The Julia programming language

Related exercise sheets

- Sheet 0: Getting Started with Julia
- Sheet 0: Basic Plotting in Julia

In this class we employ the Julia programming language for code examples and programming exercises. In this notebook you will learn why this choice has been made and we provide details on how to get started with julia.

Jump to the installation instructions

Table of Contents

About the Julia programming language

You might wonder why in this course we have opted to employ the Julia programming language, when Python seems to be the language of teaching.

What makes Julia particularly appealing for teaching topics in computational science are in my opinion four things:

1. **Accessible and mathematical syntax**: Julia code resembles mathematical equations extremely closely, sometimes literally 1:1. Julia code is thus close to pseudocode for many algorithms.
2. **Interactivity**: Julia and Pluto (the environment we use) are well-suited for interactive exploration. We will use this a lot in this class, e.g. by embedding sliders that allow you to explore for yourself.
3. **Julia can be fast**: Since the language is compiled like `C` or `Fortran` (and in contrast to Matlab or Python) basic constructs like `for` loops are surprisingly fast. For you this means that even without worring about performance, your code can actually be tested on interesting problems, which usually are the ones, where we need mathematical insight to get things working.
4. **Growing interest in industry and academic jobs**: Experience in using Julia starts to be a criterion on the Job market, e.g. LinkedIn Jobs or JuliaCon discourse

About Pluto

What makes using Julia with Pluto particularly nice is the easy interactivity. We will use this a bunch in this class. For example consider the plot of the two functions

```
f (generic function with 1 method)
```

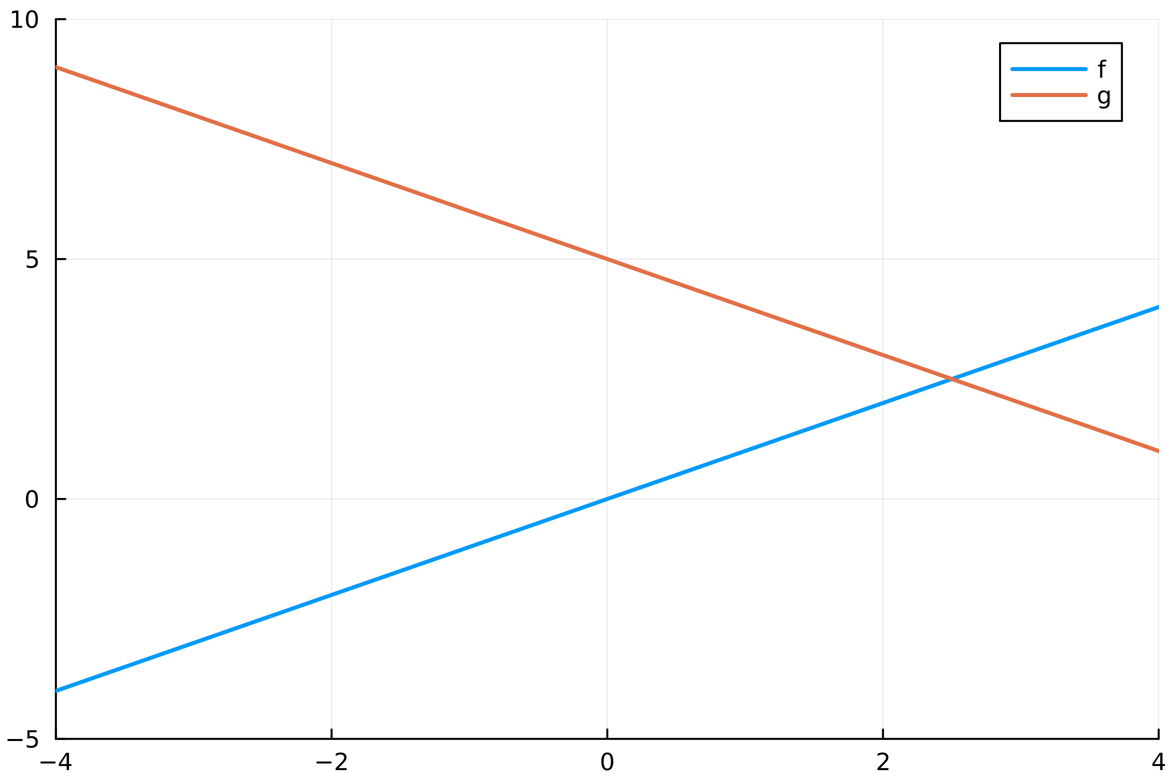
```julia
f(x) =     a * x
```

```
g (generic function with 1 method)
```

```julia
g(x) = 5 + b * x
```

where we choose the parameters via a slider

- a = 1.0
- b = -1.0



What is the point where the two curves intersect?

Exercise

Code the expression that gives the x coordinate:

```
x = missing
```

```julia
x = missing
```

Missing Response

Replace `missing` with your answer.

Hint

Equating $f(x) = g(x)$ one can easily find that

$$ax = 5 + bx \quad \Leftrightarrow \quad 0 = 5 + (b - a)x \quad \Leftrightarrow \quad x = \frac{5}{a - b}$$

[illegible]

Note how the computed intersection appeared above !

A good introduction to pluto are the featured notebooks on https://plutojl.org, for example:

- Basic mathematics with pizzas
- Markdown and text
- Comparison with Jupyter notebooks

Installing Julia

For this class, you will have to install Julia on your own computer. Assistants are there to provide support during the first week and you are welcome to ask for help. After that, Julia is assumed to be installed and working, and there will be a check in the second exercise session.

Installing on your own computer

Take a look at the installation notes of MIT's computational thinking class or watch:

Essentially this boils down to:

- Download Julia **1.12.4** for your operating system from https://julialang.org/downloads. **Windows users** can install Julia in the Microsoft store.
- Start Julia, e.g. by starting the **Julia 1.12.4** from your program menu or by executing `julia` in a Terminal.
- Install Pluto, the notebook environment that we will be using in this course. Note, that Pluto is not the only way you can run Julia, but it is particularly well-suited for the quick and interactive experiments we will perform. To install Pluto run in your Julia REPL:

  ```julia
  import Pkg
  Pkg.add(; name="Pluto", version="0.20.21")
  ```
- Run Pluto: Still in the REPL execute

  ```julia
  import Pluto
  Pluto.run()
  ```

See the installation notes of MIT's computational thinking class for more details.

The exercise notebooks are written and tested with **Julia 1.12.4** and **Pluto 0.20.21**. Prefer those.

Useful Julia resources

Get started with coding Julia

- Mathematics with Pizzas
- Some language processing
- Plots.jl tutorial

Overview of Julia learning resources

- Learning Julia

Comprehensive coverage

- Programming for Mathematical Applications course at UC Berkeley
- Think Julia: How to Think Like a Computer Scientist
- Official Julia documentation

Tips and tricks

- Modern Julia Workflows

Cheatsheets

- General syntax cheatsheet
- Comparative cheatsheet Python <-> Julia <-> Matlab
- Plots.jl cheatsheet

Getting help in Pluto

Pluto has a built-in live documentation, which is very useful. Just type in a cell `? <keyword>` to get started, e.g. to learn about the `plot` function type `?plot`.

```
# ?plot
```

Getting Julia help in general

- The EPFL-internal Matrix server (element.epfl.ch) has a Julia Lang room. You can logon to this server using your Gaspar identification and from there you should be able to join this public room.
- Julia's discourse forum is full of many helpful people and its search feature is a useful resource for answers to many beginner's questions.
- Some more advanced resources is the official JuliaLang documentation as well as the Modern Julia Workflows project.

Numerical analysis