Nonlinear Dynamics and Chaos

PHYMSCFUN12

Wladimir E. Banda Barragán

wbanda@yachaytech.edu.ec

MSc in Fundamental Physics

Yachay Tech University - 2025

Unit 1: Computational Nonlinear Dynamics
Python and computational methods

1. Computational tools for research in fundamental physics.

2. Why use free, open-source OS and languages?

3. Master Google Colab and Jupyter Notebooks for Linux and Python.

What do you need for the tutorials?

- A PC/laptop with any OS.

- Internet access.

- A Google/gmail account.

- A GitHub account (desirable, not strictly needed).

A few words on Git

Git is an **Open Source Distributed Version Control System** for tracking changes in source code or any other set of software files.

- **Control System:** Git is a content tracker.

- **Version Control System:** Code is constantly changing. Many developers can add code in parallel. Keeps history of what changes have been implemented.

- **Branching/Forks:** Also, Git provides features like branches and merges.

- **Distributed Version Control System:** Git has a remote repository stored in a server and a local repository which is stored in the computer of each developer.

About GitHub

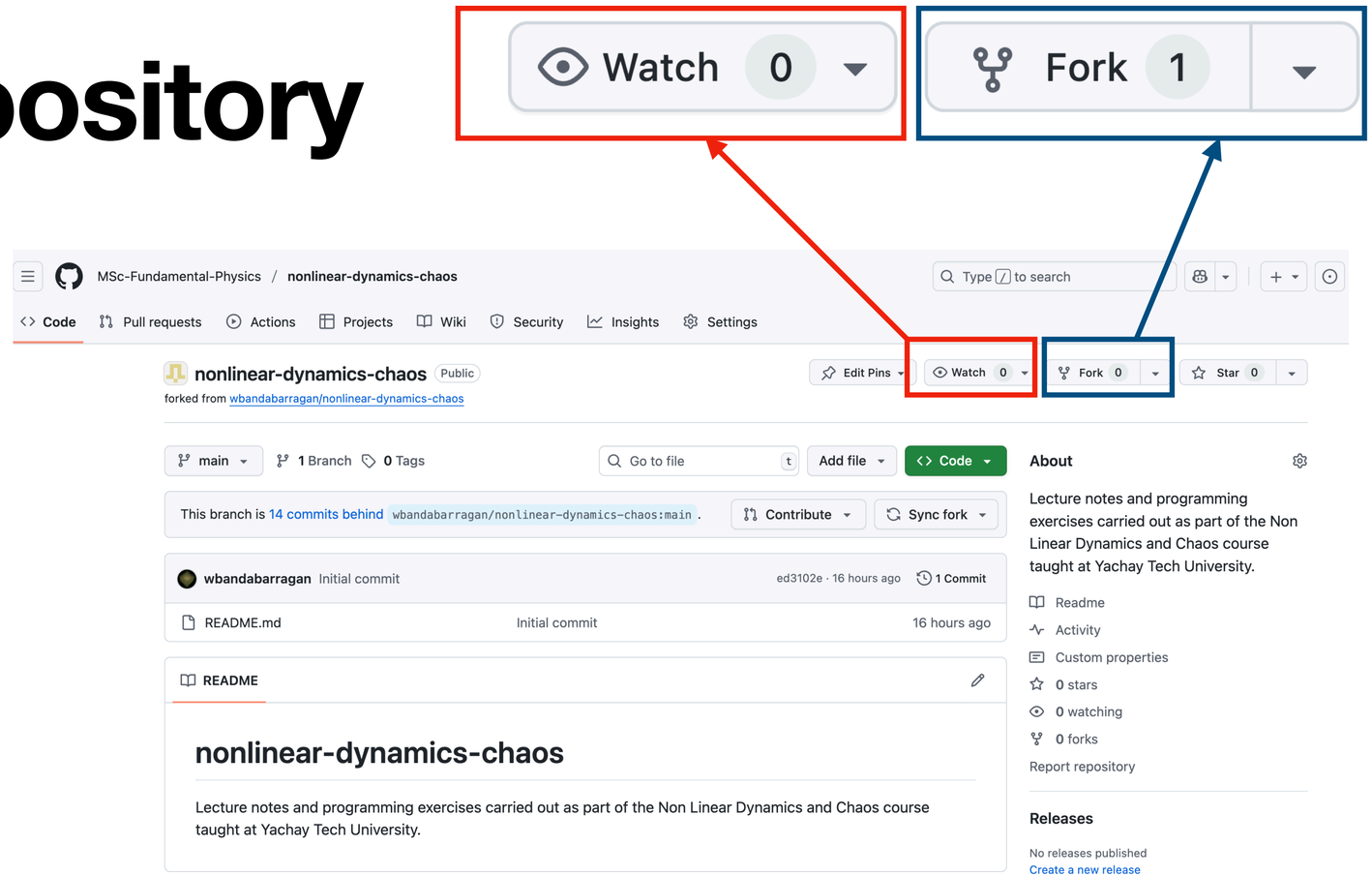
GitHub

GitHub is a web-based platform that provides hosting for software development and version control using Git.

