

Intro

- Martin
 - visual neuroscience and electrophysiology
 - Laura Busse's lab, work on mouse vision and behaviour, record neurophysiology with optogenetic manipulation
 - Python experience:
 - needed visual stimulus presentation program
 - found VisionEgg, written in Python by a neuroscientist/entomologist, flexibility, features we needed, and it was free!
 - also use Python for:
 - neural spike sorting & data analysis
 - database system for storing all our lab data
 - simpler things like renaming a bunch of files/folders in nested hierarchy
 - arithmetic, simple calculations instead of a calculator
- quick round of intros: name, field of study, programming background, data analysis wants/needs
- why programming/coding?
 - for those who don't know any programming: how do you analyze your data?
 - probably Excel, which is a mini language
 - programming gives you the flexibility to do just about anything with your data, or any data
 - use lots of small building blocks, combine them in sequences in infinite ways
 - trick is to learn which blocks to use, and when, and how best to combine them
 - can make scientific discoveries, launch spaceships, give machines the ability to learn, or swing the outcome of an election - can't do that with Excel
- why Python?
 - making science more open and reproducible:
 - open publication, open data, open hardware, and open code
 - LMU open science center: http://www.osc.uni-muenchen.de/about_us/index.html
 - Python isn't just free, it's open source: code behind Python and all of its libraries can be inspected, modified, improved by anyone
 - compare with Matlab
 - code written in Python is by default open - there's no compilation as in many other languages
 - Python is used more outside of science and engineering than within it - huge user base
 - easy to find answers to problems, or existing tools (libraries) that already do what you want to do
 - end result is fewer lines of code that *you* need to write to accomplish something
 - Python has been growing in popularity over many years, now ranked #1 depending on how you measure it - see charts at <https://stackoverflow.blog/2017/09/06/incredible-growth-python/>
 - fits your brain - Python syntax is intuitive, often reads like English
 - Guido: "programming languages are as much about communicating ideas between developers as they are about telling a computer what to do"

- batteries included - lots and lots of tools built in, and endless more that can be installed
 - bit like toolboxes in Matlab, but free and open
- makes you employable
- Python installation is relatively small, on disk and in memory
- launches fast compared to Matlab
- motivation:
 - example: load data, analyze, plot, save
 - `python LFP_analysis_example.py`
- class stuff
 - integrated lecture + in-class exercises
 - lecture: introduce concepts, exercises: practice the concepts
 - please keep your audio muted when not speaking
 - recording the Zoom meeting?

Outline

1. Python basics
2. Python basics 2
3. collections
4. [numpy](#) 1D arrays
5. numpy data types
6. numpy file operations, plotting with [matplotlib](#)
7. more matplotlib, matrices
8. image analysis
9. data analysis with [Pandas](#)
10. statistics
11. organizing code, data, results; [version control](#) with [Git](#); work on project
12. options:
 - review
 - dimension reduction & clustering
 - hierarchical indexing in pandas
 - work on project

Grades

- attendance: 25%, all or nothing
 - can miss up to 3 out of 12 classes, any more and no credit for course
- homework: 25%, 4 homework assignments, short, similar to in-class exercises
 - graded by attempt, not outcome
 - we'll go over solutions the following week
 - programming is a skill, like any other language, need practice listening/speaking/reading/writing
 - struggling and overcoming the struggle is part of the process
- final project: 50%

- meet list of requirements that are fairly easy to fulfill
- code needs to run successfully