

Quantitative Portfolio Management: Theory and Practice

Chapter 1: Managing Financial Data

Raman Uppal
EDHEC Business School

2025-2026

The big picture: Plan for the entire book

Part A: Preliminaries

Chapter 1: Managing financial data

Chapter 2: Performance measurement (especially out of sample)

Part B: Modern portfolio management

Chapter 3: Mean-variance portfolios that *ignore* estimation error

Chapter 4: Mean-variance portfolios that *adjust for* estimation error

Part C: Post-Modern Portfolio Management

Chapter 5: CAPM-based portfolios: Black-Litterman model

Chapter 6: Factor-based portfolios: Parametric portfolio policies

Chapter 7: Volatility-timed factor portfolios

Chapter 8: Portfolios exploiting systematic risk factors *and* unsystematic risk

Table of contents

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

What do we want to do in Chapter 1



In this chapter, we first take a big-picture view of the material to be covered in the book.

Then, we study the main kinds of data that one can use to construct optimal portfolios.

You will learn how to use Python to obtain this data and store it in an efficient way so that it can be accessed easily.

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
 - 2.1 Mean-variance efficient portfolios
 - 2.2 Estimation error in mean-variance portfolio weights (Focus)
 - 2.3 Portfolio weights over time
 - 2.4 Out-of-sample returns
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Overview of the entire course

- ▶ The objective of this course is to study cutting-edge methods to construct **optimal equity portfolios** that perform well **out of sample**.
- ▶ Two key questions, therefore, are
 - Q1. How should we **construct** optimal portfolios?
 - Q2. How should we **measure** portfolio performance?
- ▶ The course provides a solid foundation of the **theory** of portfolio choice and the knowledge required to **implement** this theory.
- ▶ A key part of the course is learning how to use **Python** to work with data to implement state-of-the-art portfolio-choice models.

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
 - 2.1 Mean-variance efficient portfolios
 - 2.2 Estimation error in mean-variance portfolio weights (Focus)
 - 2.3 Portfolio weights over time
 - 2.4 Out-of-sample returns
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Mean-variance efficient portfolios

- ▶ We will focus on “mean-variance efficient portfolio weights”
- ▶ These are portfolio weights that trade-off optimally
 - ▶ the return **mean** of the portfolio,
 - ▶ the return **variance** of the portfolio.
- ▶ So, for most of the course, we will not focus on
 - ▶ higher returns moments, such as skewness and kurtosis;
 - ▶ other risk measures, such as VaR, expected shortfall, etc.
 - ▶ other objective functions, such as maximizing expected utility.

Mean-variance portfolio weights when $N = 1$

- ▶ When there is **only one risky asset** (say the market portfolio), and the investor needs to choose the proportion of wealth
 - ▶ to allocate to the market portfolio
 - ▶ with the rest allocated to the risk-free asset,
- ▶ then, the optimal portfolio weight in the risky asset is

$$w_1 = \frac{1}{\gamma} \frac{\mathbb{E}[R_1] - R_f}{\mathbb{V}[R_1]}, \quad \text{where}$$

- ▶ w_1 is the proportion of wealth invested in the single risky asset
- ▶ γ is the risk-aversion of the investor
- ▶ $\mathbb{E}[R_1]$ is the expected (gross) return on the risky asset
- ▶ R_f is the (gross) return on the risk-free asset
- ▶ $\mathbb{V}[R_1]$ is the variance of return on the risky asset, $\sigma_{R_1}^2 = \sigma_{R_{11}}$.

Understanding the optimal portfolio weight

$$w_1 = \frac{1}{\gamma} \frac{\mathbb{E}[R_1] - R_f}{\mathbb{V}[R_1]}$$

- ▶ The **three** terms in the expression above make intuitive sense:
 1. If risk aversion γ increases, weight in risky asset decreases;
 2. If risk premium $\mathbb{E}[R_1] - R_f$ increases, weight in risky asset increases;
 3. If the risk $\mathbb{V}[R_1]$ increases, weight in risky asset decreases.
- ▶ Whenever we see mathematical expressions in the course, we will want to make **intuitive** sense of them.

Mean-variance portfolio weights when $N > 1$

- ▶ When there are many risky assets ($N > 1$), and the investor needs to choose the proportion of wealth
 - ▶ to allocate to the N risky assets
 - ▶ with the rest allocated to the risk-free asset,
- ▶ then, the optimal **vector** of portfolio weights in the N risky assets is

$$w = \frac{1}{\gamma} (\mathbb{V}[R])^{-1} (\mathbb{E}[R] - R_f \mathbf{1}_N), \quad \text{where}$$

- ▶ w is the N -**vector** of weights invested in each risky asset
- ▶ γ is the risk-aversion of the investor
- ▶ $\mathbb{V}[R]$ is the $N \times N$ **covariance matrix** for returns on the N risky assets
- ▶ $\mathbb{E}[R]$ is the N -**vector** of expected returns on each of the risky assets
- ▶ R_f is the (gross) return on the risk-free asset
- ▶ $\mathbf{1}_N$ is the N -vector of ones

Explicit expressions for components of optimal portfolio

- ▶ N -dimensional vectors:

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix}; \quad \mathbb{E}[\mathbf{R}] = \begin{bmatrix} \mathbb{E}[R_1] \\ \mathbb{E}[R_2] \\ \vdots \\ \mathbb{E}[R_N] \end{bmatrix}; \quad \mathbf{1}_N = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}.$$

- ▶ $(N \times N)$ matrix:

$$\mathbb{V}[\mathbf{R}] = \begin{bmatrix} \sigma_{R_{11}} & \sigma_{R_{12}} & \cdots & \sigma_{R_{1N}} \\ \sigma_{R_{21}} & \sigma_{R_{22}} & \cdots & \sigma_{R_{2N}} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{R_{N1}} & \sigma_{R_{N2}} & \cdots & \sigma_{R_{NN}} \end{bmatrix}.$$

- ▶ **Note** that

- ▶ $\sigma_{R_{nn}} = \sigma_{R_n}^2$ denotes the return variance of asset n , and
- ▶ $\sigma_{R_{nm}}$ denotes the return covariance between assets n and m .

