Welcome to BIO-210

Applied software engineering for life sciences

October 7th 2024 – Lecture 4

Prof. Alexander MATHIS

EPFL

Announcements

- As always solutions can be found on our GitHub
- You'll get the quiz results in mid week, so you can take it into account for the quiz next week.
- In person quiz: please come *in time*, there is no added on time. Submission closes at 13:35. To start, you'll need to sign in. Bring your Camipro. No notes are allowed. If you switch to a different tab from Moodle's quiz or communicate with somebody, you'll receive 0 points.
- Monday 15:15 - 16: my office hours at SV 2811

	Date	Topic	Software version	Software releases	Grading / Feedback
0	09/09/2024	Python introduction I			
1	16/09/2024	Public holiday			
2	23/09/2024	Python introduction II			
3	30/09/2024	Git and GitHub (+installation VS Code)			
4	07/10/2024	Project introduction	v1		
5	14/10/2024	Functionify	v2	v1	
6	21/10/2024	EPFL fall break			
7	28/10/2024	Visualization and documentation	v3	v2	code review (API)
8	04/11/2024	Unit-tests, functional tests	v4	v3	
9	11/11/2024	Code refactoring	v5	v4	graded (tests)
10	18/11/2024	Profiling and code optimization	v6	v5	code review
11	25/11/2024	Object oriented programming	v7	v6	graded (speed)
12	02/12/2024	Model analysis and project report	v8	v7	code review (OO)
13	09/12/2024	Work on project			
14	16/12/2024	Wrap up		v8	graded (project)

Developmental background I: pattern formation

Developmental background I: pattern formation

Credits: National Geographic

Are there simple principles that give rise to these different patterns?

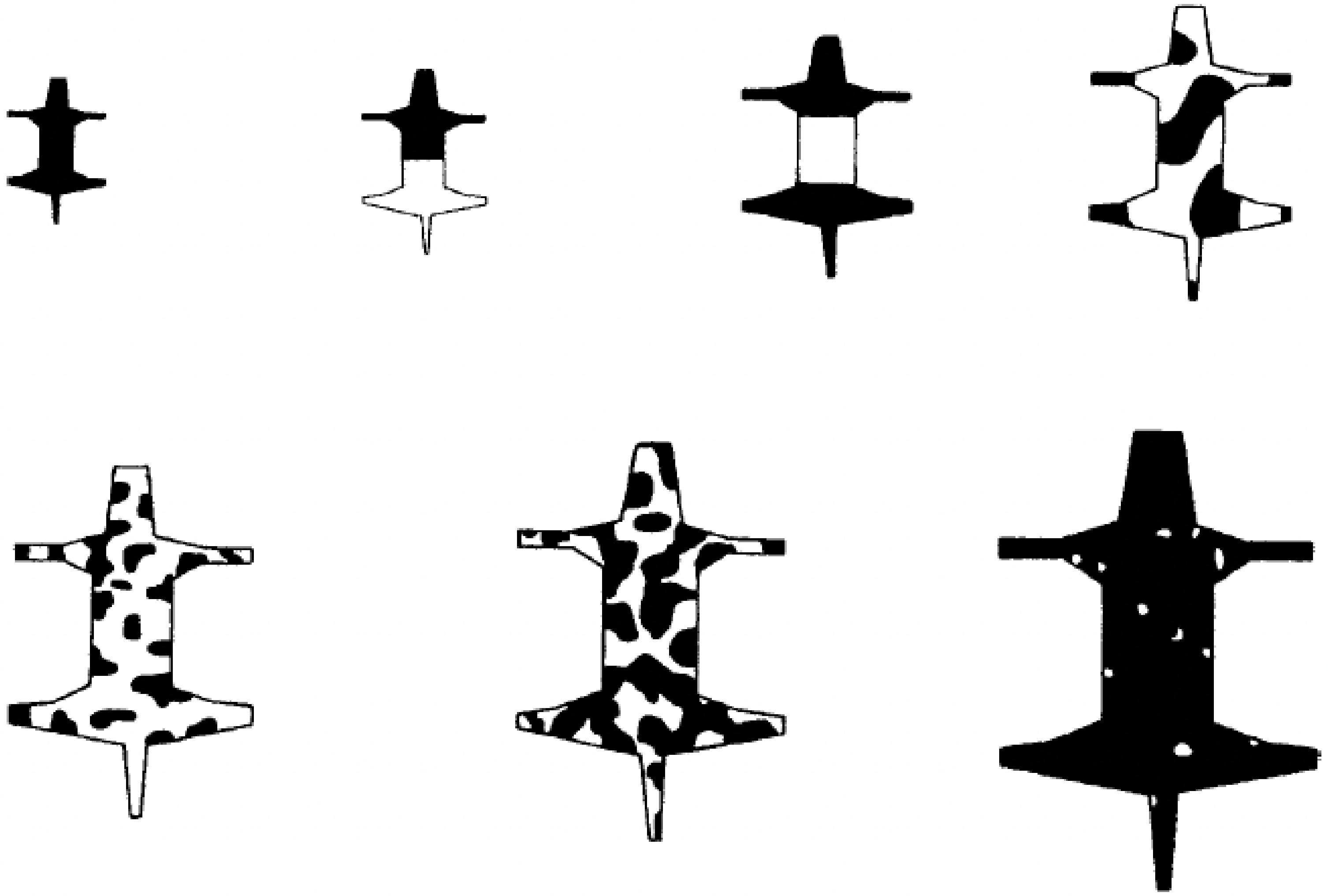


Figure adapted from Mathematical Biology II by Murray.

