Variance of one stock

In [1]:
```
import numpy as np
import pandas as pd
import time
import os
import quiz_helper
import matplotlib.pyplot as plt
```

In [2]:
```
%matplotlib inline
plt.style.use('ggplot')
plt.rcParams['figure.figsize'] = (14, 8)
```

data bundle

In [3]:
```
import os
import quiz_helper
from zipline.data import bundles
```

In [4]:
```
os.environ['ZIPLINE_ROOT'] = os.path.join(os.getcwd(), '..', '..','data','module_4_quizzes_eod')
ingest_func = bundles.csvdir.csvdir_equities(['daily'], quiz_helper.EOD_BUNDLE_NAME)
bundles.register(quiz_helper.EOD_BUNDLE_NAME, ingest_func)
print('Data Registered')
```

```
Data Registered
```

Build pipeline engine

In [5]:
```
from zipline.pipeline import Pipeline
from zipline.pipeline.factors import AverageDollarVolume
from zipline.utils.calendars import get_calendar

universe = AverageDollarVolume(window_length=120).top(500)
trading_calendar = get_calendar('NYSE')
bundle_data = bundles.load(quiz_helper.EOD_BUNDLE_NAME)
engine = quiz_helper.build_pipeline_engine(bundle_data, trading_calendar)
```

View Data¶

With the pipeline engine built, let's get the stocks at the end of the period in the universe we're using. We'll use these tickers to generate the returns data for the our risk model.

In [7]:
```
universe_end_date = pd.Timestamp('2016-01-05', tz='UTC')

universe_tickers = engine\
    .run_pipeline(
        Pipeline(screen=universe),
        universe_end_date,
        universe_end_date)\
    .index.get_level_values(1)\
    .values.tolist()

universe_tickers[1:10]
```

Out[7]:
```
[Equity(1 [AAL]),
 Equity(2 [AAP]),
 Equity(3 [AAPL]),
 Equity(4 [ABBV]),
 Equity(5 [ABC]),
 Equity(6 [ABT]),
 Equity(7 [ACN]),
 Equity(8 [ADBE]),
 Equity(9 [ADI])]
```

In [8]:
```
len(universe_tickers)
```

Out[8]:
```
490
```

In [9]:
```
from zipline.data.data_portal import DataPortal

data_portal = DataPortal(
    bundle_data.asset_finder,
    trading_calendar=trading_calendar,
    first_trading_day=bundle_data.equity_daily_bar_reader.first_trading_day,
    equity_minute_reader=None,
    equity_daily_reader=bundle_data.equity_daily_bar_reader,
    adjustment_reader=bundle_data.adjustment_reader)
```

Get pricing data helper function

In [10]:
```
from quiz_helper import get_pricing
```

get pricing data into a dataframe

In [12]:
```
returns_df = \
    get_pricing(
        data_portal,
        trading_calendar,
        universe_tickers,
        universe_end_date - pd.DateOffset(years=5),
        universe_end_date)\
    .pct_change()[1:].fillna(0) #convert prices into returns

returns_df.head()
```

Out[12]:

	Equity(0 [A])	Equity(1 [AAL])	Equity(2 [AAP])	Equity(3 [AAPL])	Equity(4 [ABBV])	Equity(5 [ABC])	Equity(6 [ABT])	Equity(7 [ACN])	Equity(8 [ADBE])	Equity(9 [ADI])	...	Equity(481 [XL])	Equity(482 [XLNX])	Equity [X
2011-01-07 00:00:00+00:00	0.008437	0.014230	0.026702	0.007146	0.0	0.001994	0.004165	0.001648	-0.007127	-0.005818	...	-0.001838	-0.005619	0.005
2011-01-10 00:00:00+00:00	-0.004174	0.006195	0.007435	0.018852	0.0	-0.005714	-0.008896	-0.008854	0.028714	0.002926	...	0.000947	0.007814	-0.00
2011-01-11 00:00:00+00:00	-0.001886	-0.043644	-0.005927	-0.002367	0.0	0.009783	-0.002067	0.013717	0.000607	0.008753	...	0.001314	0.010179	0.00
2011-01-12 00:00:00+00:00	0.017254	-0.008237	0.013387	0.008133	0.0	-0.005979	-0.001011	0.022969	0.017950	0.000257	...	0.004986	0.015666	0.01
2011-01-13 00:00:00+00:00	-0.004559	0.000955	0.003031	0.003657	0.0	0.014925	-0.004451	-0.000400	-0.005719	-0.005012	...	0.030499	-0.003217	0.00

5 rows × 490 columns

Let's look at a two stock portfolio

Let's pretend we have a portfolio of two stocks. We'll pick Apple and Microsoft in this example.

In [13]:
```
aapl_col = returns_df.columns[3]
msft_col = returns_df.columns[312]
asset_return_1 = returns_df[aapl_col].rename('asset_return_aapl')
asset_return_2 = returns_df[msft_col].rename('asset_return_msft')
asset_return_df = pd.concat([asset_return_1,asset_return_2],axis=1)
asset_return_df.head(2)
```

Out[13]:

	asset_return_aapl	asset_return_msft
2011-01-07 00:00:00+00:00	0.007146	-0.007597
2011-01-10 00:00:00+00:00	0.018852	-0.013311

Factor returns

Let's make up a "factor" by taking an average of all stocks in our list. You can think of this as an equal weighted index of the 490 stocks, kind of like a measure of the "market". We'll also make another factor by calculating the median of all the stocks. These are mainly intended to help us generate some data to work with. We'll go into how some common risk factors are generated later in the lessons.

Also note that we're setting axis=1 so that we calculate a value for each time period (row) instead of one value for each column (assets).

In [14]:
```
factor_return_1 = returns_df.mean(axis=1)
factor_return_2 = returns_df.median(axis=1)
factor_return_l = [factor_return_1, factor_return_2]
```

Factor exposures

Factor exposures refer to how "exposed" a stock is to each factor. We'll get into this more later. For now, just think of this as one number for each stock, for each of the factors.

In [15]:
```
from sklearn.linear_model import LinearRegression
```

In [16]:
```
"""
For now, just assume that we're calculating a number for each
stock, for each factor, which represents how "exposed" each stock is
to each factor.
We'll discuss how factor exposure is calculated later in the lessons.
"""
def get_factor_exposures(factor_return_l, asset_return):
    lr = LinearRegression()
    X = np.array(factor_return_l).T
    y = np.array(asset_return.values)
    lr.fit(X,y)
    return lr.coef_
```

In [17]:
```
factor_exposure_l = []
for i in range(len(asset_return_df.columns)):
    factor_exposure_l.append(
        get_factor_exposures(factor_return_l,
                             asset_return_df[asset_return_df.columns[i]]
                            ))

factor_exposure_a = np.array(factor_exposure_l)
```

In [18]:
```
print(f"factor_exposures for asset 1 {factor_exposure_a[0]}")
print(f"factor_exposures for asset 2 {factor_exposure_a[1]}")
```

```
factor_exposures for asset 1 [ 1.35101534 -0.58353198]
factor_exposures for asset 2 [-0.2283345   1.16364007]
```

Quiz: Variance of stock 1

Calculate the variance of stock 1.
$\text{Var}(f_1) = \beta_{1,1}^2 \text{Var}(f_1) + \beta_{1,2}^2 \text{Var}(f_2) + 2\beta_{1,1}\beta_{1,2}\text{Cov}(f_1, f_2) + \text{Var}(s_1)$

In [19]:
```
factor_exposure_1_1 = factor_exposure_a[0][0]
factor_exposure_1_2 = factor_exposure_a[0][1]
common_return_1 = factor_exposure_1_1 * factor_return_1 + factor_exposure_1_2 * factor_return_2
specific_return_1 = asset_return_1 - common_return_1
```

In [21]:
```
covm_f1_f2 = np.cov(factor_return_1,factor_return_2,ddof=1) #this calculates a covariance matrix
# TODO: get the variance of each factor, and covariances from the covariance matrix covm_f1_f2
var_f1 = covm_f1_f2[0,0]
var_f2 = covm_f1_f2[1,1]
cov_f1_f2 = covm_f1_f2[0][1]

# TODO: calculate the specific variance.
var_s_1 = np.var(specific_return_1,ddof=1)

# TODO: calculate the variance of asset 1 in terms of the factors and specific variance
var_asset_1 = (factor_exposure_1_1**2 * var_f1) + \
              (factor_exposure_1_2**2 * var_f2) + \
              2 * (factor_exposure_1_1 * factor_exposure_1_2 * cov_f1_f2) + \
              var_s_1
print(f"variance of asset 1: {var_asset_1:.8f}")
```

```
variance of asset 1: 0.00028209
```

Quiz 2: Variance of stock 2

Calculate the variance of stock 2.
$\text{Var}(f_2) = \beta_{2,1}^2 \text{Var}(f_1) + \beta_{2,2}^2 \text{Var}(f_2) + 2\beta_{2,1}\beta_{2,2}\text{Cov}(f_1, f_2) + \text{Var}(s_2)$

In [22]:
```
factor_exposure_2_1 = factor_exposure_a[1][0]
factor_exposure_2_2 = factor_exposure_a[1][1]
common_return_2 = factor_exposure_2_1 * factor_return_1 + factor_exposure_2_2 * factor_return_2
specific_return_2 = asset_return_2 - common_return_2
```

In [23]:
```
# Notice we already calculated the variance and covariances of the factors

# TODO: calculate the specific variance of asset 2
var_s_2 = np.var(specific_return_2,ddof=1)

# TODO: calcualte the variance of asset 2 in terms of the factors and specific variance
var_asset_2 = (factor_exposure_2_1**2 * var_f1) + \
              (factor_exposure_2_2**2 * var_f2) + \
              (2 * factor_exposure_2_1 * factor_exposure_2_2 * cov_f1_f2) + \
              var_s_2

print(f"variance of asset 2: {var_asset_2:.8f}")
```

```
variance of asset 2: 0.00021856
```

Quiz 3: Covariance of stocks 1 and 2

Calculate the covariance of stock 1 and 2.
$\text{Cov}(f_1, f_2) = \beta_{1,1}\beta_{2,1}\text{Var}(f_1) + \beta_{1,1}\beta_{2,2}\text{Cov}(f_1, f_2) + \beta_{1,2}\beta_{2,1}\text{Cov}(f_1, f_2) + \beta_{1,2}\beta_{2,2}\text{Var}(f_2)$

In [24]:
```
# TODO: calculate the covariance of assets 1 and 2 in terms of the factors
cov_asset_1_2 = (factor_exposure_1_1 * factor_exposure_2_1 * var_f1) + \
            (factor_exposure_1_1 * factor_exposure_2_2 * cov_f1_f2) + \
            (factor_exposure_1_2 * factor_exposure_2_1 * cov_f1_f2) + \
            (factor_exposure_1_2 * factor_exposure_2_2 * var_f2)
print(f"covariance of assets 1 and 2: {cov_asset_1_2:.8f}")
```

```
covariance of assets 1 and 2: 0.00007133
```

Quiz 4: Do it with Matrices!

Create matrices $\mathbf{F}$, $\mathbf{B}$ and $\mathbf{S}$, where

$\mathbf{F} = \begin{pmatrix} \text{Var}(f_1) & \text{Cov}(f_1, f_2) \\ \text{Cov}(f_2, f_1) & \text{Var}(f_2) \end{pmatrix}$ is the covariance matrix of factors,

$\mathbf{B} = \begin{pmatrix} \beta_{1,1}, \beta_{1,2} \\ \beta_{2,1}, \beta_{2,2} \end{pmatrix}$ is the matrix of factor exposures, and

$\mathbf{S} = \begin{pmatrix} \text{Var}(s_i) & 0 \\ 0 & \text{Var}(s_j) \end{pmatrix}$ is the matrix of specific variances.

Then calculate $\mathbf{BFB}^T$

In [25]:
```
# TODO: covariance matrix of factors
F = covm_f1_f2
F
```

Out[25]:
```
array([[1.02562520e-04, 9.79887017e-05],
       [9.79887017e-05, 9.44523986e-05]])
```

In [27]:
```
# TODO: matrix of factor exposures
B = factor_exposure_a
B
```

Out[27]:
```
array([[ 1.35101534, -0.58353198],
       [-0.2283345 ,  1.16364007]])
```

Hint: for diagonal matrices

You can try using numpy.diag

In [28]:
```
# TODO: matrix of specific variances
S = np.diag([var_s_1,var_s_2])
S
```

Out[28]:
```
array([[0.00021723, 0.        ],
       [0.        , 0.00013739]])
```

In [32]:
```
# TODO: covariance matrix of assets
covm_assets = B @F @ B.T + S
print(f"covariance matrix of assets 1 and 2 \n{covm_assets}")
```

```
covariance matrix of assets 1 and 2
[[2.82089786e-04 7.13296510e-05]
 [7.13296510e-05 2.18561378e-04]]
```

Solution

Solution notebook