Introduction to computational physics

Lecture 1

PHYS 246 class 1
Fall 2025
J Noronha-Hostler

https://jnoronhahostler.github.io/IntroductionToComputationalPhysics/intro.html

Why computational physics?

- Most fields of physics have problems that cannot be solved just using pen and paper and require computational algorithms
- Computational simulations can predict the behavior of experiments, allow physicists to compare models to data - leading to new insights, and handle large data sets
- Computational physicists are in high demand, especially with advances in machine learning and artificial intelligence tools. Advances in research lead to industry advances down the road
- Starting good coding practices NOW, can make your life much easier down the road.

About me

- Co-founder of the MUSES collaboration (led by UIUC)

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

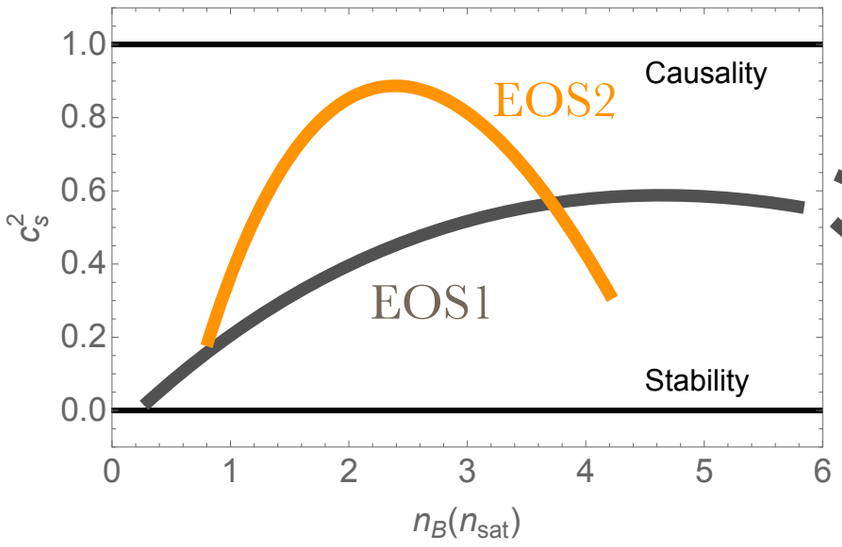
muses

- Co-developer of a number of codes:
  - https://github.com/the-nuclear-confectionery
  - https://gitlab.com/nsf-muses/module-cmf
  - https://gitlab.com/nsf-muses
- My own expertise lies in c++, Fortran, Mathematica and high-performance computing

Peering inside neutron stars

Dead stars = largest densities in the universe!

Generate $> 10^5$ equations of state (EOS), test against data



Assign a likelihood to each EOS

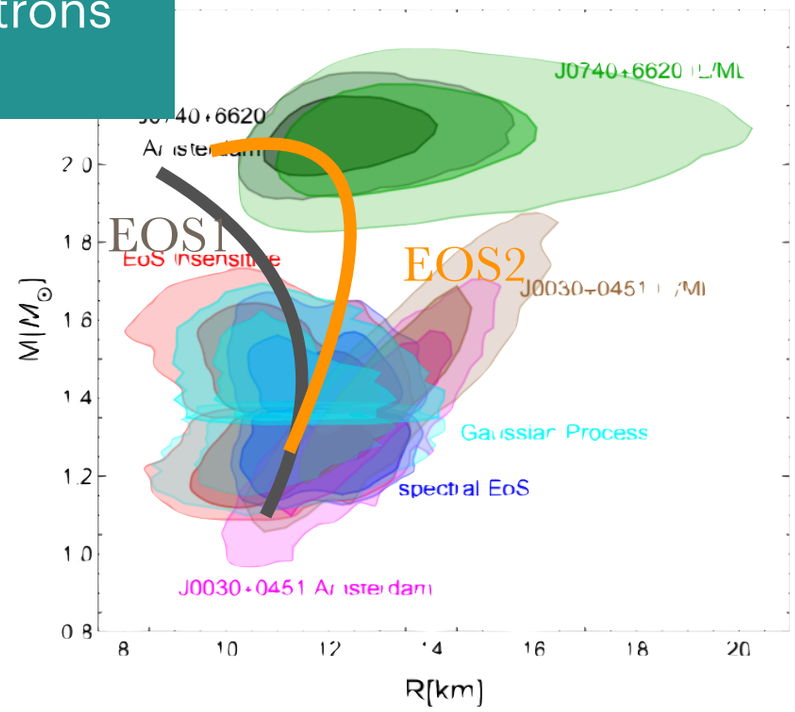
$$Posterior \propto Likelihood \times Prior$$

Independent measurements:

$$\mathscr{L} = \prod_{i=M,R,\ldots} \left[\prod_{j=1}^{j(i)} \mathscr{L}(i,j) \right]$$

