Quantitative Finance with Python

Stock Market, Bonds, Markowitz-Portfolio Theory, CAPM, Black-Scholes Model, Value at Risk and Monte-Carlo Simulations

1. Introduction

Introduction

Why Python?

Financial Models

- financial models and technical analysis are based on historical data = data from the past
- models use the features (e.g. mean, standard deviation, trend) from the past to make prediction about the future
- such models assume that the behavior in the past will be the behavior in the future
- examples of such “static” models
    - Markowitz-Portfolio Model
    - Capital Asset Pricing Model
- counter-example are “dynamic” models
    - they use non-linear relationship and evolution of features into account
    - assumptions are that the parameters are different from zero and not-constant

2. Python Set-Up

3. Stock Market Basics

Present Value and Future Value

- Time Value of Money
    - most important concept in quantitative finance is the ***time value of money***
    - receiving $x in cash today is always more desirable as receiving the same $x at some time the future
- Why?
    - because today we could invest or lend out the $x today in some way and receive $x +$r in the future
    - inversely, because there are uncertainties in the future about receiving $x such as inflation, illiquidity, default, bankruptcy, etc.
- Discrete Models
- Future Value
    - is the value a current asset will have at specified date in the future based on assumed rate of growth over time
    - $FV = x(1 + r)^n$
    - with x as the principal amount, r interest rate, n number of years (or appropriate time steps)
- Present Value

- defines how much a future amount of money is worth today given a specified rate or interest (discount rate)
- $PV = \frac{x}{(1+r)^n}$
- with x as payout amount in the future, r interest or discount rate, n number of years (or appropriate time steps)

- Continuous Models
  - we can construct a continuous model with differential equations
  - suppose we have amount $x(t)$ in the bank at time t
  - how much does this increase in value from one day to the next?
  - $x(t + dt) - x(t) = \frac{dx(t)}{dt} dt$
  - left expression is the change in the amount of money within a dt day
  - right expression is the Taylor expansion or the derivative
  - we know that the interest must be proportional to the actual $x(t)$ amount and the dt time step
  - $x(t + dt) - x(t) = \frac{dx(t)}{dt} dt = r \times x(t) \times dt$
  - we can solve
  - $x(t) = x(0)e^{r \times t}$

Time Value of Money Implementation

- we'll be implement the formulas from last lecture in Python using standard libraries
- functions from previous lecture in python

```python
from math import exp


def future_discrete_value(x, r, n):
    return x*(1+r)**n


def present_discrete_value(x, r, n):
    return x*(1+r)**-n


def future_continuous_value(x, r, t):
    return x*exp(r*t)


def present_continuous_value(x, r, t):
    return x*exp(-r*t)
```

- example

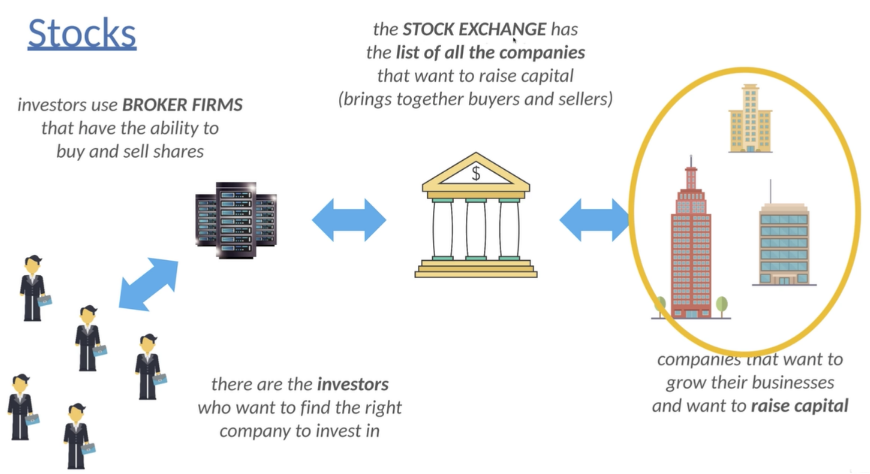
```
# example
x = 100
r = 0.05
n = 5
```

```
print(f"Future Value (discrete) = {future_discrete_value(x,r,n):.2f}")
print(f"Future Value (continuous) = {future_discrete_value(x,r,n):.2f}")

print(f"Present Value (discrete) = {present_discrete_value(x,r,n):.2f}")
print(f"Present Value (continuous) = {present_continuous_value(x,r,n):.2f}")
```

```
Future Value (discrete) = 127.63
Future Value (continuous) = 127.63
Present Value (discrete) = 78.35
Present Value (continuous) = 77.88
```

Stocks and Shares

- definition
  - stocks are the ownership of small piece of the company
  - this entitles the owner of the stock to proportion of the corporation's assets and profits equal to how much stock the own
  - shares are the units of stocks
- stocks as financing / fund raising tool
  - if you have a business, then you can raise capital by **selling off future profits** in the form of shares
  - an investor (then shareholder) gives you cash in return for a contract stating how of the company she owns
  - not all business become successful, so investors take on risk too
  - payout of realized profits is called **dividend** (paid quarterly, semi-annually, or annually)
  - the amount of the dividend depends on company and willingness to pay it (there is no duty to pay dividends)
- the stock market
  - is centralized marketplace where buyers (investors, brokers) and seller (companies) meet
  - in the past, investors had to contact brokers / broker firms through the phone, but today investors can trade through broker apps

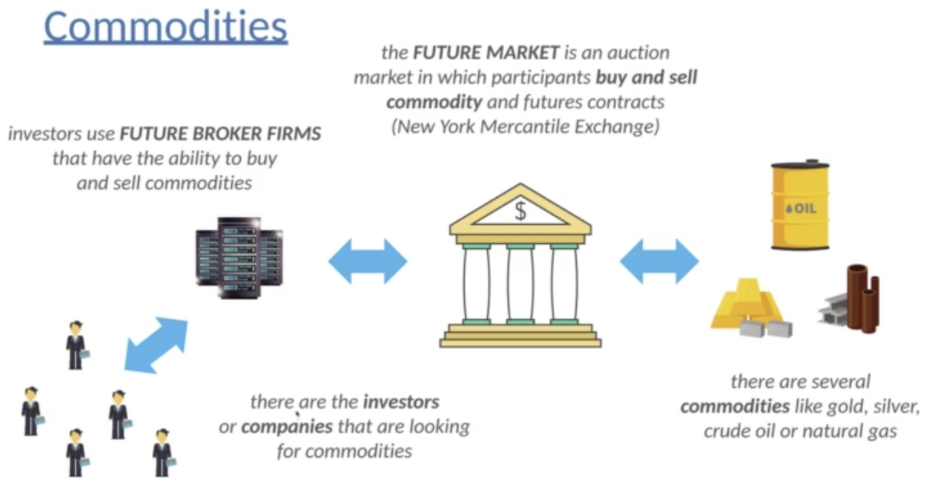


- stock price
  - stock prices have fluctuation in short and median term; and trend in long term
  - $S(t)$ is the stock price at time t where the fluctuations are similar to a **random walk** (statistical phenomenon)
  - the price (fluctuations) comes from supply and demand
  - there may be growth in the value of the stock in the long term, which is realized by selling the stock again
  - dividends can be treated as a regular cash flow
- measuring risk of a stock - volatility
  - statistical measure of the dispersion of returns for a given security
  - the amount of uncertainty (or risk) about the size of changes in the value of a given security
  - volatility is measured with **standard deviation** or **variance** of the returns
  - → the higher the volatility, the riskier a security
  - we can use the Capital Asset Pricing Model (**CAPM**) wth **β** value to approximate volatility

Commodities

- intro
  - commodities are usually raw products such as metal ores, fuels, and precious metal (e.g. gold, oil, gas)
  - investing in commodities is not simple, as you can’t simply go out and buy a barrel of oil
  - prices for most commodities are extremely volatile
  - commodities are traded through derivatives, e.g. **future contracts**
- commodity prices
  - commodity prices have fluctuation in short and mid term; and a trend in long term (just like stock prices)
  - $S(t)$ is the commodity price at time t where the fluctuations are similar to a **random walk** (statistical phenomenon)
  - fluctuation are even stronger than in stock prices; as supply and demand are driven by production and consumption cycles
- commodity futures

- future contracts are made in an attempt by producers, suppliers, and consumers of commodities to avoid market volatility (uncertainty)
    - they negotiate the price of a commodity in the future
    - for instance: a major cost for airlines is fuel expenses and fluctuating oil prices can greatly impact airlines → so they use oil future to protect themselves from rising/fluctuating prices
- commodities/ future markets
    - similar to stock markets but with different instrument
    - future markets are the central auction marketplace, where participants buy and sell commodities and futures (e.g. New York Mercantile Exchange)
    - broker firms mediate between the investors/companies and the marketplace



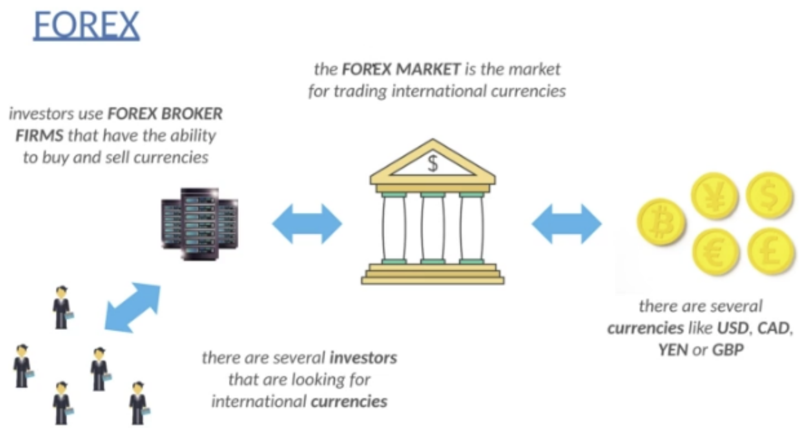
- commodities and inflation
    - commodity prices typically rise with inflation
    - thus, they can offer protection against inflation, when other assets stagnate or fall with inflation
- indirect investment in commodities
    - if cannot invest in commodities or their futures directly, we can invest into companies that rely on a given commodity
    - e.g. gold mining shares and gold price may be positively correlated
    - commodity company share prices will also be affected by operating performance
    - we can employ a “pair trading strategy”

Currencies and the FOREX

- intro
    - an exchange rate is the rate which one (national) currency will be exchanged for another
    - it tells you how much a given currency is with in another currency
    - governments and central banks can influence currencies and exchange rate (but have given up controlling them completely)
- exchange rates

	USD	EUR	GBP	CHF	CAD
USD	1	0.741	0.657	1.061	1.011
EUR	1.35	1	0.889	1.433	1.366
GBP	1.521	1.126	1	1.614	1.538
CHF	0.943	0.698	0.62	1	0.953
CAD	0.955	0.732	0.65	1.049	1

  - when a value is greater than 1, the row currency is stronger than the column currency; if less than 1, the row currency is weaker than the column currency
- exchange rates as price
  - exchange rates have fluctuation in short and mid term; and a trend in long term (just like stock prices)
  - $S(t)$ is the exchange rate at time t where the fluctuations are similar to a **random walk** (statistical phenomenon)
  - fluctuations in rates are caused by supply and demand
    - if more people want to buy a given currency, the exchange price may increase
    - if central banks increase the supply (lowering interest rates), the exchange price may fall
- the FOREX
  - the Foreign Exchange in the market for trading in international currencies
  - biggest currencies include USD, CAD, EUR, GBP, YEN
  - broker firms mediate between the investors and the marketplace
  - in recent years, similar markers have emerged for crypto currencies (without brokers as middleman)



- factors affecting exchange rates
  - *1. interest rates*
    - can be manipulated by the central banks of a country
    - investors will lend money to the banks of the given country for if interest rates increase
    - but not a deterministic relation
  - *2. money supply*
    - central banks printing currency may trigger inflation, which investors do not like and will leave the currency
    - investors leaving can push the value of currency down
  - *3. financial stability*

            - can be proxied by economic growth, unemployment rate, fiscal policy of a given country, which impacts the exchange rates
            - also includes political (in-)stability
- arbitrage on the FOREX
    - we can construct a directed graph $G(V, E)$ out of the exchange rates table with V currencies
        - the V nodes of the graph are the currencies
        - the E nodes are the relative values
    - we take the natural logarithm of the edges and multiply by -1
    - we end up with a negative edge weighted $G'(V, E)$ graph where the negative cycles are the arbitrage opportunities
    - we use the Bellman-Ford shortest path algorithm to find negatives cycles in $O(V * E)$ running time
    - arbitrage opportunities lead to risk-free profits (there should not be any arbitrage in efficient markets)

Short and Long Positions

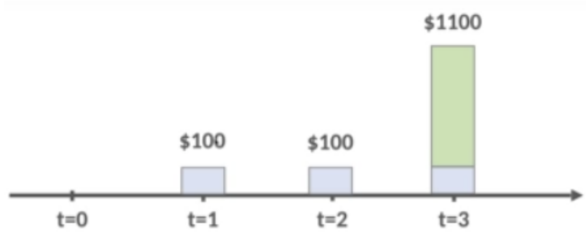
- long position
    - ~ in a security means that you buy/own the security
    - investors maintain long positions in expectation that the security **will increase in value** in the future
    - profit arises in a long position when $S(t_T) > S(t_0)$
- short position
    - ~ in a security means that you borrow and immediately sell the security
    - investors maintain short positions in the expectation that the security **will decrease in value** in the future
    - once the price has dropped, investors buy it back and return the security to the lender, pocketing the difference
    - thus, profit arises in a short position when $S(t_T) > S(t_0)$
    - short selling means you sell something that you do not own
- risks: short vs long position
    - shorting is riskier than opening long positions
    - when you open a long position then you maximum possible loss is 100%, so you may lose the invested capital
    - with short selling there is no limit how much you can lose, because there is no limit for given stock increase in value
- bullish and bearish markets
    - a bear market is when the market experiences a stable price decline
    - so when securities fall for a sustained period of time, there are opportunities for short positions
    - a bull market is when the market experiences a stable price increase
    - so when securities risk for a sustained period of time, there are opportunities for long positions

Quiz: Stock Market Basics

4. Bond Theory

What are bonds?

- definition
    - a bond is debt instrument in which an investor loans money to an entity (company or government)
        - bond is for defined period of time, called maturity
        - it can have a fixed or variable interest rate
    - bonds are fixed-income securities
    - when government or companies need to raise money to finance new projects they may issue bonds directly to investors instead of obtaining loans from a bank
        - interest rates of bond are usually a bit higher than that of banks (but bonds don't need a bank's approval)
- types of bonds
- 1. zero-coupon bonds
    - synonyms: principal amount, par value, face value, nominal value
        - denoted F or X
        - which is paid to the investor/bond holder at the maturity
    - ZCB have no other payments, i.e. cash flows
    - interest rate (premium) paid by bond issuer to investor
    - maturity (date) is the fixed date, when the borrows pays bank the lender
        - often time to maturity is denoted as T
    - present value of a ZCB is
        - $PV = \frac{F}{(1+r)^T}$
        - r is the prevailing market interest rate
    - example
        - $PV = \frac{\$1000}{(1+0.04)^2} = \$924$
        - the present value is also the price of
- 2. coupon bonds
    - also has principal amount and an interest rate
    - but it pays periodical cash amounts before the maturity, called coupons
    - Most common intervals are annually or semi-annually
    - the specific amount of the coupon is expressed as percentage of the principal (called coupon rate)



    - present value of a coupon bonds

        - $PV = \sum_{t=1}^{T} \frac{c_t}{(1+r)^t} + \frac{F}{(1+r)^T}$
        - we have to discount the coupon payments and the principal at the times in the future where they occur
        - when the market interest rate is flat, you can use the following formula
        - $PV = \frac{c}{r}\left[1 - \frac{1}{(1+r)^T}\right] + \frac{F}{(1+r)^T}$
- perpetuities (are no longer issued)
    - is a historical artifact that is no longer issued
    - it has no maturity because it is paid in perpetuity
    - it has a fictional principal to derive a coupon
    - the present value is
        - $PV = \sum_{t=1}^{\infty} \frac{c}{(1+r)^t} = \frac{c}{r}$

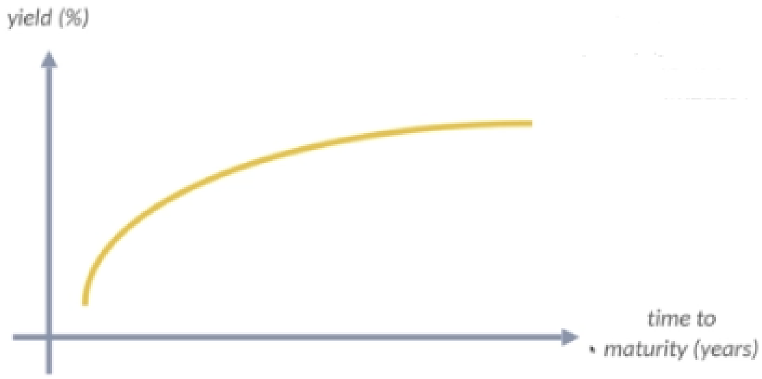
Yield and Yield to Maturity

- yield
    - yield is a metric that shows the return you get on a bond
    - we can calculate it with simple formula
    - defines how much money your investment is generating
- coupon bond
    - $y = \frac{c}{P}$
    - where c is the coupon amount and P is the bond price

    - $y = \frac{100}{1167} = 0.085 = 8.5\%$
- yield to maturity (YTM)
    - aka the internal rate of return (or overall interest rate) earned by an investor who buys the bond at $t = 0$ at the current market price V
    - we assume
        - that the bond is held to until maturity T
        - that all coupons c_i and principal payments are made on schedule
- discrete formula
    - YTM of a bond is that unique constant discount rate such that the PV of the bond's cash flows equal its market price
    - $V = \frac{c}{y}\left[1 - \frac{1}{(1+y)^T}\right] + \frac{F}{(1+y)^T}$
    - recalling that $\frac{c}{F}$ is the coupon rate, we can show
    - $y = \frac{c}{F} \Rightarrow P = F$ bond is trading at *par*

- $y > \frac{c}{F} \Rightarrow P < F$ bond is trading at discount
- $y < \frac{c}{F} \Rightarrow P > F$ bond is trading at *premium*

- continuous formula
  - discounting everything to the present with exponential functions
  - $V = \sum_{i=1}^{n} c_i e^{-y(t_i - t)} + Pe^{-y(T-t)}$
  - where
    - P is the present value of the principal amount
  - solve for y to find the YTM
- yield curve
  - longer bonds pay investors higher interest rates over time because investors expect more returns for loaning money out for longer
  - long-term bonds pose more risk than short-term bonds
  - in the long-run risks can occur such as interest rate changes, more attractive investment may arise, etc.