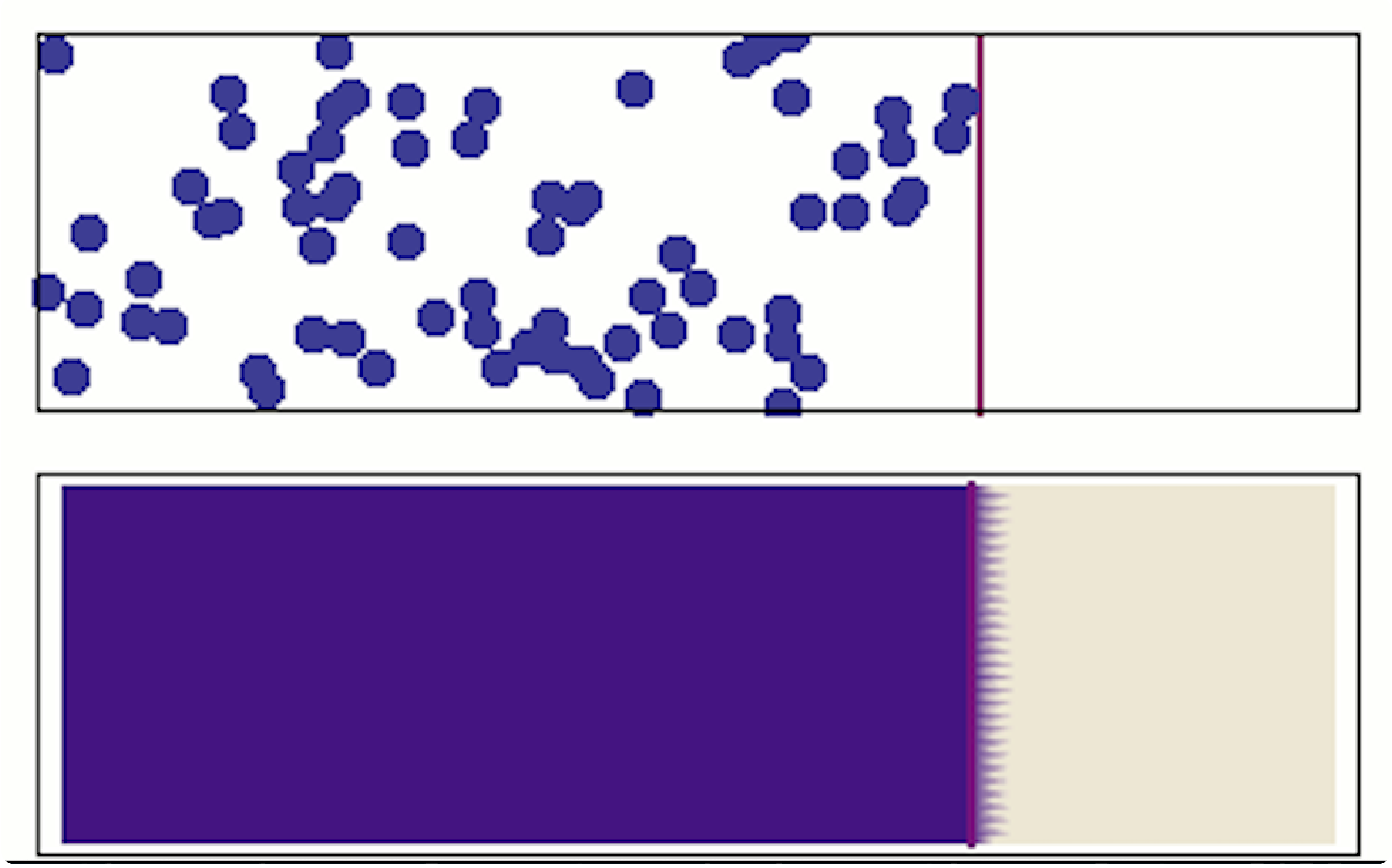
Quiz: What is this equation?

For scalar field $u(x, t)$:

$$\frac{\partial u}{\partial t} = D\nabla^2 u$$

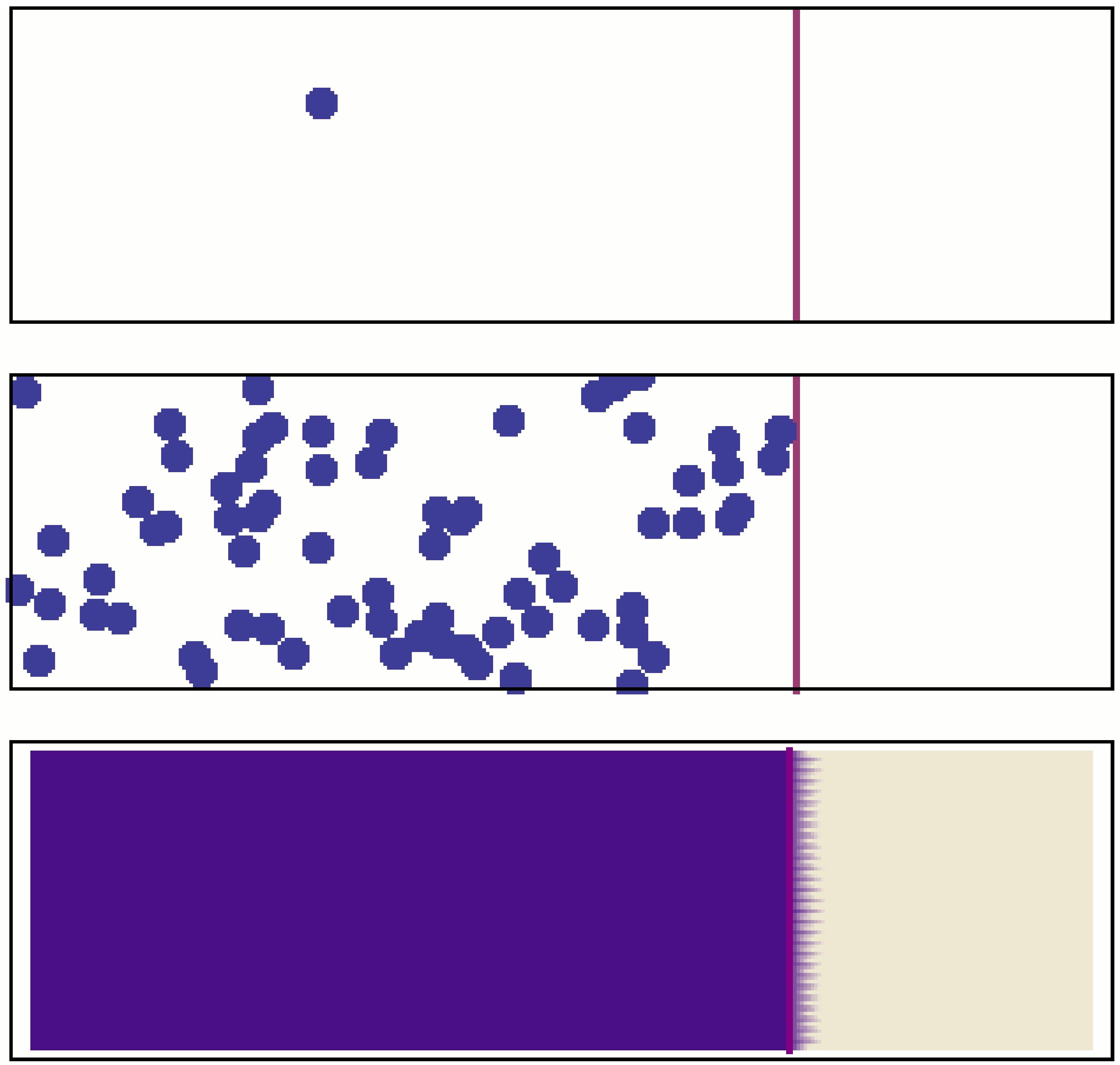
with diffusion coefficient D.