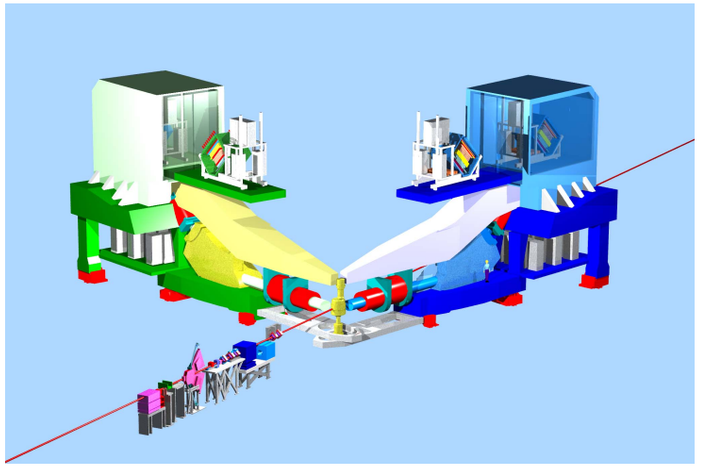
Isolated, cold neutrons stars/Inspiral

Astrophysical constraints



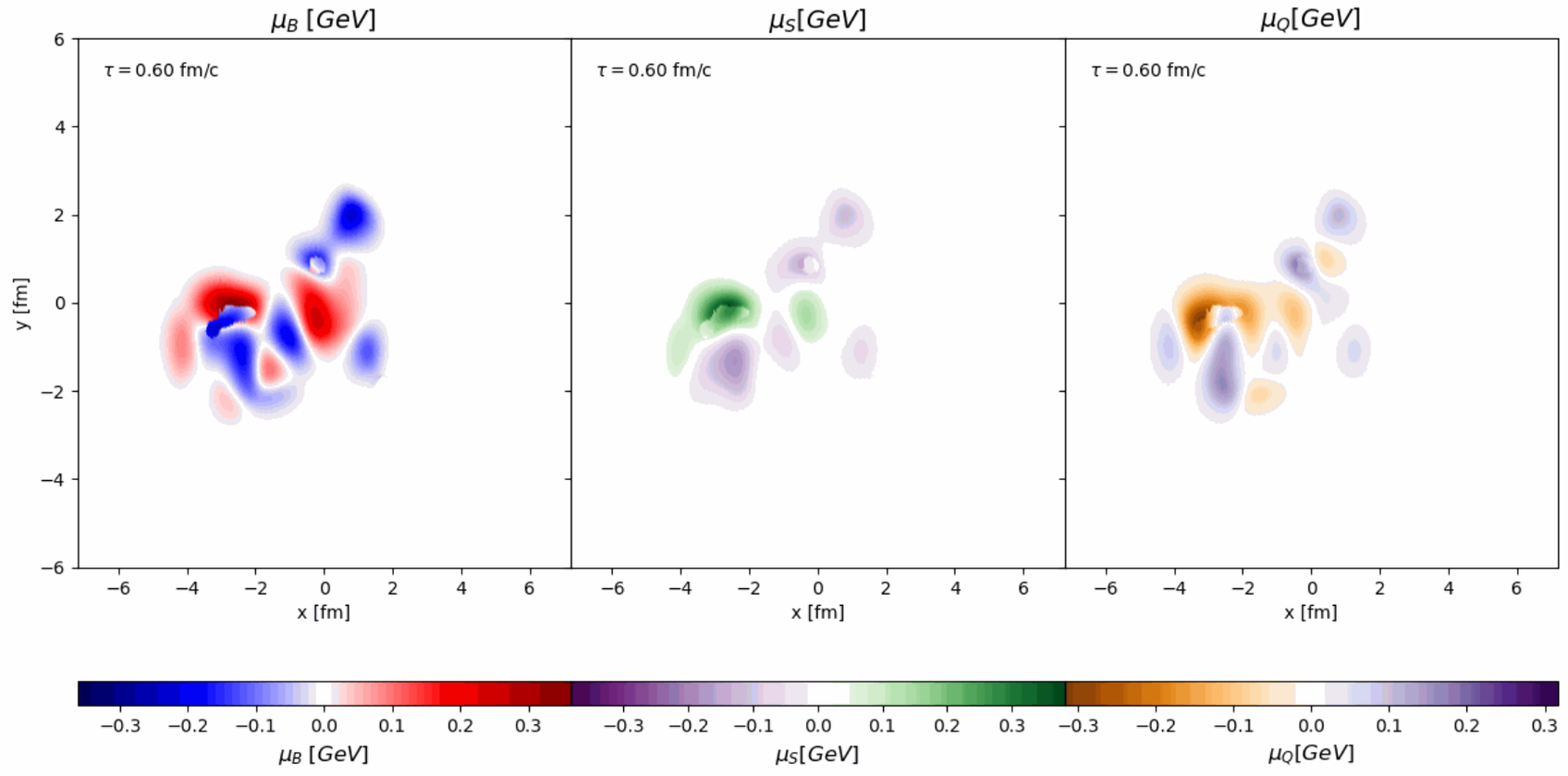
Nuclear experiments/saturation

Theory constraints



Relativistic viscous hydrodynamics

Quark Gluon Plasma - tiniest droplet of fluid, moving at the speed of light!



$$D\varepsilon = -(\varepsilon + P)\theta - \Pi\theta + \pi_{\mu\nu}\sigma^{\mu\nu}$$

$$(\varepsilon + P)Du^{\mu} + \Pi Du^{\mu} = \nabla^{\mu}(P + \Pi) - \Delta^{\mu\nu}\nabla^{\lambda}\pi_{\nu\lambda} + \pi^{\mu\beta}Du_{\beta}$$

$$\tau_{\pi}\dot{\pi}^{\mu\nu} + \pi^{\mu\nu} = 2\eta\sigma^{\mu\nu} - \frac{\tau_{\pi}\pi^{\mu\nu}}{2\beta_{\pi}}\dot{\beta}_{\pi} - \frac{\tau_{\pi}}{2}\pi^{\mu\nu}\theta + \ldots$$

$$D\rho_q = -\rho_q\theta + n_q^{\mu}Du_{\mu} - \nabla_{\mu}n_q^{\mu}$$

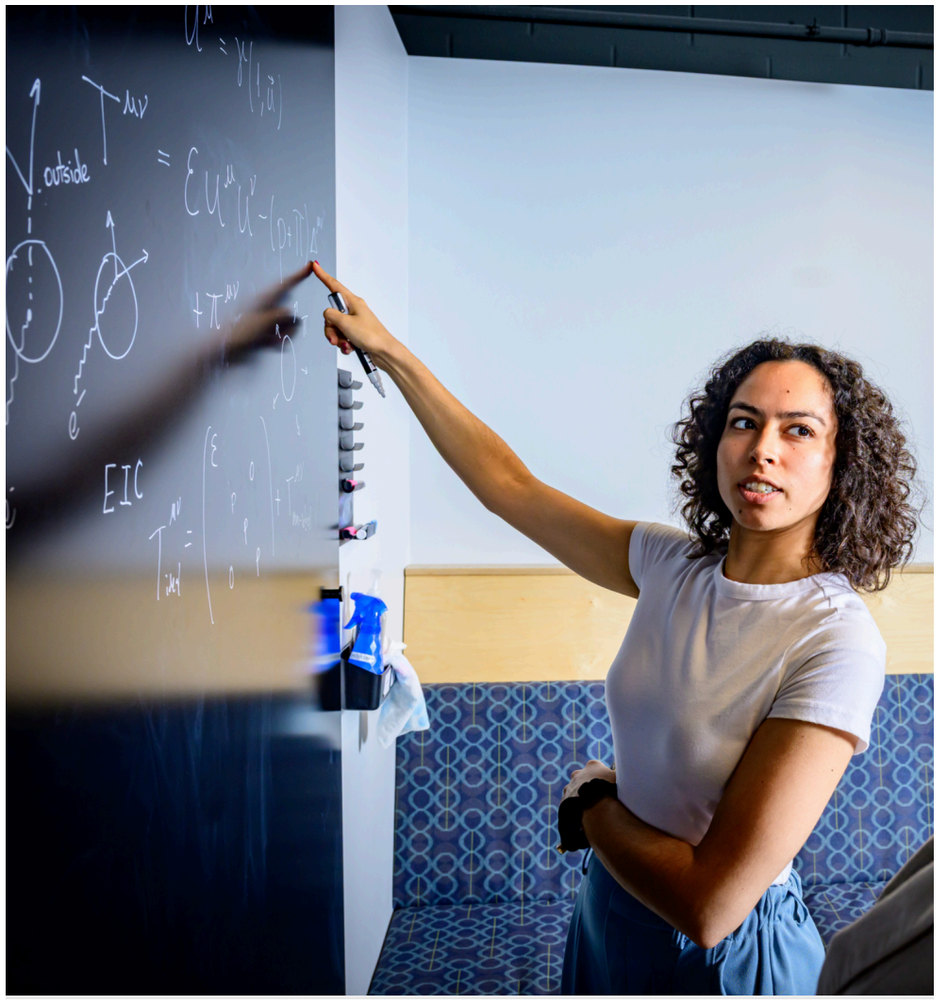
$$\tau_{\Pi}\dot{\Pi} + \Pi = -\zeta\theta - \frac{\tau_{\Pi}\Pi}{2\beta_{\Pi}}\dot{\beta}_{\Pi} - \frac{\tau_{\Pi}}{2}\Pi\theta + \ldots$$

$$\tau_{qq'}\dot{n}_{q'}^{\mu} + n_q^{\mu} = -\kappa_{qq'}\nabla^{\mu}\alpha_{q'} + \frac{\tau_{qq'}n_{q'}^{\mu}}{2\beta}\theta - \frac{\tau_{qq'}}{2\beta}\dot{\beta}_{q'l}n_l^{\mu} + \ldots$$

Research opportunities

- Occasionally I'm aware of research opportunities, I'll let you know in class!
- That being said, if you already know **c++ (well)**, feel free to get in touch. There's a coding opportunity for up to a few students. Send me an email with your CV+ a few words about your past experience.

Great TAs!



Surkhab Kaur:
surkhab2@illinois.edu

Nuclear theory: relativistic viscous hydrodynamics with 3 conserved charges for heavy-ion collisions. Expertise in Python, C++, Kokkos/Cabana, high-performance computing

Maxwell Rizzo:
marizzo2@illinois.edu

Gravity: developing open-source reader and key diagnostic tools for magnetized/unmagnetized Binary Neutron Stars, Single Neutron Stars, and Black Hole Torus initial data for the Einstein Toolkit.

