Prof. Sören Petrat, Constructor University

Lecture notes from Fall 2025

Instructor: Prof. Dr. Sören Petrat (Office 112, Res. I)

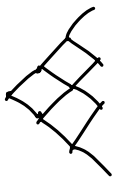
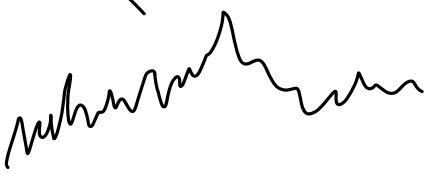
For all organizational matters and the syllabus, please take a look at the class website:

https://math.constructor.university/petrat/teaching/2025_fall_stochastic_modeling

TA: Irfan Basheer

Important:
- Come to every class session, since this is an interactive in-person class.
- Bring your laptop to every class session.

Course topics:

- Introduction to git and Scientific Python
- Basics of finance (interest, cash flows, bonds, options)
- Binomial Tree Models 
- Brownian Motion 
- Stochastic integrals and stochastic ODEs
- Black-Scholes eq.
- Time Series analysis

0. Introduction to git and Scientific Python

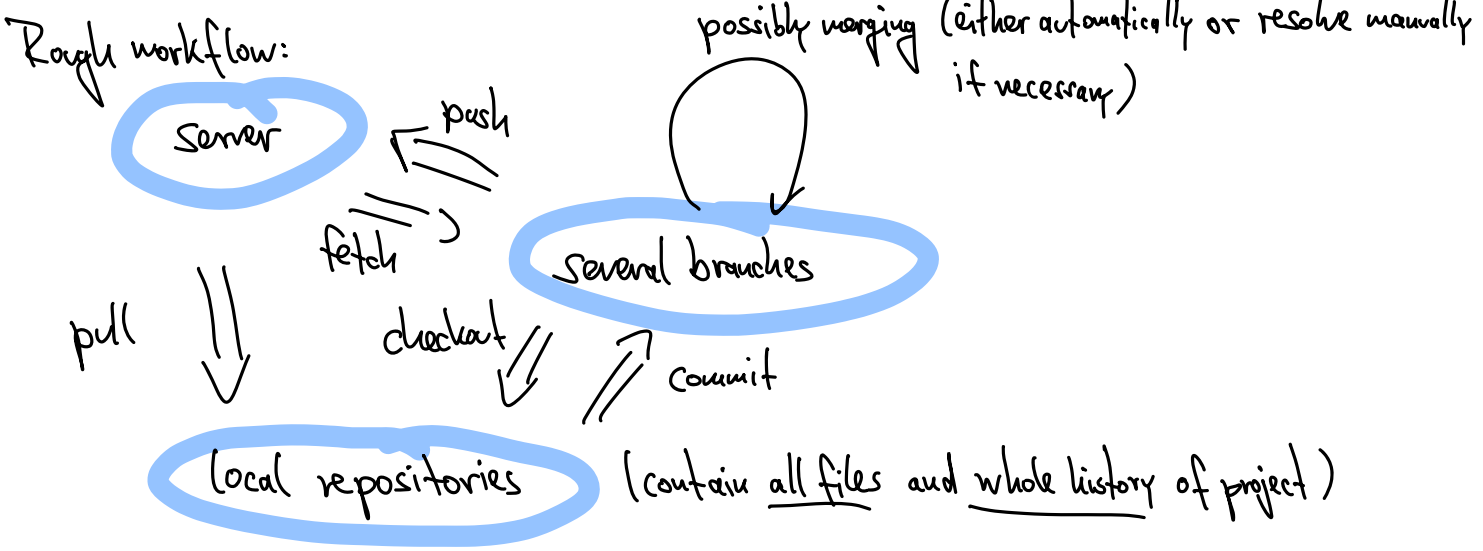
0.1 git

- software (free + open source) (locally on your computer)

