```
[7]  import pandas as pd
     import math
     import matplotlib.pyplot as plt

     import numpy as np
     import scipy.stats as ss
```

A

Risk and return are two fundamental concepts for evaluating securities. The expected return gives a measure of the (long-run) average return for the security, and the standard deviation of the return provides a measure of the "riskiness" of the security. In general, we find that the higher the expected return for a security the greater the standard deviation for that security.

The same concepts are useful for evaluating portfolios of securities. To better understand the effects of diversification, consider the following questions:

Suppose you are considering a portfolio of two securities: Microsoft stock and Intel stock. Let r1be the return on Microsoft stock per USD invested and r2 be the return on Intel stock per USDinvested. Using historical annual returns, you find that for r1 the mean is 1.1 and the standard deviation is 0.1. Similarly, r2 has a mean of 1.1 and a standard deviation of 0.1 (identical to the return on Microsoft stock).

```
[4]  .1**2 * 5000**2
```

```
250000.00000000006
```

```
[5]  math.sqrt(.1**2 * 5000**2)
```

```
500.00000000000006
```

```
[6]  u = -5000 + 5000*1.1
     print(u)
```

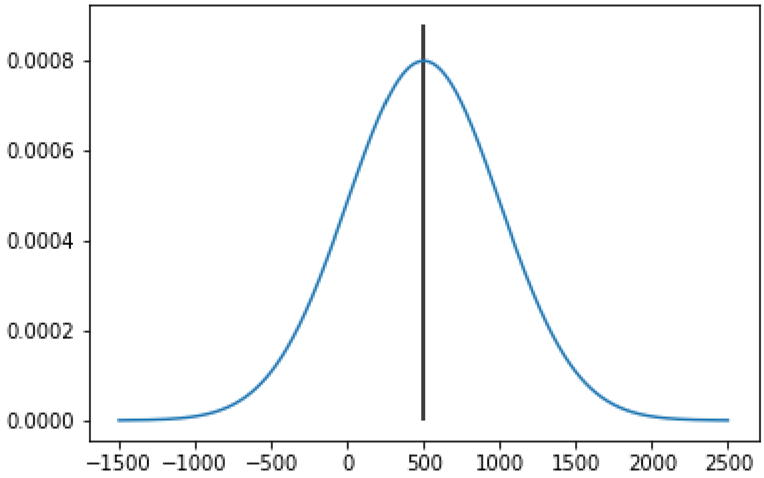
```
500.0
```

```
[8]  ss.norm(500, 500)
```

```
<scipy.stats._distn_infrastructure.rv_frozen at 0x7fa819481e48>
```

```
[20]  rvs = ss.norm(500, 500)
      xvals = np.linspace(-1500, 2500, 3000)
      plt.plot(xvals, rvs.pdf(xvals))
      plt.vlines(500, 0, 1.1*max(rvs.pdf(xvals)))
```

```
<matplotlib.collections.LineCollection at 0x7fa81886b4e0>
```



B

You decide to invest USD 2,500 in Microsoft stock and USD 2,500 in Intel stock, assuming that the correlation between r1 and r2 is 0 (the return on the two stocks are statistically independent). Note that the net profit here would be: P = 2500r1 + 2500r2-5000.

```
[21]  b_msft = 2500
      c_intc = 2500
      corr = 0

      u = -5000 + 1.1*b_msft + 1.1*c_intc
      print(u)
```

```
500.0
```

$$Var(W) = b_m^2 Var(X) + c_i^2(Y) + 2 * b_m * c_i * Corr(X, Y)$$

[22]
```
b_msft**2 * 0.1**2 + c_intc**2 * 0.1**2
```

```
125000.00000000003
```

[23]
```
math.sqrt(b_msft**2 * 0.1**2 + c_intc**2 * 0.1**2)
```

```
353.5533905932738
```

C

You decide to invest USD 2,500 in Microsoft stock and USD 2,500 in Intel stock, assuming that the correlation between r1 and r2 is 0.6 (the returns on the two stocks are positively correlated, i.e., increases in the return on one stock tend to go together with increases in the other stock).

[30]
```
corr = 0.6
# everything else is the same
cov = corr * 0.1 * 0.1
cov
```

```
0.006
```

[32]
```
(b_msft**2 * 0.1**2) + (c_intc**2 * 0.1**2) +
(2*b_msft*c_intc*cov)
```

```
200000.00000000003
```

[31]
```
math.sqrt(b_msft**2 * 0.1**2 + c_intc**2 * 0.1**2 +
2*b_msft*c_intc*cov)
```

```
447.213595499958
```

D

You decide to invest USD 2,500 in Microsoft stock and USD 2,500 in Intel stock, assuming that the correlation between r1 and r2 is -0.6 (the returns on the two stocks are negatively correlated, i.e., increases in the return on one stock tend to go together with decreases in the other stock).

```
[33]
corr = -0.6
# everything else the same
cov = corr * 0.1 * 0.1
cov
```

```
-0.006
```

```
[34]
(b_msft**2 * 0.1**2) + (c_intc**2 * 0.1**2) +
(2*b_msft*c_intc*cov)
```

```
50000.00000000003
```

```
[35]
math.sqrt(b_msft**2 * 0.1**2 + c_intc**2 * 0.1**2 +
2*b_msft*c_intc*cov)
```

```
223.60679774997902
```

E

You decide to invest USD 2,500 in Microsoft stock and USD 2,500 in Intel stock, assuming that the correlation between r1 and r2 is 1 (the returns on the two stocks are perfectly and positively correlated, i.e., increases in the return on one stock are matched by proportional increases in the return on the other stock).