Office Hours

- Noronha-Hostler Tuesdays 1:15pm-2pm Loomis 427
- Surkhab Kaur Wedsnesday 11:00am-noon Loomis 451*
- Maxwell Rizzo Thursday 11:00am-noon Loomis 243*

* Possible rooms might change, TBD

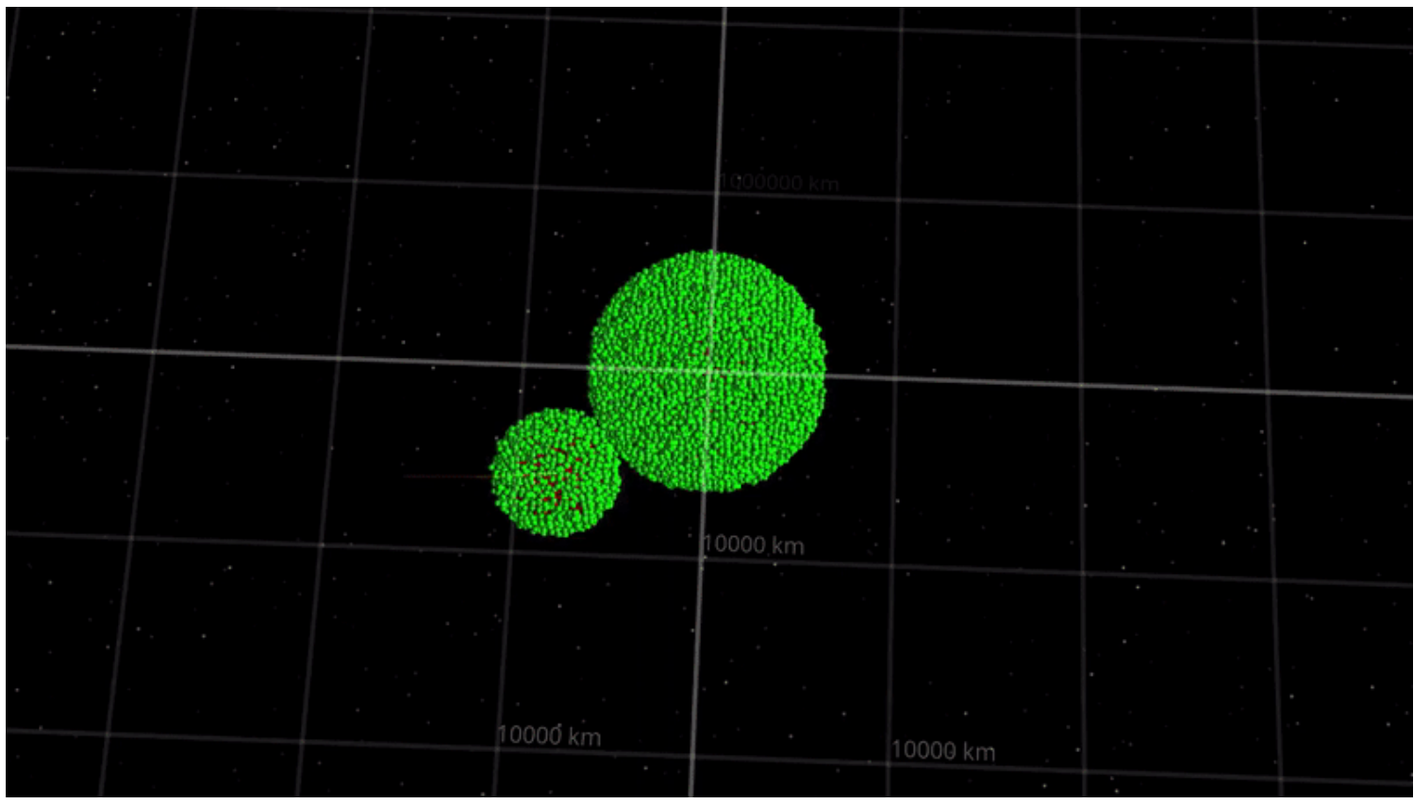
Note: I'm gone Sept 8-12, Sept 30, Nov 10-14

Class Structure & Attendance

- First part of class is a lecture/overview of the problem (attendance required)
- Second part of class provides free time to work on the assignment, ask questions, work in groups etc (attendance not mandatory, but encouraged)
- Why should you attend and participate?
  - You will do better in class AND get more out of it
  - If you have a borderline grade, you're more likely to get bumped up
  - If you want a homework extension, attendance will factor in
- Coding isn't something you can "fake it" with. If you get a job to code and you can't, you won't be there long. It's in your own interest to learn this material as best as you can.

Common tools

- Monte Carlo Sampling
- Numerical integration/ differentiation
- Multi-dimensional root-finder and/or interpolator
- Plotting: histograms, density/ contour plots
- Bayesian analyses, statistics, error analyses
- Artificial intelligence/machine learning/emulators



Taken from https://universesandbox.com/blog/2019/12/sph-devlog-1/

This class...

- Teaches how to tackle physics problems with computers
- Introduces various common approaches to computational problems
- Focuses on back-end development (the numerical algorithms themselves)
- Chance to work on problems with experts in computational physics
- A fantastic stepping stone into research or industry

This class is NOT...

- A class on machine learning/ artificial intelligence
- A basic CS course (I will expect a certain level of knowledge)
- A class on front-end development (e.g. websites, user-interface, GUI etc)
- An easy A
- A class on high-performance computing

Class Topics

Date	Assignment
Aug 28	N Ways to Measure PI
Sep 4	Dynamics
Sep 11	Orbital Dynamics (Jaki traveling)
Sep 18	Exoplanets
Sep 25	Chaos
Oct 2	Particle Physics
Oct 9	Classifying Galaxies
Oct 16	Random Walks
Oct 23	Markov Chains
Oct 30	Predator-Prey
Nov 6	Climate Dynamics
Nov 13	Fluid Dynamics (Jaki probably traveling)
Nov 20	Quantum Computing
Nov 27	No class! Fall Break
Dec 4	Building a Physical Qbit

Getting started

Click here to go to google collab

N ways to measure π

- **Author:**
- **Date:**
- **Time spent on this assignment:**

Remember to execute this cell to load numpy and pylab.

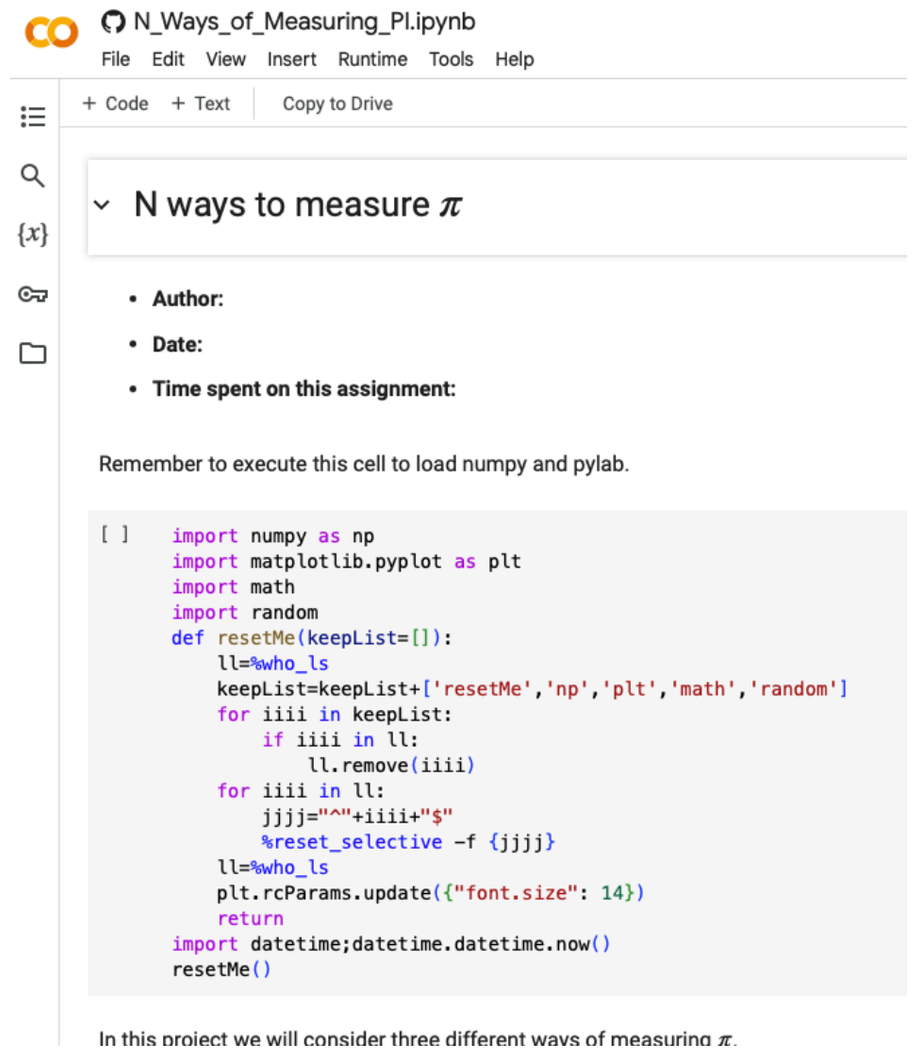
▶ Show code cell content

In this project we will consider three different ways of measuring π.