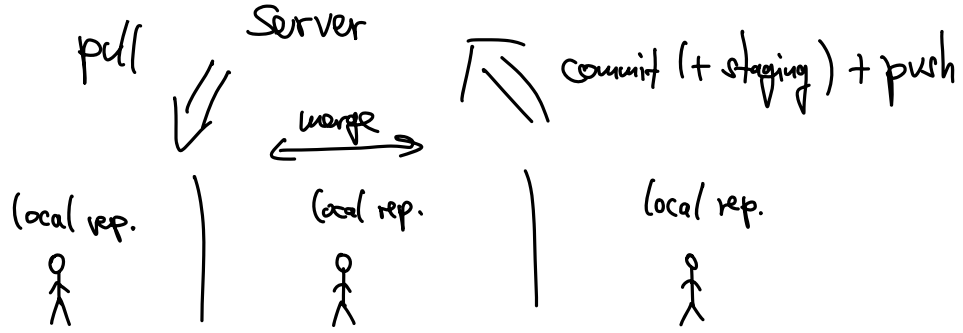
- project development software
  - ↳ version control, change tracking
  - ↳ speed, non-linear workflow (file merging; timestamps)
  - ↳ used predominantly for software development (linux, windows, some google)
  - ↳ useful for large (e.g. also scientific) collaborations

Hosting server: we use bitbucket
(alternative: github)

Rough workflow:

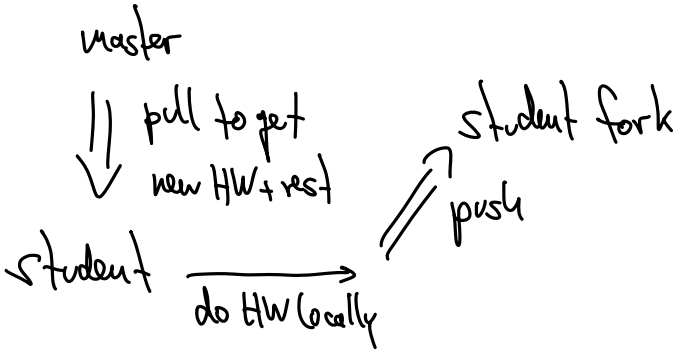


Small project (e.g. small scientific collaboration): Usually one branch sufficient



This class:
- Master branch: material + assignments (public)
- Each student: - maintains separate private branch (fork)
  - get all files from Master, but add your own
- Instructor/TA: write access to all branches

Workflow:



Steps for setting up git (see also "Intro to git for academics" on website):

- download and install git (git-scm.com)
- open command line (git CMD) to configure git (see "Intro")
- get bitbucket account (bitbucket.org), register with Constructor email address (or should already be registered)

- on bitbucket:
  - fork repository soerenpetrat/sm(_2025, mark as private!
  - under "User and group access", add spetrat@constructor.university and TA with "write" access.

- on your computer: clone your repository (via command line)
  your own branch that you just created

0.2 Scientific Python

- optimized for vectorized operations (runs near machine speed, although interpreted language), using NumPy arrays
- SciPy package comprises functionality of all aspects of scientific computing

ToDo:

Install Anaconda package
- ↳ SciPy library already included
- ↳ spyder editor (development environment) ← use this for HW submissions
- ↳ jupyter notebooks (editor + comments + run code fragments separately)

Code examples will be discussed in class, and added to git folder

For homework: always submit .py files
(If you use jupyter notebooks, export them as .py files.)

1. Basics of Financial Math

1.1 Time Value of Money

r = annual interest rate, FV = future value, PV = present value

The value of money changes over time:

$$FV_n = PV\,(1+r)^n = \text{future value after } n \text{ years of interest compounding}$$

$$\Leftrightarrow PV = FV\,(1+r)^{-n}$$

If interest is compounded m times a year: $FV = PV\left(1+\frac{r}{m}\right)^{nm}$

Terminology:
- BEY (bond equivalent yield)
  - ↳ annualized yield, compounded semiannually (twice a year, $m=2$)
- MEY (mortgage equivalent yield)
  - ↳ annualized yield, compounded monthly ($m=12$)

Q.: How to compare different compounding standards?

$\Rightarrow$ Introduce effective annual interest rate r_{eff}:

$$\left(1+\frac{r}{m}\right)^{mn} = \left(1+r_{eff}\right)^n \quad \Rightarrow \quad 1+r_{eff} = \left(1+\frac{r}{m}\right)^m$$

$$\Rightarrow \boxed{r_{eff} = \left(1+\frac{r}{m}\right)^m - 1}$$

Example: $r = 10\%$, BEY $\Rightarrow$ $r_{eff} = \left(1 + \frac{0.1}{2}\right)^2 - 1$

$= (1.05)^2 - 1$

$= 1.1025 - 1$

$= 0.1025 = 10.25\%$

If $m > 1$ is r_{eff} always greater than r

$\Rightarrow r_{eff} = 1 + m\frac{r}{m} + \underbrace{\text{rest}}_{>0} - 1 > r$ Yes!

