

Session 6: Introduction to advanced tools

October 6th, 2022 | Sandra Diaz Pier

Overview

- Versioning (GIT)
- Tests
 - Types
 - How to start testing
 - Unittests
- Debugging
 - pdb
- Interactive Development Environments

Git: Why

- Storage (backup) of source code file
- Who changed what when
- Undo / redo
- Facilitates working on multiple versions of a software
- Merge of changes from multiple developers

Git

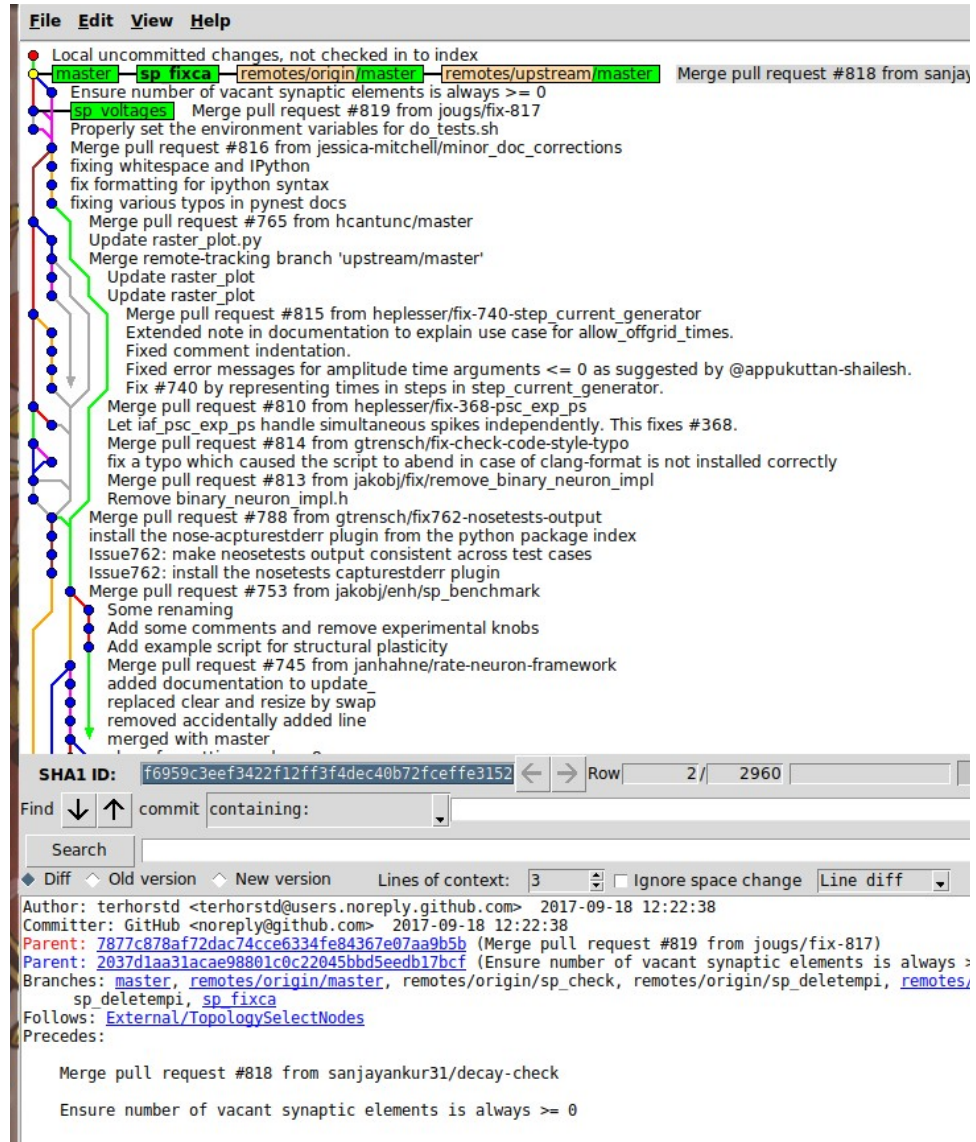
- 70s software interface
- Command line with 'intuitive' arguments
- Graphical user interfaces: Tortoise git (Windows), GitKraken (Linux)
- Integration in mature IDE's: PyCharm, Visual studio, Eclipse