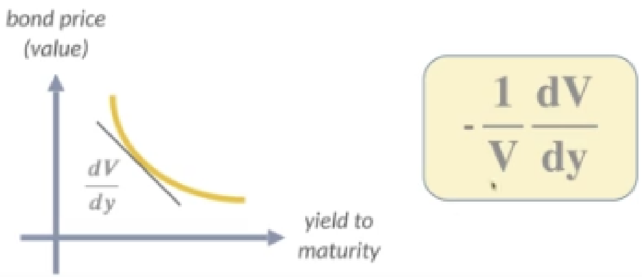


Interest Rates and Bonds

- bond price vs market interest rate → are negatively correlated
  - there is an inverse relation between bond prices and interest rates
  - if rates go upward, prices for bonds go downward leading to capital loss
- why is this relation?
  - if market interest rates go up, it is more profitable to lend money to the bank rather than buying bonds

Macaulay Duration

- definition
  - ~ describes how long it takes for the price of the bond to be repaid by the cash flows from it
  - at this point in time, the investors would get back all his invested money in the bond by the periodic coupons
  - for ZCB, the Macaulay duration is the same as the maturity T
- interest rate risk
  - bond prices and interest rates are negatively correlated
  - bond with longer maturity period are more sensitive to changes in market interest rates



- $-\frac{1}{V}\frac{dV}{dy}$
- risk management
  - *if an investment is made in a bond position and if the investment horizon equals the Macaulay duration of the invested position, then the investment is immunized against parallel shifts in the yield curve (from other course)*
  - Macaulay duration defines how sensitive the bond price is to the (little) change in the market interest rate

Risks with Bonds

- 3 crucial risks associated with bonds:
- 1. **interest rates**: the bond's price moves in opposite direction of interest rate changes; an increase in interest rates leads to a fall in the bond's price
- 2. **default risk**: the bond issuer may become unable to make coupon or principal payment on time
- 3. **inflation risk**: risk in the variation of the value of cash flows due to inflation as measured in terms of purchasing power

Stock and Bonds

- comparison

STOCKS	BONDS
• are shares in the ownership of a business • holders can vote on certain company issues • have the possibility to pay dividends • are riskier than bonds • have no risk rating	• are form of debt instrument that the issuing entity promises to repay • holders have not voting rights • no dividends, but common to have coupons • are less risky than stocks • risk is classified in ratings

Quiz: Bond Theory

5. Bond Implementation

Bond Pricing Implementation 1

- implement a class for a ZCB that takes basic parameter principal, maturity, and interest rate and has a method for computing the price

```python
class ZeroCouponBond:
    def __init__(
        self, principal: float, maturity: int, interest_rate: float
        ) -> None:
        self.principal = principal
        self.maturity = maturity
        self.interest_rate = interest_rate / 100

    def present_value(self, x: float, n: int) -> float:
        return x / (1+self.interest_rate)**n

    def calculate_price(self) -> float:
        return self.present_value(self.principal, self.maturity)
```
✓ 0.7s

```python
principal = 1000
maturity = 2
interest = 4

bond = ZeroCouponBond(principal, maturity, interest)
print(f"Price of the bond in dollars: {bond.calculate_price():.2f}")
```
✓ 0.0s

```
Price of the bond in dollars: 924.56
```

Bond Pricing Implementation 2

- implement a class for a coupon bond that takes basic parameter principal, coupon rate, maturity, and interest rate and has a method for computing the price

```python
class CouponBond:
    def __init__(
        self,
        principal: int,
        coupon_rate: int,
        maturity: int,
        interest_rate: int,
        ) -> None:
        self.principal = principal
        self.coupon_rate = coupon_rate / 100
        self.maturity = maturity
        self.interest_rate = interest_rate / 100

    def present_value(self, x, n) -> float:
        return x / (1+self.interest_rate)**n

    def calculate_price(self) -> float:
        price = 0
        # discount the coupon payments
        coupon = self.principal * self.coupon_rate
        for t in range(1, self.maturity+1):
            price = price + self.present_value(coupon, t)
        # discount principle amount
        price = price + self.present_value(self.principal, self.maturity)
        return price
```

- try on example

```python
principal = 1000
coupon_rate = 10
maturity = 3
interest = 4

bond = CouponBond(1000, 10, 3, 4)
print(f"Price of the bond in dollars: {bond.calculate_price():.2f}")
```
✓ 0.0s

```
Price of the bond in dollars: 1166.51
```

Exercise – Continuous Models for Discounting

- implement continuous discount in the present value function

```python
from math import exp

class contCouponBond(CouponBond):
    def present_value(self, x, n) -> float:
        return x*exp(-self.interest_rate*n)
```
✓ 0.0s

```python
principal = 1000
coupon_rate = 10
maturity = 3
interest = 4

bond = contCouponBond(1000, 10, 3, 4)
print(f"Price of the bond in dollars: {bond.calculate_price():.2f}")
```
✓ 0.0s

```
Price of the bond in dollars: 1164.00
```

Solution – Continuous Models for Discounting

6. Modern Portfolio Theory (Markowitz Model)

Mean, Variance, and Correlation

- **Mean**
  - the mean of **expected value** $E(x)$ or µ of a random variable x os the mean average value of the population
  - let x be a random variable with a finite number of finite outcomes $x_1, x_2, \dots, x_i$ occurring with probabilities $p_1, p_2, \dots, p_i$
  - then $E(x)$ is the defined as $E(x) = \sum_i x_i p_i$
- examples:
  - the expected outcome of a role of a die

$$E(x) = 1 \times \frac{1}{6} + 2 \times \frac{1}{6} + 3 \times \frac{1}{6} + 4 \times \frac{1}{6} + 5 \times \frac{1}{6} + 6 \times \frac{1}{6} = 3.5$$

- a roulette game consists of a wheel with 38 numbered slots
- assume for each $1 that we bet we can win $35 but lose $1
- what is our average expected outcome in roulette?

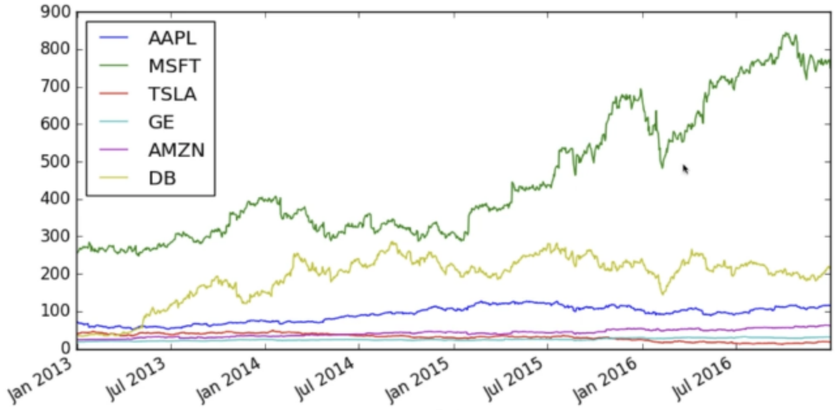
$$E(x) = -\$1\frac{37}{38} + \$35\frac{1}{38} = -\$0.0536$$

- **Variance & Standard Deviation**
  - the variance of a random variable x is the expected value of the squared deviations from the mean
  - $\sigma^2 = E[(x - \mu)^2]$
  - the variance is always positive: wen cad defined the standard deviation σ as the square root of the variance
  - both are measure of dispersion; mean and standard deviation can be used to describe the distribution of a variable on a standard normal distribution
- **Covariance**
  - the **covariance** of x and y is a measure of the joint variability of two random variables (it defines how much the move together)
  - $Cov(x, y) = E\left[(x - \mu_x)(y - \mu_y)\right]$
  - however, covariance is not a good comparative measure
    - it is a dimensional measure
    - it is not normalized
  - → correlation is preferred
- **Correlation**
  - correlation is a dimensionless measure of how two random variables vary together
  - it is computed by the covariance over the standard deviation of the two variables
  - $\delta(x, y) = \frac{cov(x,y)}{\sigma_x \sigma_Y} = \frac{E\left[(x-\mu_x)(y-\mu_y)\right]}{\sigma_x \sigma_Y}$ such that the range is always [-1,1]
  - $\delta(x, y) > 0$ is a positive linear relationship between variables
  - $\delta(x, y) = 0$ no relationship between variables
  - $\delta(x, y) < 0$ is a negative linear relationship between variables

Main Idea: Diversification

- Modern Portfolio Theory
  - first formulated by Harry Markowiz in 1952; later awarded Nobel prize in Economics
  - portfolio is defined a collection of financial investments such as stocks, bonds, commodities, and cash
  - it is a **portfolio optimization model**
  - it assists in the selection of the most efficient portfolio considering various possible portfolios of the given securities based on expected return (*mean*) and risk (*variance*)
- stock portfolio diversification
  - stocks are volatile with prices as random walk
  - investing in multiple stocks can mitigate that risk
  - or even invest in multiple assets to reduce risk as much as possible

    - combining assets is called **diversification** and the main idea behind the Markowitz model (it is also the same approach for the Black-Scholes model)
- example portfolio
    - APPL: Appel, MSFT: Microsoft, TSLA: Tesla, GE: General Electrics, AMZN: Amazon, DB: Deutsche Bank



    - with multiple stocks, individual volatilities cancel each other out making the portfolio more predictable
- assumptions of Markowitz model
    - 1. *returns are normally distributed*
        - statistical measures like mean μ and standard deviation σ show normal distribution
        - this is not always given in reality (and a problem of the model)
    - 2. *investors are risk-averse*
        - investors will only take on more risk, if they are expecting higher returns
        - but usually they like to minimize risk and maximize returns
- modern portfolio theory
    - risk and return relation
        - low risk instruments (bonds) have low returns
        - higher risk instruments (stocks) have higher returns
    - investors can construct the optimal portfolio offering the **maximum possible expected return** for a given **level of risk**
    - likewise, the efficient portfolio is a portfolio that has the **lowest level of risk for given return**

Mathematical Formulation

- investors are not allowed to to set up short position in securities; so all invested money must be divided among available assets in long positions
- example portfolio
    - allocation of investment money are the weights associated with each stock

weights = [0.2, 0.3, 0.25, 0.25]

- sum of the weights cannot be greater than 1.0 or 100%
- formalizing the variables
  - w_i is the weight for the ith asset
  - r_i is the return for the ith asset from historical data
  - μ_i is the expected return for the ith asset (mean of the return)
- returns
  - a relative measure based on a given time interval (such that $[t, t + 1]$ are neighbors on the timeline)
  - interval can be annually, daily, etc.
  - $r = \frac{S(t+1) - S(t)}{S(t)}$
- log returns
  - usually take the natural logarithm of the returns because of mathematical convenience
  - $r = ln\frac{S(t+1)}{S(t)}$

Expected Return of the Portfolio

- the model relies heavily on historical data
- historical mean performance is assumed to be the best estimator for future (expected) performance
- $\mu_{portfolio} = E(\Sigma w_i r_i) = \Sigma w_i E(r_i) = \Sigma w_i \mu_i = w_i^{\top} \cdot \mu_i$
  - the expected return of the portfolio is the sum of the expected return of the individual assets weighted by the allocation
  - a dot product also expresses the sum of the individual products allowing us to vectorize the formula for faster computation

Expected Variance (Risk) of the Portfolio

- risk of the portfolio can be measured by the volatility - that can be approximated by the standard deviation or variance
- covariance
  - $\sigma_{ij} = E\left[\left(r_i - \mu_i\right)\left(r_j - \mu_j\right)\right]$
  - covariance defines how to variables move together
  - $\sigma_{ij} < 0$ *negative covariance means returns move inversely*
  - $\sigma_{ij} > 0$ *positive covariance means returns move together*
  - must calculated for all pairs of stocks (in a matrix)
- diversification aim
  - diversification aims to reduce risk as much as possible
  - possessing assets with a high positive covariance σ_{ij} does not provide reduced risk
  - ⇒ aim is to include uncorrelated stocks in the portfolio instead
- variance
  - $\sigma_i^2 = E\left[\left(r_i - \mu_i\right)^2\right]$
  - variance is the covariance of variable with itself

    - thus, for calculating variances we don't need an extra computational step beyond our covariance matrix as it is simply the diagonal of that matrix
- covariance matrix

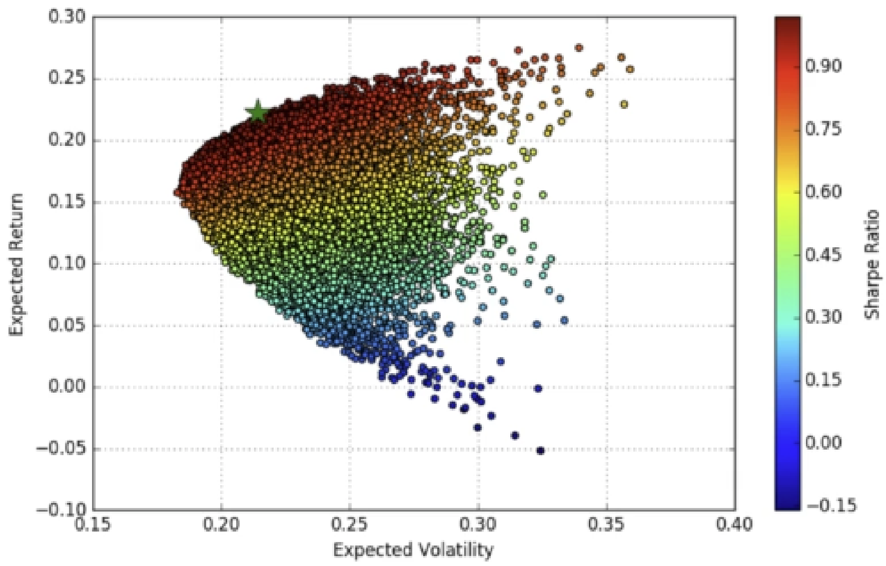
$$\Sigma = \begin{vmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \sigma_{n1} & & \cdots & \sigma_n^2 \end{vmatrix}$$

*this **covariance matrix** contains the relationship between all the assets (in this case stocks) in the portfolio*

    - the covariance matrix Σ is symmetrical, meaning $i = j$
    - expected variance of the portfolio can be expressed as
    - $\sigma^2_{portfolio} = E\left[\left(r_i - \mu_i\right)^2\right] = \Sigma^i\Sigma^j w_i w_j \sigma_{ij} = w^T \Sigma w$
    - $\sigma^2_{portfolio} = w^T \Sigma w \Rightarrow$ where Σ is the covariance matrix

Efficient Frontier

- recap:
    - we discussed how to compute the expected return and the expected volatility|risk
    - investors want
        - 1. maximum return (mean) given a fixed risk level (volatility)
        - 2. minimum risk given a fixed return
- efficient frontier
    - we can plot this data on a graph, where every dot represents different allocation weights of a given portfolio containing multiple stocks
    - generating these weights randomly is done through a Monte-Carlo Simulation
    - these portfolios show the so-called efficient frontier
    - with Markowitz model, investors can find the trade-off between the risk and the return



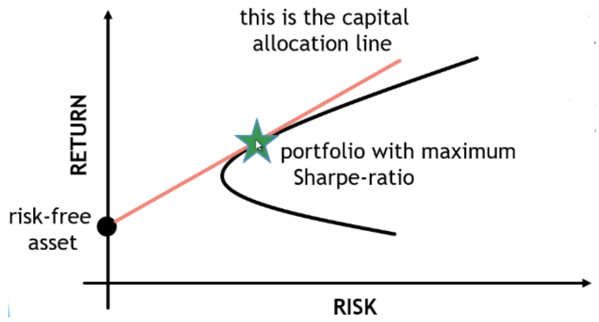
- the optimal portfolio
    - either minimizes the risk at a given level of return (moving on horizontal axis)

- or maximizes return at given level of risk (moving on vertical axis)
- and the investor can decide

Sharpe Ratio

- how can we mathematically determine the trade-off between return and risk?
- Sharpe ratio
  - defined William Sharpe
  - describes how much excess return you are receiving for extra volatility that you endure holding a riskier asset
  - $S(x) = \frac{r_x - R_f}{\sigma(x)}$
  - where
    - r_x is the average rate of return of investment x
    - R_f is the rate of return of risk-free security (like a bond or interest rate for lending)
  - $S(x) > 1$ is considered to be good,
- the optimal portfolio has the highest Sharpe ratio and lies the efficient frontier

Capital Allocation Line



- whats is the capital allocation line?
- we have learned that the optimal portfolio from the simulation is the one with maximum Sharpe ratio and one the efficient frontier
- the optimal portfolio is where the efficient frontier and the capital allocation line touch, if the portfolio contains also risk-free assets
- given a risk-free asset (such as a government bond), the expected return is non-zero but the risk is close to zero

Quiz: Markowitz Model

- 1. main idea of portfolio theory?
  - combining several assets within portfolio can minimize risk
- 2. Need historical data?
  - Yes
- 3. Most important parameter
  - mean and variance (corresponding to reward and risk)
- 4. What is the Sharpe ratio
  - measures excess return per unit of risk
- 5. Efficient portfolios (portfolios that have maximum return for a given risk, or lowest level of risk for a fixed return) are on the capital allocation line
  - Yes

7. Markowitz Model Implementation

Implementation I

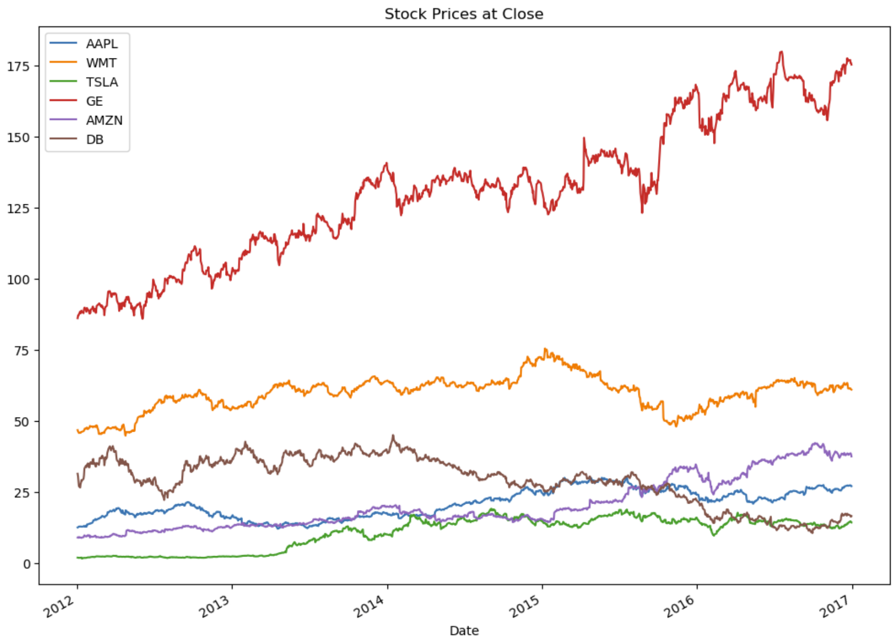
- Downloading the dataset
  - stocks AAPL, WMT, TSLA, GE, AMZN, DB
  - timeframe: 2012-2017

```python
def download_data(stocks: list[str], start_date: str, end_date: str) -> pd.DataFrame
    stock_data = {}
    for stock in stocks:
        ticker = yf.Ticker(stock)
        stock_data[stock] = ticker.history(start=start_date, end=end_date)["Close"]
    return pd.DataFrame(stock_data)
```
✓ 0.0s

```python
# on average there are 252 trading days in a year
NUM_TRADING_DAYS = 252

# stocks we are going to handle
stocks = ['AAPL', 'WMT', 'TSLA', 'GE', 'AMZN', 'DB']

# historical data - define START and END dates
start_date = '2012-01-01'
end_date = '2017-01-01'

data = download_data(stocks, start_date, end_date)
data.head()
```
✓ 2.9s

	AAPL	WMT	TSLA	GE	AMZN	DB
Date						
2012-01-03 00:00:00-05:00	12.482925	46.785137	1.872000	86.152100	8.9515	31.421204
2012-01-04 00:00:00-05:00	12.550011	46.304337	1.847333	87.090576	8.8755	30.474541

- Visualize the Stock Price

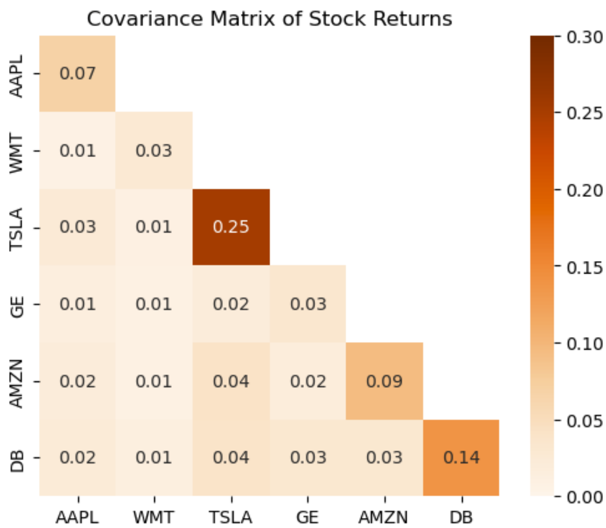


Implementation II

- calculating statistics

```python
def calculate_log_daily_return(data: pd.DataFrame) -> np.ndarray:
    # today's stock price over yesterday's stock price
    # return = ln( S(t) / S(t-1) )
    log_return = np.log(data / data.shift(1))
    return log_return[1:]


def show_statistics(returns: np.ndarray, n_trading_days: int):
    # instead of daily metrics we are after annual metrics
    # mean of annual return
    print("Returns:")
    print(returns.mean() * n_trading_days)
    print()
    print("Covariance:")
    print(returns.cov() * n_trading_days)


def show_mean_variance(returns: np.ndarray, weights: np.ndarray, n_trading_days: int) -> tuple[float, float]:
    # we want the annual return
    portfolio_return = np.sum(returns.mean() * weights) * n_trading_days
    portfolio_volatility = np.sqrt(
        np.dot(weights.T, np.dot(returns.cov() * n_trading_days, weights))
    )
    # print("Expected portfolio mean (return): ", portfolio_return)
    # print("Expected portfolio volatility (standard deviation): ", portfolio_volatility)
    return portfolio_return, portfolio_volatility
```
0.0s

