operations research
QUANTITATIVE INFRASTRUCTURE SYSTEM MODELING

Operations Research – Coding Lab

Lecture Slides

Team Operations Research

SS 2025

Technische Universität Berlin
Workgroup for Infrastructure Policy (WIP)

Agenda

1. Introduction

Agenda

Team

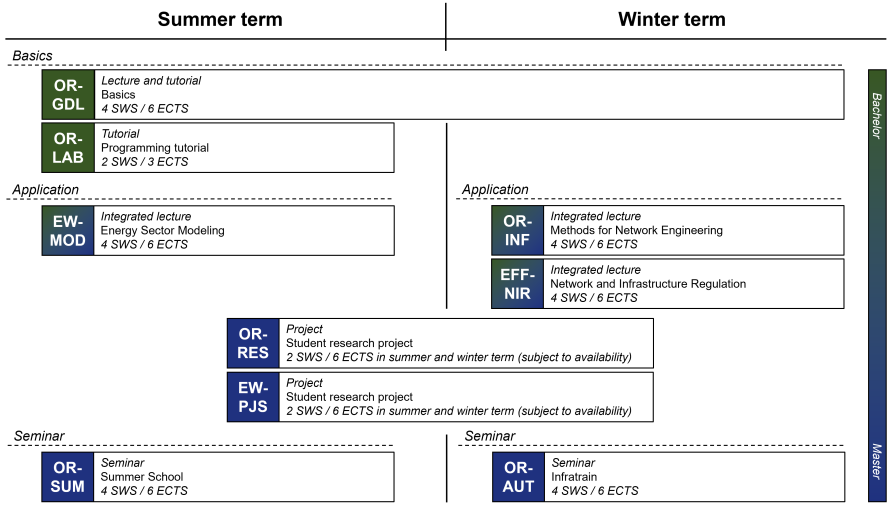
Team Assistant

- Petra Haase

Teaching Staff

- Prof. Christian von Hirschhausen
- Nikita Moskalenko
- Richard Dupke
- Lukas Barner

Operations Research - Course landscape



Content of this course

Topics

- Introduction to the programming language *Julia*
- Using the basic features of Julia
- Packages and environments
- Building optimization models with *JuMP*
- Results and visualization
- Data import, processing, and export
- Advanced topics

Aims

- Learning basics of a programming language: *Julia*
- Learning how to create numerical optimization models in *Julia/JuMP*, including pre- and post-processing steps

Examination information

- Bi-weekly lectures in **presence** on Wednesdays, 16.15 to 17.45
- Ungraded portfolio examination
- Exam registration via MTS, closes Tuesday, May 06 2025, 23:59
- 2 SWS, 3 ECTS
- Iterative individual coding assignments **[50 points]**
  - ▷ You can collaborate with up to two other students, but they must be indicated at the top of your submitted file (as shown in the template file)
  - ▷ Individual submission, only you are responsible for the contents of your file!!
- Final coding project **[50 points]**
  - ▷ Coding project with data import, data processing, modeling and result visualization
  - ▷ Hand in commented code!
- You need a total of **50**/**100 points** to pass the course

Outline of the course

Date	Topic	Assignment	Due Date
2025-04-23	Introduction, Julia Setup and Julia Basics		
2025-05-07	Julia Basics	Assignment 1	
2025-05-21	Simplex algorithm with Julia	Assignment 2	Assigment 1
2025-06-04	Packages, Introduction to JuMP	Assignment 3	Assignment 2
2025-06-18	Plotting	Assignment 4	Assignment 3
2025-07-02	Data processing	Assignment 5	Assignment 4
2025-07-16	Advanced topics, Final project	Final project	Assignment 5
2025-07-30			Final project

Table: Overview of course content.

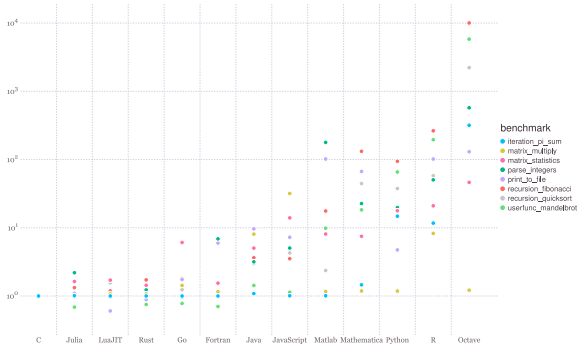
Agenda

Motivation

“Looks like Python, feels like Lisp, runs like C or Fortran”

Why Julia?

