

Lecture 1: Introduction and Setup

Coding: Introduction to Python - Applied Economics

Laboratory: Coding - Accounting, Finance and Business Consulting

Coding

- In this course you will learn the fundamentals of coding
- Coding is how we communicate with computers
 - **Allows you to tell a computer what actions to perform**
 - Making you able to exploit the opportunities offered by computers
- We will mainly focus on data analysis
 - Digital revolution brought unprecedented amount of data and analytics is the key to extract knowledge/value from it

This Course - Timetable and Office Hours

Coding: Introduction to Python

Monday 16:30 - 18:30

Laboratory: Coding

Tuesday 12:30 - 14:30

Office Hours:

Tuesday 15:00 - 17:00

Dsea, via del Santo 33, first office on the left entering the Levi Cases inner garden

This Course - Objective and Topics

This is an introductory course aiming at giving you the tools needed to do a simple data analysis using the Python programming language

Tentative list of topics:

- Setting up your python environment
- Introduction to algorithms
- Intro to data analysis

- Numpy and main data types
- Pandas and dataframes
- Data import and export
- Loops and Iteration
- Data description and visualization
- Data analysis example

This Course - Evaluation - Notions

Pass/Fail exam at the LEM

Closed-ended multiple-choice questions

- Testing your knowledge of the language
 - Given `var_a = 'True'`, what type of data is the variable `var_a` ?
 1. Boolean
 2. String
 3. Integer
 4. Float

This Course - Evaluation - Understanding

Pass/Fail exam at the LEM

Closed-ended multiple-choice questions

- Testing the understanding of the language
 - what is the output of `var_a = 1; var_b = 2; var_c = var_a + var_b; print(var_c)` ?
 1. 1
 2. 2
 3. 3
 4. "error"

This Course - Evaluation - Errors

Pass/Fail exam at the LEM

Closed-ended multiple-choice questions

- Testing the ability to spot errors in the code
 - what is the first error that will be raised by `var_a = 1; var_b = "two"; var_c = var_a + var_b; print(var_d)` ?
 1. `NameError: name 'var_d' is not defined`
 2. `TypeError: unsupported operand type(s) for +: 'int' and 'str'`

3. `TypeError: unsupported operand type(s) for +: 'float' and 'str'`
4. `SyntaxError: EOL while scanning string literal`

This Course - How to prepare?

The course is not designed to be absorbed passively

Clearly the basis is reading, understanding and being able to recall the material. The material is going to be available in Notebook format (`.ipynb` files) that include explanations, code, and code output

- You will be able to run the code and modify it to test your understanding

As you progress with the material you should experiment with the tools and syntax we cover

- Simply reading the material will not be enough
- Coding is a practical endeavour
 - You can gain "muscle" memory from doing things instead of just reading and try to memorize

I will be proposing exercises for each topic

- Solving them is very important to test your understanding
- Crucial to get prepared for the exam
- Usually teach you how to do basic (and crucial) tasks that you might need in other courses

This Course - Books and Additional Material

The course relies on material from a variety of sources, such as books, software documentation, online tutorials, blogposts, etc.

Here you can find two **great and free** books that cover the material of this course and much more: [Python for Data Analysis](#) [Python Data Science Handbook](#) by Jake VanderPlas

During the course, I will provide links to additional material that might help in understanding the topics covered

Coding and Algorithms

Coding involves crafting a set of computer-executable instructions.

- When we code we define an **algorithm** to be performed by the computer
- An algorithm is a set of instructions for solving a problem or accomplishing a task

Algorithms and Data Analysis

- A data analysis is a set of algorithms to be performed on data:
 - Importing data
 - Cleaning data
 - Describing data
 - Visualizing data
- Typically, you start this process with specific questions you seek to address through the data
 - Define the steps needed to answer those questions
 - Write and execute the code to obtain your answers
 - Through this process, you'll gain deeper insights into both the data and the questions you're investigating
 - Refine your initial questions and the methods to resolve them
 - This is an iterative process that leads to the final analysis
 - The final analysis will be a reproducible set of algorithms allowing to draw knowledge from data
 - Answers to your questions will be descriptive statistics, plots, tables, statistical tests, etc.

The Tools of Data Analysis in Python

We will start presenting the coding tools we will use in the course and how to set them up

- Main tools: Python, Jupyter Notebook
- Auxiliary tools: Anaconda, Colab, VLEM
- Focus on the role of each tool
- Learn syntax to perform basic actions

