Introduction to Python for Astronomy

Unlocking the Universe with Code

Dr. John Hoang July 12, 2025

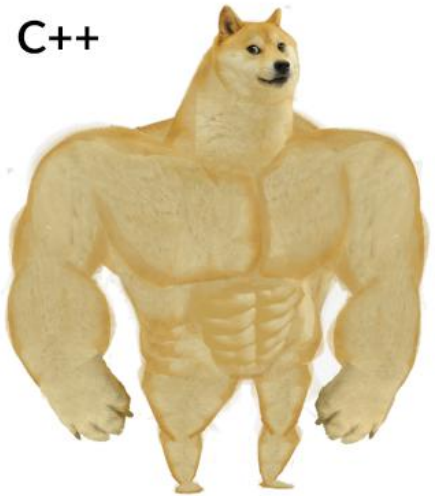
python™

Agenda

- Mini presentation
  - Introduction to Python
  - Why Python is Ideal for Astronomy
  - Essential Python Libraries for Astronomers
- Code Demos
  - Setting up and simple examples
- Coding time with real-life examples

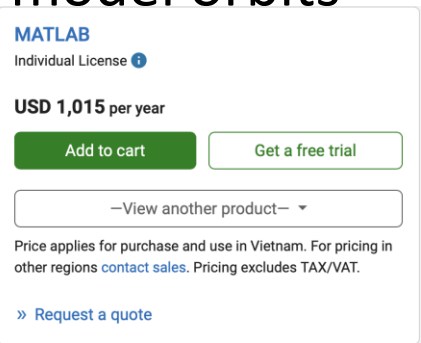
What is Python?

- High-level, interpreted language (1991, Amsterdam)
- Easy to learn, easy to use
- Huge community supports and vast libraries
- Versatile: web, data science, scientific computing, AI



Why Python in Astronomy?

- It’s free!
- Strong adoption and community support in astrophysics
  - Surveys show >90 % of astronomers use Python for data analysis
- Support for the majority of astro data formats
  - Handles massive datasets from modern telescopes & simulations
- Automation: schedule observations, analyse spectra, model orbits



Essential Python Libraries for Astronomy

Library	Purpose	Astronomy Example
NumPy	Numerical arrays & math	Handling pixel data
SciPy	Optimisation, integration, stats	Fitting spectral lines
Matplotlib	Plotting & visualisation	Graphing light curves
Pandas	Tables & time-series	Galaxy catalog analysis
Astropy	Astronomy-specific core	Coordinate transforms
TensorFlow/Pytorch	AI/Machine Learning	Object Classification

On the use of ChatGTP/Grok/Gemini

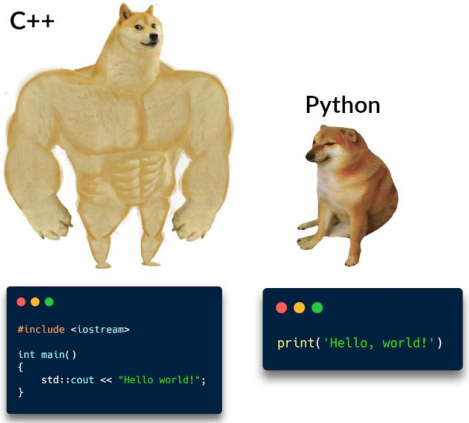
help me create a powerpoint file for a lecture on the use python in astronomy

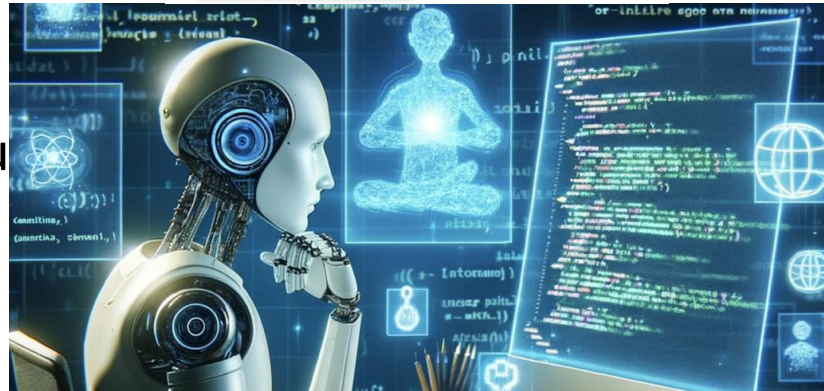
Thought for 27 seconds

- The user needs a PowerPoint file for a lecture on Python in astronomy. I'll create a .pptx with slides like Title, Python overview, data analysis libraries, Astropy, Astroquery, photutils, machine learning, examples, case studies (light curves, spectra, imaging), code snippets, and resources. I'll use the python-pptx library to create the file. If the right library is available, I'll generate the presentation and save it, then provide a link to download it.