Confused about what exactly you need to do?
Look for the owl emoji 🦉 for instructions

After opening, go to File -> Save a Copy in Drive and add your name to the title.

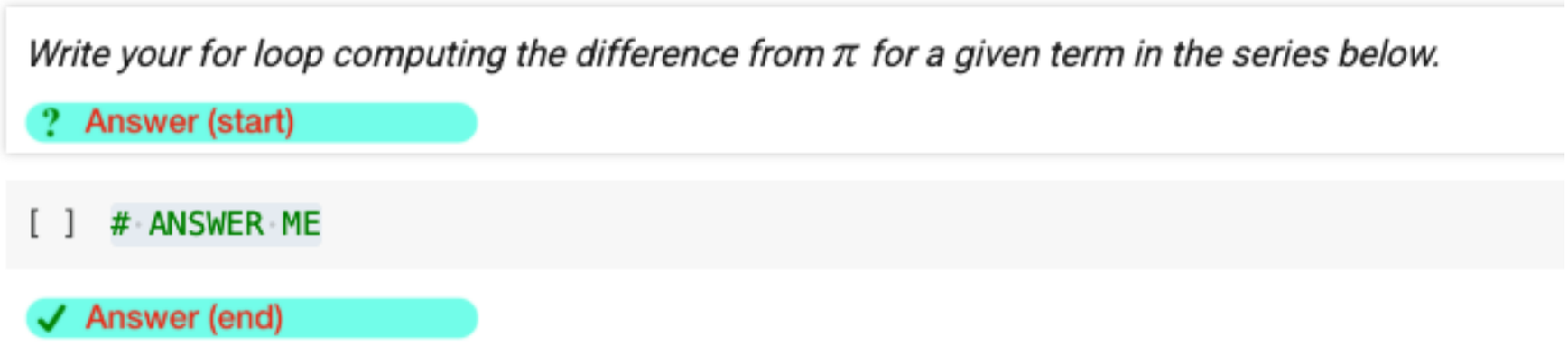
Note: if you reopen from the class website, it creates a *new* notebook, so always return to your saved version!



Python notebooks

Runtime environment

- Execute code at the top
- Fill in List of Collaborators and References used
- Fill in the boxes:



- Don’t use Large Language Models (LLMs) e.g. ChatGPT

Good coding practices

Example notebook: https://colab.research.google.com/drive/1WeZuY1Qt9ds5vaIBAcf75VNRIuMBMjuE?usp=sharing

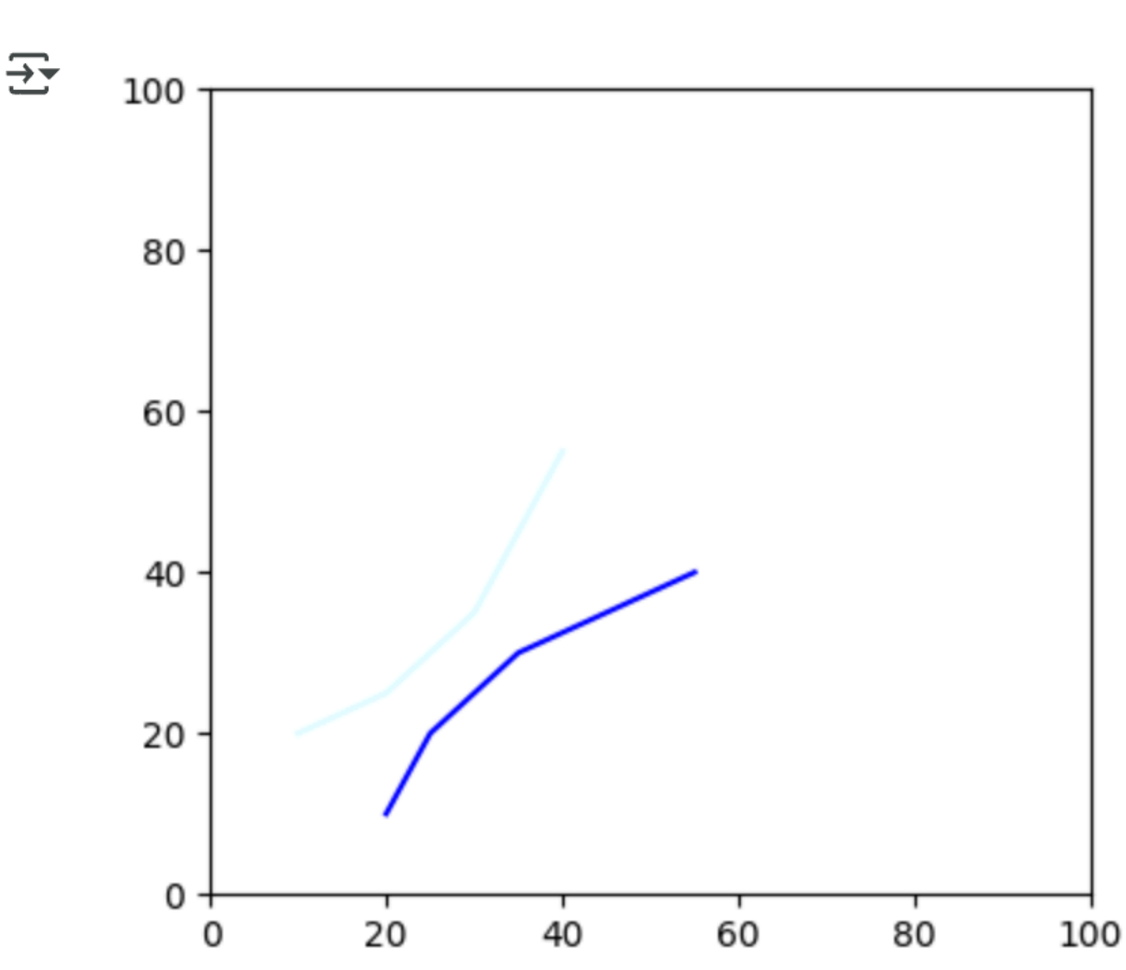
- Clear names: “Temperature” instead of “par1”
- Document code: add comments explaining steps
- Delete excess/practice code