- Visualize Covariance Matrix
    - covariance of a stock with itself is its variance (diagonal of matrix)

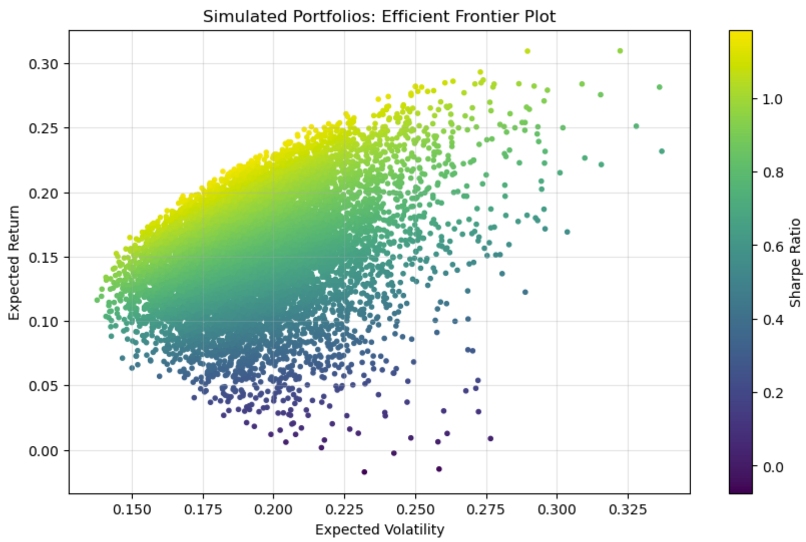


Implementation III

- generating many portfolios (N=10000)
    - randomly generate weights for each portfolio
    - compute mu, sigma for the portfolio based on its weights

```python
def generate_portfolios(returns, stocks, n_portfolios, n_trading_days):
    portfolio_means = []
    portfolio_risks = []
    portfolio_weigths = []

    for _ in range(n_portfolios):
        w = np.random.random(len(stocks))
        w /= np.sum(w) # normalize as %
        portfolio_weigths.append(w)
        mu, sigma = show_mean_variance(returns, w, n_trading_days)
        portfolio_means.append(mu)
        portfolio_risks.append(sigma)
        portfolio_weigths.append(
            np.sqrt(np.dot(w.T, np.dot(returns.cov() * n_trading_days, w)))
        )

    return (
        np.array(portfolio_means),
        np.array(portfolio_risks),
        np.array(portfolio_weigths),
    )


def sharpe_plot(returns, volatilities):
    plt.figure(figsize=(10,6))
    plt.scatter(volatilities, returns, c=returns/volatilities, marker=".")
    plt.grid(alpha=0.33)
    plt.title("Simulated Portfolios: Efficient Frontier Plot")
    plt.xlabel("Expected Volatility")
    plt.ylabel("Expected Return")
    plt.colorbar(label="Sharpe Ratio");
```
✓ 0.0s

- visualize the simulated portfolios on a plot
  - color them by their Sharpe ratio = returns / volatilities



Implementation IV

- find the optimal portfolio through optimization
  - scipy can find the minimum of a given function f(x)
  - the maximum of f(x) is the minimum of -f(x)
  - to find maximum Sharpe ratio portfolio, optimize for negative of minimum Sharpe function

```python
def portfolio_statistics(weights: np.ndarray, returns: np.ndarray) -> np.ndarray:
    portfolio_return = np.sum(returns.mean() * weights) * NUM_TRADING_DAYS
    portfolio_volatility = np.sqrt(
        np.dot(weights.T, np.dot(returns.cov() * NUM_TRADING_DAYS, weights))
    )
    # last value is Sharpe ratio = r_t / s_t
    return np.array([portfolio_return, portfolio_volatility, (portfolio_return/portfolio_volatility)])

# scipy optimize module can find the minimum of a given function
# the maximim of a f(x) is the minimum of -f(x)
def min_function_sharpe(weights: np.ndarray, returns: np.ndarray):
    return -portfolio_statistics(weights, returns)[2]

# what are the constraints? sum of weights = 1.0
# and weight bounds 0.0 <= w_i <= 1.0 for any given stock S_i
# f(x)=0 this the function to minimize
def optimize(n_stocks: int, weights: np.ndarray, returns: np.ndarray) -> optimization.OptimizeResult:
    constraints = {"type": "eq", "fun": lambda x: np.sum(x)-1}
    bounds = tuple((0, 1) for _ in range(n_stocks))
    solution = optimization.minimize(fun=min_function_sharpe, x0=weights[0],
                                     args=returns, method="SLSQP", constraints=constraints, bounds=bounds)
    # acces solution["x"]
    return solution

def print_optimal_portfolio(solution: optimization.OptimizeResult, stocks: list[str], returns: np.ndarray):
    weights = solution["x"]
    mu, sigma, sharpe = portfolio_statistics(weights, returns)
    portfolio = [(s, round(w, 3)) for s, w in zip(stocks, weights)]

    print(f"Optimal Portfolio:\n{portfolio}\n")
    print(f"Expected Return: {mu:.3f}")
    print(f"Volatility: {sigma:.3f}")
    print(f"Sharpe Ratio: {sharpe:.3f}")

```

✓ 0.0s

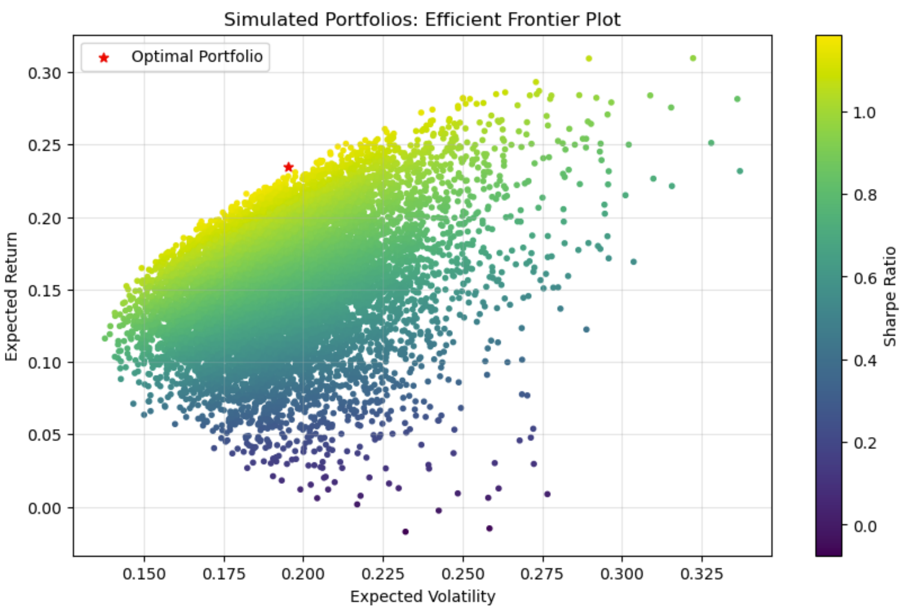
- optimal portfolio

```python
solution = optimize(len(stocks), p_weights, log_returns)

print_optimal_portfolio(solution, stocks, log_returns)
```

✓ 0.1s

```
Optimal Portfolio:
[('AAPL', 0.139), ('WMT', 0.0), ('TSLA', 0.166), ('GE', 0.373), ('AMZN', 0.321), ('DB', 0.0)]

Expected Return: 0.235
Volatility: 0.195
Sharpe Ratio: 1.201
```

- visualize the optimal portfolio



8. Capital Asset Pricing Model (CAPM) Theory

Systematic and Unsystematic Risk

- risk & risk management at the stock market
  - holding a single stock is quite risky
  - holding several stocks in a portfolio can reduce risk
  - the optimal portfolio can be found using Markowitz model
  - BUT: we cannot get rid of all risk
- risk in finance
  - ***unsystematic (specific) risk***
    - a risk specific to individual stocks
    - can be minimized by diversifying / holding multiple stocks
    - unsystematic risk component is random or uncorrelated with market movements
  - ***systematic (market) risk***
    - a risk inherent to the market or system
    - it cannot be diversified away
    - includes interest rates changes, recession, wars, etc.
    - the CAPM model measures this with **β** parameter

Capital Asset Pricing Model Formula

- we discussed the two different types of risk in finance:
  - 1) unsystematic risk → inherent to a single asset
  - 2) systematic risk → inherent to an entire market
    - CAPM model measures this with **β** parameter
- Capital Asset Pricing Model
  - first formulated by William Sharpe in 1960s
  - defines a linear relationship between any expected stock return and the market premium

$$E[r_a] = r_f + \beta_a \times (E[r_m] - r_f)$$

  - $E[r_a]$: expected return of an investment (stock or portfolio)
  - r_f : base return at risk-free rate (e.g. a government bond)
  - $\beta_a \times (E[r_m] - r_f)$: market excess return or market premium multiplied by β_a
  - a market, for instance, can be the S&P 500, i.e. top 500 stock-listed companies
- **β** parameter

$$\beta_a = \frac{cov(r_a, r_m)}{var(r_m)}$$

  - β_a : defines ***how risky your investment is*** relative to the market
  - according to CAPM model, **β** is the only relevant measure of risk
  - **β** is the investment's relative risk, i.e. how much the price of a given stock goes up or down compared to that of the whole market
  - tells investors how much risk to take for a given return

The Beta Value

- calculating **β** parameter for CAPM model

$$\beta_a = \frac{cov(r_a, r_m)}{var(r_m)}$$

- understanding the beta parameter
  - **β** defines ***how risky your investment is*** relative to the market
  - if your portfolio has no risk then your expected return is the risk-free return (same as lending out your money to a bank or government)
  - if your portfolio is more risky than the market then your expected return will also be higher
  - a portfolio less risky than the market has expected returns
- β general meaning
  - $\beta = 1$: stock is moving exactly with market
  - $\beta > 1$: stock is more volatile than the market (more expected return)
  - $\beta < 1$: stock is less volatile than the market (less expected return)
- β specific interpretation
  - $\beta = 0.5$
    - less volatile than the market
    - if stock market goes up by 50%, this stock goes up by 5%
    - if stock market falls by 2%, this stock falls by 1%
  - $\beta = 1.5$
    - more volatile than the market
    - if stock market goes up by 10%, this stock goes up by 15%
    - if stock market falls by 2%, this stock falls by 3%
- how to calculate betas for a set of stocks (portfolio)?
  - a given portfolio's β value is the weighted sum of the stocks' β within the portfolio:

$$\beta_a = w_1\beta_1 + w_1\beta_1 + ... + w_n\beta_n$$

  - these are the weights we optimized with the Markowitz model in the last section
  - assumption being that all positions are long positions of securities such that 100% of the wealth can be divided among the selected assets

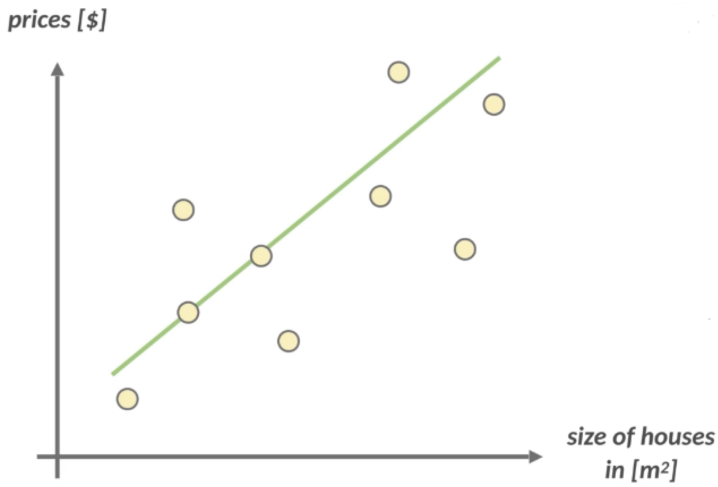
What is Linear Regression?

- linear regression is an approach for modeling the relationship between a dependent variable y and one or more explanatory variables x
- the relationship is a linear combination of coefficients β_m and the explanatory variables x_m, where m is the number of variables
- simple vs multiple linear regression

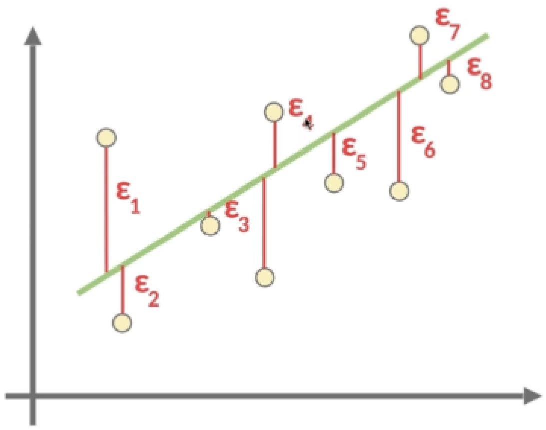
Simple Linear Regression	Multiple Linear Regression
- there is just a single x explanatory variable - e.g. we want to predict house prices based on the size of the house	- there are multiple x explanatory variable - e.g. we want to predict house prices based on the size, age, n of rooms, etc.

$y = H(x) \approx b_0 + b_1 x$	$y = H(x) \approx b_0 + b_1 x_1 + ... + b_m x_m$

- visualizing simple linear regression
  - the line describes the relationship between the size of the house and its size
  - read: the house price is a function of the house's size



- best fitting line



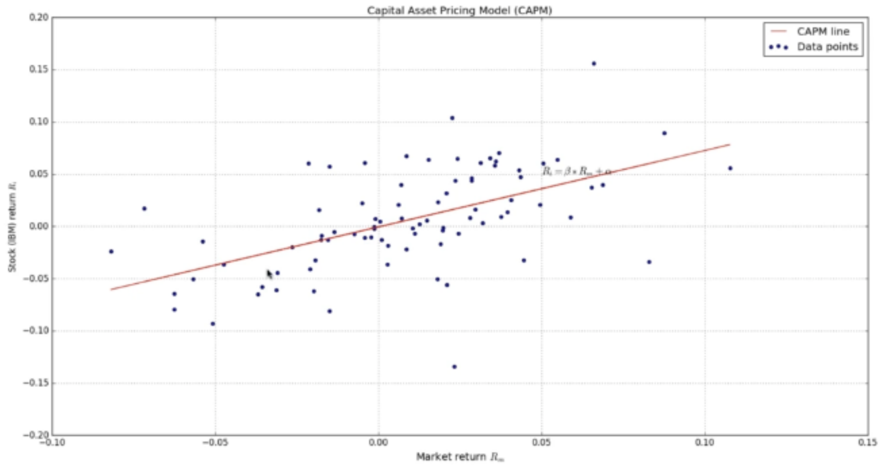
  - the best fit minimizes the difference between the actual values y and the predicted values $\hat{y}$
- MSE
  - the difference are also called error and measured in a cost function for the model, e.g. the mean squared error (MSE)
  - $MSE = \frac{1}{n}\sum_{i=1}^{n}(y_n - \hat{y}_n)^2$
  - squaring the differences helps avoid that negative and positive errors cancel each other out
  - the smaller the MSE, the better the model and vice versa
  - better for optimizing model's mathematically, but not easy to read for humans
- MAE
  - similar to MSE but instead uses the absolute difference the actual values y and the predicted values $\hat{y}$

- $MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_n - \hat{y}_n\right|$
- easier to interpret for humans

- R-squared
  - another common error metric for linear regression is the R-squared statistic
  - $R^2 = 1 - \frac{RSS}{TSS}$
  - it measured the explained variance of the model
    - $0 \le R^2 \le 1$, the closer to 1, the better the model
  - RSS → residual sum of squares
    - measures the variability left unexplained after performing the regression
    - $RSS = \sum_{i=1}^{n}(y_n - \hat{y}_n)^2$
  - TSS → total sum of squares
    - measures the total variance in y_n
    - $TSS = \sum_{i=1}^{n}(y_n - \bar{y})^2$

CAPM and Linear Regression?

- how can linear regression be used to help calculate the beta(s) of the CAPM model?
  - $E[r_a] = \beta_a \times (E[r_m] - r_f)$
  - if we subtract r_f from the capital asset pricing model
  - $E[r_a] - r_f = \alpha + \beta_a \times (E[r_m] - r_f)$
  - where β_a will be the coefficient of the linear regression (fitting without intercept, meaning $\alpha = 0$)



- **α** alpha parameter
  - alpha is the difference between the return and the expected return
  - $\alpha = E[r_a] - \left(r_f + \beta_a\left(E[r_m] - r_f\right)\right)$
  - fitting the linear model without intercept means $\alpha = 0$

Quiz: Capital Asset Pricing Model

- What are the 2 most important risks in finance?
    - systematic and unsystematic risk
- What is the main idea behind the CAPM model?
    - because we can alleviate unsystematic risk through diversification, the only relevant risk in market risk
- What is the beta parameter?
    - it measures how risk your portfolio relative to the market
- Why use linear regression for CAPM model?
    - to get beta parameter as slope of the regression

9. Capital Asset Pricing Model Implementation

Implementation I

- creating a Python class for the CAPM
    - with downloading and preparing the data methods

```python
@dataclass
class CAPM:
    start_date: str
    end_date: str
    stocks: list[str] = field(default_factory=list)
    data: pd.DataFrame = None

    def download_data(self):
        data = {}
        for stock in self.stocks:
            ticker = yf.download(stock, self.start_date, self.end_date)
            data[stock] = ticker["Adj Close"]
        # resample to monthly data at end of month
        self.data = pd.DataFrame(data)

    def prepare_data(self):
        # resample to monthly returns instead of daily returns
        stock_data = self.data.resample('M').last()

        self.data = pd.DataFrame({'s_adjclose': stock_data[self.stocks[0]],
                                  'm_adjclose': stock_data[self.stocks[1]]})

        # logarithmic monthly returns = r_[t-1] / r_[t]
        self.data[['s_returns', 'm_returns']] = np.log(self.data[['s_adjclose', 'm_adjclose']] /
                                                        self.data[['s_adjclose', 'm_adjclose']].shift(1))

        # remove the NaN values
        self.data = self.data[1:]
```

- download IBM and S&P500 data from 2010 to 2017

```python
stocks = ["IBM", "^GSPC"] # ^GSPC is the S&P500 index
start_date = "2009-12-01"
end_date = "2017-01-01"

capm = CAPM(start_date, end_date, stocks)
capm.download_data()
capm.data.head()
```

✓ 1.2s

```
[*********************100%***********************]  1 of 1 completed
[*********************100%***********************]  1 of 1 completed
```

	IBM	^GSPC
Date		
2009-12-01	76.549019	1108.859985
2009-12-02	76.112259	1109.239990
2009-12-03	76.315712	1099.920044
2009-12-04	76.136192	1105.979980
2009-12-07	76.010567	1103.250000

- Frequency of Time Series Data
    - market data can have very high frequencies depending on the traded asset
    - daily stock returns can take into account holidays and are good for short-term forecasting
    - monthly stock returns are approximately normally distributed and better for long-term models
    - most traditional statistical models like linear regression assume normally distributed data
- prepare monthly returns for the stock and the market

```python
capm.prepare_data()
capm.data.head()
```

✓ 0.1s

	s_adjclose	m_adjclose	s_returns	m_returns
Date				
2010-01-31	73.228363	1073.869995	-0.067221	-0.037675
2010-02-28	76.422630	1104.489990	0.042696	0.028115
2010-03-31	77.077713	1169.430054	0.008535	0.057133
2010-04-30	77.528442	1186.689941	0.005831	0.014651
2010-05-31	75.666618	1089.410034	-0.024308	-0.085532

Implementation II

- compute covariance matrix and then beta
    - add a method for computing covariance and beta

```python
    def calculate_beta(self):
        # covariance matrix: diagonal are the variances
        # off diagonal values are covariances
        covariance_matrix = np.cov(self.data["s_returns"], self.data["m_returns"])
        # beta = cov(r_a, r_m) / var(r_m)
        beta = covariance_matrix[0, 1] / covariance_matrix[1, 1]
        return beta
```