<https://xkcd.com/1597/>

Git

- clone
- checkout
- add
- commit
- fetch
- pull
- push
- remote
- branch



The screenshot displays the Git GUI interface. The top section shows a commit history graph with various branches (master, sp_fixca, remotes/origin/master, remotes/upstream/master) and their corresponding commit messages. The bottom section provides a detailed view of a specific commit, including its SHA1 ID, author, committer, parent, and branches.

Commit History (Top):

- Local uncommitted changes, not checked in to index
- master — sp_fixca — remotes/origin/master — remotes/upstream/master Merge pull request #818 from sanjay
- Ensure number of vacant synaptic elements is always ≥ 0
- sp_voltages Merge pull request #819 from jousg/fix-817
- Properly set the environment variables for do_tests.sh
- Merge pull request #816 from jessica-mitchell/minor_doc_corrections
- fixing whitespace and IPython
- fix formatting for ipython syntax
- fixing various typos in pyneust docs
- Merge pull request #765 from hcantunc/master
- Update raster_plot.py
- Merge remote-tracking branch 'upstream/master'
- Update raster_plot
- Update raster_plot
- Merge pull request #815 from hepler/fix-740-step_current_generator
- Extended note in documentation to explain use case for allow_offgrid_times.
- Fixed comment indentation.
- Fix #740 by representing times in steps in step_current_generator.
- Merge pull request #810 from hepler/fix-368-psc_exp_ps
- Let iaf_psc_exp_ps handle simultaneous spikes independently. This fixes #368.
- Merge pull request #814 from gtrensch/fix-check-code-style-typo
- fix a typo which caused the script to abend in case of clang-format is not installed correctly
- Merge pull request #813 from jakobj/fix/remove_binary_neuron_impl
- Remove binary_neuron_impl.h
- Merge pull request #788 from gtrensch/fix762-nosetests-output
- install the nose-apturederr plugin from the python package index
- Issue762: make nosetests output consistent across test cases
- Issue762: install the nosetests capturederr plugin
- Merge pull request #753 from jakobj/enh/sp_benchmark
- Some renaming
- Add some comments and remove experimental knobs
- Add example script for structural plasticity
- Merge pull request #745 from janhahne/rate-neuron-framework
- added documentation to update_
- replaced clear and resize by swap
- removed accidentally added line
- merged with master

Commit Details (Bottom):

- SHA1 ID: f6959c3eef3422f12ff3f4dec40b72fcef3152
- Find: commit containing:
- Search:
- Diff: Old version New version Lines of context: 3 Ignore space change Line diff
- Author: terhorstd <terhorstd@users.noreply.github.com> 2017-09-18 12:22:38
- Committer: GitHub <noreply@github.com> 2017-09-18 12:22:38
- Parent: 7877c878af72dac74cce6334fe84367e07aa9b5b (Merge pull request #819 from jousg/fix-817)
- Parent: 2037d1aa31acae98801c0c22045bbd5eedb17bcf (Ensure number of vacant synaptic elements is always ≥ 0)
- Branches: master, remotes/origin/master, remotes/origin/sp_check, remotes/origin/sp_deletempi, remotes/sp_deletempi, sp_fixca
- Follows: External/TopologySelectNodes
- Precedes:
- Merge pull request #818 from sanjayankur31/decay-check
- Ensure number of vacant synaptic elements is always ≥ 0