Start of focus

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
 - 2.1 Mean-variance efficient portfolios
 - 2.2 Estimation error in mean-variance portfolio weights (Focus)
 - 2.3 Portfolio weights over time
 - 2.4 Out-of-sample returns
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Estimation error in mean-variance portfolio weights

- ▶ Our focus will be to
 - ▶ identify **mean-variance optimal weights**
 - ▶ that perform well **out-of-sample**.

$$w = \frac{1}{\gamma} (\mathbb{V}[R])^{-1} (\mathbb{E}[R] - R_f \mathbf{1}_N) \quad \dots \text{key expression in the course}$$

- ▶ Thus, the challenge is to estimate $\mathbb{E}[R]$ and $\mathbb{V}[R]$ precisely.

The challenge in identifying an optimal portfolio

- ▶ Estimating the N -vector $\mathbb{E}[R]$ is difficult because its precision does **not** improve with additional data.

$$\begin{aligned}\text{Return from year 0 to 100} = R_{0,100} &= \frac{P_{100}}{P_0} \\ &= \frac{P_{100}}{P_{99}} \times \frac{P_{99}}{P_{98}} \times \dots \times \frac{P_2}{P_1} \times \frac{P_1}{P_0} \\ &= \frac{P_{100}}{P_0} \dots \text{intermediate prices cancel out}\end{aligned}$$

- ▶ Estimating the $(N \times N)$ -matrix $\mathbb{V}[R]$ is difficult because of the **large number of parameters** to be estimated:
 - ▶ N variance parameters on the diagonal of $\mathbb{V}[R]$, and
 - ▶ $(N \times N - N)/2$ off-diagonal (unique) covariance parameters, for
 - ▶ a total of $N + (N \times N - N)/2 = N(N + 1)/2$ parameters.
 - ▶ So, if $N = 100$, then $N(N + 1)/2 = 5050$.

The problem with poor estimates of $\mathbb{E}[R]$ and $\mathbb{V}[R]$

- ▶ When estimating the portfolio weights

$$w = \frac{1}{\gamma} (\mathbb{V}[R])^{-1} (\mathbb{E}[R] - R_f \mathbf{1}_N),$$

- ▶ if the elements of $\mathbb{E}[R]$ and $\mathbb{V}[R]$ are estimated poorly, then
 - ▶ the vector $\mathbb{E}[R]$ has large error
 - ▶ the matrix $(\mathbb{V}[R])^{-1}$ has large error (high condition number)
 - ▶ multiplying $\mathbb{E}[R]$ by $(\mathbb{V}[R])^{-1}$ magnifies the error in $\mathbb{E}[R]$
 - ▶ leading to badly estimated portfolio weights, w ,
 - ▶ which perform poorly out of sample.

The solution: Better estimates of $\mathbb{E}[R]$ and $\mathbb{V}[R]$

$$w = \frac{1}{\gamma} (\mathbb{V}[R])^{-1} (\mathbb{E}[R] - R_f \mathbf{1}_N),$$

- ▶ Throughout the course, we will focus on studying **different ways** of estimating $\mathbb{E}[R]$ and $\mathbb{V}[R]$, so as to
 - ▶ reduce the error (and dimension) in estimating $\mathbb{E}[R]$, and
 - ▶ reduce the error (and dimension) of $\mathbb{V}[R]$.

$\mathbb{E}[R]$ and $\mathbb{V}[R]$ important also for other decisions

- ▶ Note that $\mathbb{E}[R]$ and $\mathbb{V}[R]$ required also for other financial decisions
 - ▶ for example, cost of capital requires estimating $\mathbb{E}[R]$;
 - ▶ for example, risk management requires estimating $\mathbb{V}[R]$.
- ▶ A large part of the **current research** in financial economics is devoted to improving the estimation of $\mathbb{E}[R]$ and $\mathbb{V}[R]$.

End of focus

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
 - 2.1 Mean-variance efficient portfolios
 - 2.2 Estimation error in mean-variance portfolio weights (Focus)
 - 2.3 Portfolio weights over time
 - 2.4 Out-of-sample returns
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Portfolio weights over time

- ▶ We wish to identify the portfolio weights not just for one date, but for many dates, $t = \{1, 2, \dots, T\}$.
- ▶ Denote by $w_{t,n}$, with $n = \{1, 2, \dots, N\}$, the **weight** at date t on each asset, which is the proportion of wealth invested in that asset.
- ▶ Denote by w_t the N -dimensional **vector** of portfolio weights

$$w_t = \begin{bmatrix} w_{t,1} \\ w_{t,2} \\ \dots \\ w_{t,N} \end{bmatrix}.$$

- ▶ We wish to study how to choose w_t ; that is, how to choose
 - ▶ the N portfolio weights
 - ▶ at each of the t decision dates.

Using the transpose operator

- ▶ We have defined the $(N \times 1)$ -vector

$$w_t = \begin{bmatrix} w_{t,1} \\ w_{t,2} \\ \dots \\ w_{t,N} \end{bmatrix}.$$

- ▶ The transpose of w_t is given by the $(1 \times N)$ -vector

$$w_t^\top = [w_{t,1}, w_{t,2}, \dots, w_{t,N}], \quad \dots \text{where } ^\top \text{ denotes the transpose operator}$$

- ▶ which means that the **sum of the portfolio weights** can be written as

$$\sum_{n=1}^N w_{t,n} = w_t^\top \mathbf{1}_N, \quad \text{and}$$

- ▶ the expected return on a portfolio p can be written as

$$\mathbb{E}[R_{tp}] = \sum_{n=1}^N w_{t,n} \mathbb{E}[R_{t,n}] = w_t^\top \mathbb{E}[R_t].$$

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
 - 2.1 Mean-variance efficient portfolios
 - 2.2 Estimation error in mean-variance portfolio weights (Focus)
 - 2.3 Portfolio weights over time
 - 2.4 Out-of-sample returns
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Out-of-sample returns

- ▶ When choosing portfolio weights w_t , we want the portfolio to perform well in terms of its risk and return at **date $t + 1$** . That is,
 - ▶ the weights w_t are chosen based on information available until t ;
 - ▶ the portfolio performance depends on returns next period, $t + 1$.
 - ▶ Therefore, this performance is called **out of sample**.
- ▶ The return at $t + 1$ of the portfolio w_t is

$$[\text{Portfolio return}]_{t+1} = [\text{Portfolio weights}]_t \times [\text{Asset returns}]_{t+1}$$

$$R_{t+1,p} = w_t^T R_{t+1} = \sum_{n=1}^N w_{t,n} R_{t+1,n}, \quad \dots \text{"T" is transpose operator}$$

- ▶ So, our choice of w_t depends on our **views** about $R_{t+1,n}$.