Quiz: What is the solution for the scenario below, when the red barrier is removed?



$$\frac{\partial u}{\partial t} = D\nabla^2 u$$

Here is the solution...



Source: Wikipedia

Project I: Morphogenesis

THE CHEMICAL BASIS OF MORPHOGENESIS

By A. M. TURING, F.R.S. *University of Manchester*

(*Received* 9 *November* 1951—*Revised* 15 *March* 1952)

It is suggested that a system of chemical substances, called morphogens, reacting together and diffusing through a tissue, is adequate to account for the main phenomena of morphogenesis. Such a system, although it may originally be quite homogeneous, may later develop a pattern or structure due to an instability of the homogeneous equilibrium, which is triggered off by random disturbances. Such reaction-diffusion systems are considered in some detail in the case of an isolated ring of cells, a mathematically convenient, though biologically unusual system. The investigation is chiefly concerned with the onset of instability. It is found that there are six essentially different forms which this may take. In the most interesting form stationary waves appear on the ring. It is suggested that this might account, for instance, for the tentacle patterns on *Hydra* and for whorled leaves. A system of reactions and diffusion on a sphere is also considered. Such a system appears to account for gastrulation. Another reaction system in two dimensions gives rise to patterns reminiscent of dappling. It is also suggested that stationary waves in two dimensions could account for the phenomena of phyllotaxis.

The purpose of this paper is to discuss a possible mechanism by which the genes of a zygote may determine the anatomical structure of the resulting organism. The theory does not make any new hypotheses; it merely suggests that certain well-known physical laws are sufficient to account for many of the facts. The full understanding of the paper requires a good knowledge of mathematics, some biology, and some elementary chemistry. Since readers cannot be expected to be experts in all of these subjects, a number of elementary facts are explained, which can be found in text-books, but whose omission would make the paper difficult reading.

1. A model of the embryo. Morphogens

Link to the original publication

Reaction-diffusion equations

The Turing pattern formation mechanism can be modeled with two time-dependent reaction-diffusion equations, describing the evolution and the interaction between an activator $u(x, y)$ and an inhibitor $v(x, y)$ morphogen.

$$\frac{\partial u}{\partial t} = \gamma f(u, v) + \nabla^2 u$$

$$\frac{\partial v}{\partial t} = \gamma g(u, v) + d\nabla^2 v$$

with diffusion terms $\nabla^2 u$ and $\nabla^2 v$ as well as reaction terms $\gamma f(u, v)$ and $\gamma g(u, v)$.

See problem set for more details.

You’ll implement discretized equations

Reaction-diffusion equations:

$$u_{ij}^{(t+1)} = u_{ij}^{(t)} + \Delta t \left[\gamma f(u_{ij}^{(t)}, v_{ij}^{(t)}) + \frac{u_{i-1,j}^{(t)} + u_{i+1,j}^{(t)} + u_{i,j-1}^{(t)} + u_{i,j+1}^{(t)} - 4u_{ij}^{(t)}}{\Delta x^2} \right]$$

and

$$v_{ij}^{(t+1)} = v_{ij}^{(t)} + \Delta t \left[\gamma g(u_{ij}^{(t)}, v_{ij}^{(t)}) + d \frac{v_{i-1,j}^{(t)} + v_{i+1,j}^{(t)} + v_{i,j-1}^{(t)} + v_{i,j+1}^{(t)} - 4v_{ij}^{(t)}}{\Delta x^2} \right]$$

with parameters defined on the problem set.

You will analyze the model and answer two key questions:

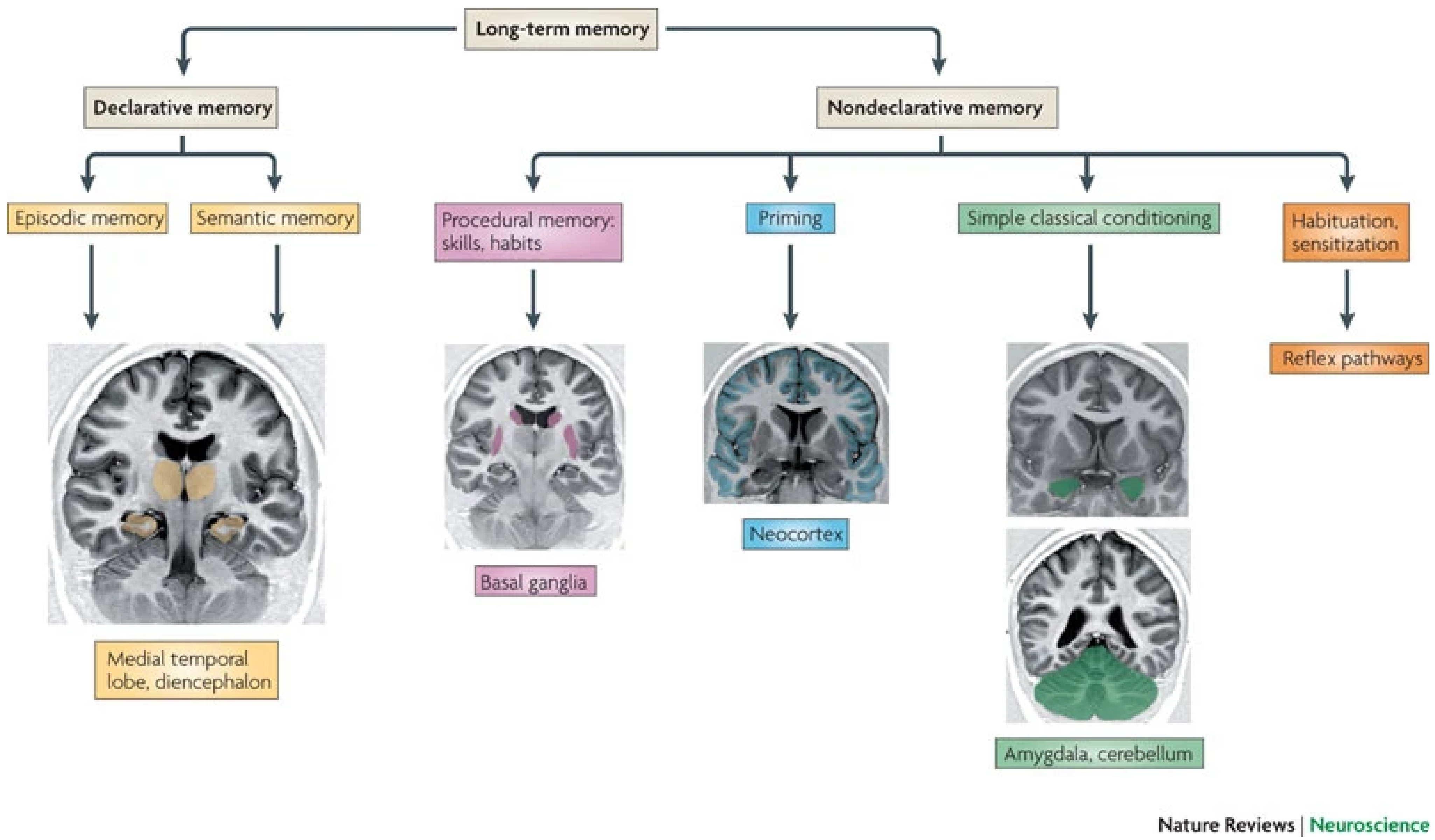
- When is the solution numerically stable?
- What type of coat patterns can emerge?

Project I:

- implement a model of morphogenesis (discretized partial differential equation)
- understand why patterns can emerge with this mechanism
- study what patterns can emerge (incl. visualization)
- integrate tests, provide specified API
- refactor (functions, oop, speed-up)
- developed on GitHub with git

Questions?

Neuroscience background for project II: memory



Source: K Henke, Nature Reviews Neuroscience 2010

- memory serves different functions (see diagram)
- different types of memory are supported by different brain areas (how do we know this?)

Quiz: Madeleine de Proust – what is this about?

"Et tout d’un coup le souvenir m’est apparu. Ce goût c’était celui du petit morceau de madeleine que le dimanche matin, à Combray (parce que ce jour-là je ne sortais pas avant l’heure de la messe), quand j’allais lui dire bonjour dans sa chambre, ma tante Léonie m’offrait après l’avoir trempé dans son infusion de thé ou de tilleul. […]"

From À la recherche du temps perdu by Proust.

Associative memory enables the formation of episodic memories based on associations between components of experienced events.

Subsequently, the partial experience of event A, leads to the recall of event A.

Project II: Hopfield networks

Proc. Natl. Acad. Sci. USA
Vol. 79, pp. 2554–2558, April 1982
Biophysics

Neural networks and physical systems with emergent collective computational abilities

(associative memory/parallel processing/categorization/content-addressable memory/fail-soft devices)

J. J. Hopfield

Division of Chemistry and Biology, California Institute of Technology, Pasadena, California 91125; and Bell Laboratories, Murray Hill, New Jersey 07974

Contributed by John J. Hopfield, January 15, 1982

ABSTRACT Computational properties of use to biological organisms or to the construction of computers can emerge as collective properties of systems having a large number of simple equivalent components (or neurons). The physical meaning of content-addressable memory is described by an appropriate phase space flow of the state of a system. A model of such a system is given, based on aspects of neurobiology but readily adapted to integrated circuits. The collective properties of this model produce a content-addressable memory which correctly yields an entire memory from any subpart of sufficient size. The algorithm for the time evolution of the state of the system is based on asynchronous parallel processing. Additional emergent collective properties include some capacity for generalization, familiarity recognition, categorization, error correction, and time sequence retention. The collective properties are only weakly sensitive to details of the modeling or the failure of individual devices.

calized content-addressable memory or categorizer using extensive asynchronous parallel processing.

The general content-addressable memory of a physical system

Suppose that an item stored in memory is "H. A. Kramers & G. H. Wannier *Phys. Rev.* **60**, 252 (1941)." A general content-addressable memory would be capable of retrieving this entire memory item on the basis of sufficient partial information. The input "& Wannier, (1941)" might suffice. An ideal memory could deal with errors and retrieve this reference even from the input "Vannier, (1941)". In computers, only relatively simple forms of content-addressable memory have been made in hardware (10, 11). Sophisticated ideas like error correction in accessing information are usually introduced as software (10).

There are classes of physical systems whose spontaneous behavior can be used as a form of general (and error-correcting)

Link to the original publication

Associative memory in the Hopfield model

Left: initial state, middle: intermediate and right: final state for 3 examples (for binary images of 130x180).

Later in the semester, you can try this out for your model.

Picture from Introduction To The Theory Of Neural Computation by J. Hertz

Neurons

Source: neuron drawing by Ramón y Cajal

A simple, computational neuron model

Our computational neuron model is subject to the following simplifications:

- binary neurons, linear integration and thresholding (σ)