✓ 0.1s

- calculate beta

Interpreting Beta

- $\beta = 1$: stock is moving exactly like the market
- $\beta > 1$: stock is more volatile than the market
- $\beta < 1$: stock is less volatile than the market

```
beta = capm.calculate_beta()
print(f"Beta for IBM with S&P500 is {beta:.2f}")
```
✓ 0.1s

```
Beta for IBM with S&P500 is 0.73
```

- beta is 0.73 which means the IBM stock is less volatile than the market but also promises less returns

Implementation III

- implementing linear regression for finding beta

CAPM Formula rewritten for Linear Regression

$$E[r_a] - r_f = \alpha + \beta(E[r_m] - r_f)$$

```
def linear_regression(data, r_f=0.0, month_in_year=12):
    # using linear regression to fit a line to the data
    # [stock_returns, market_returns] - slope is the beta
    beta, alpha = np.polyfit(data['m_returns'], data['s_returns'], deg=1) # deg is order of polynomial
    alpha = alpha if alpha else 0.0
    # calculate the expected return according to the CAPM formula
    # we are after annual return (this is why multiply by 12)
    expected_return = r_f + beta * (data['m_returns'].mean()*month_in_year - r_f)
    return (alpha, beta), expected_return
```
✓ 0.0s

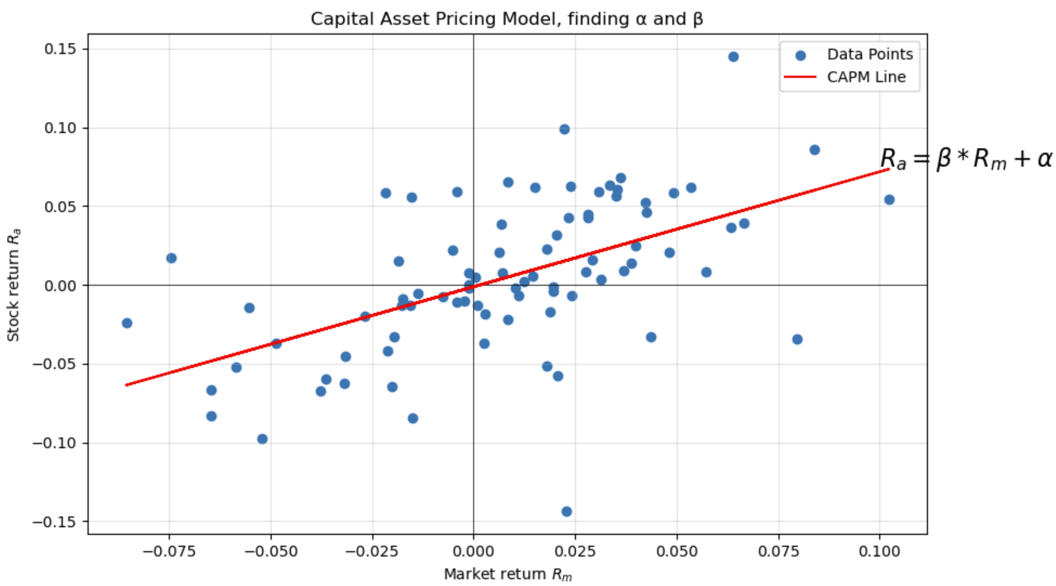
- compute alpha and beta with linear regression

```
coeff, expected_return = linear_regression(capm.data)

print(f"Alpha = {coeff[0]:.2f}")
print(f"Beta = {coeff[1]:.2f}")
print(f"Expected Return = {expected_return:.2f}")
```
✓ 0.1s

```
Alpha = -0.00
Beta = 0.73
Expected Return = 0.07
```

- plot the regression model

```
def plot_regression(data, coeff):
    alpha, beta = coeff
    _, ax = plt.subplots(1, figsize=(10, 6))
    ax.axhline(lw=0.5, c="k")
    ax.axvline(lw=0.5, c="k")
    ax.scatter(data["m_returns"], data['s_returns'], label="Data Points")
    ax.plot(data["m_returns"], beta * data["m_returns"] + alpha, color='red', label="CAPM Line")
    plt.title('Capital Asset Pricing Model, finding α and β')
    plt.xlabel('Market return $R_m$')
    plt.ylabel('Stock return $R_a$')
    plt.text(0.1, 0.075, r'$R_a = \beta * R_m + \alpha$', fontsize=16)
    plt.legend()
    plt.grid(alpha=0.33)
    plt.show()

plot_regression(capm.data, coeff)
```
✓ 0.5s

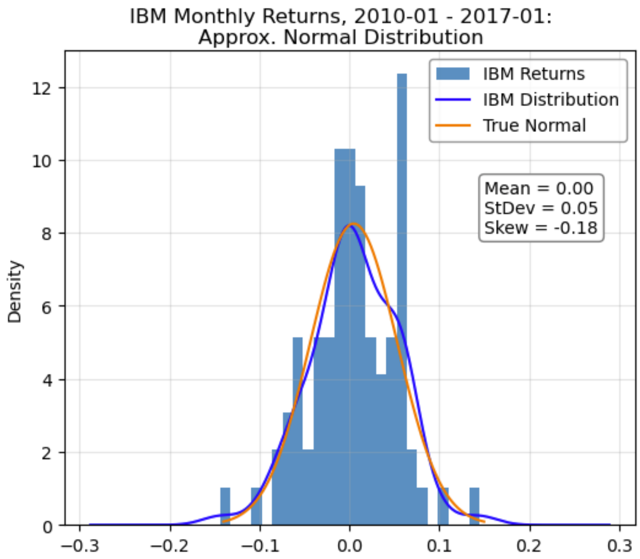


Exercise - Normal Distribution of Returns

Solution - Normal Distribution of Returns

- check if the monthly log returns of the IBM stock are approximately normally distributed
    - calculate mean, standard deviation, and skew
    - plot as histogram and PDF → compare to true normal for these parameters

```python
ibm_returns = capm.data["s_returns"]
mu = ibm_returns.mean()
sigma = ibm_returns.std()
skew = ibm_returns.skew()

fig, ax = plt.subplots(1,1, figsize=(6, 5))
fig.suptitle("IBM Monthly Returns, 2010-01 - 2017-01:\nApprox. Normal Distribution", y=0.95)

# plot IBM data
ibm_returns.plot(kind="hist", bins=25, density=True, ax=ax, alpha=0.8, label="IBM Returns")
ibm_returns.plot(kind="kde", ax=ax, color="b", label="IBM Distribution")

# plot true normal distribution
from scipy.stats import norm
x = np.linspace(mu - 3 * sigma, mu + 3 * sigma, 100)
plt.plot(x, norm.pdf(x, mu, sigma), label="True Normal")

ax.grid(alpha=.33)
props = dict(boxstyle='round', facecolor="white",  edgecolor="gray")
ax.text(0.15, 8, f"Mean = {mu:.2f}\nStDev = {sigma:.2f}\nSkew = {skew:.2f}", bbox=props)
ax.legend(edgecolor="gray");
```



10. Derivative Basics

Introduction to Derivatives

- derivative definition
    - ~ is security with a price that is **derived** from one or more **underlying assets**
    - value of the ~ is determined by fluctuation in the underlying asset
    - underlying asset may be stock, bonds, currencies, or interest rates
- benefits of derivatives
    - **1) hedging risk** in physical markets and decrease exposure
    - **2) speculation** as to increase exposure and chance of profit
    - **3) market participation** as it makes difficult items tradable
- example: hedging
    - an airline company has its largest operating cost in fuel expenses
    - fluctuating fuel (oil) prices have impact on the P&L
    - commodities such as oil are extremely volatile
        - falling costs are good for the consumers of commodity
        - rising costs are good for the producers of commodity
    - the airline may sign a contract that the oil company will sell x barrels of oil to the airline at the current market price $\$m$ in n months regardless of future prices
    - that makes operating costs more predictable for the airline
- example: speculation
    - we may want use derivative and buy oil at today's price in the future because we expect the price to increase (and want to profit)
- main types of derivatives
- 1) Forward Contract
    - a contract between two parties to buy/sell a given asset at a given price in the future (settled upon maturity)
    - they are not regulated and customizable OTC contracts
    - they are private and not available on a market place
- 2) Future Contract

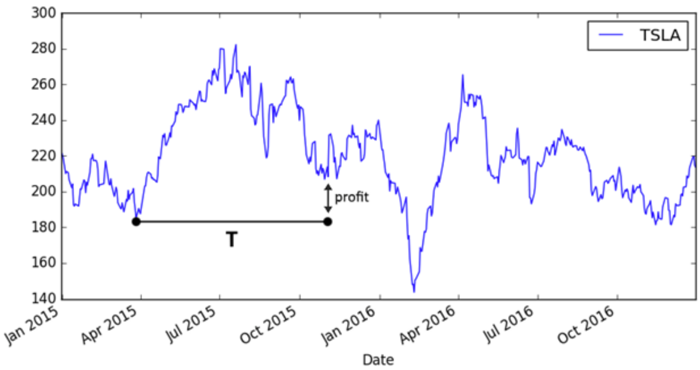
  - a contract between two parties to buy/sell a given asset at a given price (can be settled on a daily basis)
  - BUT they are standardized, regulated and can be traded on future markets
- 3) Swap
  - a contract where 2 parties agree to exchange assets or cash flows
  - most common form is the interest rate swap, where a fixed rate is exchanged against a floating rate
- 4) Options
  - a contract that gives the right (without obligation) to buy/sell a given asset at a given price on the future

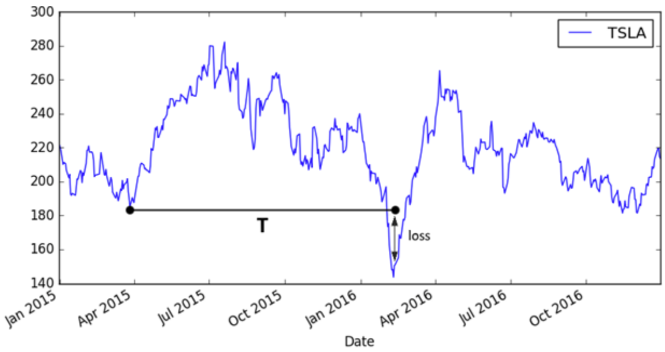
Forward and Future Contracts

- commonalities and differences forwards and futures

Forward Contract	Future Contract
- private agreement between 2 parties - buy/sell a given asset in the future of specific price - agreement is private, customizable, and not traded at an exchange - forwards are not regulated or standardized	- private agreement between 2 parties - buy/sell a given asset in the future of specific price - contract is regulated and standardized - futures are traded on exchanges and transactions are guaranteed (by exchange/clearing house)

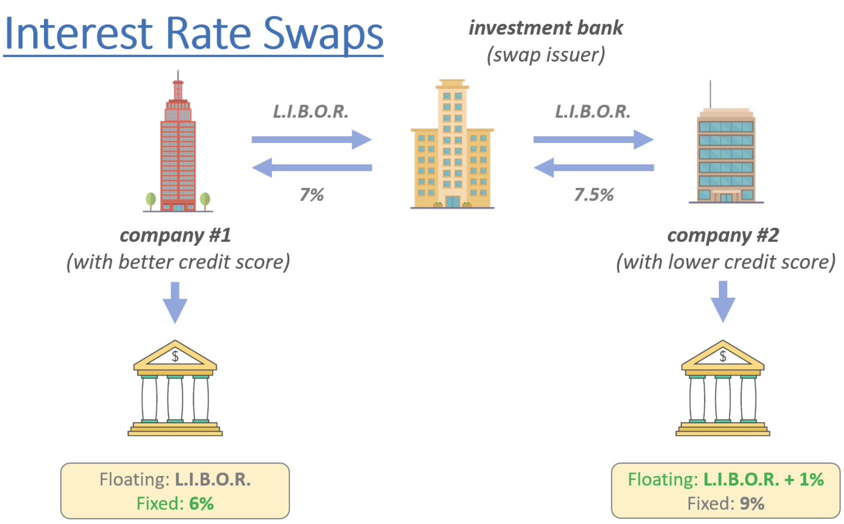
- Example: Future Contract
  - Tom owns 5,000 IBM shares valued at $50 per share
  - Tom fears that the value of the shares will decline and arranges future contract to protect the value of his stock
  - Bob speculates that the IBM stocks will rise in value and agrees with Tom to buy his 5000 shares for the current value of $50 in 1 year
  - one party engages in hedging risk while the other engages in speculation





Swaps and Interest Rate Swaps

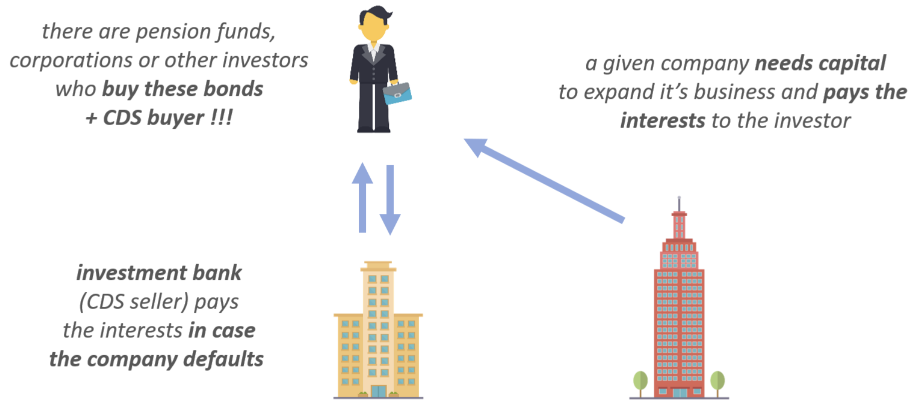
- swaps are a financial contract between two parties to exchange financial assets, cashflows or payments for a certain time
- first swap was constructed in 1981 between IBM and the World Bank
- the objective of a swap is to change one scheme of payments into another one of different nature
  - e.g. floating rate and fixed rate payments
- swap types
- 1) Interest Rate Swap
  - most common type of swap
  - 2 parties, where one pays fixed interest rate and the other floating interest rate
- 2) Currency Swap
  - both parties agree to exchange principal and interest payments on debt that is denominated in different currencies
- 3) Credit Default Swap (CDS)
  - an agreement where a party pays the principal and interest to the CDS buyer (lender) in case the original borrower default on the loan
- Interest Rate Risk & - Swaps
  - interest rate risk comes from market rates fluctuating over time
  - banks and companies that have loans depend on the interest rate, even with fixed rates loans
    - remember that bond prices fall if interest rates increase
  - a given company can borrow money from a bank or investors
  - the company agrees to pay interests to the lender (every company has different a credit score so the price of borrowing varies too)
- example



- from the main bank
  - company #1 can borrow at the LIBOR-rate or 6% fixed because of its good credit score
  - company #2 can borrow at LIBOR-rate +1% or 9% because of its lower credit scores
- at the investment bank
  - the investment bank can make a profit of 0.5% from offering the swaps
  - company #2 gets rid of the LIBOR-rate and now has to pay only 8.5% interest in total (7.5% to investment bank and 1% to main bank)
    - it benefits because 8.5 < 9
  - company #1 is now only paying LIBOR - 1%, which is better than fixed 6% or LIBOR ±0%

Credit Default Swaps

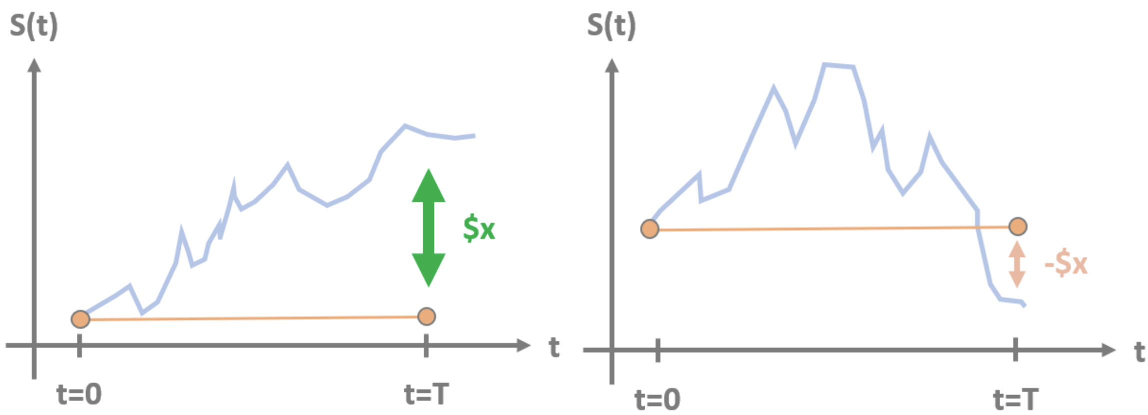
- raising funds through bonds
  - a given company need capital to expand its business
  - it issues bond that pay interest for a certain period of time
  - large investors like pension funds, corporations, etc. buy these bonds
  - then the bond issuer defaults on the bond payments
  - large investors want insurance against this risk
- CDS
  - a credit default swap is a derivative (swap contract) that the seller of the CDS will compensate the buyer in the event of debt default
  - it is a derivative that enables investors to swap credit risk
  - the buyer pays installments (fixed payments) to the seller until date of maturity of the underlying loan
  - the seller agrees to pay the interest payment (and principal) if the underlying loan defaults
- example



- ○ large investors buy CDS from investment banks at a premium: the higher the probability of default, the insurance premium
- ○ from the perspective of the investment bank, most bond issuers do not default on their payment and so the bank earns the installments

Option Basics

- options are the most popular form of derivative
- they are contracts that give the bearer the right, but not the obligation, to either buy/sell an amount of some underlying asset (usually stocks)
- with forward and future contracts, the buyer is obliged to fulfill their obligation even if the desired outcome has not arrived (and the seller must deliver)
- with an option, the buyer would only exercise their right if a profitable situation would occur
- a stock option
    - if we have bought a stock option, it gives the right to buy the underlying stock in the future
    - we will exercise the right to buy, if the stock increased in value
    - if the underlying stock decreased in value, we will not exercise the right

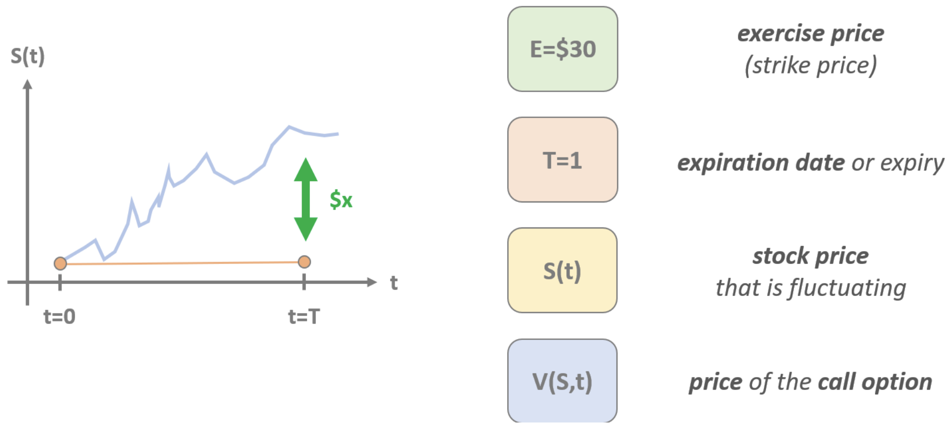


- how much would you pay for an option?
    - complicated question because the price depends on the underlying asset and the asset's price fluctuate
    - the famous **Black-Scholes model** can calculate the value of an option

