# Python for Econometrics and Operations Research

## A crash course

08-09-2025

# Desktop computer

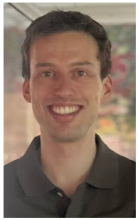
1. Start computer by pressing bottom-right button (below Intel vPRO sticker)
2. Log in with your e-mail address (initial.initial.lastname@tilburguniversity.edu) and your password.

# Team

Dr.ir. Sander Gribling

Dr.ir. Pieter Kleer

Prof.dr. Johan van Leeuwaarden



Dr.mr. Sven Polak

+

Mathijs Barkel, M.Sc.

# What is Python?

- Python is a *(computational) programming language*.
- We instruct a computer to perform operations via written text.
- The text needs to be very exact.
  - Otherwise the computer will throw a syntax (i.e. spelling/grammar) error.

# About Python

Popular programming language for data science tasks.

- Analysis of mathematical high-dimensional functions.
  - Plotting, finding roots/minima/maxima, integration, etc.
- Analysis and visualization of large-scale data sets.
  - E.g., weather or stock market predictions.

Developed by Guido van Rossum;

- Initiated at Centrum Wiskunde en Informatica (CWI), Amsterdam.
- Name comes from Monty Python's Flying Circus (British comedy series)

# Why this programming language?

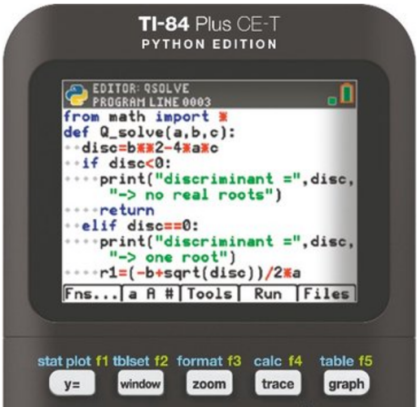
Free, open-source and most popular programming language for data science!

Many companies program in Python ...

- ... including many master thesis students who do company internships.

Good skill to have on you CV!

# Graphing calculator vs. Python




Graphing calculator can analyse one-dimensional functions like

$$f(x) = x^2 + 2x - 1.$$

**Python can analyse higher-dimensional functions like**

$$\mathbf{f(x, y) = x^2y + 2xy^2 - x - 1.}$$

# Python (and programming) in EOR curriculum

Programming courses:

- Programming for EOR (Year 2)
- Computational Aspects in Econometrics (elective, Year 3)

Assignments of:

- Linear Optimization **(Q2)**, Probability Theory, Introduction Finance and Actuarial Sciences, Quantative Finance, ….

# Other languages in curriculum

You will also see other programming languages like Matlab and R:



- Similar functionality as Python (although "syntax" is different).

AI-tools like ChatGPT can also program! Not always allowed, though.

# Plan for today

Materials (online book) at: https://pskleer.github.io/eor-python-crash-course-2025/

- Cover Python Basics (Chapter 3) using centralized explanations and Exercises 3.1-3.7.
- Lecture 2 (September 23) will cover Linear Algebra with Python!

Chapter 3 of online "book" - Python basics

# Python as a calculator

Can use +, −, *, /, and ** to perform basic arithmetic operations.

| Operation | Symbol | Example | Result |
|---|---|---|---|
| Addition | + | 2 + 3 | 5 |
| Subtraction | - | 5 - 3 | 2 |
| Multiplication | * | 2 * 3 | 6 |
| Division | / | 3 / 2 | 1.5 |
| Exponentiation | ** | 2 ** 3 | 8 |

# Variables

Suppose we want to compute the function value $f(x) = x^3 + 2x^2 + x - 1$ for $x = 5$.

```
5**3 + 2*5**2 + 5 - 1
```

179

If we want $x = 4$, we have to change $5$ to $4$ in three places. Better to define variable for $x$.

```
x = 5

x**3 + 2*x**2 + x - 1
```

179

**Try Exercises 3.1 and 3.2!**

# Lists

Can be used to store multiple variables in one *list* object.

```
z = [3, 9, 1, 7]
z
```

```
[3, 9, 1, 7]
```

Can access numbers in list by *indexing* them.

```
z[1]
```

9

Why does this give 9? **Python starts counting at $0$ when indexing.**

```
z[0]
```

3

# For-loop

```
a = [1, 4, 2, 5]

total_sum = a[0] + a[1] + a[2] + a[3]
total_sum
```

12

More efficient to use a *for-loop* for adding up numbers in $a$.

# For-loop (cont'd)

```
a = [1, 4, 2, 5]

total_sum = 0

for i in [0,1,2,3]:
    total_sum = total_sum + a[i]

print(total_sum)
```

12

| i | a[i] | total_sum after this iteration |
|---|---|---|
| 0 | 1 | $0 + 1 = 1$ |
| 1 | 4 | $1 + 4 = 5$ |
| 2 | 2 | $5 + 2 = 7$ |
| 3 | 5 | $7 + 5 = 12$ |

# For-loop (cont'd)

Suppose we only want the first two numbers in $a$: Change list with values for $i$.

```
a = [1, 4, 2, 5]
total_sum = 0

for i in [0,1]:
    total_sum = total_sum + a[i]

print(total_sum)
```

```
5
```

**Try Exercises 3.3 and 3.4!**

# Conditional statements

Carry out different lines of code depending on what statement/condition is true.

- Can be done with `if` and `else` keywords.

```
x = 5

if x > 0:
    print("x is positive")
else:
    print("x is is not positive")
```

```
x is positive
```

# Conditional statements (cont'd)

If we have more than three conditions, we can use `if`, `elif` and `else`

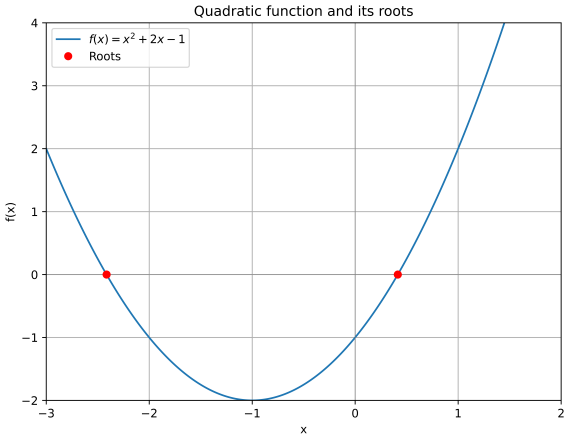
```
x = 0

if x > 0:
    print("x is positive")
elif x == 0:
    print("x is zero")
else:
    print("x is negative")
```

```
x is zero
```

**Try Exercise 3.5!**

# Mathematical example: Root finding for quadratic function.

Let $a, b$ and $c$ be given. Find $x$ such that $f(x) = a \cdot x^2 + b \cdot x + c = 0$.



# Discriminant

Number of solutions to $f(x) = a \cdot x^2 + b \cdot x + c = 0$ determined by **discriminant** $D = b^2 - 4ac$:

- If $D > 0$, the equation has two real roots $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$
- If $D = 0$, the equation has exactly one real root $x = \frac{-b}{2a}$.
- If $D < 0$, there are no real roots.

**Try Exercise 3.6 (and 3.7)!**