What could the portfolio weights depend on?

- ▶ The weight assigned to each asset $w_{t,n}$ will depend on the features of that asset. These features could include:
 - ▶ The **expected return** of this asset;
 - ▶ The **risk** of this asset, which could be measured as its variance, skewness, kurtosis, downside risk, value-at-risk (VaR), expected shortfall, tail risk, etc.;
- ▶ The asset's expected return and risk could depend on
 - ▶ **Past returns** of the asset itself and of other assets;
 - ▶ Other **characteristics** of this asset, e.g., its size, profitability, etc.;
 - ▶ The **sensitivity** of the returns of this asset to the returns of other assets, measured by covariance or correlation;
 - ▶ The sensitivity of the return of this asset to **macroeconomic factors**.

Cross-section, time-series, and panel data

- ▶ So, the first thing we need to do is to understand how to use data to compute these quantities, which can, potentially, be measured
 - ▶ across N assets (**cross-section**),
 - ▶ over T dates (**time-series**), and
 - ▶ across N assets **and** over T dates (**panel**).

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
- 3. Notation**
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Notation: N and T

- ▶ Throughout the course, our analysis will depend on two quantities:
 - ▶ N with $n = \{1, 2, \dots, N\}$, representing the **number of assets** in which we can invest.
 - ▶ T , with $t = \{1, 2, \dots, T\}$ representing the **number of observations**.

Notation: K and T^{est}

- ▶ Throughout the course, we will discuss **factors** that help us make investment decisions:
 - ▶ K with $k = \{1, 2, \dots, K\}$, representing the **number of factors** used to make investment decisions.
 - ▶ $T^{\text{est}} \leq T$, with $t^{\text{est}} = \{1, 2, \dots, T^{\text{est}}\}$ representing the **number of observations we use for estimating quantities of interest**.

Notation: N , K , T , and T^{est}

- ▶ In the papers on portfolio management,
 - ▶ the notation for N , K , and T is standard; that is, almost all papers use the same notation;
 - ▶ the notation for T^{est} is **not** standard, so you will need to be alert about the notation that is being used.
- ▶ Some of the symbols used to represent the length of the estimation window are:
 - ▶ $M \leq T$, with $m = \{1, 2, \dots, M\}$, where M stands for **months** of data used for estimation (usually 60 or 120 months);
 - ▶ $D \leq T$, with $d = \{1, 2, \dots, D\}$, where D stands for **days** of data used for estimation (usually 30, 60 or 90 days);
 - ▶ $\tau \leq T$, with $t = \{1, 2, \dots, \tau\}$.

Our notation compared to that used by other people

Our notation	Other choices in other books and papers
w	x or θ
γ	$\gamma = 1/\tau$, where τ denotes risk-tolerance
$\mathbb{V}[R]$ or \mathbb{V}_R	V or Σ_R or Σ or \mathbb{C} (for covariance matrix)
$\mathbb{E}[R]$ or \mathbb{E}_R	$E[R]$ or μ_R or μ (for mean returns)
R_f	r_f (net risk-free rate)
1_N	$\mathbf{1}$ or e or ι (iota)
T^{est}	M , D , or τ

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. **Convention regarding data**
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Convention regarding data ... I

- ▶ We will adopt the same three principles as in [Tidy Finance](#) regarding data:
 1. Every **column** is a variable.
 2. Every **row** is an observation.
 3. Every **cell** is a single value.

Convention regarding data ... II

- ▶ Consider the following table with prices for
 - ▶ T dates (in rows) and
 - ▶ N assets (in columns),
 - ▶ with the typical entry being $P_{t,n}$.

Assets Dates	$n = 1$	$n = 2$	\cdots	$n = N$
$t = 1$	$P_{1,1}$	$P_{1,2}$	\cdots	$P_{1,N}$
$t = 2$	$P_{2,1}$	$P_{2,2}$	\cdots	$P_{2,N}$
\vdots	\vdots	\vdots	\ddots	\vdots
$t = T$	$P_{T,1}$	$P_{T,2}$	\cdots	$P_{T,N}$

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
- 5. Prices and Returns**
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Prices and returns

- ▶ The price of asset n at date t , denoted $P_{t,n}$ is the most common type of data we will work with.
- ▶ We will use data on prices to compute returns:

Gross returns:
$$R_{t,n} = \frac{P_{t,n}}{P_{t-1,n}} \quad (1)$$

Net returns:
$$r_{t,n} = R_{t,n} - 1 = \frac{P_{t,n}}{P_{t-1,n}} - 1$$

Excess returns:
$$R_{t,n} - R_{t,0} = \frac{P_{t,n}}{P_{t-1,n}} - \frac{P_{t,0}}{P_{t-1,0}} \quad (2)$$

where $R_{t,0}$ is the return on some reference asset (usually the risk-free asset, whose return is denoted $R_{t,f}$).

- ▶ Note that when looking at **excess returns**, it does not matter whether we look at gross or net returns: excess gross returns are equal to excess net returns.

Multiperiod compound returns

- ▶ Multiperiod returns are called compound returns.
- ▶ The return over h periods, between date $t - h$ and date t , is

Multiperiod returns:
$$\begin{aligned} R_{t,n}(h) &= \frac{P_{t,n}}{P_{t-h,n}} \\ &= \frac{P_{t,n}}{P_{t-1,n}} \times \frac{P_{t-1,n}}{P_{t-2,n}} \times \dots \times \frac{P_{t-h+1,n}}{P_{t-h,n}} \\ &= R_{t,n} \times R_{t-1,n} \times \dots \times R_{t-h+1,n}. \end{aligned}$$

Annualized returns . . . I

- ▶ Returns are always quoted with respect to a time period.
- ▶ The convention is to quote returns **per year** (per annum).
- ▶ Monthly **net** returns are annualized by multiplying by 12.
- ▶ Monthly **gross** returns are obtained by adding 1 to the annualized net return.

Annualized returns ... II

- If each return $R_{t,n}$ is a one-year return, then the **multi h -year return** is computed as

$$R_{t,n}(h) = R_{t,n} R_{t-1,n} \cdots R_{t-h+1,n} = \prod_{i=0}^{h-1} R_{t-i,n}$$

in which case, the **annualized gross return** is

$$R_{t,n}(h)^{1/h} = [R_{t,n} R_{t-1,n} \cdots R_{t-h+1,n}]^{1/h} = \left[\prod_{i=0}^{h-1} R_{t-i,n} \right]^{1/h}$$

and the **approximate annualized gross return** is

$$\approx \frac{1}{h} \left[\sum_{i=0}^{h-1} R_{t-i,n} \right]. \quad (3)$$

Continuous compounding ... I

- Denote the natural logarithm of prices by $p_{t,n} = \ln P_{t,n}$ so that

$$\ln R_{t,n} = \ln \frac{P_{t,n}}{P_{t-1,n}} = \ln P_{t,n} - \ln P_{t-1,n} = p_{t,n} - p_{t-1,n},$$

and for the **multi**period continuously compounded return

$$\begin{aligned} \ln R_{t,n}(h) &= \ln (R_{t,n} R_{t-1,n} \dots R_{t-h+1,n}) \\ &= \ln R_{t,n} + \ln R_{t-1,n} + \dots + \ln R_{t-h+1,n} \end{aligned} \tag{4}$$

$$\begin{aligned} &= (p_{t,n} - p_{t-1,n}) + (p_{t-1,n} - p_{t-2,n}) + \dots + (p_{t-h+1,n} - p_{t-h,n}) \\ &= p_{t,n} - p_{t-h,n} \dots \text{all other (intermediate) terms cancel out} \end{aligned} \tag{5}$$

- There are two important insights from Equations (4) and (5).

Continuous compounding ... II

- ▶ Equation (5) shows that for (continuously compounded) returns, **only the first and last price observations matters**; the rest drop out!
- ▶ That is, if you are computing returns over 100 years, the return that you get is the same whether you use
 - ▶ just the data for the first price and the last price,
 - ▶ annual data on prices for all 100 years,
 - ▶ monthly data on prices for all the months in the last 100 years, or
 - ▶ daily data on prices for all the days in the last 100 years.
- ▶ That is, additional data is **not** useful for estimating the return. (This is not true for the variance.)
- ▶ Consequently, estimates of expected return are very **imprecise**. (In contrast, estimates of the variance are much more precise.)

Continuous compounding ... III

- ▶ Equation (4) shows that the multiperiod log return is the **sum** of one-period returns;
- ▶ An advantage of this is that it is easier to derive statistical properties of **sums** of random variables than products.
- ▶ However, there is one **disadvantage** of log returns:
 - ▶ while the gross return of the **portfolio**, denoted $R_{t,p}$, is a weighted sum of the gross returns of the assets in the portfolio

$$R_{t,p} = \sum_{n=1}^N w_{t,n} R_{t,n},$$

- ▶ the log return of the portfolio is **not** a weighted sum

$$\ln R_{t,p} \neq \sum_{n=1}^N w_{t,n} (\ln R_{t,n}).$$

Continuous compounding ... IV

- ▶ But, if returns are measured over short intervals, then

$$\ln R_{t,n} \approx \sum_{n=1}^N w_{t,n} (\ln R_{t,n}).$$

- ▶ When studying returns of the **cross-section** of assets, we usually use gross returns, $R_{t,n}$ or gross excess returns, $R_{t,n} - R_{t,0}$.
- ▶ When studying the **time-series** properties of returns, we usually use log returns, $\ln R_{t,n}$, or log excess returns, $\ln R_{t,n} - \ln R_{t,0}$.
- ▶ In this course, because we wish to ask how best to allocate wealth **across** assets, we will focus on the cross-section of returns.

Returns with dividend payments

- ▶ If an asset n pays dividends, $D_{t,n}$, then its gross return is

$$R_{t,n} = \frac{P_{t,n} + D_{t,n}}{P_{t-1,n}}$$

$$\ln R_{t,n} = \ln \frac{P_{t,n} + D_{t,n}}{P_{t-1,n}} = \ln(P_{t,n} + D_{t,n}) - \ln P_{t-1,n}.$$

Distribution of returns

- ▶ The return of asset n at date t is a random variable.
- ▶ Random variables are characterized by their **distribution**.
- ▶ There are at least three types of distributions of interest:
 1. Joint distribution,
 2. Conditional distribution, or
 3. Unconditional distribution.

Unconditional distribution ... I

- ▶ It is convenient to assume that gross simple returns have a distribution that is
 - ▶ IID (independent and identical distribution)
 - ▶ Lognormal
- ▶ The assumption of Lognormal returns implies that the log return has a Normal distribution.
 - ▶ log returns, $\ln R_{t,n} \sim \mathcal{N}(\mu_n, \sigma_n)$,
where μ_n is the mean and σ_n is the volatility (standard deviation).
- ▶ The Lognormal distribution has a lower bound of 0;
thus, gross returns do not violate the requirement of limited liability.
 - ▶ Note if $R_{t,n}$ has a lower bound of zero, then $\ln R_{t,n}$ takes values from $-\infty$ to $+\infty$.

Unconditional distribution ...II

- ▶ The general result is that the m -th **moment** of a Lognormally distributed variable X with mean μ and variance σ^2 is given by

$$\mathbb{E}[X^m] = \exp(m\mu + \frac{1}{2}m^2\sigma^2).$$

- ▶ If log returns have a Normal distribution, then returns themselves have the following mean and variance:

$$\mathbb{E}[R_n] = \exp(\mu_n + \frac{1}{2}\sigma_n^2) \quad \dots \text{the first moment}$$

$$\mathbb{E}[R_n^2] = \exp(2\mu_n + 2\sigma_n^2) \quad \dots \text{the second moment}$$

$$\begin{aligned} \mathbb{V}[R_n] &= \mathbb{E}[R_n^2] - (\mathbb{E}[R_n])^2 && \dots \text{variance, second central moment} \\ &= \exp(2\mu_n + 2\sigma_n^2) - \exp(2\mu_n + \sigma_n^2) && \dots [\text{second moment}] - [\text{first moment}]^2 \\ &= \exp(2\mu_n + \sigma_n^2) (\exp(\sigma_n^2) - 1) && \dots \text{collecting/rearranging terms} \end{aligned}$$

- ▶ You can read more about the Lognormal distribution at:
 - ▶ [Wikipedia](#).

Are asset returns Normally distributed?

- ▶ At short horizons, asset returns are **not** Normal; they have
 - ▶ weak skewness, and
 - ▶ strong excess kurtosis (fat tails).
- ▶ If a random variable X has mean μ and variance σ^2 , then:

Skewness:

$$S[X] = \mathbb{E} \left[\frac{(X - \mu)^3}{\sigma^3} \right]$$

Kurtosis:

$$K[X] = \mathbb{E} \left[\frac{(X - \mu)^4}{\sigma^4} \right].$$

- ▶ For a Normally distributed random variable, $S[X] = 0$ and $K[X] = 3$.
- ▶ Excess kurtosis is then $K[X] - 3$.

Distribution of skewness and kurtosis

- ▶ If you have T with $t = \{1, \dots, T\}$ time-series observations, then

Estimate of skewness:
$$\hat{S}[X] = \frac{1}{\hat{\sigma}^3} \frac{1}{T} \sum_{t=1}^T [(x_t - \hat{\mu})^3]$$

Estimate of kurtosis:
$$\hat{K}[X] = \frac{1}{\hat{\sigma}^4} \frac{1}{T} \sum_{t=1}^T [(x_t - \hat{\mu})^4],$$

where:
$$\hat{\mu} = \frac{1}{T} \sum_{t=1}^T x_t$$
$$\hat{\sigma}^2 = \frac{1}{T} \sum_{t=1}^T [x_t - \hat{\mu}]^2.$$

- ▶ When T is large, then we have the following distribution result:
 $\hat{S}[X] \sim \mathcal{N}(0, 6/T)$ and $\hat{K}[X] \sim \mathcal{N}(3, 24/T)$.

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
- 6. Obtaining financial and macroeconomic data**
 - 6.1 Stock-price data
 - 6.2 Stock-characteristics data
 - 6.3 Macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Financial and macroeconomic data

- ▶ As mentioned before, we will be working with several types of data:
 1. Stock-price data
 2. Stock-characteristics data
 3. Macroeconomic data
- ▶ We look at obtaining these three kinds of data as explained on the [Tidy Finance website](#).

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
 - 6.1 Stock-price data
 - 6.2 Stock-characteristics data
 - 6.3 Macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Stock-price data

Packages we will use to access data

```
# *** IMPORTANT ***  
# Note that before you can import these packages,  
#     you will need to install them on your computer  
# How to install packages depends on your Python environment  
  
import pandas as pd    #For more information, see pandas  
import numpy as np     #For more information, see NumPy  
import yfinance as yf  #For more information, see yfinance
```

Example of downloading price data

Price data for Apple from Yahoo Finance

```
prices = (yf.download(tickers="AAPL",  
                      start="2000-01-01",  
                      end="2024-12-31")  
  
.reset_index()  
.assign(symbol = "AAPL")  
.rename(columns = {"Date": "date",  
                   "Open": "open",  
                   "High": "high",  
                   "Low": "low",  
                   "Close": "close",  
                   "Adj Close": "adjusted",  
                   "Volume": "volume"  
})  
  
)  
prices.head()
```

Output: Price data for Apple

	date	open	high	low	close	adjusted	volume	symbol
0	2000-01-03	0.936384	1.004464	0.907924	0.999442	0.848323	535796800	AAPL
1	2000-01-04	0.966518	0.987723	0.903460	0.915179	0.776801	512377600	AAPL
2	2000-01-05	0.926339	0.987165	0.919643	0.928571	0.788168	778321600	AAPL
3	2000-01-06	0.947545	0.955357	0.848214	0.848214	0.719961	767972800	AAPL
4	2000-01-07	0.861607	0.901786	0.852679	0.888393	0.754065	460734400	AAPL

- ▶ The **adjusted** prices are corrected for anything that might affect the stock price after the market closes, e.g., stock splits and dividends.
- ▶ Because these actions affect quoted prices but have no direct impact on investors holding the stock, we typically use adjusted prices.
- ▶ What was the opening price of Apple stock on
 - ▶ 2000-01-04? \$0.9665;
 - ▶ 2023-01-04? \$126.98.

History of stock splits for Apple

Date	Stock split
1987 JUN 16	2-for-1
2000 JUN 21	2-for-1
2005 FEB 28	2-for-1
2014 JUN 09	7-for-1
2020 AUG 28	4-for-1

- Aside: To read more about Apple's stock splits, see [Apple stock split 2020: what you need to know](#).

From prices to returns

- ▶ We will be working with returns.
- ▶ So we need to convert prices to returns.
- ▶ From the daily price data, we can get daily **net** returns:

$$r_t = \frac{P_t}{P_{t-1}} - 1.$$

From prices to returns for AAPL

Converting prices to returns

```
returns = (prices
    .sort_values("date")
    .assign(ret = lambda x: x["adjusted"].pct_change())
    .get(["symbol", "date", "ret"])
)
returns.head()
```

	symbol	date	ret
0	AAPL	2000-01-03	NaN
1	AAPL	2000-01-04	-0.084310
2	AAPL	2000-01-05	0.014633
3	AAPL	2000-01-06	-0.086539
4	AAPL	2000-01-07	0.047369

From a single stock to multiple stocks ... I

- ▶ The return computation above is for a single stock.
- ▶ It is straightforward to extend the analysis to multiple stocks.
- ▶ For example, the [Tidy Finance website](#) shows how to compute the returns for **all** current constituents of the Dow Jones Industrial Average (**DJIA**) index. To do this,
 1. first download the ticker symbols for all the current constituents of the DJIA, and
 2. then replace in the old code `symbol = "AAPL"` with the **list** of **all** tickers: `symbol = ["AAPL", ..., ...,]`

From a single stock to multiple stocks ... II

- ▶ You should learn how to compute returns for stock constituents of:
 - ▶ **S&P 500**; see
 - ▶ [S&P 500 data loader by Hamed-Ahangari](#), or
 - ▶ [TowardsDataScience](#), or
 - ▶ [InsiderFinanceWire](#).
 - ▶ **Nasdaq**; see
 - ▶ [Nasdaq data link](#), or
 - ▶ [TowardsDataScience](#).
 - ▶ **DAX**; use
 - ▶ either the same approach as for constituents of DJIA or Nasdaq;
 - ▶ or see Section 9.5 of the book by Brugière ([2020](#)).
(Also, Sections 1.5, 3.4, and 5.4 of the book may be useful.)

Sample code

- ▶ Sample Python code for [analyzing World Stock Indices Performances](#)

Sample code from [Intan Dea Yutami's website](#)

```
# Import packages that we will use
import numpy as np
import pandas as pd
import yfinance as yf

# Retrieve list of world major stock indices from Yahoo! Finance
df_list = pd.read_html('https://finance.yahoo.com/world-indices/')
majorStockIdx = df_list[0]
majorStockIdx.head()
```

Sample code (continued)

Sample code from Intan Dea Yutami's website (continued)

```
# Get historical price data for all the stock indices
stock_list = []
for s in majorStockIdx.Symbol: # iterate for every stock indices
    # Retrieve data from Yahoo! Finance
    tickerData = yf.Ticker(s)
    tickerDf1 = tickerData.history(period='1d', start='2010-1-1', end='
    2020-9-30')
    # Save historical data
    tickerDf1['ticker'] = s # don't forget to specify the index
    stock_list.append(tickerDf1)
# Concatenate all data
msi = pd.concat(stock_list, axis = 0)
```

- ▶ From this point on, you can repeat the earlier steps to calculate returns from prices, etc.
- ▶ Or you can look at [Intan Dea Yutami's website](#), in which case, notice also the nice plots (important to learn *data-visualization* skills).

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
 - 6.1 Stock-price data
 - 6.2 **Stock-characteristics data**
 - 6.3 Macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Stock characteristics ... I

- ▶ Our objective is to obtain good estimates of $\mathbb{E}[R]$ and $\mathbb{V}[R]$.
- ▶ One simple way to do this is to use the **sample counterparts** of these moments, i.e.,

$$\hat{\mu}_{R_n} = \frac{1}{T} \sum_{t=1}^T R_{t,n} \quad \dots \text{sample mean}$$

$$\hat{\sigma}_{R_{nn}} = \frac{1}{T} \sum_{t=1}^T [R_{t,n} - \hat{\mu}_n]^2 \quad \dots \text{sample variance}$$

$$\hat{\sigma}_{R_{nm}} = \frac{1}{T} \sum_{t=1}^T [(R_{t,n} - \hat{\mu}_n)(R_{t,m} - \hat{\mu}_m)] \quad \dots \text{sample covariance}$$

Stock characteristics ... II

- ▶ But, one could also use **other characteristics of each stock** to estimate $\mathbb{E}[R_n]$ and $\mathbb{V}[R_n]$. These characteristics could be:
 - ▶ Size
 - ▶ Value (ratio of book value to market value)
 - ▶ Profitability
 - ▶ Investment
 - ▶ etc.
- ▶ A website used by many academic researchers with data on the above firm characteristics is [Ken French's data library](#).
- ▶ On the next few pages, we see how one can use Python to download data from Ken French's website.

Using Python to work with stock-characteristics data

- ▶ This material is from the section on “Financial Data” on the [Tidy Finance website](#).
- ▶ The website makes the eminently sensible suggestion to organize and store data in a **single** database.
- ▶ We have already loaded the packages “pandas” and “numpy”
- ▶ In addition to that, we define the **date range** for which we will be downloading the data.

Load packages and define data range

```
import pandas as pd      #For more information, see pandas
import numpy as np       #For more information, see NumPy

start_date = "1960-01-01"
end_date = "2022-12-31" #You can change this to the current date
```

Using Python for Fama-French data

- ▶ For stock characteristics, we will use data from Ken French's data library.
 - ▶ We will study how the Fama-French factors are constructed later on in the course.
- ▶ To read this data, we will use “pandas-datareader”

Load package for reading data

```
import pandas_datareader as pdr  
#For more information, see pandas-datareader
```

Fama-French data on 3 factors: mkt, size, and value

- ▶ The dataset “Fama/French 3 Factors” contains the return time series of the **market** (mkt_excess), **size** (smb), and **value** (hml) alongside the risk-free rates, rf.
- ▶ The code below reads the data and does some steps to correctly parse all the columns and scale them appropriately.

Code to download Fama-French 3 factors

```
factors_ff3_monthly_raw = pdr.DataReader(  
    name="F-F_Research_Data_Factors",  
    data_source="famafrench",  
    start=start_date,  
    end=end_date)[0]  
  
factors_ff3_monthly = (factors_ff3_monthly_raw  
    .divide(100)  
    .reset_index(names="month")  
    .assign(  
        month = lambda x: pd.to_datetime(x["month"].astype(str))  
    )  
    .rename(str.lower, axis="columns")  
    .rename(columns = {"mkt-rf" : "mkt_excess"})  
)
```

Fama-French data on 5 factors

- ▶ The [Tidy Finance website](#) also provides code to download the **five** Fama-French factors (2x3), which includes the return time series of **profitability** (rmw) and **investment** (cma) factors.

Code to download Fama-French 5 factors

```
factors_ff5_monthly_raw = pdr.DataReader(
    name="F-F_Research_Data_5_Factors_2x3",
    data_source="famafrench",
    start=start_date,
    end=end_date)[0]

factors_ff5_monthly = (factors_ff5_monthly_raw
    .divide(100)
    .reset_index(names="month")
    .assign(
        month = lambda x: pd.to_datetime(x["month"].astype(str))
    )
    .rename(str.lower, axis="columns")
    .rename(columns = {"mkt-rf" : "mkt_excess"}))
```

Daily data and data on other factors

- ▶ The [Tidy Finance website](#) provides Python code to download also:
 - ▶ **Daily** data for the Fama-French factors;
 - ▶ Returns on **10 industry factors** from Ken French's data library;
 - ▶ Factors for the **q-factor** model in Hou, K., C. Xue, and L. Zhang. 2015. Digesting anomalies: An investment approach. *Review of Financial Studies* 28 (3): 650–705.

Other code and data on stock characteristics

- ▶ Python code to access Fama-French data and also other data is available from a number of other websites:
 - ▶ [PyAnomaly](#) (efficient data download from WRDS using `asynco`).
 - ▶ [pyassetpricing](#).
 - ▶ [getFamaFrenchFactors](#).
- ▶ An excellent website with data on 331 predictors is [Open Source Asset Pricing](#); for details, see the paper Chen and Zimmermann (2022).

Sources for data on non-US stocks

- ▶ [investpy](#) is a free Python package to retrieve data from [Investing.com](#), which provides data retrieval for up to
 - ▶ 39952 stocks,
 - ▶ 82221 funds,
 - ▶ 11403 ETFs,
 - ▶ 2029 currency crosses,
 - ▶ 7797 indices,
 - ▶ 688 bonds,
 - ▶ 66 commodities,
 - ▶ 250 certificates, and
 - ▶ 4697 cryptocurrencies.
- ▶ [investpy](#) allows you to download both recent and historical data from all the financial products indexed at [Investing.com](#).
 - ▶ It includes data from countries such as the US, France, Germany, India, Russia, and Spain, among many others.

