Tutorial 1

Tutorial outline

Here is a **tentative** outline of our plans for this semester:

1. Setup *git* and python

2. Introduction to python and *jupyter*:

   Investigate the harmonic oscillator problem with an ODE solver

3. Basic finite difference numerical methods:

   Discretise and implement a code to solve the simple harmonic oscillator problem with the Euler and Störmer-Verlet methods

4. Symbolic programming and dimensional analysis:

   Solve a few problems with the Buckingham-Π theorem

5. Finite difference method to solve hyperbolic problems (part 1):

   Discretise and implement a code to solve the 1D Burgers equation with the Lax-Friedrichs method

6. Finite difference method to solve hyperbolic problems (part 2) and python debugging:

   Discretise and implement a code to solve the 1D and 2D shallow water equations with the Lax-Wendroff method

7. Finite volume method and more advanced python programming:

   Discretise and implement a code to solve the 1D and 2D shallow water equations with a MUSCL-type scheme and a Riemann solver

My goal for this tutorial

We work towards a conducive environment in which we can feel safe to communicate, make mistakes, and learn at our own pace.

Introduction to *git*

git is the most widely used[1] version control system by programmers. As a version control system, *git* helps us manage changes to our code even if we are working alone. Knowledge of how to use a version control system should be in every programmer's toolbox.

Figure 1 below introduces the fundamentals of *git* along with some of the terms used in version control systems. We will cover *git branches* in a separate session.

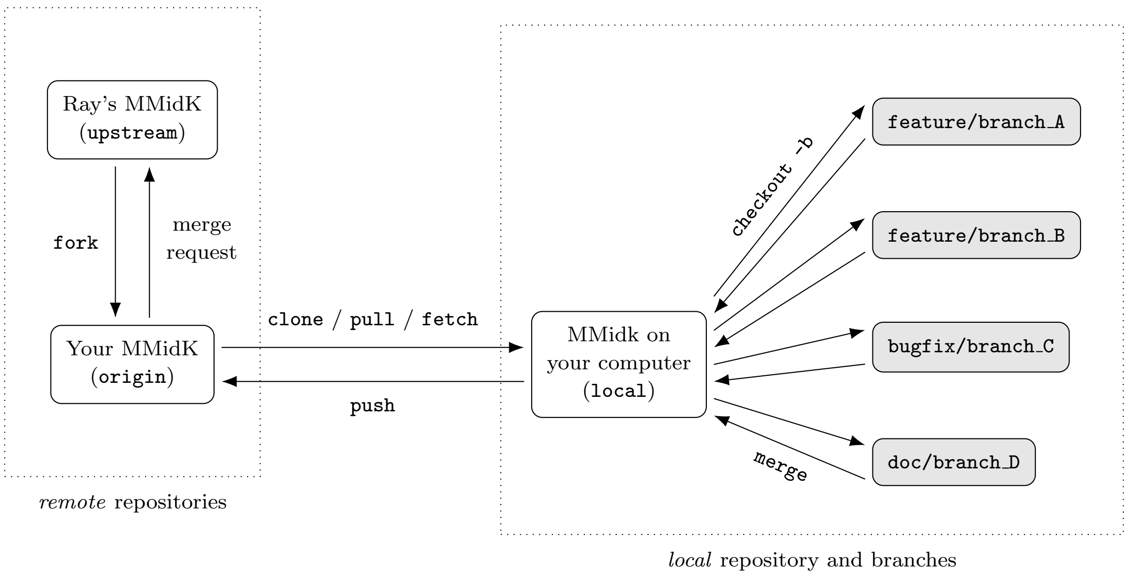

Figure 1: An overview of how we use *git* and the associated commands and terms.

Practical exercise:

1. Install *git*
2. Fork the `MMidk_WS2122` repository
3. Clone the forked repository to your local machine

```
$ git clone https://git.imp.fu-berlin.de/your_name/mmidk_ws2122.git
```

4. In your local repository, create a folder with your name and create a blank file, e.g. on a Unix system,

```
$ touch file.txt
```

5. Look at the changes "seen" by *git*

```
$ git status
```

6. Add and commit this blank file

```
$ git add ./file.txt
# The first -m should be a concise commit message
# The optional second -m can include a more detailed text
$ git commit -m "first commit" -m "more text"
```

7. Push the commit to your remote repository

```
$ git push origin
```

8. Now, go to your remote repository and look at the changes there. Explore and play around with *git*.

[1]If we were to trust wikipedia...

Installing the necessary packages

Apart from *git* where you will have to submit your assignments, **you are free to choose the language and tools to use. However, I will only be providing support for the "recommended" way of doing things.**

This course will be taught with *python3*, and I will be using *jupyter* as an interactive notebook and Visual Studio Code as an integrated development environment (IDE)[2]. Furthermore, I will use Anaconda to manage my python packages and virtual environments.

For now, we will need the following packages:

- python3
- numpy
- scipy
- matplotlib
- jupyterlab

Recommended way to install the packages

1. Install Anaconda
2. Create a virtual environment, e.g.

   ```
   $ conda create --name mmidk
   ```

3. Activate the virtual environment

   ```
   $ conda activate mmidk
   ```

   Now you are in the `mmidk` virtual environment, and all the packages you install below are only accessible within `mmidk`.

4. Now, we install the packages, e.g. for *numpy*,

   ```
   $ conda install numpy
   ```

   A nice thing about Anaconda is that it takes care of all the package dependencies and conflicts for you.

5. Now that you have installed all the packages, you might want to check which version of python you are using

   ```
   $ python --version
   ```

6. Finally, we want to open jupyter lab by typing in the terminal

   ```
   $ jupyter lab
   ```

Some other administrative matters

- You are to submit the assignments by submitting a **merge request** on *git* to the upstream repository.
- You may work in groups for your assignments. There are a couple of ways to work together. The recommended way to do this will be to invite your group members as members to one of your repositories.
- From time to time, you will be asked to present your solution during class.

In the next tutorial, we will solve the linear oscillator problem with the help of *jupyter* notebooks and the python `scipy` library.

[2]Again, if we were to trust wikipedia, Visual Studio Code is the most used IDE among programmers surveyed...