($(1+x)^m \geq 1 + mx \quad \forall x > -1$ Bernoulli's inequality)

↳ $r_{eff} > r$ even true for $\frac{r}{m} > -1$

One useful idealization: continuous compounding ($m \to \infty$)

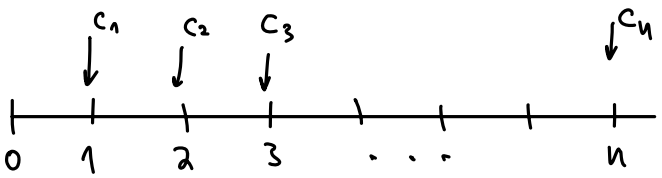
Then $\boxed{FV = PV \lim_{m\to\infty} \left(1 + \frac{r}{m}\right)^{nm}}$

$= PV\, e^{rn}$ $\quad (e^{rn} = \exp(rn) = \text{exponential fct.})$

1.2 General Cash Flows

n years, r yearly interest rate

Suppose at end of year j, we have a given cash flow C_j



$$PV = \frac{C_1}{1+r} + \frac{C_2}{(1+r)^2} + \dots \frac{C_n}{(1+r)^n} = \sum_{j=1}^{n} \frac{C_j}{(1+r)^j}$$

(= present value or "price" of this financial instrument)

Now suppose we want to implement this in python (write code given r and $C_1, \dots, C_n$, and give as output the present value). Need to evaluate polynomial $\sum_{j=1}^{n} C_j x^j$, with $x = \frac{1}{1+r}$.

- explicit "for loop" (discouraged, very slow)

- best: vectorized operation

  ↳ define vector $(1, \dots, n)$: $j = \text{arange}(1, n+1)$

  ↳ given vector $C = (C_1, \dots, C_n)$

  ↳ now implement x^j as a vectorized operation:

  In python $\underbrace{x{*}{*}j}_{x^j}$ is def. componentwise: $x{*}{*}j = (x^1, x^2, x^3, \dots, x^n)$

  ↳ now use dot (or scalar) product to evaluate sum: $PV = \text{dot}(C, x{*}{*}j)$

- Horner's scheme:

$$PV = \Big(\dots \big((C_n x + C_{n-1}) x + C_{n-2} \big) x + \dots + C_1 \Big) x$$

(requires the fewest operations)

(E.g.: $\big((C_3 x + C_2) x + C_1 \big) x \quad , n=3$)

- polyval (fct. from SciPy) uses optimized Horner's scheme

Special cash flow: Annuity, i.e., yearly payment, all $C_j = C$ are the same

Ordinary annuity: pays C at end of year

(There is also the "annuity due", which pays at the beginning of the year.)

Here: general annuity with $C_j = C$ and m payments per year (n given)

$$PV = \sum_{j=1}^{mn} \frac{C}{(1+\frac{r}{m})^j} = C \sum_{j=1}^{mn} \left(\frac{1}{1+\frac{r}{m}}\right)^j$$

Recall geometric series:

$$\sum_{j=0}^{N} x^j = \frac{1-x^{N+1}}{1-x}$$

$$= C \left(\frac{1}{1+\frac{r}{m}}\right) \underbrace{\sum_{j=0}^{mn-1} \left(\frac{1}{1+\frac{r}{m}}\right)^j}$$

$$= \frac{1-(1+\frac{r}{m})^{-mn}}{1-(1+\frac{r}{m})^{-1}}$$

$$= C \frac{1-(1+\frac{r}{m})^{-mn}}{1+\frac{r}{m}-1}$$

$$= C \frac{m}{r}\left[1-(1+\frac{r}{m})^{-mn}\right]$$

Note: • $FV = C \sum_{i=0}^{mn-1} (1+\frac{r}{m})^i = C \frac{m}{r}\left[(1+\frac{r}{m})^{mn} - 1\right]$

• Perpetual annuity: $n \to \infty$

↳ $PV_{perp} = C\frac{m}{r}$