[36]
```
corr = 1
cov = corr * 0.1 * 0.1
cov
```

```
0.010000000000000002
```

[37]
```
(b_msft**2 * 0.1**2) + (c_intc**2 * 0.1**2) +
(2*b_msft*c_intc*cov)
```

```
250000.00000000006
```

[38]
```
math.sqrt(b_msft**2 * 0.1**2 + c_intc**2 * 0.1**2 +
2*b_msft*c_intc*cov)
```

```
500.00000000000006
```

F

You decide to invest USD 2,500 in Microsoft stock and USD 2,500 in Intel stock, assuming that the correlation between r1 and r2 is -1 (the returns on the two stocks are perfectly and negatively correlated, i.e., increases in the return on one stock are matched by proportional decreases in the return on the other stock).

[39]
```
corr = -1
cov = corr * 0.1 * 0.1
cov
```

```
-0.010000000000000002
```

[40]
```
(b_msft**2 * 0.1**2) + (c_intc**2 * 0.1**2) +
(2*b_msft*c_intc*cov)
```

```
0.0
```

[41]
```
math.sqrt(b_msft**2 * 0.1**2 + c_intc**2 * 0.1**2 +
2*b_msft*c_intc*cov)
```

```
0.0
```

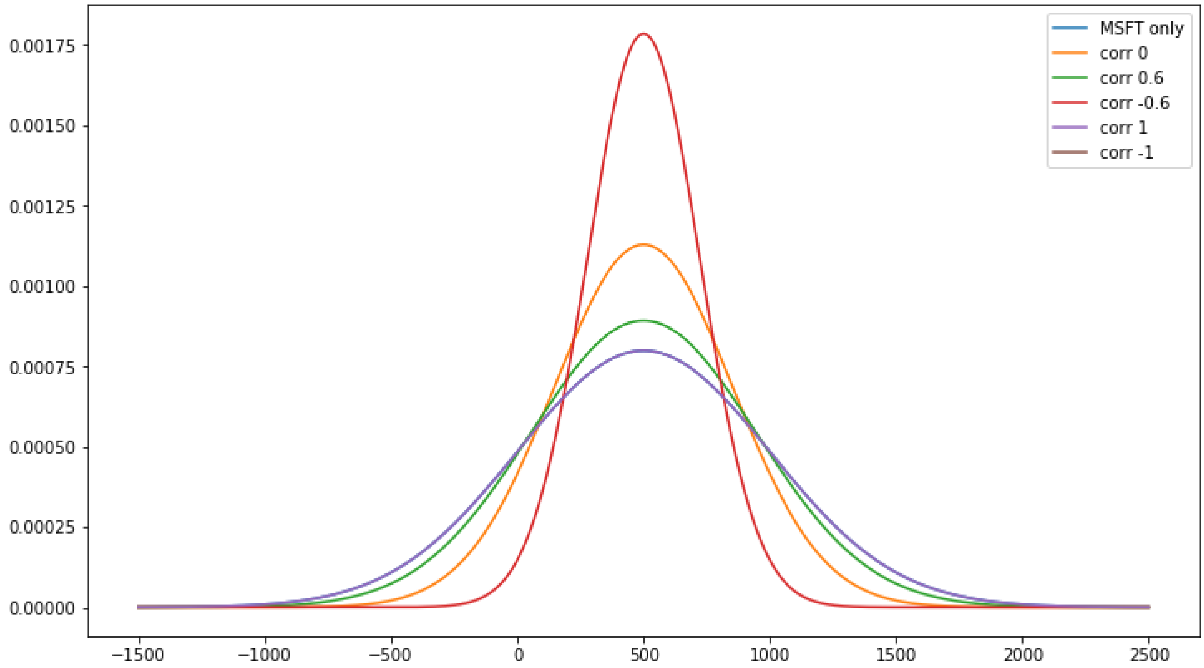
The comparison histograms

[43]
```
sd_a = 500
sd_b = 353.5533905932738
sd_c = 447.213595499958
sd_d = 223.60679774997902
sd_e = 500.00000000000006
sd_f = 0
```

[44]
```
u = 500
```

[52]
```
plt.figure(figsize=(12,7))
plt.plot(xvals, ss.norm(500, sd_a).pdf(xvals), label="MSFT only")
plt.plot(xvals, ss.norm(500, sd_b).pdf(xvals), label="corr 0")
plt.plot(xvals, ss.norm(500, sd_c).pdf(xvals), label="corr 0.6")
plt.plot(xvals, ss.norm(500, sd_d).pdf(xvals), label="corr -0.6")
plt.plot(xvals, ss.norm(500, sd_e).pdf(xvals), label="corr 1")
plt.plot(xvals, ss.norm(500, sd_f).pdf(xvals), label="corr -1")
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7fa817dd2e10>
```



What general tips would you suggest for constructing a portfolio of securities? Is it possible to construct a portfolio with zero standard deviation? If so, how? If not, why not?

It is important to create a sufficiently diversified portfolio of contrary-movement stocks, since doing so will help you maintain a more predictable expected return as a result of reduced volatility. However, it is not possible to create a perfectly 0 standard deviation portfolio since asset classes do not move perfectly with negative correlation and even if you were able to find these asset classes at any given point in time, it is very unlikely that their volatility and hence covariance will remain perfectly -1 over time

```
[ ]
```