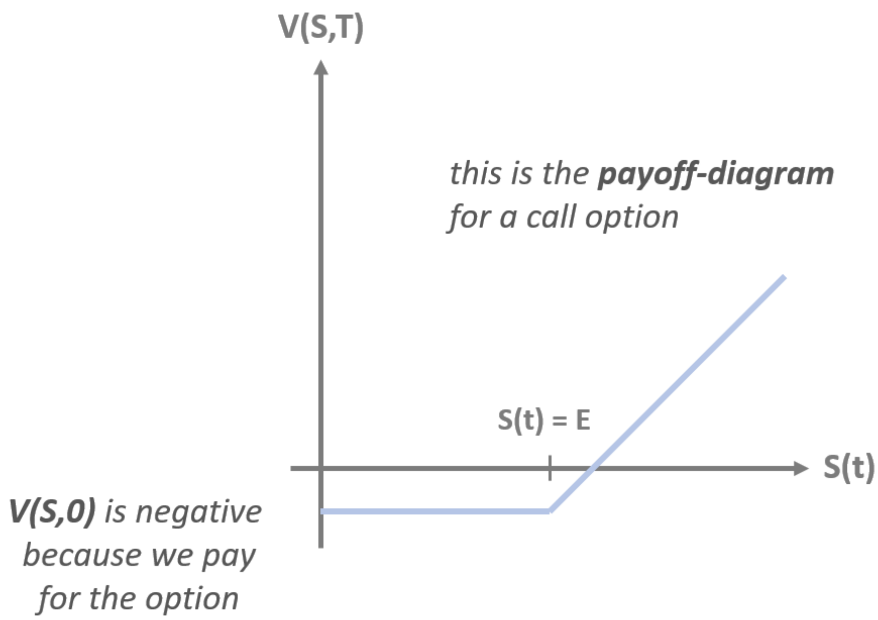
Call Option

- for coming lectures, we'll assume the underlying asset is always a stock

- a call option is the right to buy a particular asset for an agreed price E (exercise or strike price) at the specific time T in the future
- thus, with call option, the buyer speculates on rising price of underlying
- example
  - you can buy a call option on IBM shares, with right to buy a share at $E = \$30$ in 1 month; today's price is \$25

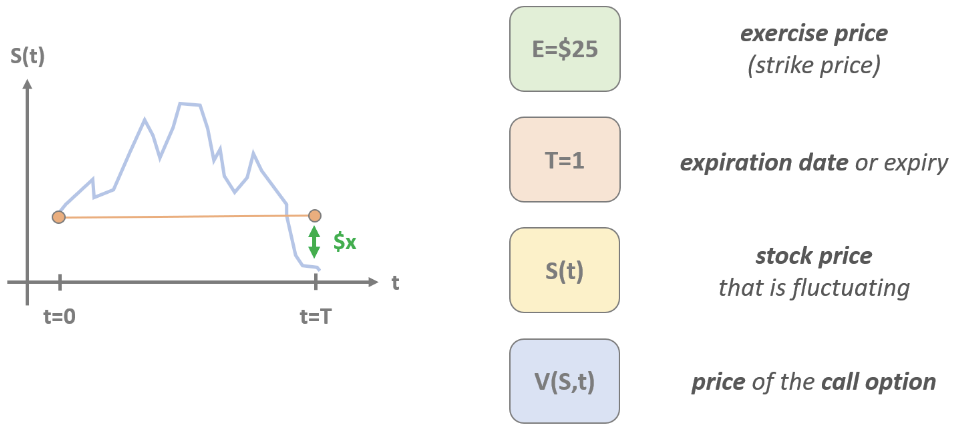


  - pricing an option is complicated because the price of the underlying asset fluctuates
  - we'll learn about stochastic differential equations to calculate the value of the options $V(S, t)$
- pay-off diagram
  - at $t = 0$, the pay-off of for the call-option is negative, because you just paid the price of the option and the stock price is less than the strike price $S(0) < E$
  - a profit starts when the stock price rises over the strike price $S(t) > E$ and the profit will be $S(t) - E$
  - at expiry T, the price of the option is $V = max(S(T) - E, 0)$
  - if the stock price never exceeds the strike price, we would not exercise the call-option

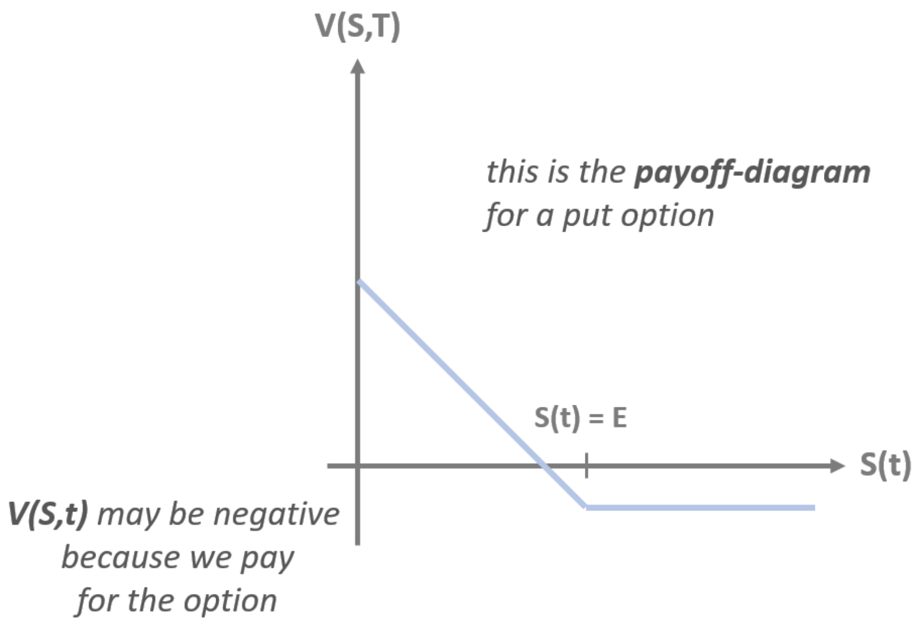


Put Option

- a put option is the right to sell a particular asset for an agreed price E at a specific time T in the future
- thus, with put option, the buyer speculates on falling price of underlying
- example
  - you can buy a put option on IBM shares, with right to sell a share at $E = \$25$ in 1 month; today's price is \$25



- pay-off diagram
  - at expiry T, the price of the option is $V = max(E - S(T), 0)$
  - we make a profit if the stock price falls below the strike price $S(t) < E$ and and the profit will be $E - S(t)$



American vs European Option

- the time when you have the right to exercise an option
- 1) American Option: may be exercised at any time before the expiration $0 < t < T$
- 2) European Option: may be exercised only at the expiration date T
- 3) Bermudan Option: may be exercised only at set of predefined points until expiration $0 < \{t_1, t_2, ...\} < T$

Quiz: Derivative Basics

- What is the main difference between options and futures?
    - holder of future contract is obliged, while holder of option has right to exercise
- What is a call option?
    - right to buy an underlying asset at maturity
- What is a put option?
    - right to sell an underlying asset at maturity
- What is the difference between American and European options?
    - American option may be exercised at any $t < T$ but European option may only be exercised at expiration T
- How to calculate the price $V(S, t)$ of an option before expiry?
    - Black Scholes Model

11. Random Behavior in Finance

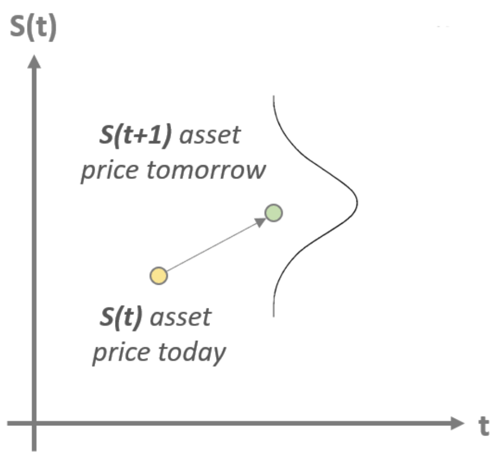
Types of Analysis

- 3 main types
    - 1) Fundamental Analysis
    - 2) Technical Analysis
    - 3) Quantitative Analysis
- 1. Fundamental Analysis
    - is about in-depth study of a given company
    - there are many factors to consider: management style, products and services, balance sheets, income statements, etc.
    - we predict whether the stock is over-/undervalued based on the intrinsic value of the company
    - we may have a good model for the value of a company; but the rest of the world should see the mispricing too
    - fundamental analysis usually consists of checking business metrics
        - assets: cash, properties, equipment
        - liabilities: debts and loans
        - income statement: revenues and expenses, cashflows
        - *net income = revenues - expenses*
        - company can share the net income with shareholder (dividends) or reinvest them
    - a good cash position means very low probability of default
    - P/E ratio (price to earnings) is powerful tool of fundamental analysis
        - $EPS = \frac{Net\ Income}{\#shares}$ earnings per share
        - $P/E = \frac{stock\ price}{EPS}$ price to earning → the lower the better
- 2. Technical Analysis
    - opposite approach to fundamental analysis as it assumes all information are contained with the stock (price)
    - it is all about analyzing historical data

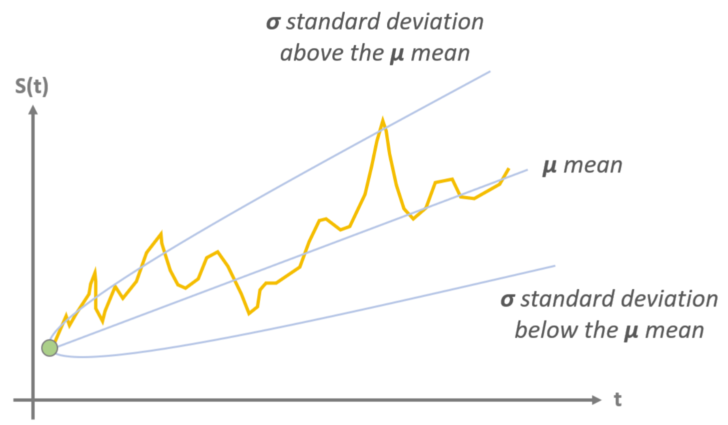
- we interpret historical data and look for specific patterns in the stock price $S(t)$ to make predictions accordingly
    - it can be done with time series models and machine learning
- 3. Quantitative Analysis
    - assumes that all financial quantities such stock prices $S(t)$ or interest rates $r(t)$ have **random behavior**
    - this tries to model the behavior of financial assets while accounting for randomness
    - using randomness in models requires stochastic calculus and differential equations
    - Black-Scholes model is example of quantitative analysis

Random Behavior of Returns

- fluctuations in financial quantities such stock prices $S(t)$ or interest rates $r(t)$ are random
- we have seen that daily or monthly returns have an approximately normal distribution
    - $R(t) = \frac{S(t)-S(t-1)}{S(t-1)}$
    - normal distributions can be defined by two parameters: **μ** mean and **σ²** variance
- thus, daily or monthly returns can be defined with these parameters as well
    - $R(t) = \mu + x * \sigma$
    - returns as random variable are drawn from a normal distribution
- modeling random behavior
    - stock price $S(t)$ can be described by a random walk $W(t)$, the so-called Wiener-Process
    - $W(t)$ has a continuous sample path
    - it has independent and normally distributed increments



- the stock price will fluctuate within the parameters of the normal distribution



Wiener-Processes and Random Walks

- recap: normal distribution
    - the returns of a given stock price is $N\left(\mu, \sigma^2\right)$ normally distributed
    - what is the distribution of the stock prices?
    - standard normal distribution is not working fine as stock prices can not have negative values (and stock values keep increasing in the long-term)
    - → stock prices follow log-normal distribution
    - in probability theory a **log-normal distribution** is a continuous probability distribution of a random variable whose logarithm is normally distributed
    - so if the random variable X is log-normally distributed, then $Y = ln(X)$ has a normal distribution
    - thus, we can incorporate a long-term deterministic trend and random component of price fluctuations
- Wiener Process
    - $W(t)$ is called the Wiener-Process
    - has independent increments:
        - future $W(t + dt) - W(t)$ increments are independent of past values
    - has Gaussian increments
        - $W(t + dt) - W(t)$ is normally distributed with a $\mu = 0$ and $\sigma^2 = dt$
        - $W(t + dt) - W(t) \sim N(0, dt)$
    - we can use this to model the random component of price fluctuations
- Geometric Random Walk

$$dS = \mu \cdot S \cdot dt + \sigma \cdot S \cdot dW$$

    - where
        - $dS = S(t + dt) - S(t)$ is the change in the stock price
        - $\mu \cdot S \cdot dt$ is the deterministic drift
        - $\sigma \cdot S \cdot dW$ is the stochastic part
        - $dW = W(t + dt) - W(t)$ is the change in the Wiener-process
    - dW is random variable drawn from a normal distribution $N(0, dt)$
    - the formula a continuous model of asset prices and stochastic differential equation
    - it is fundamental for modern financial models

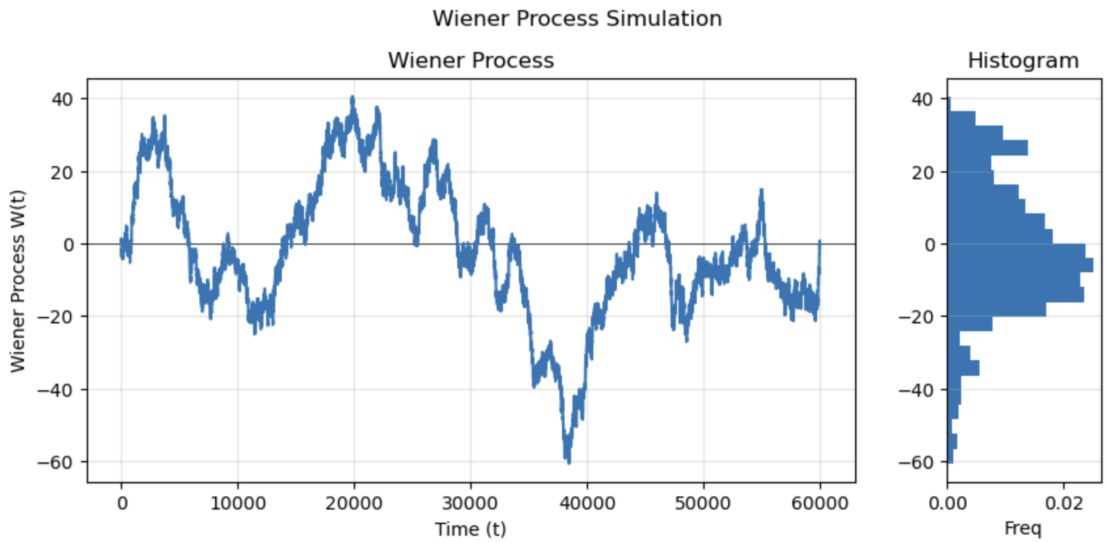
$$\underbrace{dS}_{\substack{\textit{this is the } \boldsymbol{S(t+dt)-S(t)} \\ \textit{change in the} \\ \textit{stock price}}} = \underbrace{\mu\, S\, dt}_{\substack{\textit{deterministic} \\ \textit{part – the } \boldsymbol{drift}}} + \underbrace{\sigma\, S\, dW}_{\substack{\boldsymbol{stochastic} \textit{ part with} \\ \textit{Wiener-process}}}$$

Wiener-Process Implementation

- implement the simulation of the Wiener process
    - set array of all zeros to be filled with the values
    - create a timeline with timesteps dt
    - Wiener process is the cumulative sum of the values drawn from $N(0, \sqrt{dt})$

```python
def wiener_process(dt=0.1, x0=0.0, n=1000) -> tuple[np.ndarray, np.ndarray]:
    """
    Wiener Process Simulation
    Args:
        dt: is change in time
        x0: is initial value
        n: is number of samples
    """

    # initialize W(t) with all zeros
    W = np.zeros(n+1)
    # create the time steps
    t = np.linspace(x0, n, n+1)

    # every new step is drawn from normal dist with N(0,dt)
    # where N(0,dt) = sqrt(dt)*N(0,1)
    W[1:n+1] = np.cumsum(
        npr.normal(0, np.sqrt(dt), n)
    )

    return t, W
```

- plot the simulation



Stochastic Calculus Introduction

- In order to solve stochastic differential equations, we need to learn stochastic calculus

- ordinary functions (with deterministic variables)
  - deterministic function & variables are denoted by lowercase letters

$$f(x) = x^2$$

  - finding the derivative is straight forward

$$\frac{\delta f(x)}{\delta x} = 2x$$

$$\delta f(x) = \frac{\delta f(x)}{\delta x}\delta x = 2x\delta x$$

  - with stochastic random variables this function is not valid
- stochastic calculus
  - random functions & variables are denoted by uppercase letters

$$F(X) = X^2$$

  - $X(t)$ may take random values samples from certain distributions and certain probabilities
  - we have to use **Ito's lemma** to deal with stochastic random variables

$$\delta F(X) = \frac{\delta F(X)}{\delta X}\delta X + \frac{1}{2}\frac{\delta^2 F(X)}{\delta X^2}\delta t$$

  - the change in the function is equal to the derivative of the first order function multiplied by change in X plus the second order derivative term
  - we can get help from the **Taylor series** and **expansion** to solve for $F(X, \delta X)$

$$F(X, \delta X) = F(X) + \frac{\delta F(X)}{\delta X}\delta X + \frac{1}{2}\frac{\delta^2 F(X)}{\delta X^2}\delta t$$

  - for the Wiener-process, we assume that $E[\delta X] = 0$ and $E\left[\delta X^2\right] = dt$ because it has independent increments
- as an exercise for the reader, solve to find

$$\delta F(X) = 2X\delta X + \delta T$$

Ito's Lemma in Higher Dimensions

- why are we talking about stochastic calculus and differential equations
- pricing an option outside of $t = 0$ and $t = T$ is difficult because the underlying asset, i.e. stock price $S(t)$ can fluctuate randomly
- we learned that the price fluctuations have deterministic part and stochastic part, which can be modeled by the Wiener process
  - $dS = \mu \cdot S \cdot dt + \sigma \cdot S \cdot dW$
- for convenience, we can rewrite this formula as
  - $dS(t) = a(S, t)dt + b(S, T)dX$
- If we have another variable $V(S, t)$ depending $S(t)$ for the price of the option, what stochastic equations describes the change in $V(S, t)$
  - a higher-dimension of Ito's lemma is needed

Ito's lemma in higher dimensions:

Suppose we have **dS(t) = a(S,t) dt + b(S,t) dX**. How to define the change in **V(S,t)**? (**S** defines the underlying asset's price, **V** is the option price)

$$V(S+\Delta S,t+\Delta t) = V(S,t) + \frac{\partial V}{\partial t}\Delta t + \frac{\partial V}{\partial S}\Delta S + \frac{1}{2}\Big(\underbrace{\frac{\partial^2 V}{\partial t^2}\Delta t^2 + 2\frac{\partial^2 V}{\partial S \partial t}\Delta S\,\Delta t}_{\text{these terms are small so we can omit them}} + \frac{\partial^2 V}{\partial S^2}\Delta S^2\Big)$$

$$dV(S,t) = \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}dS + \frac{1}{2}\frac{\partial^2 V}{\partial S^2}dS^2$$

$$dS^2 = (a\,dt + b\,dX)^2 = \underbrace{a^2dt^2 + 2\,a\,b\,dt\,dX}_{\text{these terms are small so we can omit them}} + b^2dX^2 = b^2dX^2$$

- we can omit too small terms and summarize the change in the stock price as
  - $dS^2 = b^2 dX^2$
  - $E\left[dX^2\right] = dt$ do that $dX^2 \sim dt$
- now the approximate change in the option price can be modeled as
  - $dV(S,t) = \frac{\partial V}{\partial t}dt + \frac{\partial V}{\partial S}dS + \frac{1}{2}b^2\frac{\partial^2 V}{\partial S^2}dt$

Solving the Geometric Random Walk Equation

- we have derived the geometric random walk formula as the change in the stock price
- $dS(t) = \mu \cdot S(t) \cdot dt + \sigma \cdot S(t) \cdot dW(t)$
- divide both sides by the stock price
- $\frac{dS(t)}{S(t)} = \mu \cdot dt + \sigma \cdot dW(t)$
- remember that we applied the logarithm to get normal distribution
- $d(\log S(t)) = \mu \cdot dt + \sigma \cdot dW(t)$
- on the lower order level we get
- $F(X) = \log S(t)$
- applying Ito's lemma
- $dF(S) = \frac{1}{S}dS - \frac{1}{2}\frac{1}{S^2}dS^2$
- for dS, we can substitute the formula on top
- for dS^2, we have derived the formula in the last lecture
- $dF(S) = \frac{1}{S}(\mu\, S\, dt + \sigma\, S\, dW) - \frac{1}{2}\frac{1}{S^2}\left(\sigma^2 S^2 dt\right)$
- if we rearrange (canceling out S and S^2), we get
- $dF(S) = \left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma\, dW$
- now we apply the inverse transformation of the logarithm because $F(X) = \log S(t)$, which is the exponential function
- $S(t) = S(0)e^{\left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W}$

