Intro

- about 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!
  - now use Python for:
    - arithmetic, simple calculations instead of a calculator
    - simpler things like renaming a bunch of files/folders in nested hierarchy
    - acquiring video frames from an eye-tracking camera
    - neural spike sorting & data analysis
    - managing a database for storing all our lab data
- 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, give machines the ability to learn, or fly a helicopter on Mars - can't do that with Excel
- why Python?
  - like Matlab, Python is an "interpreted" language as opposed to a "compiled" language - no extra compilation step as in many other languages, developing code is easier and quicker
  - Python installation is relatively small, on disk and in memory, launches faster than Matlab
  - Python is free, and also open source: code behind Python and all of its libraries can be inspected, modified, improved by anyone - not so with Matlab
    - using Python can make science more open and reproducible:
      - ultimately, science needs open publication, open data, open hardware, and open code
      - LMU open science center: https://www.osc.uni-muenchen.de/
  - Python is used more outside of science and engineering than within it - huge user base
    - easy to find answers to problems, easy to find 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
    - https://stackoverflow.blog/2017/09/06/incredible-growth-python/
    - https://insights.stackoverflow.com/survey/2020#technology-programming-scripting-and-markup-languages

    - 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
    - helps make you employable
    - motivation:
        - example: load data, analyze, plot, save
        - `python LFP_analysis_example.py`

- class stuff
    - integrated lecture + in-class exercises
        - lecture: introduce concepts wth live coding, saved to a history.txt file uploaded afterwards
        - exercises: practice the concepts
    - please remember to keep your audio muted when not speaking
    - online vs. in-person? room E02.054 is currently assigned

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. dimension reduction & clustering
12. organizing code, data, results; version control with Git; 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 more or less successfully