It is an ideal platform to share code with colleagues.

Course GitHub Repository

https://github.com/MSc-Fundamental-Physics/nonlinear-dynamics-chaos



Tutorial Time:

If you don’t have a **GitHub account**, please create a free one now.

Then, fork the course repository in your own accounts.

I will now change something, you should Sync your repos.

Explore, create branches, etc.

https://github.com/MSc-Fundamental-Physics/nonlinear-dynamics-chaos

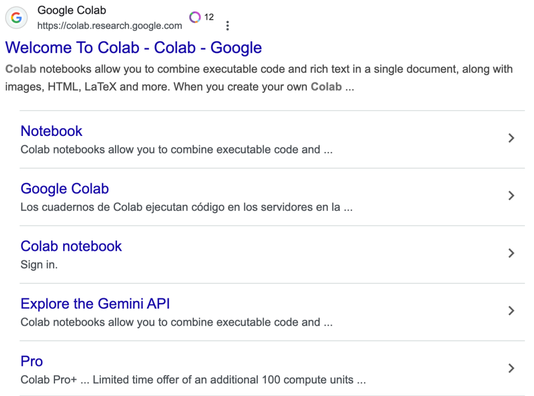
Google Colab and Notebooks

Google Colab allows us to access remote Linux servers owned by Google, where python is already installed.

With Google Colab we can access such servers via a browser, and this means we do not need to have python installed in our laptops/ PCs to use it.

The disadvantage is that we are bound to the RAM memory and disc space available in the remote server.

For short programming tasks and small computing loads, this is not a problem, but this could be a limitation if we are doing heavy computing, e.g. if we are analysing large data sets.



https://colab.research.google.com

Tutorial Time:

You should go to the course GitHub repository and click on the **Colab notebook** there.

https://github.com/MSc-Fundamental-Physics/nonlinear-dynamics-chaos

Linux and the need for open source software

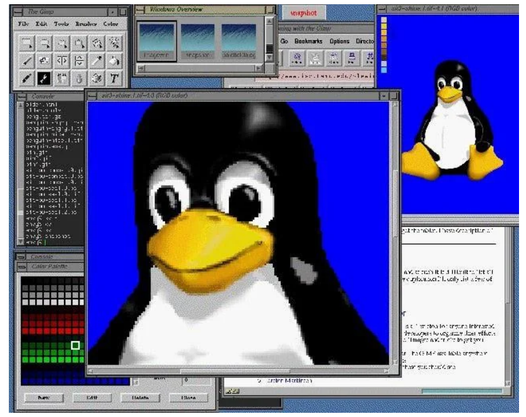
It is an operating system based on Unix, which was developed by Ken Thompson and Dennis Ritchie (at AT&T Bell Laboratories) during the 60's/70's.

Unix was highly portable, so it was adopted, copied, and modified by many companies and universities.

The source code was available, but modification and redistribution were restricted and its commercial version was too expensive.

Finnish student Linus Torvalds decides to create a new free operating system kernel called Linux.

Linux released its first version in 1991.



https://www.reddit.com/r/aggies/comments/vintc1/tamu_linux_released_in_1992_it_was_the_first/

Reference: https://www.greenteapress.com/thinkpython/thinkCSpy/html/chap01.html

Advantages of Linux OS

Linux OS are free and open source.

You can modify the source code and adapt it to your applications at will.

Linux provides security (much harder to hack, nearly no viruses).

https://www.geeksforgeeks.org/blogs/kde-vs-gnome/

Linux distributions come with an in-built platform to do programming.

There are two types of desktops:
KDE and GNOME.

Flavours of Linux OS

You have many options, the most popular ones in physics are:

1. Ubuntu
2. Fedora
3. Debian
4. CentOS

Linux can run on virtual machines / co-exist with other OS.

Linux is installed in (pretty much all) large-scale, high-performance supercomputers.

Linux is the OS of cloud servers (Google Colab).