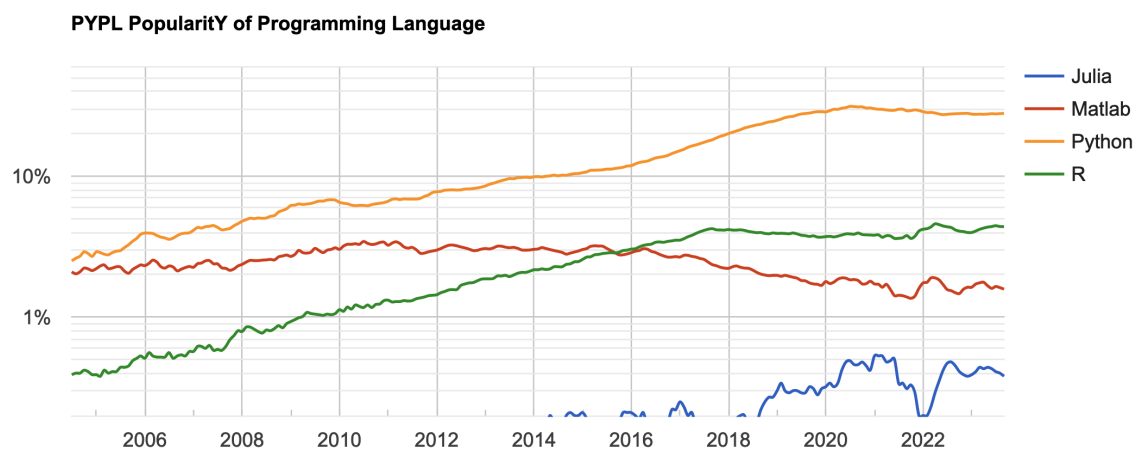
Python

Python is a general-purpose programming language. Created in 80s, it is now one of the most popular languages in the world.

- Not the fastest but easy to use and very versatile

- Can be used for web development, videogames, edit videos/music, create Large Language Models (LLM) such as Chat GPT, manage networks, etc.
 - Used and supported extensively by Internet services and high-tech companies: Google, Netflix, Meta, Amazon, Reddit
- Python versatility and huge community led to the development of endless extensions
 - Crucial role in today's data analysis landscape thanks to many data analysis tools (to make it easy to write/debug/share of analysis) and libraries (to clean/analyze/visualize data)
- Python developers are in huge demand in the job market
 - Knowing python can improve your efficiency and efficacy in any job that involve a computer
 - Most of the Deep Learning and LLM tools that revolutionized the data analysis environment in later years are developed in Python

Python popularity



Based on tutorial searches on Google (more info [here](#))

Tools for Data Analysis - Python

Python: Provides a command-line Python interpreter

- Takes your code and translates it to the processor so that can be executed
- Its usual inputs are scripts that perform a set of pre-defined actions
- Not well suited to perform data analysis that involves continuous exploration
 - Read the data, analyze it to understand some aspect, visualize the result, perform sub-analysis on the result or on the original data based on what has been learnt from the results and so on

Tools for Data Analysis - Jupiter

IPython/Jupyter Notebooks: Extend the Python interpreter making it more usable for scientific computing

- Provides an interactive command-line terminal for Python
- Allows read-eval-print loop (REPL) to ease the interaction with the data
- In 2015, the IPython became Jupyter Notebook that provides a browser-based control panel
- While Jupyter notebooks are run in a browser, they connect to a computational engine (the kernel)
 - can be used not only with Python but also with R, Julia, SageMath
- Offers a more convenient text editor, includes formatted text, static and dynamic visualizations, mathematical equations, JavaScript widgets, and much more.
- Eases data analysis reproducibility by facilitating other people to execute the code on their own systems
- This presentation and the course book are Jupyter notebooks

Tools for Data Analysis - IDEs

IDEs (Integrated Development Environment): are applications making programming easier by combining a suite of tools to develop code: code highlighting, support for writing (AI-completion), rendering, debugging, etc.

- We will have a look at [Microsoft Visual Studio code \(VS code\)](#) because it is integrated with [Copilot](#), an AI-tool to write code

Setting up your coding environment

We will see three ways to set up your coding environment:

- A local installation on your laptop
- Using the VLEM of the dSEA
- Using Google Colab

Set up the coding environment on your laptop

We will see how to do so using **Anaconda**, a famous python and R distribution for data science

- Need to install all the software on your laptop
 - **PROs** No internet needed to run the code, no constraints on what to install, all your files readily available, all the computational power of your laptop
 - **CONS** Installation requires a bit of work, some knowledge of how a computer works and can entail unexpected problems
- Probably the best option

- Is the most flexible and powerful, opens the door to many other tools
- You will develop a bit of practice on some valuable computer science skills

Use an environment in the cloud

- Rely on a remote server that has **some** of the needed software already installed
 - **PROs** Avoid all the installation difficulties and start to code "right away"
 - **CONS** Requires an internet connection, less flexible
- We will see two options:
 - Aula Virtuale **VLEM** is a platform developed by the dSEA to support the teaching of programming and data analysis
 - **Google Colab** is a free service offered by Google that provides a Jupyter notebook environment in the cloud

Use the coding environment on the VLEM

The dSEA provides a Virtual Learning Environment that allows you to run Jupyter notebooks on a remote server

Our IT has set up a cloud instance with all the needed tools for this course

- The instance you should use is the VLEM-BASE
- Information on the VLEM is available [here](#) (in Italian, but google translate works well)
- To access the VLEM is necessary to [book a slot](#)
- **PROs** No need to install **anything** (is it a pro if your aim is learning?)
- **CONS** Need an internet connection, need to book a slot, limited time (30 minutes), cannot store anything and any modification to the environment is lost