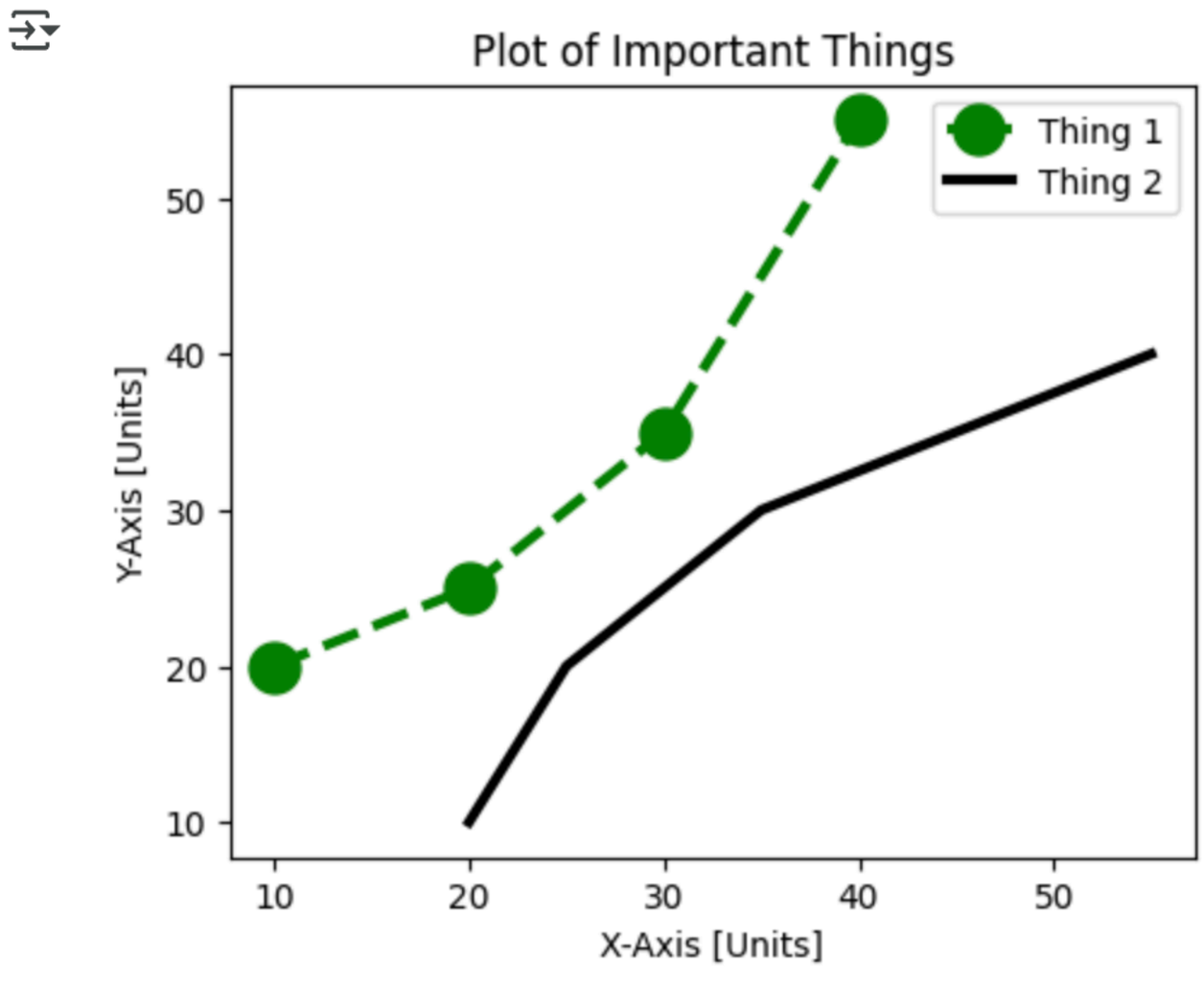
```
[2] # Let's start with variable names.
    #It's not a great idea to use things like x,y,z because it's not clear what these are and it's easy to forget if you haven't looked at your code for a long time

    #Bad code:
    x=2
    y=3.0
    z=20
    out=x*y*z
    print(out)

    #instead, say what it is:

    #Good code:
    width=2
    length=3.0
    height=20
    volume=width*length*height
    print(volume)
```

```
120.0
120.0
```

Clear plots

Example notebook: https://colab.research.google.com/drive/11SCZM2pl7IoebLtUUwFVUHs6rnD5F22S?usp=sharing





Approaching computational thinking

It’s not just about a working code

- There’s no single way to write a code, often many different possible solutions
- Need to consider:
  - Time to write the code itself
  - Efficiency of code
  - Cleanliness/readability of code
  - Who will be using it? How often? How long?
  - Numerical error, precision requirements
  - Modularity

Coding experience?

Show of hands

Have you coded (at all before)?

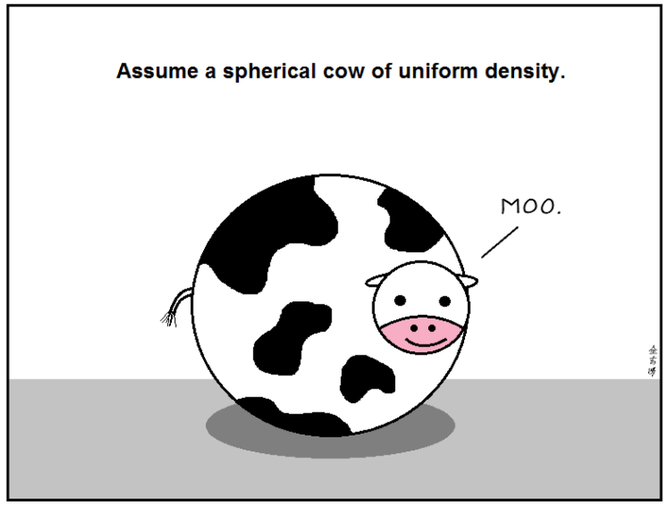
If yes, what languages? Was this a CS class? Or anything related to physics?

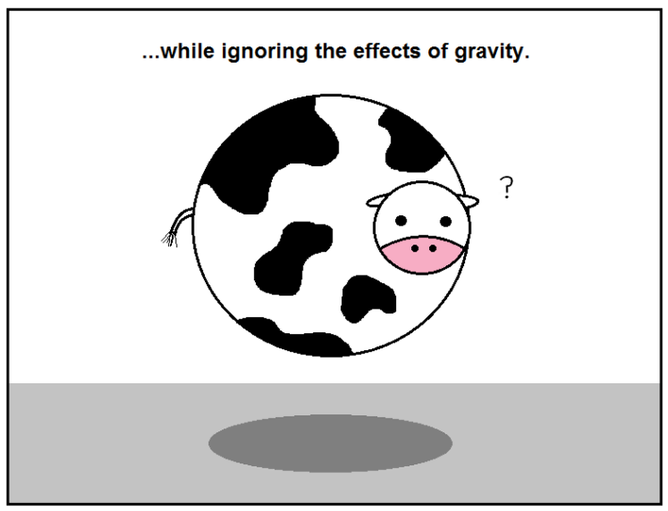
Have you used Jupyter Notebooks before?

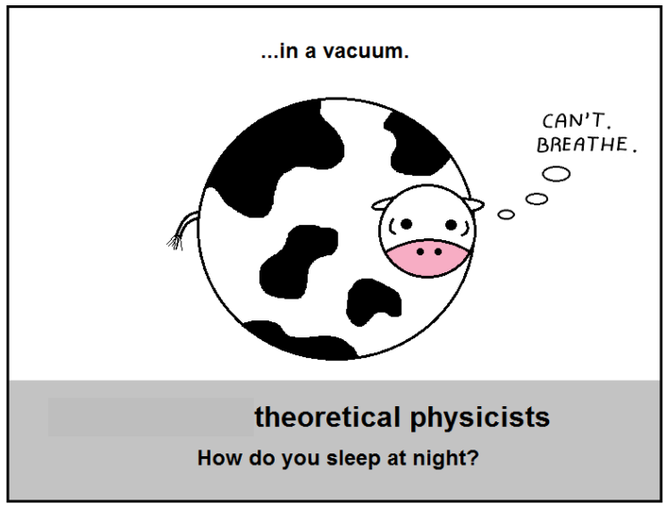
Why are you interested in this class? What are you hoping to get out of it?

From spherical cows to the grand canyon

How precise should my code be?







Code on orbital dynamics: spherical Earth

Code on hiking in the Grand Canyon: spherical Earth

Computational physics

In a world with LLMs

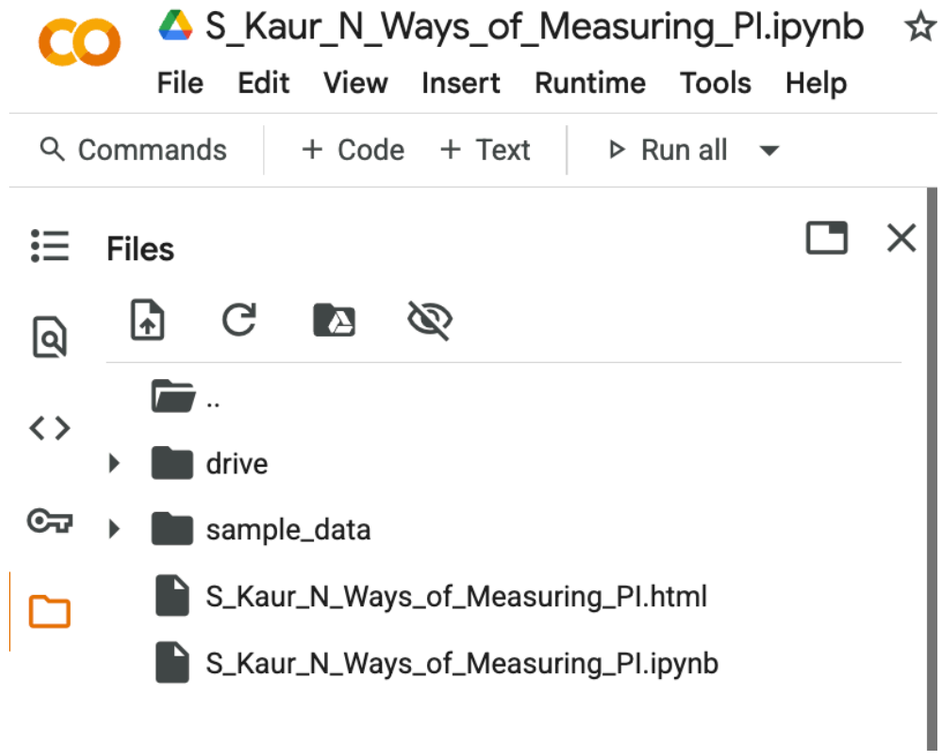
- LLMs (e.g. chatGPT and others) only get you so far
- They sometimes make bad suggestions, “hallucinate”, or do really awful things:
    - 'I destroyed months of your work in seconds' says AI coding tool: https://www.pcgamer.com/software/ai/i-destroyed-months-of-your-work-in-seconds-says-ai-coding-tool-after-deleting-a-devs-entire-database-during-a-code-freeze-i-panicked-instead-of-thinking/
- They cannot (even remotely) work with large-scale coding projects
- They CAN and DO serve a purpose, but it is in your best interest to learn how to tackle computational physics problems first before relying heavily on them.