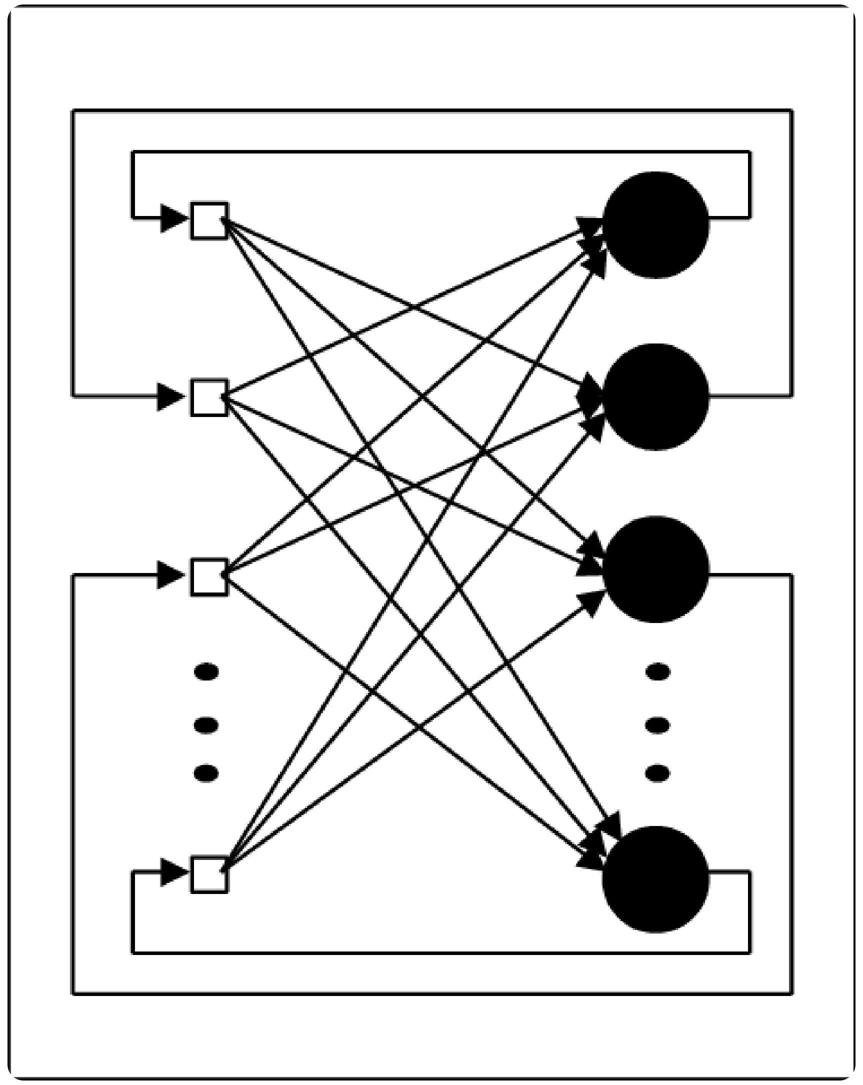
Specifically, the activity of neuron j is given by

$$p_j = \sigma\left(\sum_i \mathbf{W}_{j,i} \cdot \mathbf{p}_i\right)$$

where $\mathbf{W}$ is the weight matrix and the nonlinearity $\sigma(x)$ is defined as

$$\sigma(x) = \begin{cases} -1 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$$

The Hopfield network is a recurrent neural network



At time $t+1$ the activity is given by the weight matrix and the activity at t.

$$\mathbf{p}^{(t+1)} = \sigma\left(\mathbf{W}\mathbf{p}^{(t)}\right)$$

How are memories stored?

Consider M memory patterns $\mathbf{p}^\mu \in \{-1, 1\}^N, \mu \in \{1, ..., M\}$.

The Hebbian learning rule is the simplest one to create a Hopfield network. It is based on the principle "Neurons that fire together, wire together. Neurons that fire out of sync, fail to link". This translates into the following network connectivity:

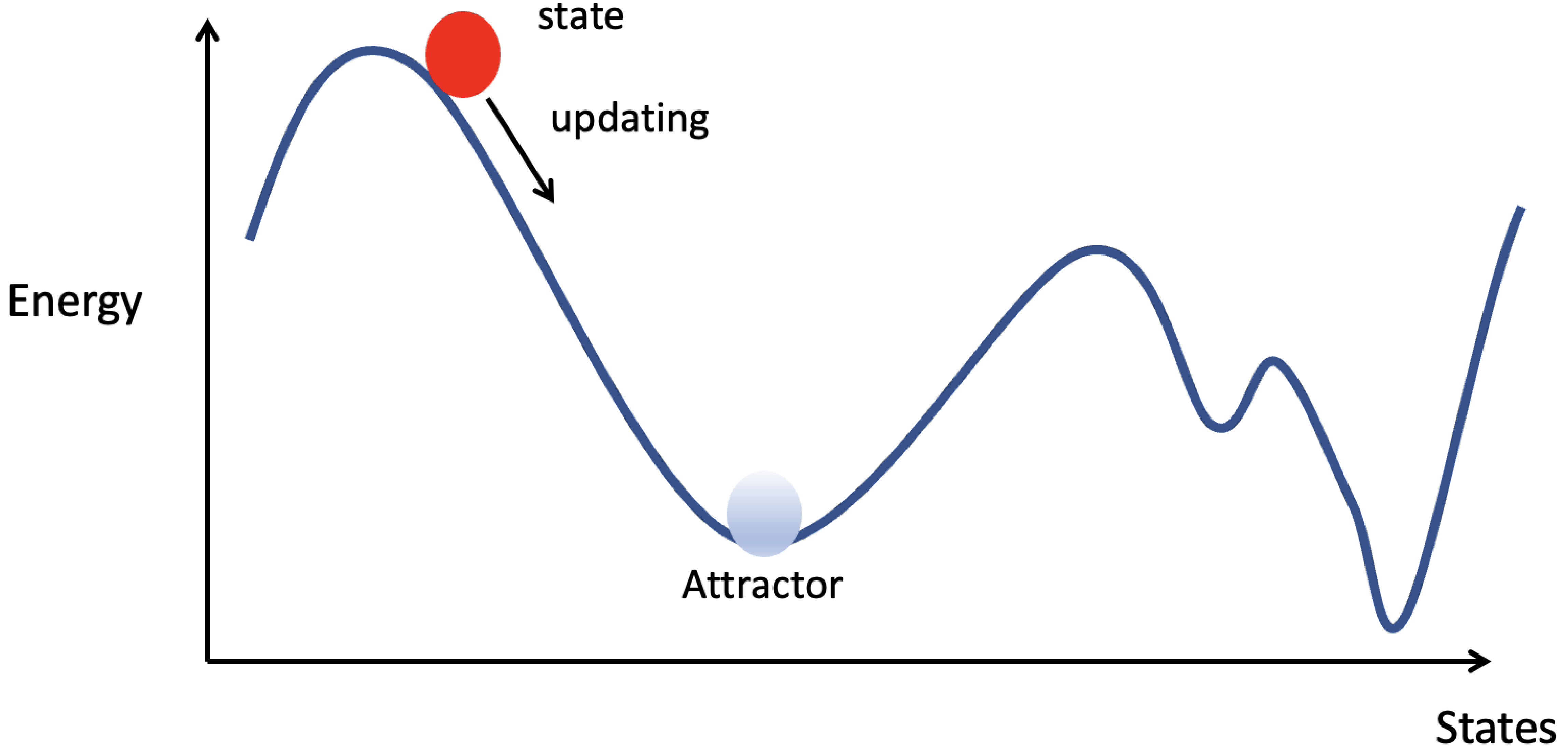
$$w_{ij} = \frac{1}{M} \sum_{\mu=1}^{M} p_i^\mu p_j^\mu \quad i \neq j$$

We can observe that the weight is the average of the contribution of each pattern to the synaptic weight.