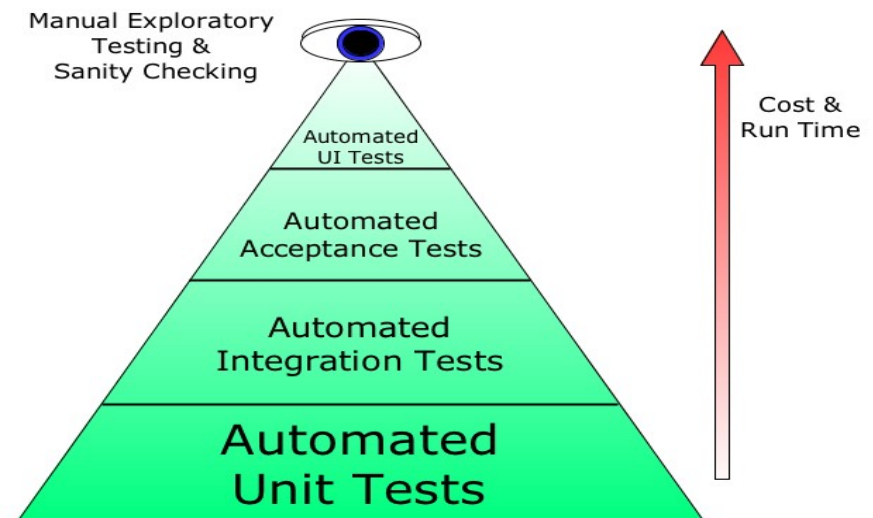
Testing

- Automatic programs or checklist assessing the correction functioning of software.
- Prevent introduction of errors when adding features.
- But also:
 - Tests as documentation
 - Leads to better design: loose coupling
 - In larger projects, improved development speed (mostly due to reduction in bugs to be solved)

Testing pyramid

- Major types of tests:
 - Manual testing
 - Data driven delta testing / regression testing
 - Component testing
 - Unit testing

The concepts are fuzzy and there is overlap and different names for the same thing



<http://willhamill.com/2013/08/12/automated-testing-and-the-evils-of-ice-cream>

How to start testing?

- Writing down the **manual tests** you already do
 - Doubles as documentation
- Create an **data driven delta** test
 - Create test data
 - Forces you to think about ‘user’ interactions
 - Doubles as introductory how-to
- Pick a single important **component** and disconnect it from the rest.
 - And continue doing this till you end up with:
- **Unit test** for small parts of the code that do one and only one thing.

Python: unittest

- Based on the xunit standard
- Setup -> test -> teardown
 1. Create files, etc. needed to run the component
 2. Run individual function and test the correct output eg:
 - `assertEqual`
 - `assertTrue`
 - `assertExceptionThrown`
 3. Delete used resources

<http://pythontesting.net/framework/unittest/unittest-introduction/>

Python: unittest

```
import unittest
```

```
def function(parameter):  
    return parameter
```

```
class TestSomething(unittest.TestCase):
```

```
    def setUp(self):  
        pass
```

```
    def test_fail(self):  
        self.assertEqual(function(13), 12)
```

```
    def test_succes(self):  
        self.assertEqual(function(12), 12)
```

```
    def tearDown(self):  
        pass
```

```
if __name__ == '__main__':  
    unittest.main()
```

```
wouter@WKLIJNWORK:/mnt/c/work$ python3 unittester.py  
F.
```

```
=====
```

```
FAIL: test_something_fail (__main__.TestSomething)
```

```
-----
```

```
Traceback (most recent call last):
```

```
  File "unittester.py", line 15, in test_fail
```

```
    self.assertEqual(function(13), 12)
```

```
AssertionError: 13 != 12
```

```
-----
```

```
Ran 2 tests in 0.001s
```

```
FAILED (failures=1)
```

Debugging

- Debug print statements
- Use binary search to find the problem. If you know your program this is often the fastest
- If the program is big, or not your own, it's a hard problem:

```
python -m pdb program.py
```

Debugging: pdb

| Command | action |
|---------------|---|
| n | Execute the next command |
| enter | Repeat the last command |
| q | Hard exit (with a signal / exception) |
| p <var>,<var> | Print the value of the variable |
| c | Continue with program (until trace_point) |
| s | Step into a function |
| r | Continue till end of function |
| list <n1,n2> | Print surrounding code, include (n1, n2) |

Debugging: pdb cont.