- It is open source and free
- It is fast (similar to C or Fortran)
- It is comparatively easy to learn and use (similar to Python or R)
- It features a powerful library called “JuMP” which enables us to build large optimization problems



Source: https://julialang.org/benchmarks/

General Coding Remarks

How to learn Julia in this course

- If you have questions which could be relevant for other participants please use the ISIS discussion forum!
- Most problems can be solved by a quick search or directly through the respective documentation.
  - It's impossible to remember every function call or command...
  - Developing the ability to search effectively is an essential skill you'll gain in this course
- Whenever something doesn't work, return to the last point where it did and investigate what might be causing the issue.

Troubleshooting

- Julia Discourse is a helpful forum to look for help or inspiration
- Stackoverflow
- Search the internet to solve your problems! This is usually the most efficient and fastest way of solving your problems.

Agenda

Some commands to use in the PowerShell*/Terminal

Windows*

- Shows the current working directory: *pwd*
- Move to a directory: *cd **
- List the files in your current directory: *ls*
- Create a new folder: *md **
- Copy a file: *cp **

Mac OS X

- Shows the current working directory: *pwd*
- Move to a directory: *cd **
- List the files in your current directory: *ls*
- Create a new folder: *mkdir **
- Copy a file: *cp **

- *Attention: PowerShell $\neq$ CMD Terminal
- *We recommend using the PowerShell

Agenda

Installing Julia

- Version A:
  - See the juliaup documentation, or the blue boxes below.

Windows

- winget install −−name Julia −−id Julialang.Juliaup −e −s winget

Mac & Linux

- curl -fsSL https://install.julialang.org | sh
- brew install juliaup (Homebrew)
- zypper install juliaup (openSUSE)

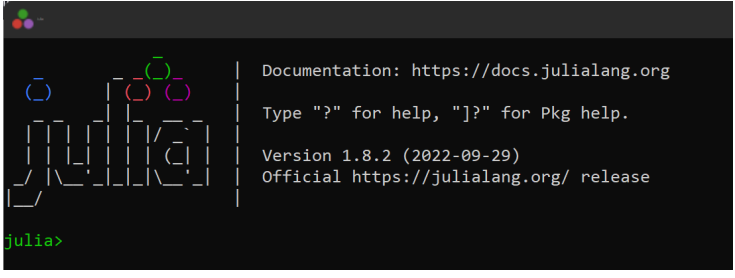
- Version B:

Install Julia manually

- On the official Julia page, download the installer for your specific operation system
- Follow the installation instructions
- Important: Make sure to tick the box when asked to add Julia to your path!

Starting Julia from the commandline/terminal

- Open the terminal of your choice.
- Navigate to the directory you want to work in (cd "directory").
- If Julia was successfully installed, you should be able to start Julia by typing *julia*.



What is VSCode

- Essentially a text editor with a lot of useful features
- You can run code by using extensions
- A lot of quality of life features are included (Git, debugging, syntax highlighting, auto completion)
- Download installer on `https://code.visualstudio.com/`

Using Julia in VSCode

- Start VSCode.
- Inside VSCode, go to the Extensions view by clicking View on the top menu bar and then selecting Extensions.
- In the Extensions view, search for the term "julia" in the Marketplace search box, then select the Julia extension (julialang.language-julia) and press the Install button.
- Restart VSCode.

Starting Julia in VSCode

- Open the terminal of your choice.
- Navigate to the directory you want to work in.
- Use *code .* to start VSCode in the current working directory. (If you are in a different directoy you can also replace . with the directoy you want VSCode to start from.)
  - Alternatively, select the directory in the VSCode explorer
- Create a file that ends with *.jl*. VSCode will automatically realize that this file contains Julia code.
- There are different ways to execute the code:
  - Pressing the “Play“-button in the top right
  - Depending on the default assigment you can use *Shift + Enter*, *Alt + Enter*, or *CTRL + Enter* to run the selected code in the REPL