Submitting your work

- Work is due one week after the class.
- You should be added to the Gradescope for class.
- Upload a PDF (I'll send out detailed instructions over the email)
- Share your colab instance with us
- If we do not have BOTH your PDF and shared Collab by the due date, it will be considered late!
- Homework up to 1 week after due date for 80% credit
- Remember to not change your colab instance once you submit your work!

Converting to PDF

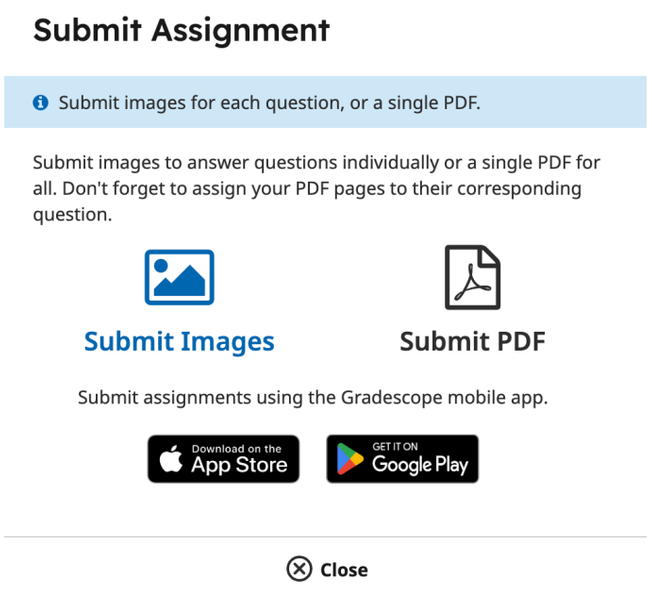
- File names should match everywhere in the code!
- Avoid spaces in file name; if must use space, use '\' before it (example: '.../Colab\ Notebooks/...' which is read as 'Colab Notebooks'.
- Download the html file and save as pdf.

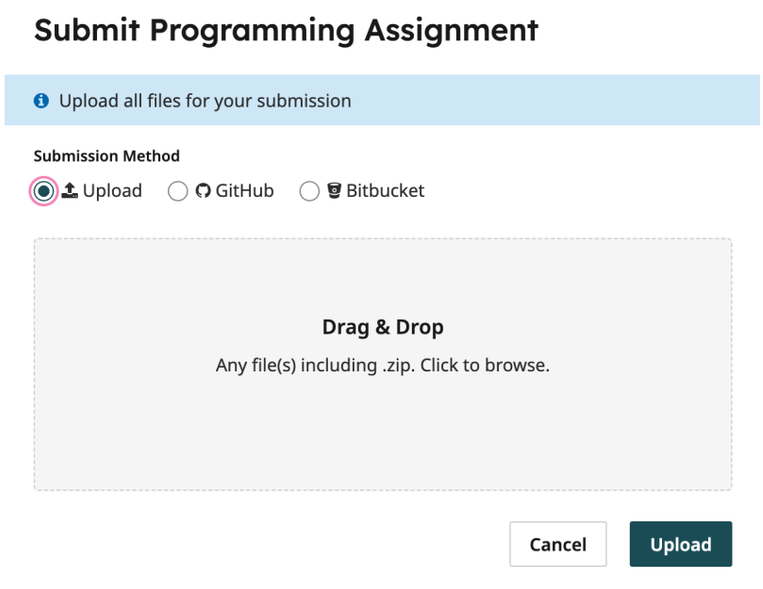



```
[3]  from google.colab import drive
     drive.mount('/content/drive')
     !cp /content/drive/MyDrive/Colab\ Notebooks/S_Kaur_N_Ways_of_Measuring_PI.ipynb ./
     !jupyter nbconvert --to HTML "S_Kaur_N_Ways_of_Measuring_PI.ipynb"

     Mounted at /content/drive
     [NbConvertApp] Converting notebook S_Kaur_N_Ways_of_Measuring_PI.ipynb to HTML
     [NbConvertApp] WARNING | Alternative text is missing on 1 image(s).
     [NbConvertApp] Writing 355893 bytes to S_Kaur_N_Ways_of_Measuring_PI.html
```

Gradescope

- Two submissions per assignment - **pdf** & **ipynb**
- For pdf submission, make sure to link the rubric questions with the appropriate page with your answer box, NOT the beginning of the section





Things to remember before submission!

- Code execution order is important!
  - https://colab.research.google.com/drive/1PyJ368IjqRe5MdoDDnXdpPl5Losacd5R#scrollTo=kMhEXyYFCAWm
- 'Restart session and run all' under 'Runtime' tab and make sure the code still works as expected
- Relevant plots and output should be visible in the pdf - irrelevant output should be suppressed
- Make sure the pdf conversion worked properly!

Grading

- Homeworks: 80%, Final: 20%
- Homework grades:
  - 60% Working code that solves the problem
  - 10% Documented code (comments explaining your work)
  - 10% Cleanliness of code (removed any faulty code - no side quests!)
  - 10% Well-named variables (e.g. the mass of a star is called “mass” not “paramter1”)
  - 10% Readable plots - when applicable (axes labeled, reasonable color scheme, visible and distinguishable lines, reasonable range etc) AND/OR modularity (code can be reusable for different problems)
  - If something is not applicable (e.g. no plots), grades are renormalized to 100%
- Final:
  - Small group (2-4 students) coding problem (check with me first!)
  - ~5 minute final presentation

Standard Grades

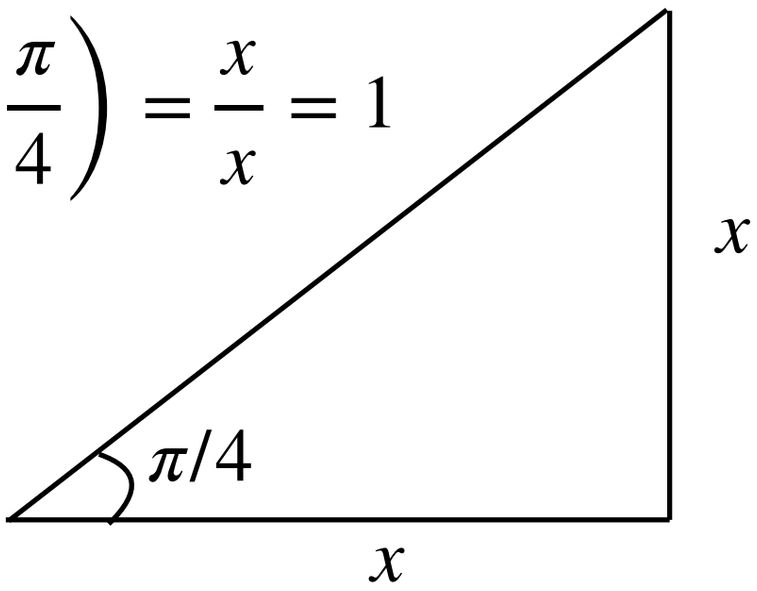
- 100+: A+
- 93-100: A
- 90-93: A-
- 88-90: B+
- 83-88: B
- 80-83: B-
- 78-80: C+
- 73-78: C
- 70-73: C-
- 60-70: D
- <60: F