- PDB starts your program and halts at the first statement.
- For large programs you can add trace points:

```
import pdb  
pdb.set_trace()
```
- Execution will drop into debugging mode

When doing interactive development:

- `pdb.run('statement to evaluated')`

<https://pymotw.com/2/pdb/>

Debugging: pdb advanced

Interactive development:

- `pdb.run('statement to evaluated')`
- Postmortem:
 - `pdb.pm()`
“Debugging of the `sys.last_backtrace`”
 - Could be use in combination with `except`
- For more in-depth information:
 - <https://pymotw.com/3/pdb/index.html>

<https://pymotw.com/2/pdb/>

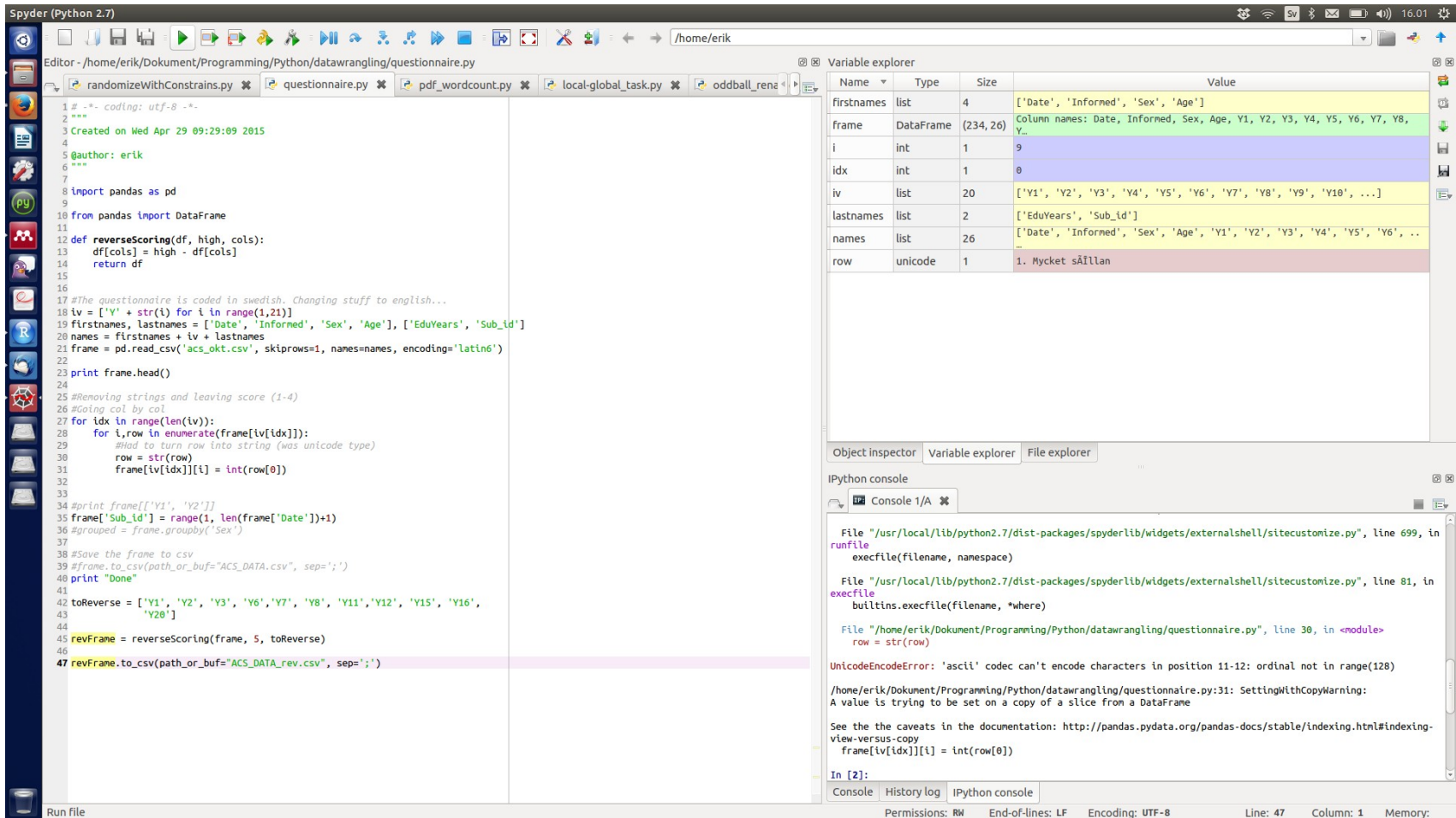
IDE

- The biggest difference between python and Matlab is the Integrated Development Environment (IDE)
- Python is typically interacted with via code or console.
- Selecting an IDE is an ‘important’ choice.
 - It takes time to get use to a IDE
 - Operating system
 - Features

IDE

- Spyder: MATLAB like interface
 - Available on most operating systems
 - Python centric
- Visual Studio: python development tools
 - Windows
 - Prepared for later C++ development (Cython)
- Eclipse JAVA based but supports most languages
 - Available on most operating systems
 - Prepared for later C++ development
- PyCharm. Python centric IDE

IDE: Spyder



The screenshot displays the Spyder Python IDE interface. The main window is divided into several panels:

- Editor:** Shows a Python script named `questionnaire.py` with the following code:


```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Apr 29 09:29:09 2015
4
5 @author: erik
6 """
7
8 import pandas as pd
9
10 from pandas import DataFrame
11
12 def reverseScoring(df, high, cols):
13     df[cols] = high - df[cols]
14     return df
15
16
17 #The questionnaire is coded in swedish. Changing stuff to english...
18 tv = ['Y' + str(i) for i in range(1,21)]
19 firstnames, lastnames = ['Date', 'Informed', 'Sex', 'Age'], ['EduYears', 'Sub_id']
20 names = firstnames + tv + lastnames
21 frame = pd.read_csv('acs_okt.csv', skiprows=1, names=names, encoding='latin6')
22
23 print frame.head()
24
25 #Removing strings and leaving score (1-4)
26 #Going col by col
27 for idx in range(len(tv)):
28     for i,row in enumerate(frame[iv[idx]]):
29         #Had to turn row into string (was unicode type)
30         row = str(row)
31         frame[iv[idx]][i] = int(row[0])
32
33
34 #print frame[['Y1', 'Y2']]
35 frame['Sub_id'] = range(1, len(frame['Date'])+1)
36 #grouped = frame.groupby('Sex')
37
38 #Save the frame to csv
39 #frame.to_csv(path_or_buf='ACS_DATA.csv', sep=';')
40 print "Done"
41
42 toReverse = ['Y1', 'Y2', 'Y3', 'Y6', 'Y7', 'Y8', 'Y11', 'Y12', 'Y15', 'Y16',
43             'Y20']
44
45 revFrame = reverseScoring(frame, 5, toReverse)
46
47 revFrame.to_csv(path_or_buf='ACS_DATA_rev.csv', sep=';')
```
- Variable explorer:** Displays a table of variables in the current namespace:

| Name | Type | Size | Value |
|------------|-----------|-----------|--|
| firstnames | list | 4 | ['Date', 'Informed', 'Sex', 'Age'] |
| frame | DataFrame | (234, 26) | Column names: Date, Informed, Sex, Age, Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y... |
| i | int | 1 | 9 |
| idx | int | 1 | 0 |
| iv | list | 20 | ['Y1', 'Y2', 'Y3', 'Y4', 'Y5', 'Y6', 'Y7', 'Y8', 'Y9', 'Y10', ...] |
| lastnames | list | 2 | ['EduYears', 'Sub_id'] |
| names | list | 26 | ['Date', 'Informed', 'Sex', 'Age', 'Y1', 'Y2', 'Y3', 'Y4', 'Y5', 'Y6', ...] |
| row | unicode | 1 | 1. Mycket sällan |
- IPython console:** Shows the execution of the script, including an error message:


```
File "/usr/local/lib/python2.7/dist-packages/spyderlib/widgets/externalshell/sitecustomize.py", line 699, in
runfile
execfile(filename, namespace)

File "/usr/local/lib/python2.7/dist-packages/spyderlib/widgets/externalshell/sitecustomize.py", line 81, in
execfile
builtin.execfile(filename, *where)

File "/home/erik/Dokument/Programming/Python/datawrangling/questionnaire.py", line 30, in <module>
row = str(row)

UnicodeEncodeError: 'ascii' codec can't encode characters in position 11-12: ordinal not in range(128)

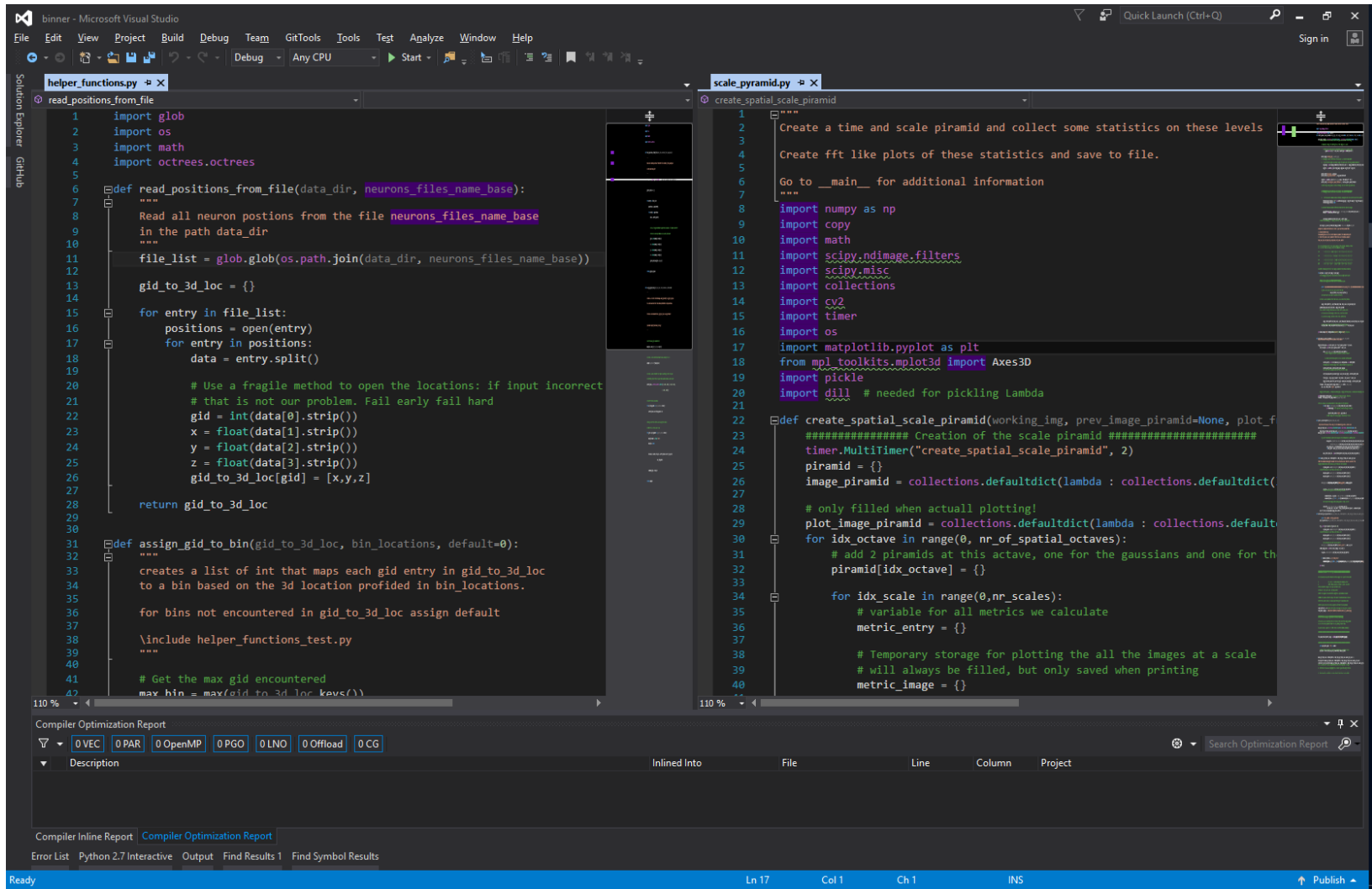
/home/erik/Dokument/Programming/Python/datawrangling/questionnaire.py:31: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html#indexing-view-versus-copy
frame[iv[idx]][i] = int(row[0])

In [2]:
```

<https://www.marsja.se/rstudio-like-python-ides-rodeo-spyder/>

IDE: Visual Studio



The screenshot displays the Microsoft Visual Studio IDE with two Python files open. The left pane shows `helper_functions.py` with a function `read_positions_from_file` that reads neuron positions from a file. The right pane shows `scale_pyramid.py` with a function `create_spatial_scale_pyramid` that creates a time and scale pyramid. The bottom of the window features a **Compiler Optimization Report** window with tabs for `0 VEC`, `0 PAR`, `0 OpenMP`, `0 PGO`, `0 LNO`, `0 Offload`, and `0 CG`. The status bar at the bottom indicates the current line (Ln 17), column (Col 1), and character (Ch 1).

```

1  import glob
2  import os
3  import math
4  import octrees.octrees
5
6  def read_positions_from_file(data_dir, neurons_files_name_base):
7      """
8      Read all neuron positions from the file neurons_files_name_base
9      in the path data_dir
10     """
11     file_list = glob.glob(os.path.join(data_dir, neurons_files_name_base))
12
13     gid_to_3d_loc = {}
14
15     for entry in file_list:
16         positions = open(entry)
17         for entry in positions:
18             data = entry.split()
19
20             # Use a fragile method to open the locations: if input incorrect
21             # that is not our problem. Fail early fail hard
22             gid = int(data[0].strip())
23             x = float(data[1].strip())
24             y = float(data[2].strip())
25             z = float(data[3].strip())
26             gid_to_3d_loc[gid] = [x,y,z]
27
28     return gid_to_3d_loc
29
30
31 def assign_gid_to_bin(gid_to_3d_loc, bin_locations, default=0):
32     """
33     creates a list of int that maps each gid entry in gid_to_3d_loc
34     to a bin based on the 3d location provided in bin_locations.
35
36     for bins not encountered in gid_to_3d_loc assign default
37
38     \include helper_functions_test.py
