# Computer Exercise 8: Bootstrap

In [ ]:
```
# Plotting
import plotly.graph_objects as go
import plotly.express as px
import plotly.subplots as sp
import plotly.io as pio
pio.renderers.default = "notebook+pdf"
pio.templates.default = "plotly_dark"

# Utilities
import numpy as np
```

## Part 1 - Exercise 13 in Chapter 8 of Ross

### (a) How to Estimate $p$ using Bootstrap

Given the data, we sample $n$ samples with replacement. Using these samples, we calculate deviation from the mean as seen in the exercise. We then repeat the sampling and calculation $r$ times. Using the $r$ values of the deviation we can estimate the probability that the deviation falls between a and b.

### (b) Estimating $p$ for an Example

Now we are given an array of data and use the above mentioned method to estimate the probability that $\sum_{i=1}^{n} X_i/n - \mu$ falls between $a$ and $b$.

In [ ]:
```
n1 = 10
data1 = np.array([56, 101, 78, 67, 93, 87, 64, 72, 80, 69])
a = -5
b = 5

# Number of bootstrap samples
r = 100000

# Bootstrap
samples1 = np.random.choice(data1, (r, n1))
means1 = np.mean(samples1, axis=1)
mu1 = np.mean(means1)
deviations1 = means1 - mu1

# Compute the probability
count = np.histogram(deviations1, [a, b])[0][0]
prob = count/r
print("p =", prob)
```

```
p = 0.76774
```

We found that the probability is approximately 0.76.

## Part 2 - Exercise 15 in Chapter 8 of Ross

In this exercise, we are given a data set and we are asked to estimate the variance of the data set. We use the bootstrap method to estimate the variance of the data set.

In [ ]:
```
n2 = 15
data2 = np.array([5, 4, 9, 6, 21, 17, 11, 20, 7, 10, 21, 15, 13, 16, 8])

# Bootstrap
samples2 = np.random.choice(data2, (r, n2))
means2 = np.mean(samples2, axis=1)
deviations2 = means2 - np.mean(means2)

mu2 = np.mean(means2)
res = (np.sum((samples2 - mu2)**2, axis=1))/(n2-1)

var = np.var(res)

print(f"s^2 = {var:.2f}")
```

```
s^2 = 55.90
```

We found the variance of the data set to be approximately 55.9.

# Part 3 - Median Bootstrap Variance

In this exercise we wish to define a bootstrad method to estimate the median of a dataset and the variance of the median. We will use samples from the pareto distribution as our data set.

## (a, b, c) Bootstrap Statistics

In [ ]:
```
def bootstrap(data, r):
    n = len(data)

    sample = np.random.choice(data, (r, n))
    medians = np.median(sample, axis=1)
    means = np.mean(sample, axis=1)

    mu = np.mean(means)
    var_mu = np.var(means, ddof=1)
    median = np.median(medians)
    var_median = np.var(medians, ddof=1)

    return mu, var_mu, median, var_median

N = 200
r = 100

beta = 1
k = 1.05

pareto = beta*(np.random.uniform(0,1,N)**(-1/k)-1)

# bootstrap
mu, var_mu, median, var_median = bootstrap(pareto, r)

print(f"μ = {mu:.2f}\nvar(μ) = {var_mu:.2f}\nmedian = {median:.2f}\nvar(median) = {var_median:.2f}")
```

```
μ = 8.81
var(μ) = 20.16
median = 0.79
var(median) = 0.02
```

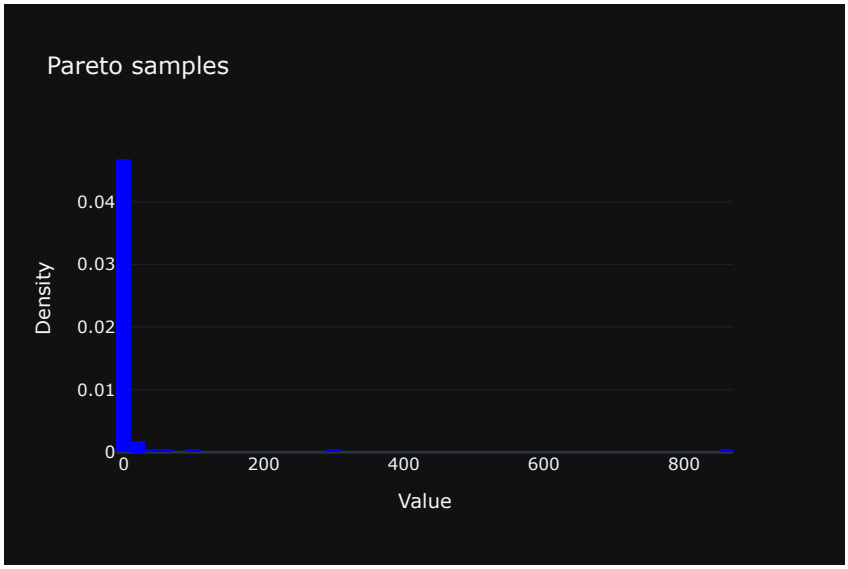
After having implemented a function for the bootstrap method, we generate 200 samples from the pareto distribution and calculate the median and variance of the median using the bootstrap method.

We found the median of the data set to be approximately 0.88 and the variance of the median to be approximately 0.01. So we are quite confident in the value of the median.

## (d) Precision

The pareto distribution does not have a mean when $k = 1$. Consequently, the bootstrap estimate becomes increasingly unstable when $k$ approaches 1. This is not the case for the median, which is well defined for all values of $k$. The explains why the estimate for the median is much more stable than that of the mean when $k = 1.05$.

In [ ]:
```
# Plot pareto
fig = go.Figure(go.Histogram(x=pareto, histnorm='probability density', marker=dict(color='Blue')))
fig.update_layout(title="Pareto samples", xaxis_title="Value", yaxis_title="Density", width=600, height=400)
fig.show()
```


Pareto samples
Density
0.04
0.03
0.02
0.01
0
0
200
400
600
800
Value