Each pattern contributes positively to a certain weight, if the state of the two connected neurons is the same, and negatively otherwise.

Hopfield network

One can show that the dynamics of the Hopfield system minimize a certain energy.



You will analyze the model and answer two key questions:

- How robust are stored memories?
- How many memories can be stored?

Great interview with J. Hopfield

Somewhat recent podcast with Prof. J. Hopfield

I recommend the whole podcast, but in particular the part about associative memory and Hopfield nets (that minute is linked).

Project II:

- implement Hopfield nets for a given set of memories (dynamics and learning)
- understand the Hopfield network
- study dynamics, memory capacity (incl. visualization)
- integrate tests, provide specified API
- refactor (functions, oop, speed-up)
- developed on GitHub with git

See problem set for more details.

Questions?

Logistics of the projects

- we already formed groups of 2-3 students
- each team works on one project (for the whole semester)
- each team was already invited to their private GitHub repo, with the naming convention: `` `https://github.com/EPFL-BIO-210/BIO-210-24-team-ABC` ``.
- Only one dedicated student assistant, the TAs and me will have access

Please put your project choice in the usual form

Do so, before the exercises begin and you **will find your room in the form.**

Problem set and exercise session

As usually, more details can be found on our github website, in particular on this week's problem set that will be discussed in the exercise session

Assessment

Final grade comprises three independent scores:

- 40% individual assessments (short quizzes: 3/5/7/9/11 week via Moodle)
- 35% evaluation of the group project (tests, profiling and project report)
- 25% individual contributions to the group project (participation and individual contributions to code)

Questions?

Quiz: how do you run the code from the first commit of a repository?

Quiz: how do you run the code from the first commit?

```
# Look up the log (and find the first commit)
alex@mac demo-repo % git log --all --graph --decorate --oneline
*   b0337d2 (master) Conflicts fixed and greeting Alberto
|\
| * a7ed742 (feature/new_context) Greeting Lucas
* | 799c9ac Greeting Alberto
|/
* f7d2e70 Greeting program expanded
* 6baa498 (HEAD) new greeting
* 5be7f54 Adding script
alex@mac demo-repo % git checkout 5be7
Previous HEAD position was 6baa498 new greeting
HEAD is now at 5be7f54 Adding script
alex@mac demo-repo % python3 script.py
Hello world!
alex@mac demo-repo % cat script.py
print('Hello world!')
```

Recap: git

- we discussed the data model
- git stores snapshots containing trees, and blobs as commits.

At least in the lecture, we did not discuss distributed git much (so far). But you had your first experiences in the exercise session. Let’s start with a quick reminder.

Reminder: Basic git commands

For more details, see git's reference manual.

- `git help <command>`: get help for a git command
- `git init`: creates a new git repo, with data stored in the .git directory
- `git status`: tells you what's going on
- `git add <filename>`: adds files to staging area
- `git commit`: creates a new commit (**write good commit messages!**)
- `git log`: shows a (flattened) log of history
- `git log --all --graph --decorate`: visualizes history as a DAG
- `git diff <filename>`: show changes you made relative to the staging area
- `git diff <revision> <filename>`: shows differences in a file between snapshots
- `git checkout <revision>`: switch branches or restore working tree files

Quiz: What is a branch and how do you create a new one?

Reminder: Git branching and merging

- `git branch`: shows branches
- `git branch <name>`: creates a branch
- `git checkout -b <name>`: creates a branch and switches to it. This is the same as `git branch <name>`; `git checkout <name>`
- `git merge <revision>`: merges the commit `<revision>` into current branch

