# Python for Astronomy

ASTR 302

mjuric@astro.washington.edu

## Today

- Why Python for Astronomers?
- Administrivia
- Getting set up (Anaconda & Slack)

# About Me

Artist's impression: 👇

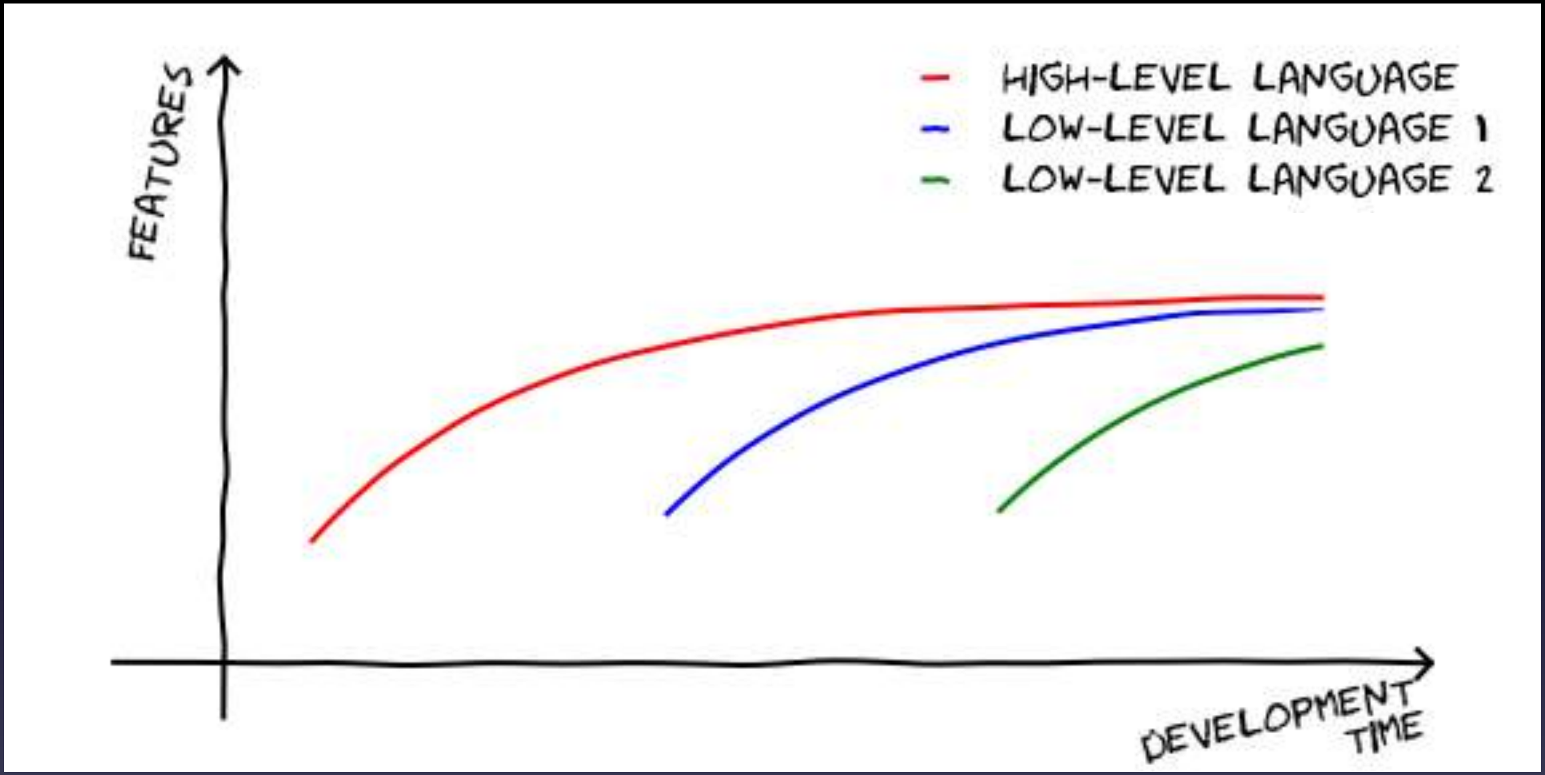
- Mario Juric (mar-ee-oh you-rich)
  - Astronomy Prof & DiRAC Institute Director
  - Office: C320, mjuric@astro.washington.edu
- What I do:
  - Rubin Observatory / LSST
  - Astronomical algorithms and software research
  - Science derived from large surveys: Galactic structure, properties of the solar system