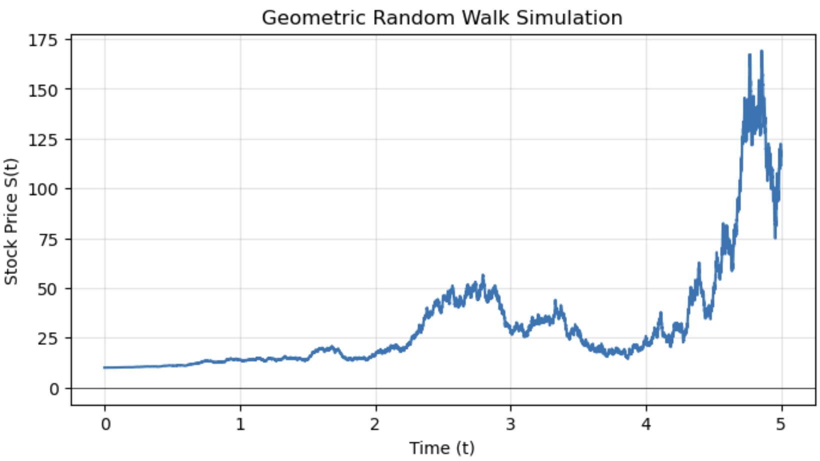
Geometric Brownian Motion Implementation

- implement the geometric brownian walk

- create a deterministic term and a stochastic term
- put them through the exponential function

```python
def simulate_geometric_random_walk(S0=10, T=2, N=1000,
    mu=0.1, sigma=0.05)-> tuple[np.ndarray, np.ndarray]:
    # single timestep
    dt = T/N
    # all timesteps
    t = np.linspace(0, T, N)
    # wiener process ~ N(0,1)
    W = npr.standard_normal(size=N)
    W = np.cumsum(W) + np.sqrt(dt)
    # geometric random walk from lecture
    S = (mu - 0.5 * sigma ** 2) * t * sigma * W
    S = S0 * np.exp(S)
    return t, S
```

- plot the brownian walk
    - notice that that value cannot fall below zero, closer to how stock prices work than just the Wiener-process by itself



Quiz: Random Behavior

- Daily returns (and monthly returns) have approximate normal distribution?
    - True
- Stock prices $S(t)$ have _ distribution?
    - log-normal
- What is the distribution of the (W(t+dt)-W(t)) increments?
    - normal distribution with $N(0, dt)$ with mean=0 and variance=dt

12. Black-Scholes Model

Black-Scholes Model Introduction - The Portfolio

- recap
    - we discussed why random variables are good reflection of price fluctuations
    - in order to model random variables, we learned stochastic calculus and differential equations
    - we modeled stock price fluctuations with the Wiener-process and the Geometric Brownian Walk

- break-through of Black-Scholes model
  - published in 1973 by FIsher Black, Robert Merton, & Myron Scholes
  - they derived a model that can yield the option price $V(S, t)$ for $t < T$
  - **they showed that combining risky assets can eliminate risk itself**
  - Markowitz-Model had shown including several stocks in portfolio can reduce unsystematic risk
  - CAPM showed that because we can eliminate unsystematic risk only relevant risk is market risk β (market risk cannot be diversified away)
  - **market-neutral strategies: delta-hedging and pairs-trading can eliminate all risk**
- option pricing
  - the option price is a function of various parameters $V(S,\ t,\ \mu,\ \sigma, E,\ T,\ r)$
  - $S,\ t$ = underlying stock price and time
  - μ, σ = mean and volatility of associated stock
  - $E,\ T$ = strike price and expiry
  - r = risk-free rate
- call option
  - right to buy an asset for an agreed price at a specific time in future
  - call option price should rise in value if underlying asset rises; and fall if it falls
  - the greater the stock price $S(t)$, the greater the pay-off at expiry because $V(T) = max(S - E, 0)$
  - positive correlation between the two instruments
- put option
  - right to sell an asset for an agreed price at a specific time in future
  - put option price should rise in value if underlying asset falls; and fall if it rises
  - the lower the stock price $S(t)$, the greater the pay-off at expiry because $V(T) = max(E - S, 0)$
  - negative correlation between the two instruments
- exploiting correlation to create a risk-free portfolio
  - $\pi = V(S, t) - \Delta S$
  - the portfolio π contains a long position on the option $V(S, t)$ and short position on the underlying stock ΔS (where Δ is the quantity of the underlying asset)

Black-Scholes Model Introduction - Dynamic Delta Hedge

- previously we verified the assumption that underlying asset's price follows a log-normal random walk
  - $dS = \mu S\, dt + \sigma S\, dX$
- thus, the portfolio's change in value can be expressed as
  - $d\pi = dV(S, t) - \Delta dS$
- we can solve this stochastic equation with Ito's Lemma
  - $dV(S, t) = \frac{\partial V}{\partial t} dt + \frac{\partial V}{\partial S} dS + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} dt$
  - substitute in for change in option price
  - $d\pi = \frac{\partial V}{\partial t} dt + \frac{\partial V}{\partial S} dS + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} dt - \Delta dS$
- factor out dt and dS

- $d\pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)dt + \left(\frac{\partial V}{\partial S} - \Delta\right)dS$
    - deterministic termis $\left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)dt$
    - stochastic term is $\left(\frac{\partial V}{\partial S} - \Delta\right)dS$
- **Delta Hedging**
    - $\frac{\partial V}{\partial S} = \Delta$: if we choose delta equal to the changes in prices, the risk is reduced to zero
    - the elimination of risk through the opposite movements in the 2 assets' prices is called hedging
    - delta hedging is the mathematically perfect elimination of risk because of exploiting the correlation between 2 assets
    - delta hedging is also dynamic, which means that hedge must be continually rebalanced at all times
    - $d\pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)dt + (0)dS$
    - if we achieve continual rebalancing then, the stochastic term cancels out and leaves the deterministic term exclusively
    - this makes the portfolio predictable and risk-neutral

Black-Scholes Model Introduction - No Arbitrage Principle

- the no-arbitrage principle
    - states that efficient markets are arbitrage-free and one cannot make money from arbitrage
    - no arbitrage = no investor can make in a profit without risk
    - how does this work with our assumption of the risk-free portfolio?
- the risk-free rate r_f
    - $d\pi = \left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)dt$
    - the change of the risk-free portfolio $d\pi$ must be the same rate as we would get from a risk-free investment (lending money to the bank)
    - $d\pi = r_f \cdot \pi \cdot dt$
- from discrete to continuous model
    - suppose we have amount $x(t)$ in the bank at time t
    - how much does this change in value from today to tomorrow?
    - the change in the amount is equal to the Taylor-expansion of the derivatives
    - $x(t + dt) - x(t) = \frac{dx(t)}{dt}dt$
    - we also say that the actual amount today $x(t)$ must be proportional to the interest rate r and the timestep dt
    - $x(t + dt) - x(t) = \frac{dx(t)}{dt}dt = r \cdot x(t) \cdot dt$
    - we can solve this for $x(t)$
    - $x(t) = x(0)e^{r\times t}$
- let's assume the no-arbitrage principle is false
    - we can make a riskless profit through arbitrage
    - we can get greater return with the delta-hedged portfolio:

            - we borrow money from the bank and invest it into delta-hedged stock+option portfolio
            - return from the portfolio would be greater than the interest rate, we can keep the difference
    - but arbitrage does not exists (or only for short time)
- deriving Black-Scholes model
    - $\left(\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}\right)dt = r\left(V - S\frac{\partial V}{\partial S}\right)dt$
    - solve for 0
    - $\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + r\,S\frac{\partial V}{\partial S} - rV = 0$
    - it is a parabolic portal differential equation
    - linear model: sum of solutions is also a solution
    - financial equations are usually parabolic and differential like heat and diffusion equations of physics

The Black-Scholes Equation

- Black-Scholes Equation

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + r\,S\frac{\partial V}{\partial S} - rV = 0$$

- solution to depending on put or call
    - assuming no dividend yields on the underlying

CALL	PUT
$S(0)N(d_1) - Ee^{-r(T-t)}N(d_2)$	$-S(0)N(-d_1) + Ee^{-r(T-t)}N(-d_2)$
$d_1 = \frac{\log\left(\frac{S(0)}{E}\right)+\left(r+\frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}$ $d_2 = d_1 - \sigma\sqrt{T-t}$	

- where
    - standard normal distribution is
        - $N(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{z^2}{2}}dz$
- this is the easy way
    - no need to find partial derivatives, Ito's lemma or other stochastic-differential calculus
    - we just need to find d_1 and d_2

The Greeks

- in finance, particularly for option pricing, common metrics are abbreviated by Greek letters
- 1) Delta Δ
    - delta of an option/portfolio is the sensitivity of the option to the underlying

        - $\Delta = \frac{\partial V}{\partial S}$
    - for a portfolio, it is the sum of deltas of all individual positions
    - for delta-hedging, we want to ensure that delta is equal to the derivative
- 2) Gamma Γ
    - the gamma of an option/portfolio is the second derivative of the position with respect to the underlying
        - $\Gamma = \frac{\partial^2 V}{\partial S^2}$
    - or the sensitivity of the delta Δ
    - gamma measures how often a position must be re-hedged in order to maintain a delta-neutral position
- 3) Theta Θ
    - theta is the rate of change of the option price with time
    - $\Theta = \frac{\partial V}{\partial t}$
- 4) Vega ν
    - measures sensitivity to volatility
    - derivative of the option value with respect to the volatility of the underlying asset
    - $\nu = \frac{\partial V}{\partial \sigma}$
- 5) Lambda or Omega
    - is the percentage change in option value per percentage change in the underlying price
    - measure of leverage, sometimes called gearing
    - $\Lambda = \Omega = \Delta \cdot \frac{S}{V} = \frac{\partial V}{\partial S} \cdot \frac{S}{V}$
- Plug in the Greeks into Black-Scholes Equation

$$\underbrace{\frac{\partial V}{\partial t}}_{\text{theta}} + \frac{1}{2}\sigma^2 S^2 \underbrace{\frac{\partial^2 V}{\partial S^2}}_{\text{gamma}} + r S \underbrace{\frac{\partial V}{\partial S}}_{\text{delta}} - r V = 0$$

    - turns into:

$$\Theta + \frac{1}{2}\sigma^2 S^2 \Gamma + r S \Delta - r V = 0$$

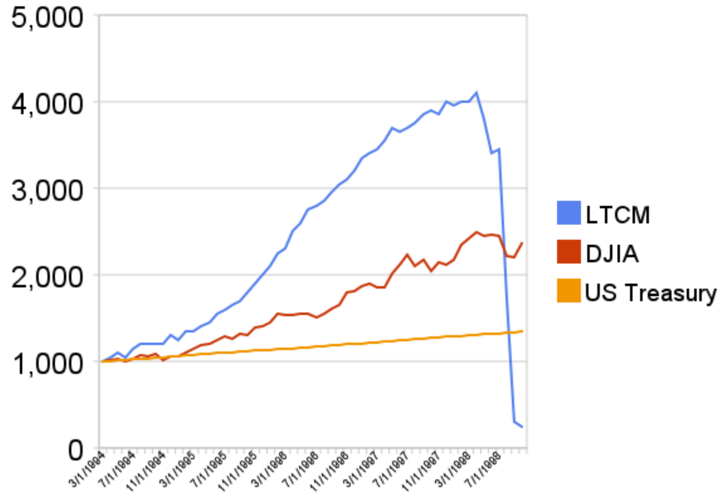
How to make money with Black-Scholes Model

- purpose of Black-Scholes model
    - making money is not the objective on Black-Scholes model
    - remember that no-arbitrage principle of efficient markets means there is no risk-free profit anyways
    - you can also use the Black-Scholes model to find mispriced option (and hope that market will correct creating a profit for you)
    - many investors use this model hedge and eliminate risk on existing positions

Long-Term Capital Management (LTCM)

- LTCM was a hedge fund founded in 1994 by former vice-chairman of Salomon Brothers: John Meriwether

- lots of members were academics: like Nobel-prize winner Myron Scholes and Robert Merton
    - they used complex mathematical models like pair trading or Black-Scholes



- at first, they were very successful 40%
- there success came from market-neutral strategies
    - these would profit regardless of market-trend and from arbitrage opportunities
    - for a while, they could hedge even market risk (systematic risk)
    - but with Asian financial crisis in 1997 and Russian financial crisis in 1998 LTCM collapsed
- Pairs-Trading strategy
    - combining long and short positions in a pair of highly correlated financial instruments
    - it is a form of statistical arbitrage strategy developed in the 1980s
    - if the correlation between 2 stocks weakens, investors should long the underperforming stock and short the overperforming stock
    - the stocks are expected to converge, thus getting profit on both positions

Quiz: Black-Scholes Model

- What is the main idea behind Black-Scholes model?
    - by combining correlated assets we can eliminate risk
- Risk-neutral strategies try to eliminate market risk and try to profit from both increasing or decreasing stock prices
    - True
- What is the mathematical aim of Black-Scholes model?
    - to calculate the price of an option at $t < T$
- Can we always use correlations (pos or neg) between assets to eliminate risk?
    - True

13. Black-Scholes Implementation

Black-Scholes Model Implementation

- imports

```python
from scipy import stats
from numpy import log, exp, sqrt
# alias the normal distribution cdf  as N
N = stats.norm.cdf
```
✓ 0.0s

- implement the call and put option pricing formulas from the lecture to calculate the option price at $t = 0$

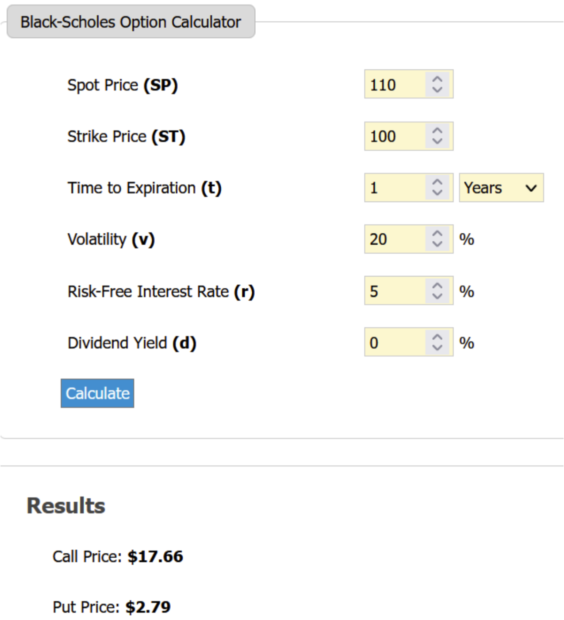
```python
def call_option_price(S0, E, T, rf, sigma):
    # calculate parameters d1 and d2
    d1 = (log(S0/E) + (rf + sigma * sigma / 2.0) * T) / (sigma * sqrt(T))
    d2 = d1 - sigma * sqrt(T)
    # use standard normal dist N(x) to calculate price
    V = S0 * N(d1) - E * exp(-rf*T) * N(d2)
    return V

def put_option_price(S0, E, T, rf, sigma):
    # calculate parameters d1 and d2
    d1 = (log(S0/E) + (rf + sigma * sigma / 2.0) * T) / (sigma * sqrt(T))
    d2 = d1 - sigma * sqrt(T)
    # use standard normal dist N(x) to calculate price
    V = -S0 * N(-d1) + E * exp(-rf*T) * N(-d2)
    return V
```
✓ 0.0s

- apply the formulas with the given parameters

```python
# underlying stock price at t=0
S0 = 110
# strike price
E = 100
# expiry 1year=365days
T = 1
# risk-free rate
rf = 0.05
# volatility of the underlying stock
sigma = 0.2

V_call = call_option_price(S0, E, T, rf, sigma)
print(f"Call option price according to Black-Scholes model: {V_call:.2f}")

V_put = put_option_price(S0, E, T, rf, sigma)
print(f"Put option price according to Black-Scholes model: {V_put:.2f}")
```
✓ 0.0s

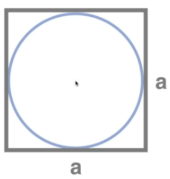
```
Call option price according to Black-Scholes model: 17.66
Put option price according to Black-Scholes model: 2.79
```

- verify on Black Scholes Calculator



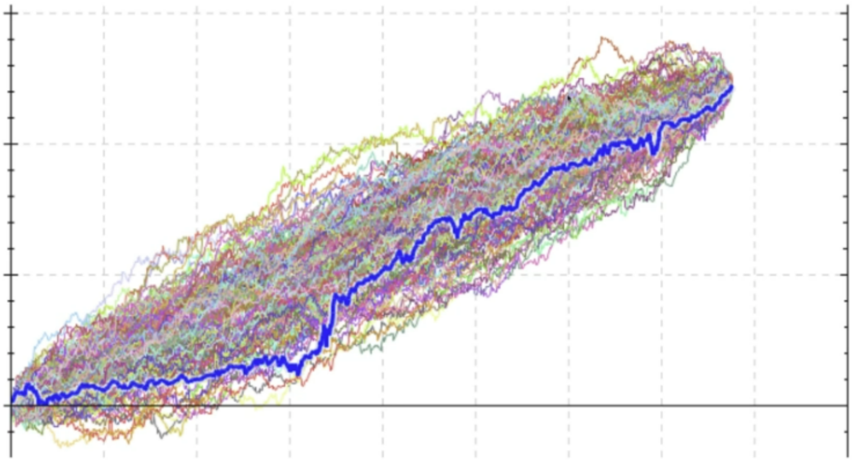
What is a Monte-Carlo Simulation?