39     """
40
41     # Get the max gid encountered
42     max_gid = max(gid_to_3d_loc.keys())

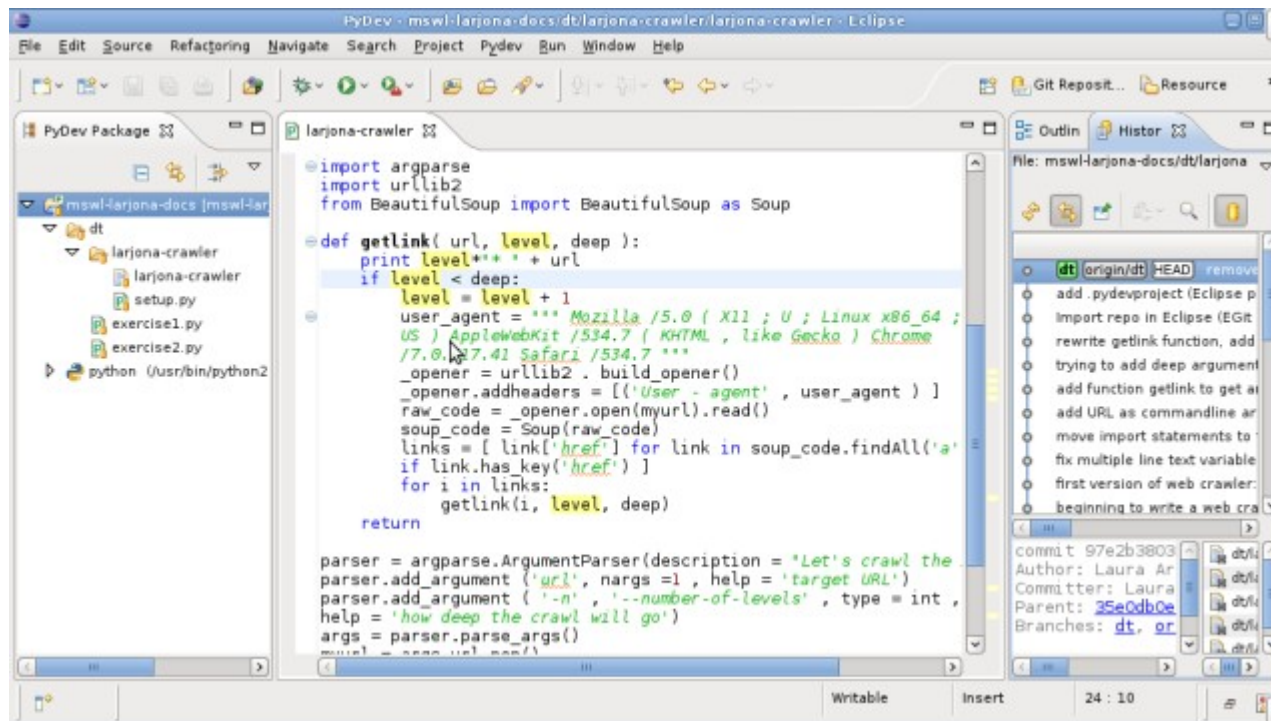
```

```

1  """
2  Create a time and scale pyramid and collect some statistics on these levels
3
4  Create fft like plots of these statistics and save to file.
5
6  Go to __main__ for additional information
7  """
8
9  import numpy as np
10 import copy
11 import math
12 import scipy.ndimage.filters
13 import scipy.misc
14 import collections
15 import cv2
16 import timer
17 import os
18 import matplotlib.pyplot as plt
19 from mpl_toolkits.mplot3d import Axes3D
20 import pickle
21 import dill # needed for pickling Lambda
22
23 def create_spatial_scale_pyramid(working_img, prev_image_pyramid=None, plot_f
24     """##### Creation of the scale pyramid #####"""
25     timer.MultiTimer("create_spatial_scale_pyramid", 2)
26     pyramid = {}
27     image_pyramid = collections.defaultdict(lambda : collections.defaultdict(
28
29     # only filled when actually plotting!
30     plot_image_pyramid = collections.defaultdict(lambda : collections.default
31     for idx_octave in range(0, nr_of_spatial_octaves):
32         # add 2 pyramids at this octave, one for the gaussians and one for the
33         pyramid[idx_octave] = {}
34
35         for idx_scale in range(0, nr_scales):
36             # variable for all metrics we calculate
37             metric_entry = {}
38
39             # Temporary storage for plotting the all the images at a scale
40             # will always be filled, but only saved when printing
41             metric_image = {}

```

IDE: Eclipse



<https://larjona.wordpress.com/2011/09/27/first-steps-with-python-and-eclipse-ide/>

Thank you for your attention

References and further reading:

<https://www.slideshare.net/phpcodemonkey/introduction-to-version-control-presentation>

<https://pythonconquerstheuniverse.wordpress.com/2009/09/10/debugging-in-python/>