Learns by trial and error: 👉

## Why Python for Astronomers?

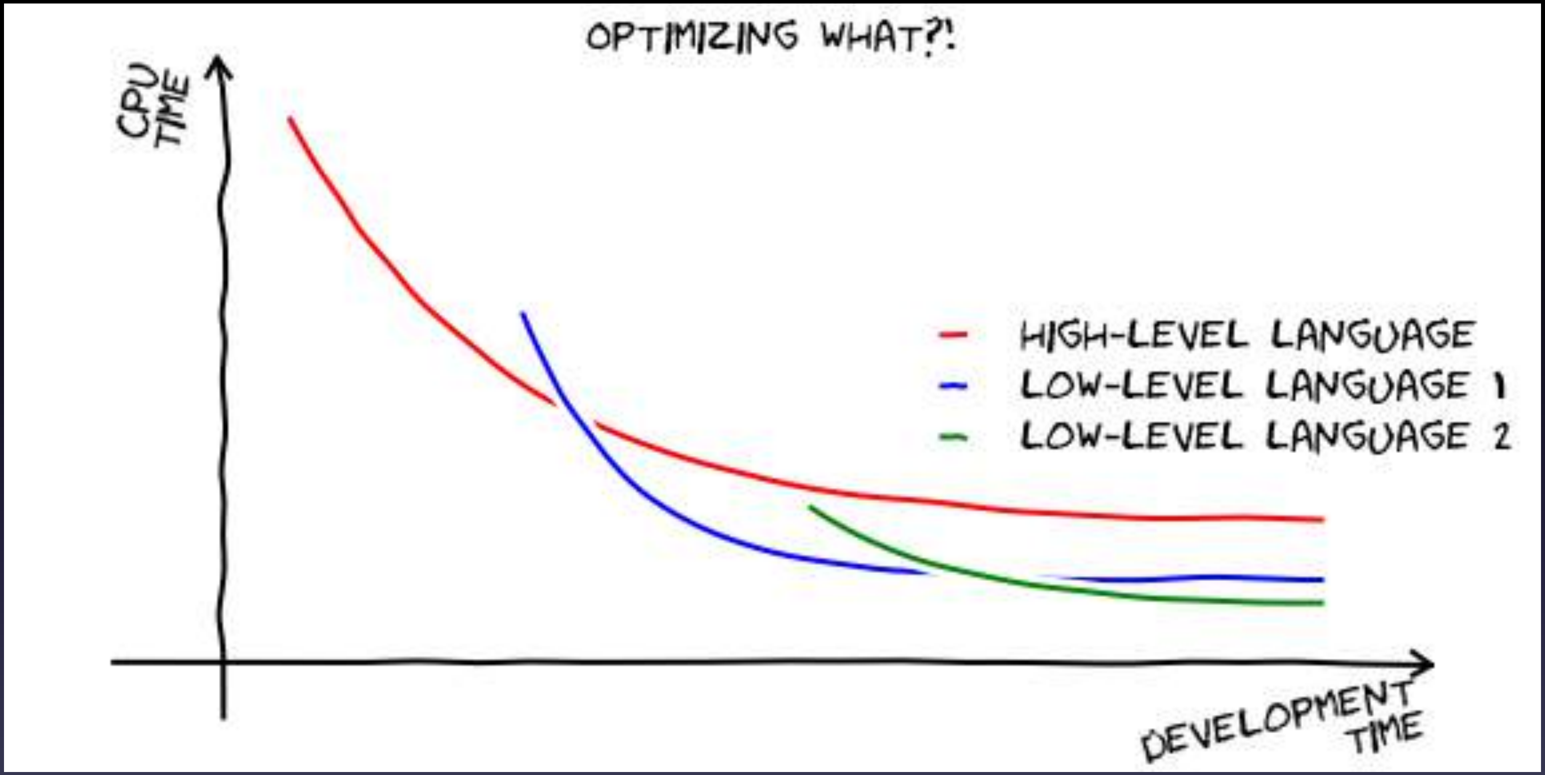
- ***Python has recently become the common language of astronomy (specifically, astronomical data analysis).***
- The goal of this course is to teach how to effectively use the tools available in the Python ecosystem (the language, the libraries, and *the concepts underpinning both*) for your research. Think of it as a part of a series: ASTR 300 -> 302 -> 324 -> 497 (Astro Surveys & Big Data).
- While motivated by astronomy, the skills learned here should be broadly useful. It should easily transfer to more general data-science work.