- Monte-Carlo simulation history
  - first used in 1940 during the Manhattan Project at Los Alamos
  - John v Neumann, Stanilaw Ulam & Nicholas Metropolis applied this method nuclear physics (combining deterministic and stochastic processes)
  - Monte-Carlo method solves a deterministic problem using probabilistic methods
  - delivers good approximations: the more iterations, the better the prediction converges on the correct result
- Simple Example: Finding π
  - we have a circle with diameter a and square with a, and they have the same center



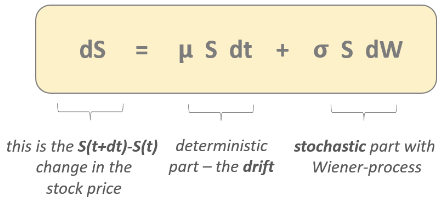
  - we can generate random points within the their two-dimensional plane
  - then we count the points inside the circle and inside the square
  - rearranging their area formulas for π, we get
  - $\pi = 4 * \frac{circle\ points}{square\ points}$
- Basic Principles
  - we can estimate the outcome of an **uncertain event**
  - generate several possible outcomes and then calculate the average of the outcomes (in stochastic called expected outcome)
  - delivers good approximations: the more iterations, the better the prediction converges on the correct result
- Stock Price Simulation
  - $dS = \mu S\, dt + \sigma S\, dX$

- we already know that asset prices $S(t)$ (such as stocks) follow a log-normal random walk
- with the parameters $S(0)$, μ, σ we can simulate multiple stock price developments
- with modern computing power, creating simulations has become cheap
- the average of these simulations yields the future stock price $S(t)$



Predicting Stock Prices with Monte-Carlo Simulation

- we cannot foresee the price of an asset in the future, but we can simulate many possible developments with the **geometric random walk**
- the mean of these simulations yields $S_B(t)$, the path with the highest probability in the future
  - → this a Monte Carlo simulation
- geometric random walk: recap



  - $dS = \mu \cdot S \cdot dt + \sigma \cdot S \cdot dW$
  - where
    - $dS = S(t + dt) - S(t)$ is the change in the stock price
    - $\mu \cdot S \cdot dt$ is the deterministic trend
    - $\sigma \cdot S \cdot dW$ is the stochastic part
    - dW independent increment drawn from a normal distribution $N(0, dt)$
  - the formula a continuous model of asset prices and stochastic differential equation
- we have also proven that the solution w.r.t. $S(t)$ of the geometric random walk is
  - $S(t) = S(0)exp^{\left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W}$
- imports: pandas, numpy, matplotlib
- implement Monte-Carlo-based stock price simulation
  - calculate value in day-by-day frequency

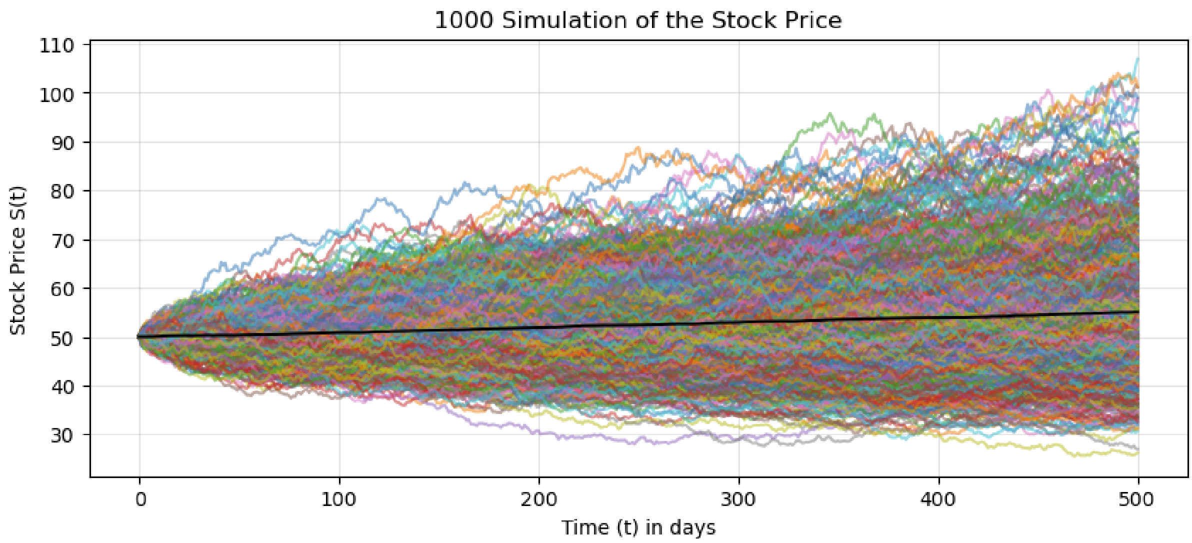
```python
N_SIMULATIONS = 1000

def stock_price_monte_carlo(S0: float, mu: float, sigma: float, N: int=500):
    results = []
    for _ in range(N_SIMULATIONS):
        prices = [S0]
        for _ in range(N):
            # we simulate the day by day, i.e. no multiplicatio by t
            stock_price = prices[-1] * \
                np.exp((mu - 0.5 * sigma ** 2) + sigma * np.random.normal())
            prices.append(stock_price)
        results.append(prices)

    simulation_data = pd.DataFrame(results).T
    return simulation_data
```
✓ 0.0s

- simulate the data and compute the highest probability path (mean across all days)

```python
# simulate data reproducably
np.random.seed(2020)
simulation_data = stock_price_monte_carlo(50, 0.0002, 0.01)
# get mean across all simulated days
expected_path = simulation_data.mean(axis=1)
simulation_data
```
✓ 1.2s

	0	1	2	3	4	5	6	7	8	9	...
0	50.000000	50.000000	50.000000	50.000000	50.000000	50.000000	50.000000	50.000000	50.000000	50.000000	...
1	49.130722	49.638927	50.337555	50.086140	50.027547	50.354706	50.494668	49.876039	49.654977	50.988747	...
2	49.175231	48.451300	50.944970	50.017814	49.740483	50.323570	51.444916	49.609366	50.409165	50.643006	...
3	48.629667	47.485513	49.921747	49.373112	50.262860	50.263137	50.855846	48.799193	51.111129	50.562572	...
4	48.321156	46.958813	49.545630	48.675825	50.962267	50.025427	50.507879	48.322173	51.171961	50.760205	...
...	...	...	...	...	...	...	...	...	...	...	...

- plot all simulated paths and highlight highest probability path



MC Simulation & BL Model Implementation I

- for an option, the underlying asset follows a geometric random walk
- by simulating these stochastic processes, we can determine the price of a financial derivative (option)
  - $dS = \mu \cdot S \cdot dt + \sigma \cdot S \cdot dW$
  - with the solution that we used in the last lecture
  - $S(t) = S(0)exp^{\left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma W}$

- we can obtain the $log\, S(t)$ because we know that stock prices cannot be negative
    - we use Ito's lemma with $F(S) = log\, S(t)$
    - $d\, log\, S(t) = \left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma\, dW$
    - $log\, S(t) = log\, S(0) + \left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma \int_0^t dW$
    - the increments of the Wiener-process are independent and can be sampled from a normal distribution $N(0, t)$ or $\sqrt{t} \cdot N(0, 1)$
- risk-neutral assumption in Black-Scholes model
    - the drift μ becomes the risk-free interest rate r
    - so that the exponential function can define the stock price at expiry $S(T)$
    - $S(T) = S(0) \cdot exp^{\left[\left(r - \frac{1}{2}\sigma^2\right)T + \sigma\sqrt{T}N(0,1)\right]}$
- with Monte-Carlo simulation, we generate a large amount of stock price estimates using this equation
    - the option price is the expected value of a pay-off function (put or call pay-off) plus a **discount factor** that brings the future value back to the present
    - call option = $max(S - E, 0)$
    - put option = $max(E - S, 0)$

MC Simulation & BL Model Implementation II

- implement a class for option pricing with Monte-Carlo + Black-Scholes
    - take input parameter and simulate the stock prices

```python
class OptionPricing:
    def __init__(self, S0, E, T, rf, sigma, iterations) -> None:
        self.S0 = S0
        self.E = E
        self.T = T
        self.rf = rf
        self.sigma = sigma
        self.iterations = iterations
        # we have 2 columns: 1st with 0s & 2nd with with pay-offs
        self.option_data = np.zeros([self.iterations, 2])

    def monte_carlo_simulation(self):
        # dimensions: 1 dimensional array with as many items as the itrations
        rand = np.random.normal(0, 1, [1, self.iterations])

        # equation for the S(t) stock price at T
        stock_price = self.S0*np.exp(
            self.T*(self.rf - 0.5*self.sigma**2)+self.sigma*np.sqrt(self.T)*rand
            )

        return stock_price
```

    - next use the simulated stock prices to find option price with the pay-off formulas

```python
    def call_option_simulation(self):
        option_data = self.option_data
        stock_price = self.monte_carlo_simulation()

        # we need S-E because we want to calculate the max(S-E,0)
        option_data[:,1] = stock_price - self.E

        #average for the Monte-Carlo method
        #np.amax() returns the max(0,S-E) according to the formula
        average = np.mean(np.amax(option_data, axis=1))

        #have to use the discount factor exp(-rT)
        return np.exp(-1.0*self.rf*self.T)*average

    def put_option_simulation(self):
        option_data = self.option_data
        stock_price = self.monte_carlo_simulation()

        # we need S-E because we want to calculate the max(E-S,0)
        option_data[:,1] = self.E - stock_price

        #average for the Monte-Carlo method
        #np.amax() returns the max(E-S,0) according to the formula
        average = np.mean(np.amax(option_data, axis=1))

        #have to use the discount factor exp(-rT)
        return np.exp(-1.0*self.rf*self.T)*average
```

- input the parameters and run 10,000 simulations

```python
np.random.seed(2020)#seed for reproducibility
S0=110              #underlying stock price at t=0
E=100               #strike price
T = 1               #expiry
rf = 0.05           #risk-free rate
sigma=0.2           #volatility of the underlying stock
iterations = 10_000 #number of iterations in the Monte-Carlo simulation

mc_bs_prices = OptionPricing(S0, E, T, rf, sigma, iterations)
V_call = mc_bs_prices.call_option_simulation()
V_put = mc_bs_prices.put_option_simulation()

print(f"Number of Simulations: {iterations:,}")
print(f"Call option price acc. to Monte-Carlo Black Scholes model: {V_call:.2f}")
print(f"Put option price acc. to Monte-Carlo Black Scholes model: {V_put:.2f}")
```

✓ 0.0s

```
Number of Simulations: 10,000
Call option price acc. to Monte-Carlo Black Scholes model: 17.78
Put option price acc. to Monte-Carlo Black Scholes model: 2.80
```

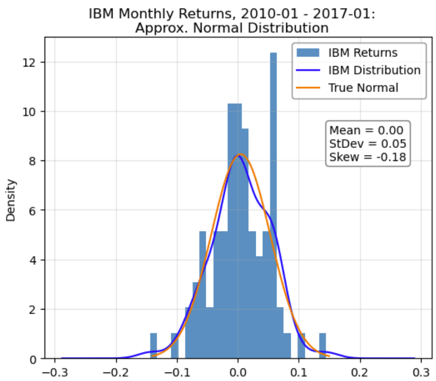
  - results are close to the Black-Scholes model with Normal Distribution CDF implementation

14. Value-at-Risk

What is the Value-at-Risk (VaR)?

- recap
  - return is easier to measure than risk
  - there is a trade-off between risk and return
  - Markowitz Model and CAPM model gave us simple tools to measure and diversify against risks
  - in Black Scholes model, risk was assessed through stochastic variables
- but there are more measures for assessing risk
  - another probabilistic measure is the **Value-at-Risk VaR**
- Returns
  - $\frac{S(t+1)-S(t)}{S(t)} = R(t)$ are the returns for the interval $[t, t+1]$
  - as we have seen, the daily or monthly returns are approximately normally distributed (over great enough time span)

- we have also observed that the return distribution has "fat tails" → outliers occur more often than in truly normal data
- normal distributions can be defined by the two parameters mean and variance $N(\mu, , \sigma^2)$
- we can re-define returns as function of these parameters too
- $R(t) = \mu + x * \sigma$
- thus, returns can be treated as random variable drawn from a normal distribution



- **Value-at-Risk** Definition
- 1) VaR
  - is number measured in price units ($, €, £)
  - tells in a large percentage of cases, your portfolio is likely to not lose more than that amount of money
  - "Measures amount of potential loss that could happen in an investment (or a portfolio of investments) over a given period of time (with a given degree of confidence)"
- 2)
  - it is easy to understand and easy to interpret
  - standard deviations and betas are not as straightforward
- 3) you can compare
  - different types of assets or portfolios with VaR
  - Profitability and risk of different units and make decisions accordingly
  - "risk budgeting"

Value-at-Risk Mathematics

- there two ways to measure the VaR
  - 1) variance method and 2) Monte-Carlo simulation
- 1) **Variance Method**
- assumes returns are normally distributed
- the probability density function (PDF) of a normal distribution is

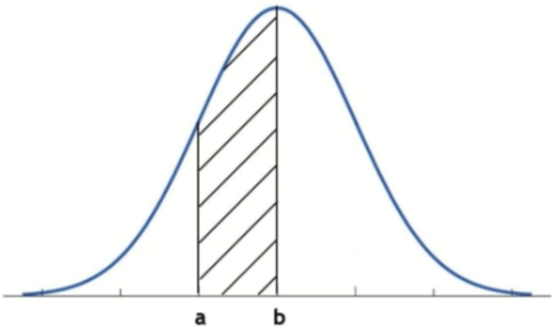
$$PDF(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- the cumulative density function (PDF) of a normal distribution is

$$CDF = \frac{1}{2}\left[1 + erf\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right]$$

- PDF helps us talk to about probability
- the integral of the function yields the probability

$$P(a > x > b) = \int_a^b f(x)dx$$



the definite integral is the area under the bell-shaped curve

- we would like to ensure that the loss is not going to be greeted than a predefined value
  - $Prob\{\delta\pi \leq -\$5Mio\} = 0.05$ (or confidence level is 95%)
  - $Prob\{\delta\pi \leq VaR\} = 1 - c$
- z defines the number of standard deviations from the mean
  - 99% confidence level is $z = 2.33$
  - 95% confidence level is $z = 1.64$
- VaR time period considerations
  - $VaR = position(\mu_{period} - z \cdot \sigma_{period})$
  - but the time interval for VaR is not necessarily 1 day (like what is the VaR in a week, in a month, etc.)
  - $\sigma_{n-day} = \sigma_{daily} \cdot \sqrt{n}$
  - $\mu_{n-day} = \mu_{daily} \cdot n$
- VaR final calculation

$$VaR = \Delta S\left[\mu \cdot \delta t - \sigma \cdot \sqrt{\delta t} \cdot \alpha(1 - c)\right]$$

  - if we calculate VaR for tomorrow, δt is so small that we can omit it
  - if we calculate VaR for long periods, we have to take into account the drift too
  - $\alpha(c - 1)$ is the inverse cumulative distribution function for the standardized normal distribution (to get to desired confidence level)
- thus, we can measure that the loss is not going to be greeted than a predefined value with a certain confidence level

Value-at-Risk Implementation

- create functions
  - download a stock data needed to calculate VaR from historical data

$$VaR = \Delta S\left[\mu \cdot \delta t - \sigma \cdot \sqrt{\delta t} \cdot \alpha(1 - c)\right]$$

  - calculate VaR for single day (without δt)
  - calculate VaR for n days

```python
def download_data(stock: str, start_date: datetime, end_date: datetime) -> pd.DataFrame:
    data = {}
    ticker = yf.download(stock, start_date, end_date)
    data[stock] = ticker["Adj Close"]
    return pd.DataFrame(data)


def calculate_var(position: float, mu: float, sigma: float, conf: float = 0.95):
    # this is the value-at-risk for tomorrow
    var = position * (mu - sigma * norm.ppf(1 - conf))
    return var


def calculate_var_n(position: float, mu: float, sigma: float, n_days: int, conf: float = 0.95):
    # this is the value-at-risk for any future period
    var = position * (mu*n_days - sigma*np.sqrt(n_days) * norm.ppf(1 - conf))
    return var
```

- apply the functions to data
    - download CityGroup data
    - caculate log daily returns
    - drop NaN values

```python
# Citigroup Inc. Stock
stock = "C"
start_date = datetime(2014, 1, 1)
end_date = datetime(2018, 1, 1)

# download data
data = download_data(stock, start_date, end_date)

# compute log daily returns
data["returns"] = np.log(data["C"] / data["C"].shift(1))
data = data[1:]
data
```

- compute
    - mean and standard deviation for VaR
    - VaR for a $1Mio position

```python
# log returns parameter
mu = np.mean(data["returns"])
sigma = np.std(data["returns"])

# investment position $1Mio
S = 1e6

var = calculate_var(S, mu, sigma)
print(f"For an Investment of ${S:,} in CityGroup, tomorrow's VaR = ${var:,.2f}")
```

```
For an Investment of $1,000,000.0 in CityGroup, tomorrow's VaR = $25,396.73
```

```python
n_days = 5
var = calculate_var_n(S, mu, sigma, n_days)
print(f"For an Investment of ${S:,} in CityGroup, {n_days} days VaR = ${var:,.2f}")
```

```
For an Investment of $1,000,000.0 in CityGroup, 5 days VaR = $57,834.33
```

Value-at-Risk with Monte-Carlo Simulation I & II

- recap:
    - stock prices follow Geometric Brownian Walk
    - $dS(t) = \mu S(t)\, dt + \sigma S(t)\, dW$
    - we can obtain $log\, S(t)$ because the stock prices cannot be negative
    - use Ito's lemma with $F(S) = log\, S(t)$
    - $d\, log\, S(t) = \left(\mu - \frac{1}{2}\sigma^2\right)dt + \sigma\, dW$
    - solve for $log\, S(t)$
    - $log\, S(t) = log\, S(0) + \left(\mu - \frac{1}{2}\sigma^2\right)t + \sigma \int_0^t dW$
    - $\sqrt{t} \cdot N(0, t)$ for the increments Wiener-process
- solution to this stochastic differential equation
    - $S(t) = S(0)\exp^{\left[\left(\mu + \frac{1}{2}\sigma^2\right)t + \sigma\sqrt{t}\, N(0,1)\right]}$
    - this exponential function defines the stock price at a given time t
    - we also used it in the Monte-Carlo simulation for Black Scholes
- Monte-Carlo simulation
    - we generate a large amount of stock price estimates with this formula
    - then we sort the stock prices from smallest to largest
    - if we need VaR with 99% confidence level, we take the 1% lowest percentile $S'(t)$ in this series
    - $VaR_{99\%} = S(t) - S'(t)$
- implement a class for VaR simulation
    - simulate the potential changes in the stock price using the exponential function from above
    - sort the values and subtract corresponding the percentile corresponding to the confidence level

```python
class ValueAtRiskMC:
    def __init__(self, P, mu, sigma, conf = 0.95, n_days = 1, iterations = 100_000) -> None:
        self.P = P
        self.mu = mu
        self.sigma = sigma
        self.conf = conf
        self.n_days = n_days
        self.iterations = iterations

    def simulation(self):
        rand = np.random.normal(0, 1, [1, self.iterations])

        # equation for the S(t) stock price
        stock_price = self.P * np.exp(
            self.n_days * (self.mu - 0.5 * self.sigma ** 2) \
                + self.sigma * np.sqrt(self.n_days) * rand
                )

        # we have to sort the stock prices to determine the percentile
        stock_price = np.sort(stock_price)

        # it depends on the confidence level: 95% -> 5 and 99% -> 1
        percentile = np.percentile(stock_price, (1 - self.conf) * 100)

        # VaR = S(t) - S'(t)
        return self.P - percentile
```

- use the stock data from CitiGroup as before to populate the Monte-Carlo simulation
    - with 95% confidence level, we can say that at most we would lose tomorrow from our investment is ~ \$24.5k

```python
# investment position $1Mio
P = 1e6

# log returns parameter
mu = np.mean(data["returns"])
sigma = np.std(data["returns"])

model = ValueAtRiskMC(P, mu, sigma, conf=0.95, n_days=1)
var = model.simulation()

print(f"For an Investment of ${P:,} in CityGroup, tomorrow's VaR = ${var:,.2f}")
```

✓ 0.0s

```
For an Investment of $1,000,000.0 in CityGroup, tomorrow's VaR = $24,488.52
```

Quiz: Value-at-Risk

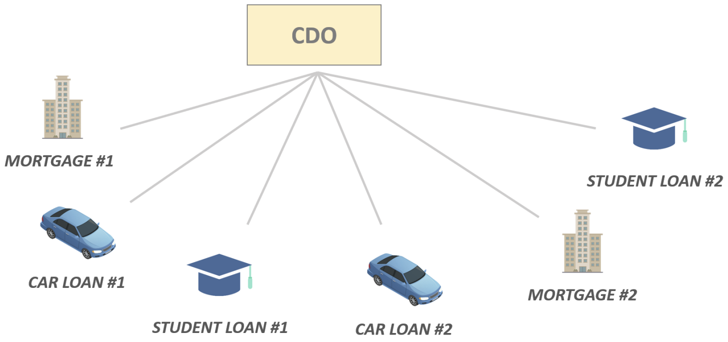
- What is the fundamental assumption when using Value at Risk?
    - daily returns are approx. normally distributed (over long run)
- Value-at-Risk needs historical data.
    - True
- If we know standard deviation for today $\sigma_{t=0}$, then what is the standard deviation in ten days $\sigma_{t=10}$?
    - $\sigma_{t=10} = \sigma_{t=0} * \sqrt{10}$

15. Collateralized Debt Obligations (CDOs) and the Financial Crisis

What are CDOs?