Use a coding environment on Google Colab

There are many options for coding in Python on the cloud

- Google [Google Colab](#) is one of the best
- Offers a free tier with your Google account (that also include GPUs) and its paid tier has an extremely rich offer (GPU for AI)
- Google Colab instances come with only a minimal set of pre-installed packages
- Google instances are temporary, if you stay idle for too much or you close your window your instance gets deleted
 - But you can connect it to your Google Drive to save your notebooks and coding environment
- Most of the slides/notebooks we will use include a "Launch notebook" button that will directly run them in Colab

- **PROs** Always ready with one click, possibility of storing the notebooks and personalizing the environment, a framework used by professionals (great for your experience)
- **CONs** Need an internet connection, need some set up to have a semi-permanent environment (but we will see it together), the interface is a bit different from the standard Jupyter notebook

Setting up Python and Jupyter on your computer using Anaconda

Anaconda is a distribution (a collection of software that is pre-configured to work together) for Python and R that is focused on data science

- Simplifies deployment and is focused on scientific computing and data analysis
- Has a graphical interface to manage libraries and environments
- Also installs Jupyter, so that you edit notebooks
- Provides many other tools for data science
- Has a comprehensive and detailed [documentation](#), a lively [community](#) plus a massive knowledge base on [Stackoverflow](#)

Installing Anaconda

We will now see how to install Anaconda on your computer following the [installation docs](#)

Command Line Interface and Graphical Interface

Anaconda can be used through the command line interface (CLI) or the graphical interface (GUI)

- Each have their advantages and disadvantages
 - The CLI is more powerful and more flexible
 - The GUI is easier to use and more intuitive
- For this course the GUI is enough, but start getting your head around the CLI, we will need to use it later on

Anaconda Navigator

Anaconda Navigator is the GUI of Anaconda

- It allows you to manage your environments and libraries

To get started with Navigator we will use this [anaconda doc](#)

Setting up a python environment

An environment is the context in which a Python program runs, it specifies:

- What is the interpreter to be used
 - In this case the default version of Python
- Defines its fundamental settings
 - Where the libraries are stored

Setting up the environment for this course: codepy

To set up the environment to be used in this course

1. Open Anaconda Navigator
2. Click on the Environments tab on the left shoulder
3. Click on 'create', name the new environment `codepy` select Python and the latest version
4. Click on 'create' and wait for the environment to be created (it may take a while, it is downloading and installing a lot of stuff)
5. Click on the 'Home' tab on the left shoulder
6. Select the `codepy` environment from the drop down menu
7. Click on 'install' on the Jupyter Notebook tile
8. Click on 'install' on the VS Code tile (we will use it later on)

Congrats! You have set up your first Anaconda Python environment

Creating a Jupyter notebook

To create your first Jupyter Notebook

1. Open Anaconda Navigator
2. Go to the home tab
3. Select the `codepy` environment
4. Click on 'launch' on the Jupyter Notebook tile
5. A new tab will open in your browser listing the files in your home directory
6. It is convenient to store all your notebooks in a single directory
 - Click on 'New' on the top right corner and create a new directory
 - Select the "Untitled folder" just created and click the rename button on the top left
 - Rename it `codepy`
 - Click on the `codepy` directory
7. Click on 'New' on the top right corner and select "codepy" from the drop down menu
8. You can now start editing your first notebook

Install a package in Anaconda

1. Open Navigator and select the environment tab
2. Select the `codepy` environment
3. Click on the dropdown menu and select not installed
4. Search for the library you are interested in, let's say `rise`
5. Click on the checkbox on the left of the `rise` package and apply at the bottom of the window
6. Now if you change the dropdown menu to installed you will see the `rise` package listed
7. You can now import the package in your notebook and use all its functionalities

Install the packages needed for this course in Anaconda

The packages needed for this course are:

- `numpy`
- `pandas`
- `matplotlib`
- `seaborn`

install them in the codepy environment following the steps above

Setting up the environment in Colab

If you are going to use Colab in this course (even just on occasion), it is worth to set up a semi-permanent environment

While the set-up of Anaconda is a standard procedure and there is official documentation, what we are doing is something a bit more custom, so we will make follow [this](#) tutorial.

Homework

Next week we will start coding, so you need to set up your environment

My Earnest advice is to do all three set-ups

- Each has its pros and cons
 - You will be able to choose the one you like the most
 - You will be able to use the one that is more convenient for the task at hand
 - Some things we will see are only available in one of the three
- You will learn a lot by doing so
 - On topics that are crucial for your future career
 - On tools that can widen your horizon and make you more efficient

Notes

Author: Duccio Gamannossi degl'Innocenti

Web: <https://www.dgdi.me/>

Mail: duccio.gamannossi@unipd.it

Office Hours:

Tuesday 15:00 - 17:00

Dsea via del Santo 33, first office on the left entering the Levi Cases inner garden