# Why Python?



Higher-level languages, in general, shorten your time to research results. Python is one such language that is easy to learn, freely available, similar to legacy languages (IDL, FORTRAN, C/C++), and was mature enough at just the right time to spur adoption.

# As usual, lunches don't come free



Computational efficiency is the price you pay for speed of development. But that is typically a good trade to make.

## How Many of you are comfortable with…

Writing a command-line Python script (.py)

Using Python lists()

Using Python dicts()

Using tuples

Writing Python functions

Writing Python classes

Writing a Python module

Working with numpy arrays

Making a mpl scatter plot

Plotting an image w. mpl

Exception handling

Writing a list comprehension

Writing a lambda function

Using decorators

Writing a decorator

Utilizing ABC classes

*"Hey, look, a survey!"*

## “The Only Thing That Is Constant Is Change”

*- typically misattributed to Heraclitus*

- In astronomical data analysis (and related computing disciplines), things change too quickly to be learned only once. Learning is a process that never ends.

- I will try to teach you what is currently the best (IMNSHO!) set of tools and techniques to have in your toolbox.

- But know this will change. So the major emphasis of this course will be on *understanding the concepts*, and learning how to continuously keep your knowledge up-to-date.

R

julia

Scala

*Languages may change,*
*but concepts transfer!*

## Learning how to Learn

- We'll also learn ***how*** to learn: how to discover information, keep your knowledge up-to-date, find leads that help you solve problems, evaluate their worthiness.

- This may be the most valuable piece to take away from this class.

# Lectures

- When: MW, 2:30pm-3:50pm
- Were & how:
  - Zoom

- Lecture structure (will change from week to week):
  - About an ~hour for introducing new material. **!!! WILL BE HANDS-ON !!!**
  - Some weeks may be fully flipped (w. pre-recorded lectures/material to read before class).
  - Leave ~20 minutes to work on homeworks, questions, open-ended discussion
  - Interrupt and ask questions at any time!

## Course Materials

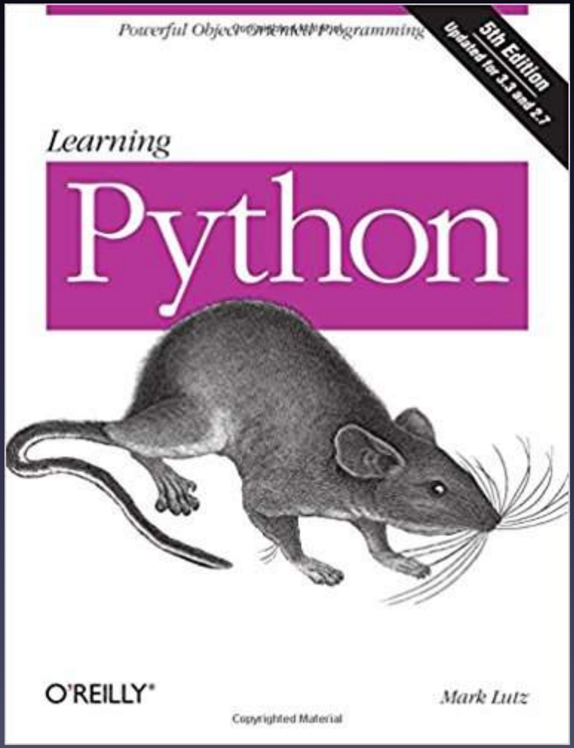
- I'll be adding most of what we need to the following repository on GitHub:

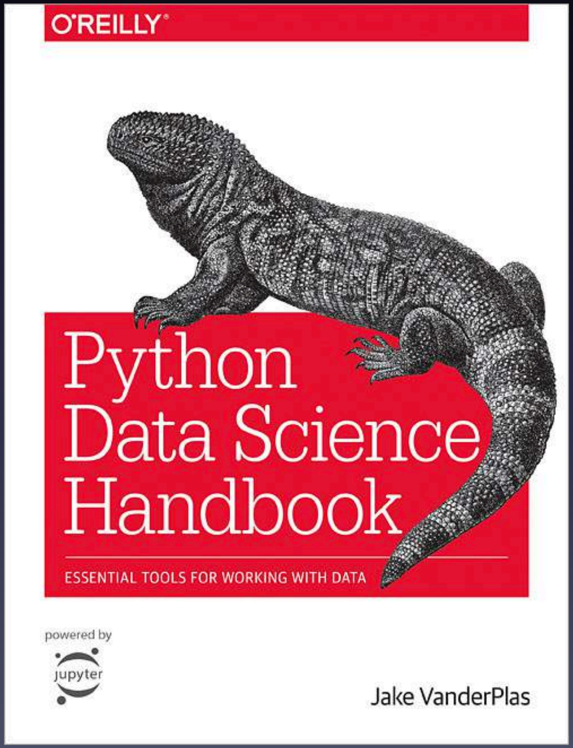
`https://dirac.us/git302`

*Action: Let's make sure everyone can access this repository.*

# Textbooks & Reference Material

- There is a wealth of material available online and as eBooks (for UW students, via UW Libraries & Seattle Public Library). The two I'd recommend to begin with:




O'REILLY
Python Data Science Handbook
ESSENTIAL TOOLS FOR WORKING WITH DATA
powered by
jupyter
Jake VanderPlas

+ many (many) online writeups / blog posts that I will point you to over the next few weeks.

## Syllabus

- Available at: https://dirac.us/1tq
  - (this will take you to the class github repository).

- Things to note:
  - The syllabus *will* change as we go through the quarter. Your feedback will be invaluable!

# Grading, Homeworks, Etc.

**Homeworks (60% of the grade):**

– Designed to exercise what we've learned recently.

**Final project (40% of the grade):**

– You will propose a piece of software (or analysis) to build using the techniques and libraries we'll learn about in the course. Keep an eye out for ideas!

*Advice: Turn in your homeworks on time!*

# Communication: Slack

- In this class, we won't be using a mailing list but an instant messaging (-like) tool called Slack (http://slack.com). Slack is heavily used today by many research & technology companies and projects.
- Request to join our department Slack at:
  - `https://uw-astronomy.slack.com`
  - `Then join the #astr-302 channel`
- What to use it for:
  - Asking questions, discussing the class, exchanging snippets of code, collaborating on projects (when appropriate).
  - Please prefer Slack to sending me e-mails. Two reasons:
    - Everyone can benefit from the question and answer.
    - Your colleagues may be able to help!

*Action: Let's make sure everyone's on Slack.*

## Our working environment: class JupyterHub

[https://dirac.us/hub302](https://dirac.us/hub302)

- A UW-managed Jupyter
- Allows you to run Jupyter on a remote computer w/o having to have anything installed locally (like Google Docs, but for Jupyter)
- Everything "Just works" ™

*Action: Let's try it out*

## Getting Python: Miniconda

- Locally, you can install the Miniconda Python Distribution.

*https://conda.io/miniconda.html*

Note: we don't need it for this class, but it'll be useful for you to learn how to set up a Python environment from scratch.

a) it's easy, and b) you won't always have sysadmins available to do it for you.

ANACONDA®

## Next time

- A Walk Through Python!

*Note: Please refresh your ASTR 300 memory by working through the notebooks at*

[https://github.com/UWashington-Astro300](https://github.com/UWashington-Astro300)

*(the ASTR 300 github repository)*