The backend of GitHub relies heavily on **Linux**.

Basic Linux OS commands

1. man — offline manual, get help about any commands
2. which — find out where a command is defined.
3. <command> --help — Find help on any command
4. cd — Change the current directory (folder)
5. ls — List files in a directory
6. mkdir — Make/create a new directory
7. pwd — Print current directory
8. cp — Copy files and directories
9. rm — Delete files and directories
10. cat file.txt — see contents of file.
11. head file.txt — see the first 10 lines of a file
12. tail file.txt— see the last 10 lines of a file.
13. chmod — change permissions of a file or directory for 3 user groups: user (owner) permission, group permission, and other permission.
14. diff file1.txt file2.txt— show differences between two files
15. file — show the type of a file
16. less — browse the contents of a file, exit with q
17. locate — find files with names matching a pattern
18. touch — Create a new file or update an existing one
19. top — See what is going on, what processes are running, exit with q
20. ping server — check to see if a server is alive
21. df — show free disk space
22. du — show disk space usage
23. uname -a — information on Linux kernel
24. uptime — how long the system has been running
25. date — show current date/time

Thanks to modern coding tools, there is no need to install Linux /Python.

A browser and Colab is all we need!

Tutorial Time

Tutorial Time:

You should go to the course's GitHub repository and click on the **Linux notebook** there.

https://github.com/MSc-Fundamental-Physics/nonlinear-dynamics-chaos

What is computer science?

Computer science combines some of the best features of mathematics, engineering, and natural science.

Computer science uses formal languages to denote ideas (specifically computations).

Software engineering:
Design and assembling of components into systems.
Evaluate advantages/disadvantages among a pool of alternatives.

Science:
Observe the behaviour of complex systems, form hypotheses, and test predictions.

Reference: https://www.greenteapress.com/thinkpython/thinkCSpy/html/chap01.html

Problem Solving

The single most important skill for a computer scientist is **problem solving**.

- Ability to formulate problems.

- Think creatively about solutions.

- Express a solution clearly and accurately.

- The process of learning to program is an excellent opportunity to practice problem-solving skills.

- Learning to program, while using programming as a means to understand physics.

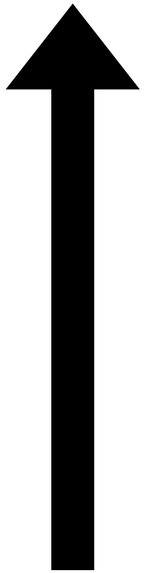
Reference: https://www.greenteapress.com/thinkpython/thinkCSpy/html/chap01.html

High-level vs. Low-level languages

Human

High-level

Python, Julia, IDL/GDL,
Mathematica, Matlab



C++, java

C89/90, Fortran77/90

Computer

Low-level

Binary (01001000)

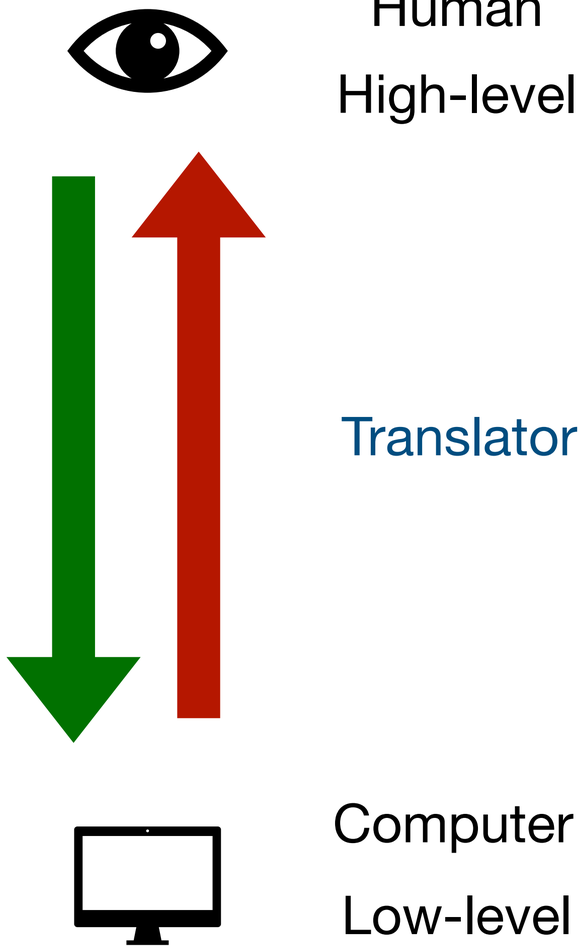
High-level vs. Low-level languages

Programs written in a high-level language **have to be processed** before they can run.