Sources for data on characteristics of other asset classes

- ▶ Code for other characteristics and asset classes other than stocks
 - ▶ [Martin Waibel's code in Python](#) to reproduce the results in Gu, Kelly, and Xiu (2020) on machine learning. The code also includes [options'](#) features used in the paper Bali, Beckmeyer, Moerke, and Weigert (2023).
 - ▶ The paper by Gu, S., B. Kelly, and D. Xiu. 2020. Empirical asset pricing via machine learning. *The Review of Financial Studies* 33 (5): 2223–2273 is an excellent starting point to learn [machine learning](#).
 - ▶ Code for [bond pricing](#) is available from [Open Source Bond Asset Pricing](#) with details in Dickerson, Mueller, and Robotti (2023).
 - ▶ See also the paper Dick-Nielsen, Feldhütter, Pedersen, and Stolborg (2023) that constructs an error-free dataset for corporate bonds.
 - ▶ Python code to clean academic TRACE data following the procedure outlined in the paper by Dick-Nielsen and Poulsen (2019) is available from [Martin Waibel's package "PyCleanTrace"](#).

EDGAR: Accessing financial data on companies ... I

- ▶ In the US, companies are required by law to file forms with the Securities and Exchange Commission ("SEC").
- ▶ **EDGAR**, the Electronic Data Gathering, Analysis, and Retrieval is a database system that automates the collection, validation, and indexing of the information submitted by companies.
 - ▶ The database is freely available to the public.
- ▶ Details of EDGAR can be read on [this US SEC website](#).
- ▶ The types of data available on EDGAR [are described here](#).
 - ▶ The different **forms** that companies have to file are [described here](#).

EDGAR: Accessing financial data on companies ... II

- ▶ Python APIs (Application Programming Interface) to access EDGAR are available from:
 - ▶ [SEC Edgar Downloader](#).
 - ▶ You can build a master index of SEC filings with [python-edgar](#).
 - ▶ [OpenEDGAR: Open Source Software for SEC EDGAR Analysis](#).
 - ▶ [EDGAR Tools](#) to access **and** analyze SEC filings.
 - ▶ [Analyzing stock index data with Python and EDGAR](#).
 - ▶ [SEC EDGAR API](#) to stream new filings in real time.
 - ▶ There is also commercial software code to access EDGAR. [sec-api.io](#).
- ▶ A paper that explains how to [scrape EDGAR With Python](#).
- ▶ [Background information for accessing EDGAR](#).

EDGAR: Accessing financial data on companies ... III

- ▶ You can get basic information for French companies from [Registre du commerce et des sociétés](#).
- ▶ Information about UK companies is stored on [Companies house](#) and [a second website](#), but these are less detailed than EDGAR.
- ▶ For EU companies, you can get basic information from:
 - ▶ [European Business Register \(EBR\)](#) and [a second website](#);
 - ▶ [first website](#) and [a second website](#) for Germany;
 - ▶ [Ireland](#);
 - ▶ [Italy](#);
 - ▶ [Switzerland](#).
- ▶ The Canadian version of EDGAR, is called [SEDAR](#).

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
 - 6.1 Stock-price data
 - 6.2 Stock-characteristics data
 - 6.3 Macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

Macroeconomic data

- ▶ The last kind of data we discuss is **macroeconomic** data.
- ▶ Macroeconomic variables are often used as predictors for the equity premium on the market portfolio.
- ▶ Welch and Goyal (2008) reexamine the performance of a large number of predictor variables considered in the academic literature:
 - ▶ dividend-price ratio, dividend yield, earnings-price ratio, dividend-payout ratio,
 - ▶ three-month US treasury-bill rate, long-term yield, long-term government bond returns, the term spread (the difference between the long-term yield on government bonds and the Treasury bill), default yield spread (the difference between BAA and AAA-rated corporate bond yields),
 - ▶ stock variance, inflation, and net-equity expansion (net issues divided by total end-of-year market capitalization of NYSE stocks).

Macroeconomic predictors on Amit Goyal's website

- ▶ Amit Goyal maintains the data for these macroeconomic predictors in an XLSX-file stored on a public Google drive.

Details of file with macroeconomic data on Amit Goyal's website

```
sheet_id = "1g4LOaRj4TvwJr9RIaA_nwrXXWT0y46bP"  
sheet_name = "macro_predictors.xlsx"  
macro_predictors_link = (  
    "https://docs.google.com/spreadsheets/d/" + sheet_id +  
    "/gviz/tq?tqx=out:csv&sheet=" + sheet_name  
)
```

- ▶ The [Tidy Finance website](#) explains how to download this data.

Macro predictors

Macro predictors from Amit Goyal's website

```
macro_predictors = (  
    pd.read_csv(macro_predictors_link, thousands=",")  
    .assign(  
        month = lambda x: pd.to_datetime(x["yyyymm"], format="%Y%m"),  
        IndexDiv = lambda x: x["Index"] + x["D12"],  
        logret = lambda x: (np.log(x["IndexDiv"]) -  
                             np.log(x["IndexDiv"].shift(1))),  
        Rfree = lambda x: np.log(x["Rfree"] + 1),  
        rp_div = lambda x: x["logret"] - x["Rfree"].shift(-1),  
        dp = lambda x: np.log(x["D12"]) - np.log(x["Index"]),  
        dy = lambda x: (np.log(x["D12"]) -  
                        np.log(x["D12"].shift(1))),  
        ep = lambda x: np.log(x["E12"]) - np.log(x["Index"]),  
        de = lambda x: np.log(x["D12"]) - np.log(x["E12"]),  
        tms = lambda x: x["lty"] - x["tbl"],  
        dfy = lambda x: x["BAA"] - x["AAA"]  
    )  
    .get(["month", "rp_div", "dp", "dy", "ep", "de", "svar",  
         "b/m", "ntis", "tbl", "lty", "ltr", "tms", "dfy",  
         "infl"])  
    .query("month >= @start_date and month <= @end_date")  
    .dropna()  
)
```

Other macroeconomic data

- ▶ There is a large amount of macroeconomic data available on the internet, most of which can be accessed efficiently using Python.
- ▶ For example, the Federal Reserve Bank of St. Louis provides the Federal Reserve Economic Data (FRED), an extensive database for macroeconomic data, with 817,000 US and international time series from 108 different sources.
- ▶ The [Tidy Finance website](#) explains how to download this data using Python.

