Outroduction

Programming for Economists

Jeppe Druedahl
Brigitte Hochmuth

Plan

1. Learning goals
2. Exam
3. Repetition

Learning goals

Summary

1. Summarize and visualize empirical data
   $\Rightarrow$ gain flexibility and automation compared to Excel
   $\Rightarrow$ build upon Descriptive Economics
   $\Rightarrow$ working with »big data«
2. Numerically solve and simulate economic models
   $\Rightarrow$ less strict assumptions than with math and easy visualization
   $\Rightarrow$ improve your understanding of Micro and Macro
   $\Rightarrow$ from model-user to *model-builder*

Course overview

- **Week 37-38: Basics** (types, containers, views vs. copies, conditionals, loops, functions, floating point numbers, classes, numpy)
- **Week 39-41: Data** (print, plot, save/load, online data, descriptive economics, structure and documentation, workflow and debugging, git)
- **Week 43-44: Tools** (optimization, root-finding, random numbers, simulation)
- **Week 45-48: Models** (Solow, Walras, AS-AD, calibration/estimation)
- **Week 49-50: Perspectives** (dynamic optimization, speed and parallization, artificial intelligence)

Learning phases

1. **Basics:** A million new concepts and syntax.
   How does it all fit together!?!?! I will never understand this.
2. **Data:** Arghhh! It is also hard to use in practice.
   But I begin to be able to do some cool stuff.
3. **Models:** Hard to understand math and code simultaneously.
   But the combination is really powerful.
4. **Perspectives:** I have learned a lot! And so many possibilities ahead for working with data and models.

Knowledge

1. Describe the differences between fundamental data types (e.g. strings, booleans, integers and floats)
2. Describe the differences between data containers (e.g. lists, dicts and arrays)
3. Explain the use of conditionals (if-elseif-else)
4. Explain the use of loops (for, while, continue, break)
5. Explain the use of functions, methods and classes
6. Describe the difference between views and copies of objects
7. Explain how to use numerical optimizers and root-finders
8. Explain how to use (pseudo) random numbers
9. Explain how to use linear interpolation

Skills

1. Setup a Python environment
2. Write Python scripts, functions and notebooks
3. Structure and document code
4. Test and debug code
5. Use version control system (Git)
6. Import and export data and use online databases (APIs)
7. Summarize and visualize data
8. Use numerical optimizers and equation solvers
9. Solve economic models numerically
10. Simulate economic models
11. Calibration economic models to data

Competencies

1. Write well-structured and well-documented code
2. Work collaboratively on code projects
3. Present and discuss results of a numerical analysis of empirical data and economic models

Exam

Exam

- **Portfolio exam**
  - **Part 1** (40%): Project 1+2 revised based on received feedback
  - **Part 2** (60%): 48 hour home assignment
  - **Grading:** Pass/Fail. *The fail grade is given when the assignments jointly show that many learning goals has not been adequately understood*.
- **Dates:** 17th-19th of January 2026
- **Copilot and LLMs:** Allowed, even encouraged $\Rightarrow$ state how you have used in the »GAI declaration«

  Example: »I have used the GitHub Copilot in VSCode. I have both used it to generate suggestions for new code and correct and improve code I have written.«

Home assignment

- **Structure:**
  - 3 problems with 3-6 sub-questions.
  - Working with data or on solving and simulating economic models.
  - Easier than problem sets and project assignments
- **Curriculum:** Lecture notebooks (except 13+14 in Perspectives)
- **Packages:** *No new packages are required, and using non-standard packages are actively discouraged*
- Exam from **Introduction to Programming** (2019-2024) [link]
  1. Curriculum was broader
  2. Exam problems were harder

Answering

1. **Focus on answering the questions** - nothing more, nothing less
2. Explain your **method in words** (or with a step-by-step algorithm)
3. **Structure and comment your code!**
4. Explain your **results in words**
5. **Partial answers, attempts and considerations** are also **awarded** (something on everything is better than a lot on a few questions)

   If you think there is an error in your code, but you can't/don't have time to fix it, addressing this in words is a **huge** plus.

Disclaimer: Solving the full exam project in depth might be hard.

Hand-in in Digital Exam

- **You should hand-in a single zip-file named with your groupname only.**
- The zip-file should contain:
  1. A general README.md for your portfolio
  2. Your data analysis project (in the folder 01_dataproject)
  3. Your model analysis project (in the folder 02_modelproject)
  4. Your exam project (in the folder 03_examproject)
- If you've uploaded everything to GitHub, you can download a zip of your repository from there (press the green 'code'-button)

Repetition

Repetition

Based on your survey answers