A multiple-developer git workflow

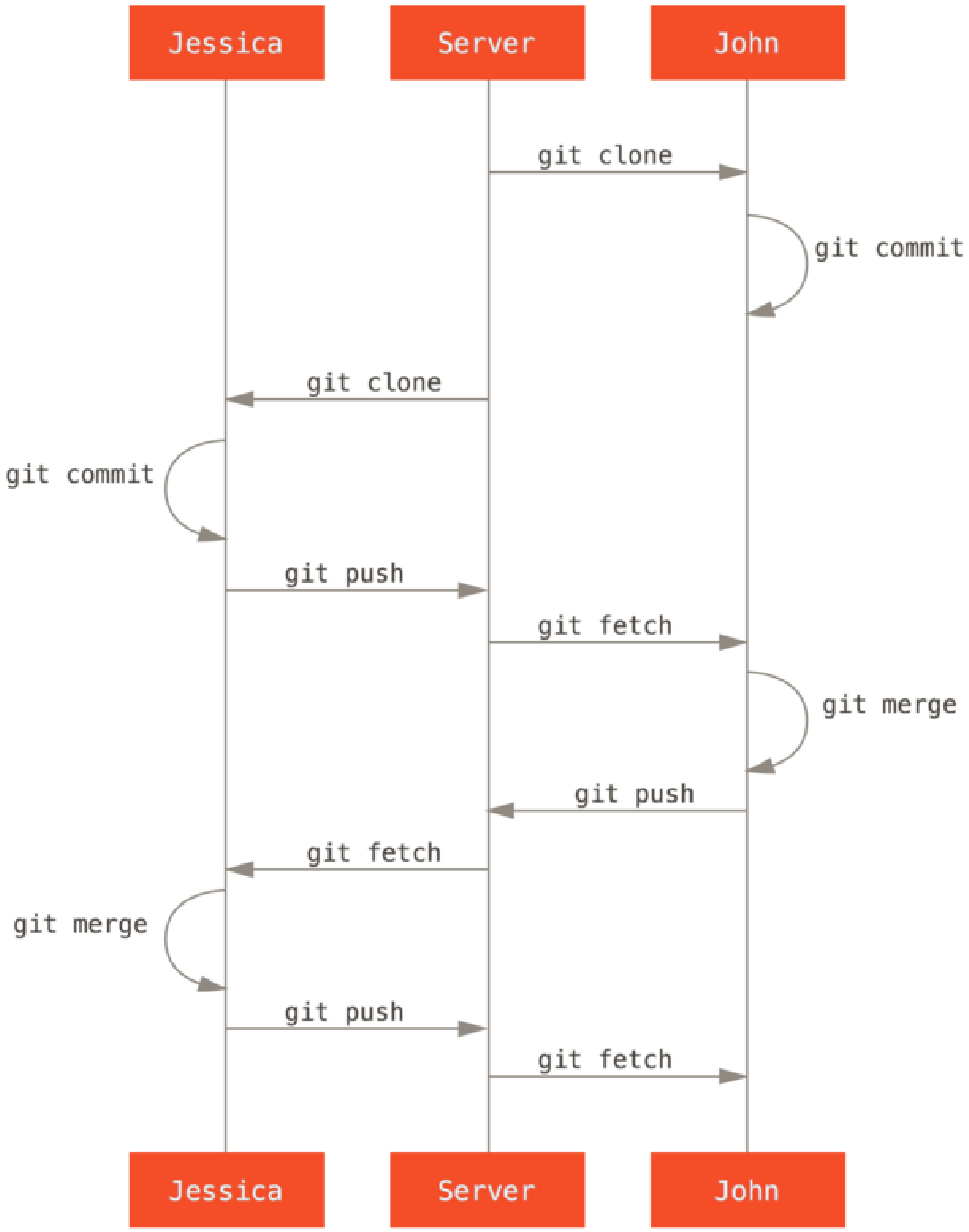


Image source: git docs

Git remote commands

- `git clone <url>`: download repository from remote
- `git fetch`: retrieve objects/references from a remote
- `git push <remote-name> <branch-name>`: upload local repository content to a remote repository, docs; e.g. `git push origin main`
- `git pull`: fetch from and integrate with another repository or a local branch; same as git fetch; git merge

See this git cheat sheet (in French) and English

Git beyond the command line?

- there are many graphical interfaces for git
- git is also integrated in many integrated (code) development environment (IDEs, e.g., Visual Studio Code (our focus), atom, PyCharm, etc.)

GitHub provides web-based software development and version control using git

Advanced topics

- vast resources in the git-book
- Oh Shit, Git!?! (some nice solutions to git-hell)
- removing sensitive data from git
- avoiding accidental commits

Project week I passed …

- Last week, as part of version 1 (released this morning) you implemented a discrete dynamical simulation for either the Turing/Hopfield project (with some parameters)
- this week the task is to re-factor your code by creating an interface that we specifiy in the problem set
- release this v2 by next Monday at 10am.
- we will then review your code and give feedback (this is not graded!)

After lunch:

- Monday 13 - 15: exercises (4 groups) in rooms CO4, CO5, CO6 and CO260 according to teams you formed. See here

Please note that we altered the room assignment slightly!

- Monday 15:15 - 16: my office hours at SV 2809

Today’s summary

- deeper dive into functions: `` `def` ``, `` `return` ``
- scoping, namespaces, LEGB rule, global, globals()
- discussion of Python’s argument matching modes

Try out the commands in the Python shell/notebooks!

After lunch:

- Monday 13 - 15: exercises. *5 groups* in rooms CO4, CO5, CO6 and CO260, CO23. according to teams you formed. See here for finding out your room. But first let us know what project you want to do!
- Monday 15:15 - 16: my office hours at SV 2811