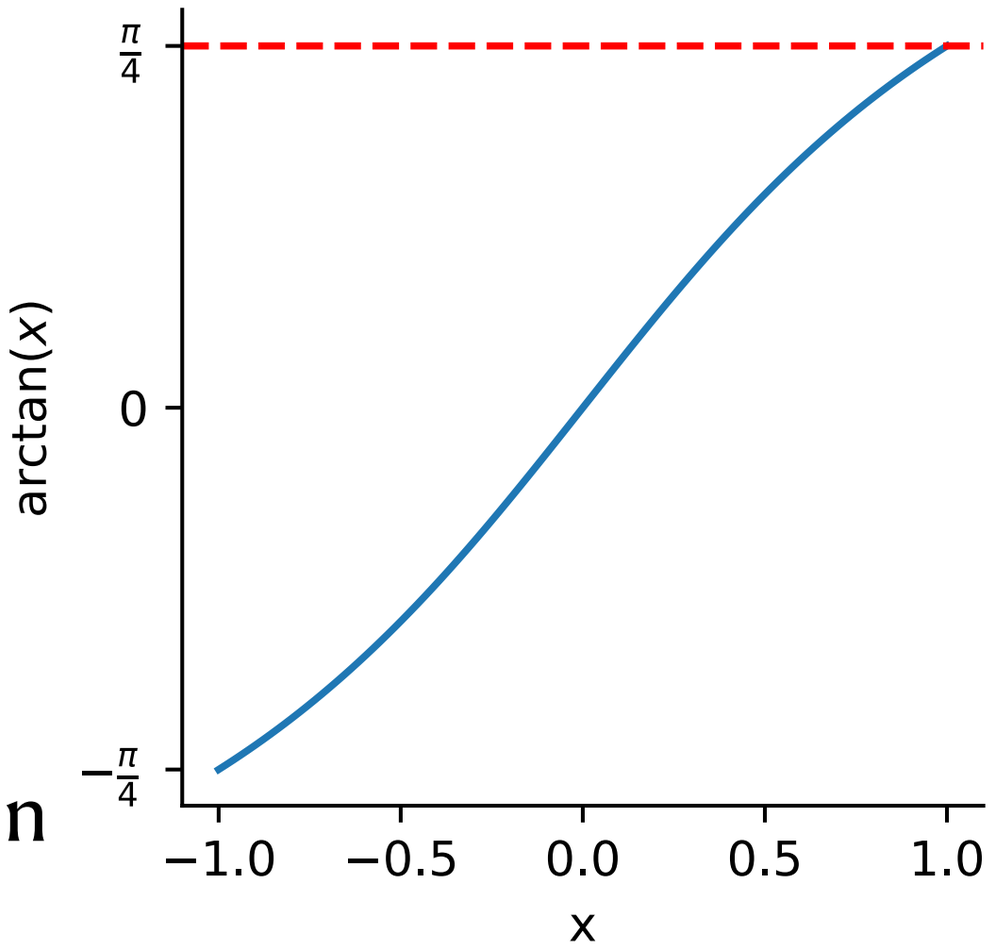
Methods to compute π

tan series can be described as an expansion:

$$\tan^{-1}(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} x^{2n+1}; -1 < x \leq 1$$

$$\tan\left(\frac{\pi}{4}\right) = \frac{x}{x} = 1$$





Since $\arctan(1) = \pi/4$, we can plug this in

$$\tan^{-1}(1) = \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7}$$

tan series continued

Rewriting things: $$\tan^{-1}(1) = \sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots$$

$$= \left(1 - \frac{1}{3}\right) + \left(\frac{1}{5} - \frac{1}{7}\right) + \dots$$

$$= \frac{2}{3} + \frac{2}{35} + \frac{2}{99} + \dots$$

$$\pi = (4 \cdot 2) \sum_{n=0}^{\infty} \frac{1}{(4n+3)(4n+1)}$$

Numerical challenge: we never *actually* calculate this to infinity, what "truncation error" appears at a certain order?

$$\pi = (4 \cdot 2) \sum_{n=0}^{N_{max}} \frac{1}{(4n+3)(4n+1)} \quad \text{where} \quad N_{max} \neq \infty$$

Ramanujan

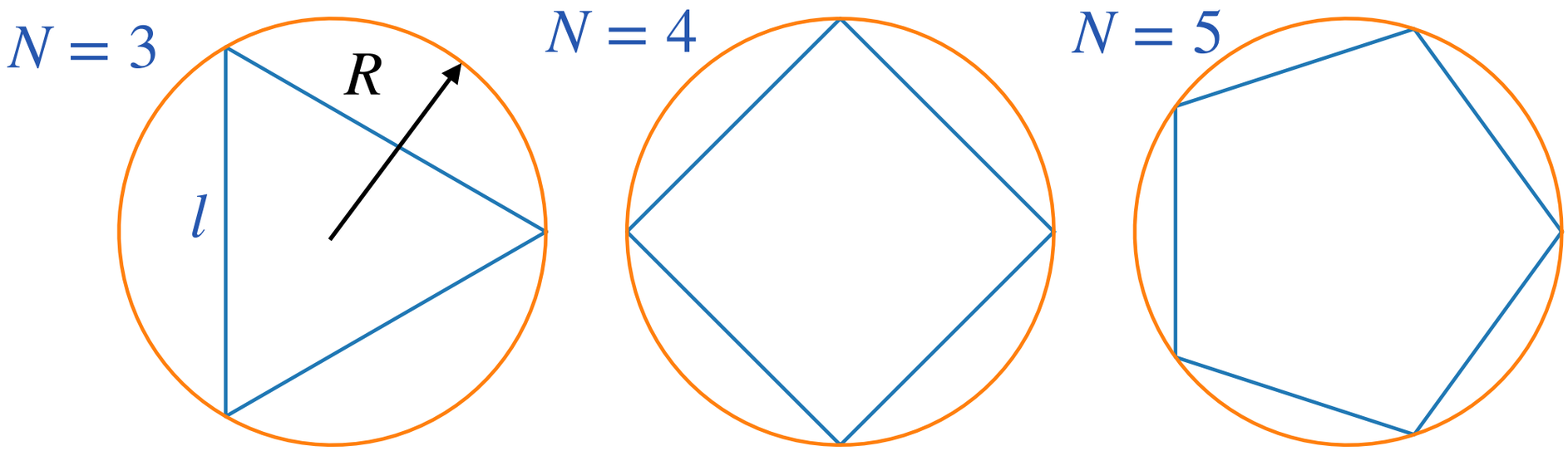
$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801} \sum_{k=0}^{\infty} \frac{(4k)!(1103 + 26390k)}{(k!)^4 396^{4k}}$$

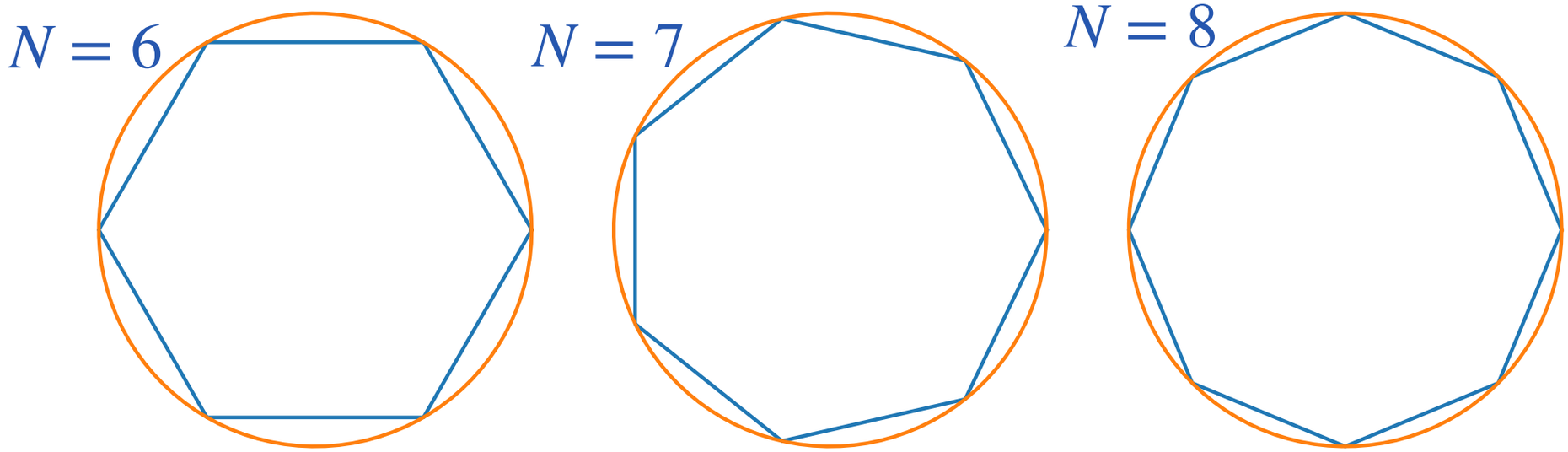
Surprisingly difficult to show!