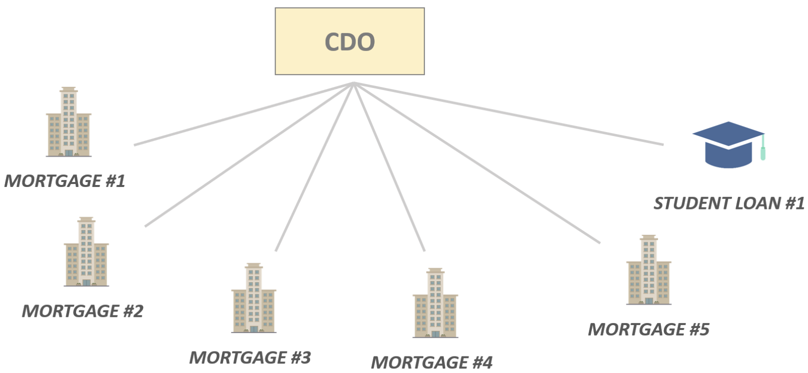
- Collateralized Debt Obligations
    - CDOs first constructed in 1987 by bankers at Drexel Burnham Lambert Inc.
    - within 10 years, they became major force in financial markets
    - CDO is **complex structured-finance product** that is **backed by a pool of loans** and other assets
    - acc. to Warren Buffet CDOs and similar derivatives are financial **weapons of mass destruction**
- valuation of CDOs is complicated cannot follow a mathematical model
    - there are no standard rules when and what underlying collaterals are called
    - especially, with pooled-asset calling on a individual debt is already complicated
- example: a US commercial bank can issue several types of consumer loans
    - student loans: 65% of students take a loan and graduate with average debt of \$30k
    - credit card loans: the average family/household credit card debt is \$6000
    - mortgage: around 60% of homeowners have mortgage debt
    - car loans: around 100m car loan accounts
- an investment bank can buy these debts and create financial products called CDOs

- the CDO is sold by investment banks to investors seeking higher returns than a treasury bill of corporate bond



CDOs and Diversification

- the good intention behind a CDO is diversification
  - the interest payments of each individual loan are now transferred to the the investment bank as owner of the underlying loans
  - pooling servers as risk-minimizing tool that if a single borrower defaults, the entire CDO product is still served by the rest of the loan borrowers
- ideally, then for **best diversification** an underlying pool should be comprised of loans or assets that are **uncorrelated**
  - aim of diversification is eliminate fluctuation in the long-term
- a pool possessing assets with high positive covariance or correlation does not provide good diversification
- thus, high risk of the CDO collapsing occurs when most of its underlying loans are of the same type, like a pool only made of mortgage loan



CDO Tranches

- when a CDO is constructed its underlying assets are classified into tranches
- senior tranche
  - is the safest tranche, with AAA ratings from rating agency
  - with lower returns (~4%) and lower risk
  - it gets payouts first coming from the loans
- junior tranche
  - is a bit riskier that the senior tranche (BBB ratings)
  - with higher returns (7%) and higher risk
  - it gets payouts second coming from the loans
- risky tranche

    - it is the riskiest tranche
    - with highest returns (~10%) and highest risk
    - it gets payouts last (only after senior and junior tranche are covered)
- investors can decide which tranche to buy based on investment and risk profile

The Financial Crisis of 2007-08

- this system was working extremely well for everyone involved - commercial banks, investment banks and investors
- the success of this product created a demand for more loans and mortgages to be able sell more CDOs
- why mortgages were seen as a safe type of loan
    - if a homeowner defaults and goes bankrupt, then the lender (bank) can repossess the house
    - usually, house prices are increasing, so selling the house would cover the costs of the bank
- this and the demand for CDOs is why commercial banks approved more risky mortgages
- a **subprime mortgage** is a type of loan granted to individuals with poor credit scores who do not qualify for conventional mortgages
- the subprime mortgage had a higher chance of default to begin and were pooled into CDOs with little diversification
- as subprime mortgage borrowers began to default on the regular, the CDOs they backed started to dry up and eventually collapsed
- but given the initial expectation, investment banks and investors were holding leveraged positions on their expected CDO payouts
- the collapse of the CDOs led to very large loses and confidence in the mortgage and loan markets

Quiz: CDOs

- A collateralized debt obligation is
    - a complex financial product that is backed by a pool of loans and other assets
- Is it necessary to diversify pooled assets in a CDO?
    - yes
- What is a subprime mortgage?
    - type of loan granted to individuals with poor credit scores

16. Interest Rate Modeling: Vasicek Model

Why use interest rate models?

- interest rates are not only essential for pricing loans and cashflow instruments but also serve as a reference for other assets with the risk-free rate
- the main characteristic with market interest rates $r(t)$ is their fluctuation
- the fluctuation affects both floating and fixed rates instruments in their prices, and thus constitutes interest rate risk (Zinsrisiko)
- recap
    - discounting bonds as stand-in for cashflow instrument pricing
    - zero-coupon bond: $PV = \frac{F}{(1+r(t))^n}$

- coupon bond: $PV = \sum_{i=1}^{n} \frac{c}{(1+r(t))^{i}} + \frac{F}{(1+r(t))^{n}}$

- unlike stock prices, which have a drift, the stochastic process for interest rate modeling has a mean-reversion effect
- but the good news is that deriving and solving the interest rate models is mathematically very similar to the stock price models

Ornstein-Uhlenbeck Process Introduction

- recap Wiener-Process
  - $W(t)$ has independent increments: future $W(t + dt) - W(t)$ increments are independent of past values
  - $W(t)$ has Gaussian increments, i.e. $W(t + dt) - W(t)$ is normally distributed with $\mu = 0$ and $\sigma = dt$
  - also written as $W(t + dt) - W(t) \sim N(0, dt)$
  - the Wiener-Process has a continuous path
- Ornstein-Uhlenbeck Process
  - $dx_t = \theta(\mu - x_t)dt + \sigma\, dW_t$
  - this process is the basis for many short-term interest rate models such as the Merton, Vasicek or Hull-White (also used for currency exchange rates)
  - this process is stationary while Brownian motion is not
  - one application of this process is in pairs-trading strategy

$$dx_t = \underbrace{\theta(\mu-x_t)}_{\substack{\boldsymbol{\mu} \text{ is the equilibrium or the} \\ \text{mean value and } \boldsymbol{\theta} \text{ is the rate by which the} \\ \text{process reverts towards the mean}}}dt + \overbrace{\sigma\, dW_t}^{\substack{\boldsymbol{\sigma} \text{ is the degree of volatility} \\ \text{around the } \boldsymbol{\mu} \text{ mean}}}$$

- Pairs-Trading Strategy
  - relies on the idea of mean-reversion effect
  - we open a long or short positions when the given time series $x(t)$ is far away from the mean under the expectation that $x(t)$ will revert to the mean
  - if PEP goes up a significant amount while KO stay the same pairs trader should buy KO stock and sell PEP stock
  - pair traders assume that companies will return to their historical balance point
  - to model this strategy, the traders construct the time series $x(t)$ from $S_{PEP}(t)$ and $S_{KO}(t)$

Ornstein-Uhlenbeck Process Implementation

- rewrite the Ornstein-Uhlenbeck formula for simulation

Ornstein-Uhlenbeck Process:

$$dx_t = \theta * (\mu - x_t) * dt + \sigma * dW_t$$

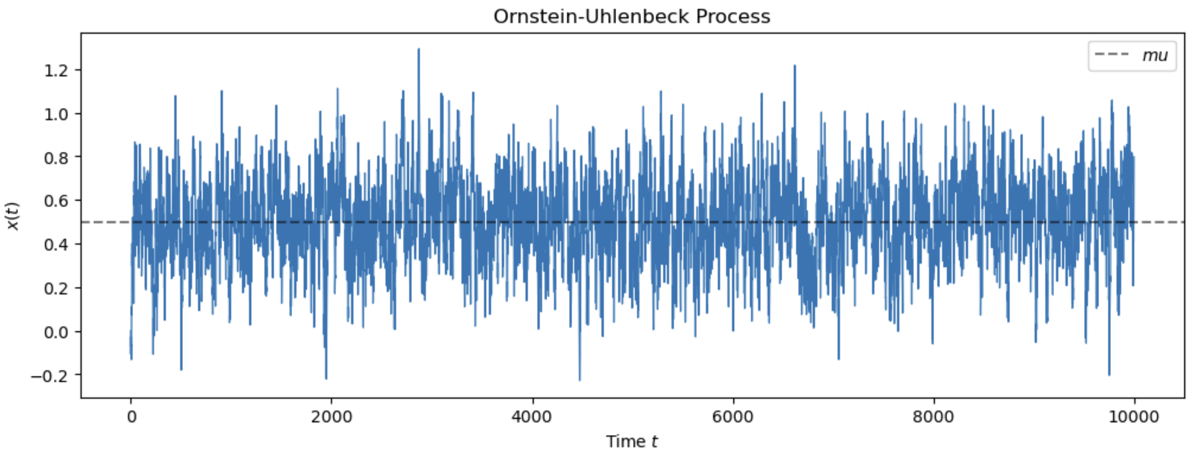
In Simulation:

$$x_t = x_{t-1} + \theta * (\mu - x_{t-1}) * dt + \sigma * \mathcal{N}(0, \sqrt{dt})$$

- implement the process as stepwise simulation

```python
def orn_uhl_process(dt=0.1, theta=1.2, mu=0.5, sigma=0.3, n=10_000):
    # initialize x(t) with all zeros
    x = np.zeros(n)

    for t in range(1, n):
        x[t] = x[t-1] + theta*(mu-x[t-1])*dt + sigma * N(0, np.sqrt(dt))

    return x
```

- plot the simulated data for parameter mu=0.5 and sigma=0.3



Vasicek Model Introduction

- this model was first introduced in 1977 by Oldrich Vasicek
- it is a simple short-rate mode
- bonds, mortgages and credit derivatives are quite sensitive to interest rate changes
- interest rate is a complex topic because it is affected by many factors that are hard to model: political decisions, government interventions, economic developments
- the Vasicek model assumes that the interest rates $r(t)$ follow a mean reverting Ornstein-Uhlenbeck process
- formula
    - $dr(t) = \kappa(\theta - r(t))dt + \sigma\, dW(t)$
    - this model allows for interest rates to be negative
    - the rate $r(t)$ fluctuates around θ
    - κ is the speed of mean reversion (kappa)
    - low κ is slow reversion and high is fast reversion
    - stochastic random noise is defined by $\sigma\, dW(t)$
- we can populate this process with Monte-Carlo simulation and thus predict interest rate developments and better price bonds

Vasicek Model Implementation

- rewrite the model from the lecture for simulation

  Vasicek Model:

$$dr(t) = \kappa * (\theta - r(t)) * dt + \sigma * dW(t)$$

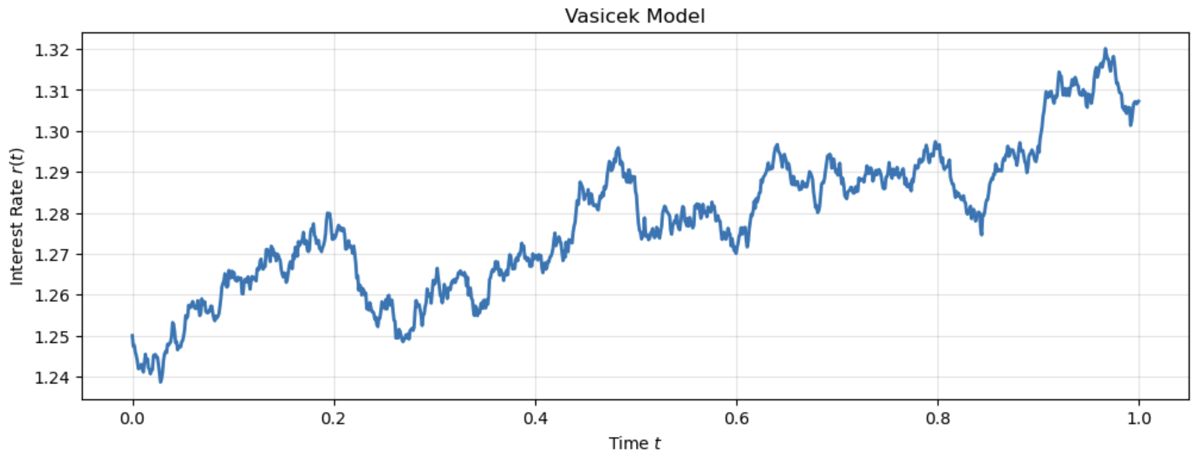
  In Simulation:

$$r(t) = r(t-1) + \kappa * (\theta - r(t-1)) * dt + \sigma * \mathcal{N}(0, \sqrt{dt})$$

- implement the process as stepwise simulation

```
def vasicek_model(r0, kappa, theta, sigma, T=1.0, n=1000):
    # timestep
    dt = T/float(n)
    # timeline
    t = np.linspace(0, T, n+1)
    # interest rates time series
    rates = [r0]
    # simulation
    for _ in range(n):
        dr = kappa * (theta - rates[-1])* dt  + sigma * N(0, np.sqrt(dt))
        rates.append(rates[-1] + dr)

    return t, rates
```

- plot the simulated data with the parameter
  - r0=1.25, kappa=0.9, theta=1.4, sigma=0.05



Quiz: Interest Rate Modeling

- Ornstein-Uhlenbeck process is a
  - stochastic mean-reverting process
- What is the theta parameter in the Vasicek model?
  - a parameter defines the long-term mean of the interest rate
- If kappa is small in the Vasicek, then the stochastic process reverts to the theta mean extremely fast
  - False

17. Pricing Bonds with Vasicek Model

Bond Pricing with Vasicek Model I

- recap

- discounting bonds as stand-in for cashflow instrument pricing
- zero-coupon bond: $PV = \frac{F}{(1+r(t))^n}$
- coupon bond: $PV = \sum_{i=1}^{n} \frac{c}{(1+r(t))^i} + \frac{F}{(1+r(t))^n}$

- example
  - 2-year ZCB with $1000 principal and 10% market interest
  - $PV = \frac{F}{(1+r(t))^n} = \frac{1000}{(1.1)^2} = 826,44$
- construct continuous models with differential equations
  - suppose we amount $x(t)$ in the bank for a time t
  - how does it increase in value from one day to the next?
  - we can use Taylor expansion
  - $x(t + dt) - x(t) = \frac{dx(t)}{dt}dt$
  - the interest received must be proportional to the actual amount $x(t)$
  - $\frac{dx(t)}{dt} = r\, x(t)$
  - the change in capital w.r.t. the change in time equals the interest time capital
  - solving this differential equation for $x(t)$ (as the future value) we get an exponential function
  - $x(t) = x(0)\exp(r \cdot t)$
  - for the present value of future cashflows we use a negative sign for the exponent
- Vasicek model and bond pricing
  - here we assume that
    - 1) interest rates are not flat
    - 2) follow a stochastic process
  - then the present value calculated with Vasicek can be described as
  - $PV = x \cdot \exp\left(-\int_t^T r(s)ds\right)$
  - more on [Vasicek model bond pricing]
- we will using Monte-Carlo simulations for pricing
  - the mean of the simulations will yield the most likely interest rate (development) in the future

Bond Pricing with Vasicek Model II

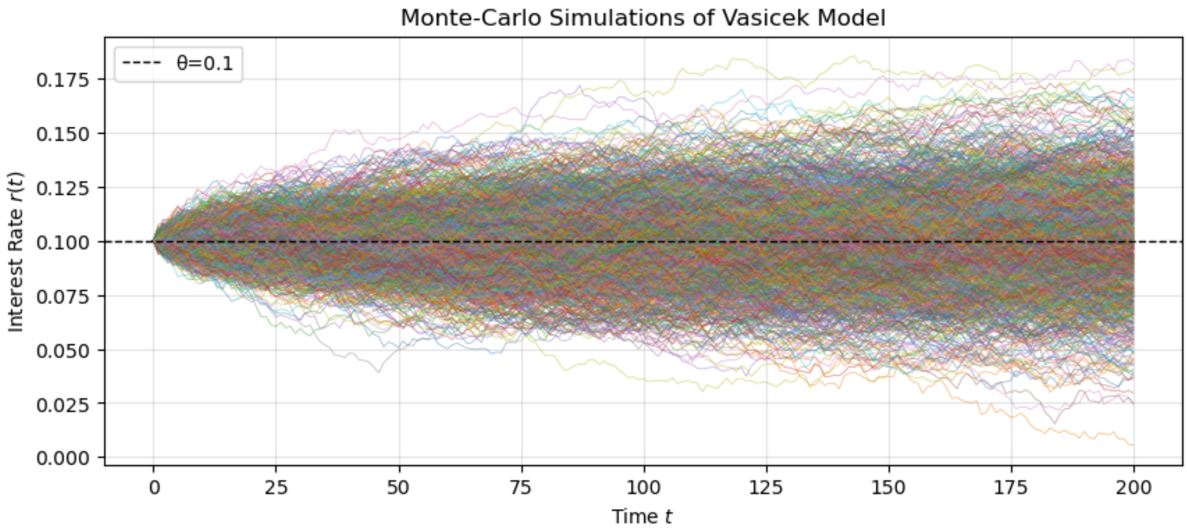
- create Monte Carlo simulation of the Vasicek model
  - use the formula from the previous section for simulation
  - implement a loop for the number of simulations and number of points into the future

```python
# we will simulate N interest rate processes
N_SIMULATIONS = 1000
# we will simulate N steps into future
N_POINTS = 200


def vasicek_monte_carlo(x, r0, kappa, theta, sigma, T=1.0):
    # timestep
    dt = T/float(N_POINTS)
    # stores results of simulations
    results = []

    for _ in range(N_SIMULATIONS):
        rates = [r0]
        for _ in range(N_POINTS):
            # Ornstein-Uhlenbeck
            dr = kappa * (theta - rates[-1])* dt  + sigma * N(0, np.sqrt(dt))
            rates.append(rates[-1] + dr)
        results.append(rates)

    simulations = pd.DataFrame(results).T

    return simulations
```

- simulate with parameters
  - r0=0.1, kappa=0.3, theta=0.1, sigma=0.03

```python
r0=0.1      # initial interest rate
kappa=0.3   # speed of mean reversion
theta=0.1   # long-term mean
sigma=0.03  # volatility

simulations = vasicek_monte_carlo(x, r0, kappa, theta, sigma)
```

- plot the simulations



Bond Pricing with Vasicek Model III

- for the implementation of bond pricing, we need to calculate the integral
  - the integral is the area under the curve of the function $f(x)$
  - we can approximate the area with rectangle and sum them up

```python
def vasicek_bond_pricing(x, simulations, T=1.0):
    # timestep
    dt = T/float(N_POINTS)
    # calcaute the integral of the r(t) based on simulated paths
    integral_sum = simulations.sum() * dt
    # present value of future cashflow
    present_integral_sum = np.exp(-integral_sum)
    bond_price = x * np.mean(present_integral_sum)

    return bond_price
```

- use the simulated data to price the bond

```python
x = 1e3     # principal investment

pv = vasicek_bond_pricing(x, simulations)

print(f"Bond Price based on Monte-Carlo simulations is = ${pv:,.2f}")
```

✓ 0.0s

```
Bond Price based on Monte-Carlo simulations is = $904.38
```

18. Long-Term Investing

Value Investing

- Long-Term Investing
    - algorithmic trading is about making money quickly using quantitative methods to outperform the market
    - long-term investing is the opposite
- Value-Investing
    - strategy of Benjamin Graham and Warren Buffet
    - selects stocks that traded for less than their intrinsic value
    - this is determined through fundamental analysis
    - value investors believe markets overreact to news
    - good or bad news result in stock price movements that do not correspond with a company's long-term fundamentals
- recap of fundamental analysis
    - is about in-depth study of a given company
    - there are many factors to consider: management style, products and services, balance sheets, income statements, etc.
    - we predict whether the stock is over-/undervalued based on the intrinsic value of the company
    - we may have a good model for the value of a company; but the rest of the world should see the mispricing too
- long-term investing
    - value investors have the tendency to invest in companies not in stocks (companies with good long-term prospects)
    - they buy stocks when the market price is significantly below true value
    - value investing is a long-term strategy "buy and hold"
    - it does not use shorts, no quantitative methods, no hedging, or machine learning

    - value investors ignore the crowd: most people buy shares when the given stock price arises + sell when prices declines (buy high sell low)

Efficient Market Hypothesis

- Efficient Market Hypothesis
    - “it is impossible to beat the market”
    - because stock market efficiency causes existing stock prices to always reflect all relevant information
    - that means according EMH stocks always trade at their fair value (making it impossible to buy under- or overpriced stocks)
    - it is impossible to outperform the market with stock selection or market timing
    - the only way to higher returns is with higher risk
- implications
    - the movement of stock prices $S(t)$ is totally random (>0)
    - stock price changes $dS(t)$ occur in an unpredictable way
    - all available information is reflected in the stock price $S(t)$
    - we can check the autocorrelation of stock returns
    - returns should have low autocorrelation according to the EMH
- companies beating the market
    - Fidelity
        - Peter Lynch managed it from 1977 to 1990
        - achieved annual average return of 29%
    - Renaissance
        - founded by mathematician James Simons in 1982
        - achieved annual average return of 34%
    - Warren Buffet
        - is a long-term value investor
        - achieved annual average return of 22%

Next Courses

- https://www.udemy.com/course/quantitative-finance-algorithmic-trading-ii-time-series/?referralCode=FF9A28EEDCB3D9B0BD8B
- https://www.udemy.com/cart/subscribe/course/1001438/
-