END

Welcome to BIO-210

Applied software engineering for life sciences

October 7th 2024 – Lecture 4

Prof. Alexander MATHIS

EPFL

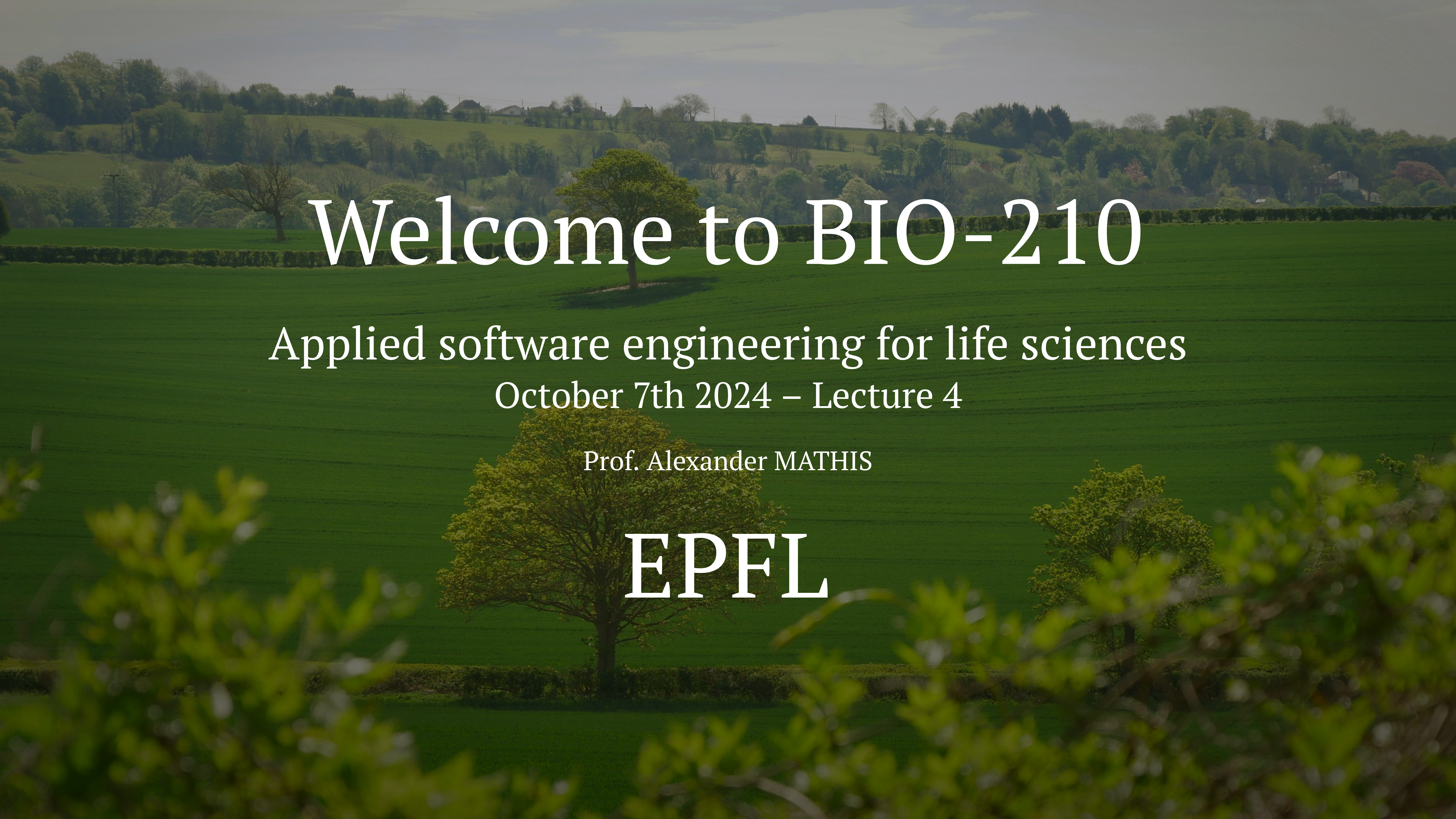
Welcome to BIO-210

Applied software engineering for life sciences

October 7th 2024 – Lecture 4

Prof. Alexander MATHIS

EPFL



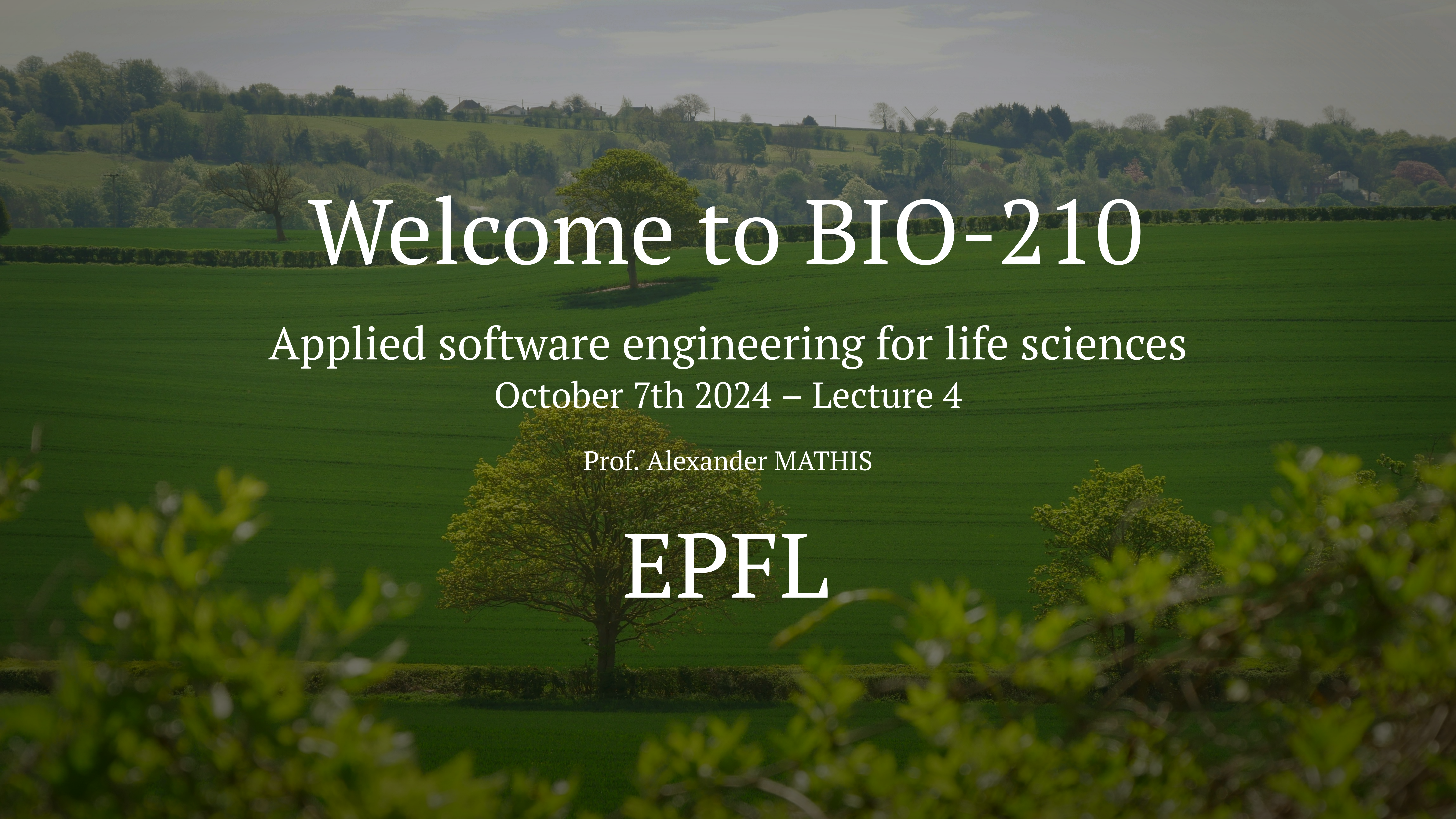
Welcome to BIO-210

Applied software engineering for life sciences

October 7th 2024 – Lecture 4

Prof. Alexander MATHIS

EPFL



Welcome to BIO-210

Applied software engineering for life sciences

October 7th 2024 – Lecture 4

Prof. Alexander MATHIS

EPFL

Welcome to BIO-210

Applied software engineering for life sciences

October 7th 2024 – Lecture 4

Prof. Alexander MATHIS

EPFL

Welcome to BIO-210

Applied software engineering for life sciences

October 15th 2024 – Lecture 5

Prof. Alexander MATHIS

EPFL