On the use of ChatGTP/Grok/Gemini

- Python saves your (developing) time from C++!
  - Risk: you need “better” hardware for Python
- AI-assisted coding saves you even more time!
  - Risk: AI can and will make mistakes (hallucination)
- Do **NOT** trust them 100%!
  - Use when you can, avoid when you mu





Conclusion

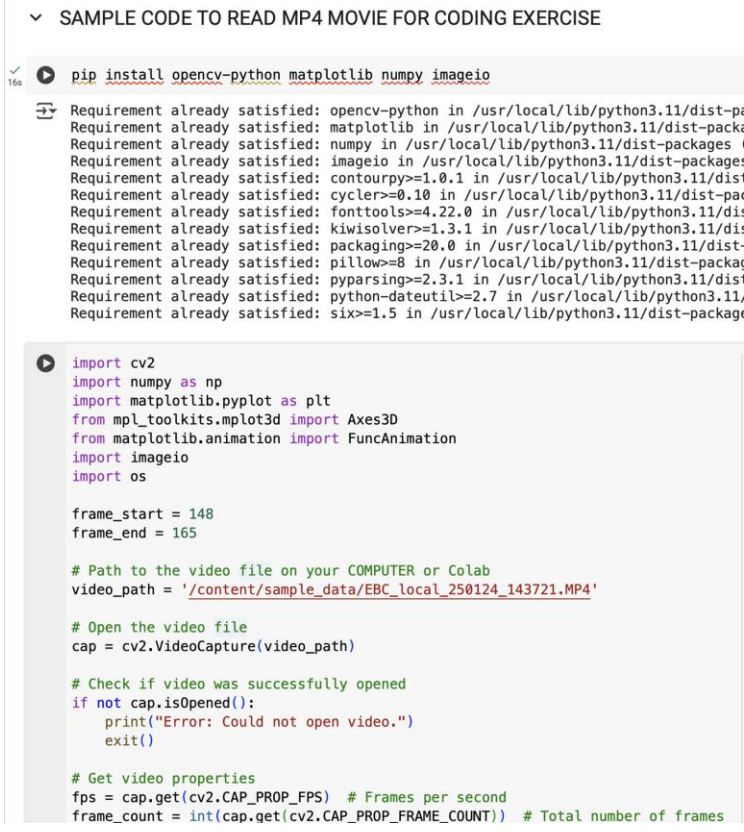
- Python is a must-have for any astronomer!
- It’s easy. Start coding today!
- Questions?

Coding project: moon rise

- Above is a video of a “moon rise” captured by a special type of camera
- Saved as MP4 movie file
- Your tasks:
  - Open the file and use python to plot an image of the moon from the file
  - Align the obtained image with the full_moon_image to identify the landmark

Coding project: moon rise

- Work in pair/ team of 3
- Instructions
  - Go to school's github: https://github.com/johnkimdinh/sagi_summer_school_2025
- Download the MP4 and JPEG file
- Use the sample code to install necessary library and help you get started with reading the file.
- Remember: AI when you can, your brain when you must!
- Happy coding!



SAMPLE CODE TO READ MP4 MOVIE FOR CODING EXERCISE

```
pip install opencv-python matplotlib numpy imageio
```

```
Requirement already satisfied: opencv-python in /usr/local/lib/python3.11/dist-pa
Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packa
Requirement already satisfied: numpy in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: imageio in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-pac
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dis
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dis
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packag
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-package
```

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib.animation import FuncAnimation
import imageio
import os

frame_start = 148
frame_end = 165

# Path to the video file on your COMPUTER or Colab
video_path = '/content/sample_data/EBC_local_250124_143721.MP4'

# Open the video file
cap = cv2.VideoCapture(video_path)

# Check if video was successfully opened
if not cap.isOpened():
    print("Error: Could not open video.")
    exit()

# Get video properties
fps = cap.get(cv2.CAP_PROP_FPS)  # Frames per second
frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))  # Total number of frames
```