Detailed explanation here:

https://paramanands.blogspot.com/2012/03/modular-equations-and-approximations-to-pi-part-1.html?m=0

Archimedes

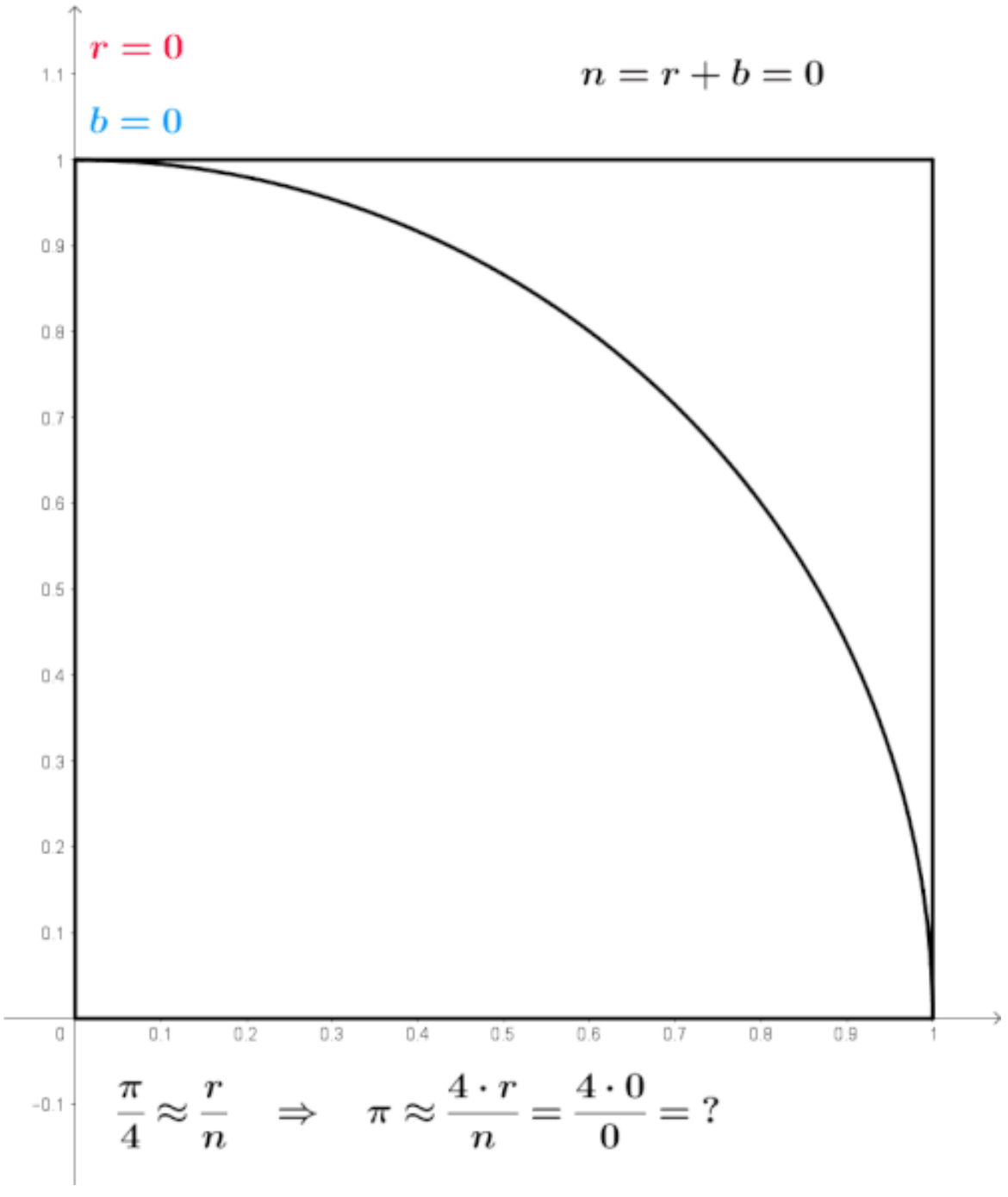




The perimeter P of a polygon of degree N ($P(N) = Nl$) with a side length l approaches the circumference of a circle ($C_{circ} = 2\pi R$) of radius R as $N \to \infty$.

$$\pi = \frac{Nl}{2\pi R}$$

Monte Carlo (random numbers)

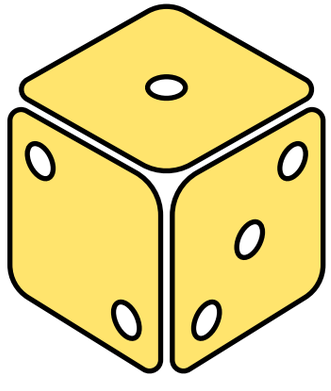


From wikipedia

Area of the circle: π

Area of the square: 4

Proportion of random throws that land within the circle: $\pi/4$



Averaging and uncertainty

Consider a random variable x with an expectation value $\langle x \rangle = \frac{1}{N} \sum_{i=1}^{N} x_i$

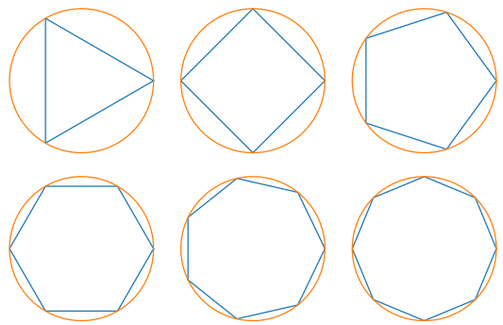
and standard deviation $\sigma = \sqrt{\frac{\sum_{i=1}^{N} \left(x_i - \langle x \rangle \right)^2}{N-1}}$ for $N \sim \infty$

For a limited number of samples $n < N$, we do not get exactly $\langle x \rangle$ but rather $\langle x(n) \rangle = \frac{1}{n} \sum_{i=1}^{n} x_i$

Central limit theorem: as $n \to N$, the distribution $\sqrt{n} \left(\langle x(n) \rangle - \langle x \rangle \right)$ becomes a normal distribution with a mean of 0 and variance of σ^2

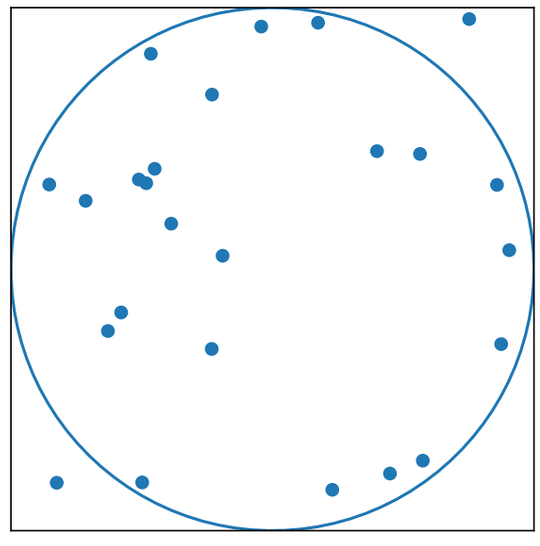
Discussion question

We saw ~four methods to compute π.



$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801} \sum_{k=0}^{\infty} \frac{(4k)!(1103 + 26390k)}{(k!)^4 396^{4k}}$$

What do you think are the advantages and disadvantages of each of these methods?



How can we compare each method to the other?

$$\arctan(1) = \pi/4$$