Using Python to download data from FRED

- ▶ As an illustration, we reproduce the instructions from the [Tidy Finance website](#) to get consumer price index (CPI) data that can be found under the CPIAUCNS key.

Downloading CPI from FRED

```
import pandas_datareader as pdr

cpi_monthly = (pdr.DataReader(
    name="CPIAUCNS",
    data_source="fred",
    start=start_date,          # This was defined earlier
    end=end_date               # This was defined earlier
))
cpi_monthly.reset_index(names="month")
cpi_monthly.rename(columns = {"CPIAUCNS" : "cpi"})
cpi_monthly.assign(cpi=lambda x: x["cpi"] / x["cpi"].iloc[-1])
```

- ▶ To download different data, we just need to find its key on FRED; e.g., the key for the producer price index for gold ores is PCU2122212122210.

Setting up a database (optional)

- ▶ It is extremely useful to store downloaded data in a database.
- ▶ The Tidy Finance website explains [how to set up an SQLite database](#) (see the bottom of the web page).
- ▶ For our course, it will **not** be required to store data in an SQLite database (but you may still wish to learn how to do this).

Python code for setting up an SQL database (optional)

Three steps for setting up an SQL database

```
# Step 1: Import sqlite3
import sqlite3

#Step 2: Create an SQLite database ‘tidy_finance.db’
tidy_finance = sqlite3.connect("data/tidy_finance.sqlite")

#Step 3: Convert dates, create remote table, copy table to database
(factors_ff3_monthly
 .assign(                                     # convert dates to UNIX integers
    month = lambda x:
        ((x["month"]- pd.Timestamp("1970-01-01"))
         // pd.Timedelta("1d"))
 )
 .to_sql(                                     # ‘to_sql()’ creates the remote table
    name="factors_ff3_monthly", # import monthly Fama-French data
    con=tidy_finance,
    if_exists="replace",
    index = False)
)
```

Python code for accessing the SQL database

- ▶ To access data stored in an SQL database, follow two steps:
 1. Establish the connection to the SQLite database, and
 2. Execute the query to fetch the data.
- ▶ The [Tidy Finance website](#) provides the code below for how to access an SQLite database (see bottom of the web page).

Accessing an SQL database

```
import pandas    # package to query the database
import sqlite3   # package to connect to the database

tidy_finance = sqlite3.connect("data/tidy_finance.sqlite") # connection
factors_q_monthly = (pd.read_sql_query(                    # query
    sql="SELECT * FROM factors_q_monthly",
    con=tidy_finance,
    parse_dates={"month": {"unit": "D", "origin": "unix"}})
)
```

WRDS, CRSP, and Compustat (optional)

- ▶ [WRDS \(Wharton Research Data Services\)](#) is a convenient interface to access asset- and firm-specific data in [CRSP \(Center for Research in Security Prices\)](#) and Compustat.
- ▶ The data at WRDS is organized in an SQL database, although they use the PostgreSQL engine.
- ▶ The [Tidy Finance website](#) explains how to use WRDS to access CRSP and Compustat data and store the data in an SQL database.
- ▶ We will **not** have time to study how to use WRDS in this course.

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

What we plan to do in the next chapter



What's
next?

In the next chapter, we will study how to measure the performance of a portfolio.

Our focus will be on understanding that performance should be measured *out-of-sample*.

We will then study various performance metrics used in finance.

To do for next class

- ▶ Readings
 - ▶ Please read the section on [Financial Data](#) in the book by Scheuch, Voigt, Weiss, and Frey (2024), available online at [Tidy Finance](#).
- ▶ Assignment
 - ▶ You can also start reading (and working) on the first assignment, even though the assignment also depends on the material to be covered in the next chapter.

Road map

1. Overview of this chapter
2. Overview of entire course: Mean-variance efficient portfolios (Focus)
3. Notation
4. Convention regarding data
5. Prices and Returns
6. Obtaining financial and macroeconomic data
7. To do for next class: Readings and assignment
8. Bibliography

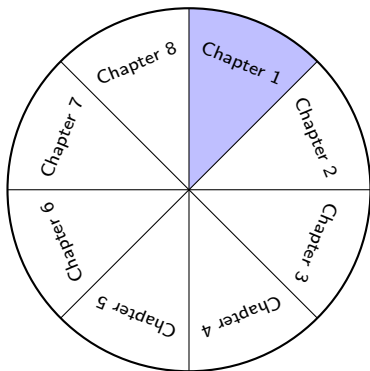
Bibliography ... I

- Bali, T. G., H. Beckmeyer, M. Moerke, and F. Weigert. 2023. Option return predictability with machine learning and big data. *Review of Financial Studies* 36 (9): 3548–3602. (Cited on page [76](#)).
- Brugière, P. 2020. Quantitative portfolio management. *Springer Texts in Business and Economics*. (Cited on page [63](#)).
- Chen, A. Y., and T. Zimmermann. 2022. Open source cross-sectional asset pricing. *Critical Finance Review* 11 (2): 207–264. (Cited on page [74](#)).
- Dick-Nielsen, J., P. Feldhütter, L. H. Pedersen, and C. Stolborg. 2023. Corporate bond factors: replication failures and a new framework. [Available at SSRN 4586652](#). (Cited on page [76](#)).
- Dick-Nielsen, J., and T. K. Poulsen. 2019. How to clean academic TRACE data. [Available at SSRN 3456082](#). (Cited on page [76](#)).
- Dickerson, A., P. Mueller, and C. Robotti. 2023. Priced risk in corporate bonds. Forthcoming in *Journal of Financial Economics*. (Cited on page [76](#)).
- Gu, S., B. Kelly, and D. Xiu. 2020. Empirical asset pricing via machine learning. *The Review of Financial Studies* 33 (5): 2223–2273. (Cited on page [76](#)).
- Hou, K., C. Xue, and L. Zhang. 2015. Digesting anomalies: An investment approach. *Review of Financial Studies* 28 (3): 650–705. (Cited on page [73](#)).

Bibliography ... II

Scheuch, C., S. Voigt, P. Weiss, and C. Frey. 2024. *Tidy finance with Python*. Chapman-Hall/CRC. Available online from [this link](#). (Cited on page 92).

Welch, I., and A. Goyal. 2008. A comprehensive look at the empirical performance of equity premium prediction. *The Review of Financial Studies* 21 (4): 1455–1508. (Cited on page 81).



End of Chapter 1