This **extra processing takes some time**, which is a small disadvantage of high-level languages.

Low-level programs can run on only one kind of computer and **have to be rewritten to run on another.**

Roughly speaking, computers **can only execute programs written in low-level languages.**

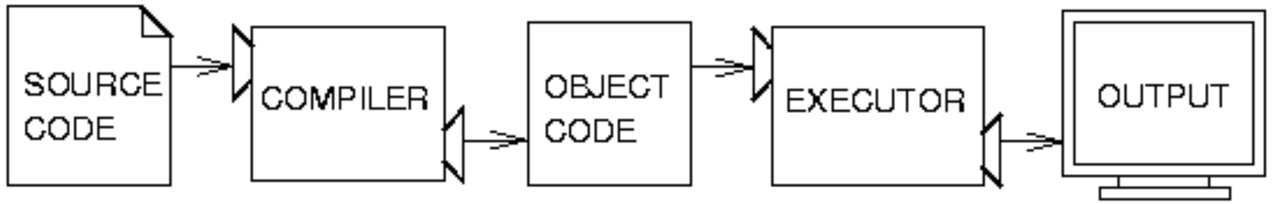


Interpreter vs. Compiler

Two kinds of programs process high-level languages into low-level languages:
interpreters and **compilers**.

An **interpreter** reads a high-level program and executes it. It processes the program alternately reading lines and performing computations.

A **compiler** reads the program and translates it completely before the program starts running. The program is the **source code**, and the translated program is called the **object code** or the **executable**. Once compiled, you can execute it repeatedly without further translation.



Why python?

Python is considered an interpreted language because Python programs are executed by an interpreter.

The advantages of high-level languages are enormous:

- It is much easier to program in a high-level language.
- Programs written in a high-level language take less time to write.
- They are shorter and easier to read, and they are more likely to be correct.
- They are portable, meaning that they can run on different kinds of computers with few or no modifications.

Low-level languages are used only for a few specialised applications.

Package Managers

pip (python-specific package installer)

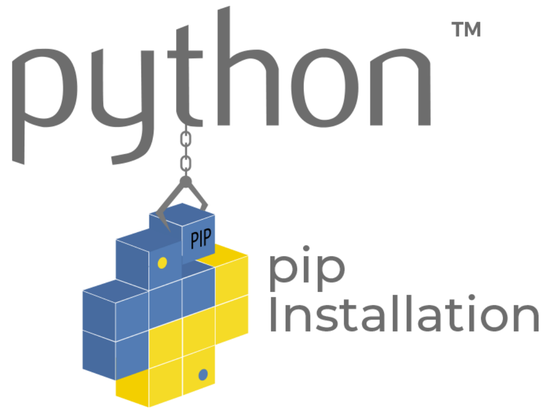
conda (language-agnostic package and environment manager)

- **Anaconda** (Full version)
- **Miniconda** (Reduced version)

source installation

- **Python Executable:**
  /usr/bin/python3

- **Standard Library & Third-Party Packages** (`site-packages`)
  /usr/lib/python3.x



Tutorial Time:

You should go to the course GitHub repository and click on the **Python Package Managers notebook** there.

https://github.com/MSc-Fundamental-Physics/nonlinear-dynamics-chaos

Programming Editors and IDE Tools

We can classify the tools for programming into 3 categories:

Command line editors:
vi/vim
nano
emacs

Integrated Development Environments (IDEs):
Visual Studio Code
Eclipse
Xcode

Web-based Interactive Environments:
Google Colaboratory (Colab)
JupyterLab / Jupyter Notebook

Programming Editors and IDE Tools

An IDE (Integrated Development Environment) is an application that provides comprehensive tools for software development.

An IDE consists of a source code editor, build automation tools (like compilers and interpreters), and a debugger.

Popular IDE choices are:

- Visual Studio Code (VS Code)
- Sublime Text
- Spyder
- Android Studio
- Xcode (Apple)
- **Google Colab** (cloud-based Jupyter notebook environment with IDE-like features)

Ideal Scientific Programming Setup

CLI Editor

IDE Editor

Web Editor

Cloud Editor

Package managers

Python Language

Other kernels

Tutorial Time:

You should go to the course GitHub repository and click on **Editors & IDEs for Coding**.

https://github.com/MSc-Fundamental-Physics/nonlinear-dynamics-chaos