FYS-4096 (2017) Exercise 1: Setting up your numerical experiment part 1

Summary

In this exercise you will set up your first numerical experiment and refresh your memory on Git, Python, and command line usage.

Return you solution following instructions at the end of this file to GitLab before Friday 5 am.

Problem 1: Setting up Ubuntu and GitLab

1. Follow the instructions on the lecture slides and install Ubuntu Linux 17.10 on a virtual machine on your computer.

- If you already have Linux on your PC, make sure that you have installed
  - Python >= 3.6
  - Numpy >= 1.13
  - Scipy >= 1.0
  - Matplotlib >= 2.1.0
  - Mayavi for Python 3 >= 4.5.0
  - Scikit-learn >= 0.19.1
  - Git
  - OpenSSH client

  Note that unfortunately not all distributions have a working version for Mayavi on Python 3.
- After this, you're supposed to do all your work during this course on Linux.

2. Create a GitLab account at gitlab.com and email janne+compphys@solanpaa.fi your Gitlab username and the corresponding email address. You will get a private GitLab group for returning your exercises and access to the course materials.

3. Open up Gnome Terminal from the sidebar.

4. Select your command-line text editor with `$ . select_editor.sh`. It gives you two options: Nano and Vim. Vim is better but with a steeper learning curve.

5. Set up SSH keys for GitLab following https://docs.gitlab.com/ce/ssh/README.html

6. (Optional) Get acquainted with your command-line text editor of choise. For Vim, the command `$ vimtutor` will give you a nice tutorial. For Nano, the wiki at gentoo.org is a good starting point https://wiki.gentoo.org/wiki/Nano/Basics_Guide

Problem 2: Creating a git repository for the first exercise

1. Open up Gnome Terminal from the sidebar.
2. Create an empty folder in your home directory (`$ mkdir exercise1`) and change into it (`$ cd exercise1`).
3. Initialize an empty Git repository on your local machine from command line with `$ git init .`.
4. Open up, e.g., Geany, a light-weight development environment, and create a file README.md in your repository. Write up a short description of the contents and purpose of the directory. You can use Geany for all your future programming needs.
5. Place the newly created README.md under version control with `$ git add README.md`.
6. Make a commit (snapshot) of the repository from command line with `git commit`. This opens up your default command-line text editor.
7. Write a descriptive commit message and save it.
8. Create a GitLab project/repository under your **pre-generated** GitLab group (gitlab.com/comp_phys/2017/< your student number >). Name it 'exercise1'. Push the local 'exercise1' repository from your computer to GitLab following the guide on your newly generated project page (gitlab.com/comp_phys/2017/< your student number >/exercise1).

Problem 3: Programming exercise

Your first task is to write a Python function which estimates numerically the first derivative of any given single-argument PYTHON function. The function signature should be

```
def eval_derivative( function, x, dx )
```

and placed, e.g., in a file called 'differentiation.py' (thanks Rost).

One should be able to use it like

```
from num_calculus.differentiation import eval_derivative

def fun(x): return 3*x*x

der = eval_derivative( fun, 2.0, 0.0001)
```

Create this within a multi-file Python project structured the same way as the example in the lectures. For more details, see http://python-packaging.readthedocs.io/en/latest/minimal.html Please name your Python module 'num_calculus' (in the example it was 'taylor_approximation').

Criteria for passing this exercise: * Everything must be under version control (git) and in your GitLab group for this course. Name the GitLab project for the first exercise as 'exercise1'. * Your Python package should have: * modular design * reasonable naming conventions * compliance with PEP 8 (THE Python style guide) (check program `autopep8`) * unit-tests * short documentation * README.rst -file * a license file * Unit tests should be runnable with `python3 setup.py test` * The package should to be installable with `pip3 install .`, i.e., it needs to have `setup.py`-file

Hint: For this first exercise we provide a preliminary helper tool at https://gitlab.com/comp_phys/2017/material/blob/master/exercises/ex1/helper.py. Place it at the root of your git repo and run it with `python3 helper.py`. It'll give you hints where your code might not match the exercise assignment. Note that it might complain about some irrelevant stuff also.

Problem 4: Numerics exercise

Within the same git repository, `exercise1`, create a Python script called `1st_derivative_convergence_check.py`. This script should illustrate the convergence properties of your `eval_derivative` -function.

Select a non-trivial test function for which you know the first derivative analytically. 1. Plot the absolute error of the numerically estimated derivative vs the spacing dx for a single value of 'x'. 2. How does the error scale with respect to the spacing dx? 3. What is the expected scaling for your implementation?

Use the Python library `matplotlib` for plotting (see Slack for an example). Write the answers to `README.md` together with your name and student number.

Finally, Remember to include also the 'numerics exercise' part of this assignment in your repository.

Returning your exercise

1. Create a file `problems_solved` at the root of your git repo. Inside it, write a comma separated list of problems you have solved, e.g., `1,2,3`
2. Make sure the all your source files are under version control and push them to GitLab.
3. Tag your final solution (git commit) with `final` tag `$ git tag final`
4. Push your commits and the `final` tag to GitLab before Friday 5 AM sharp. `$ git push --all && git push --tags`