^* = \hat{\theta}) = 1 - \left(1 - \frac{1}{n}\right)^n \underset{n\to\infty}{\rightarrow} 1 - 1/e$$

In [43]:
```
1-1/np.exp(1)
```

Out[43]:
```
0.6321205588285577
```

In [ ]: