Computer Vision II

Julia Introduction

24.04.2019

TECHNISCHE UNIVERSITÄT DARMSTADT

visual inference

Group assignments

- You will be working in groups of exactly two people
- Please form groups until next Wednesday
- Link for group assignment will be up today

- If you have not found a partner yet
  - Find him right after the exercise class
  - Use the discussion board

Why Julia?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Previously used Matlab for CV1
- Easy to learn and use
- Many built-in tools for image processing and optimization
- License hassle
- Students needed to be in our lab / pool or get there “own” license

- Julia promises to combine ease-of-use of Matlab with fast execution and open source license.
- We will use Julia **v1.1.0**

Tutorial with IJulia notebook

Try it: www.juliabox.com

Package Manager

- `Pkg.add("pkgname")` to install packages
- `Pkg.update()` to update all packages

- `using pkgname` to import packages into namespace

- Standard packages:
  - PyPlot package for plotting
  - Images package for loading/storing images
  - JLD package for loading/storing generic data to containers

Differences to Matlab

- Indexing via square brackets: e.g. `x[1]`
- Loops are fast
- Broadcasting is simple: cf. broadcasting
- Multiple functions can be declared in a single file
- Packages need to be imported

Pitfalls

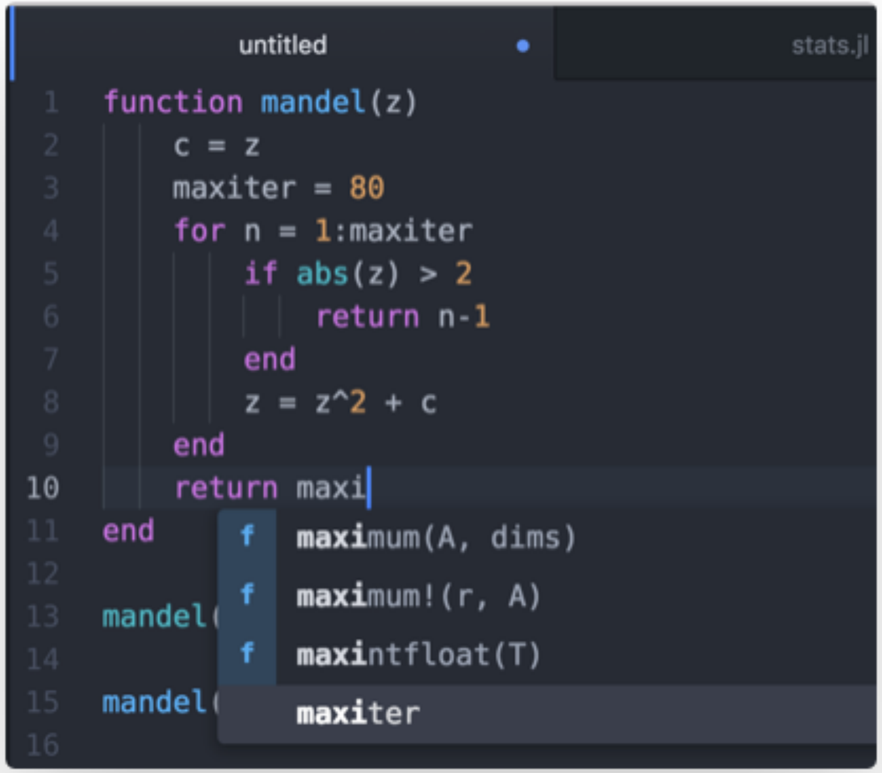
- Take care of the current path and working directory
  - relative vs absolute path
- Double-check when indexing multi-dimensional arrays
  - row, column
- Dynamic typing and implicit conversions can lead to unexpected data types
  - use `typeof(data)` to check for correct type
- Broadcasting is implicit and may yield unexpected results

Developing with Julia

- Popular IDE: http://junolab.org

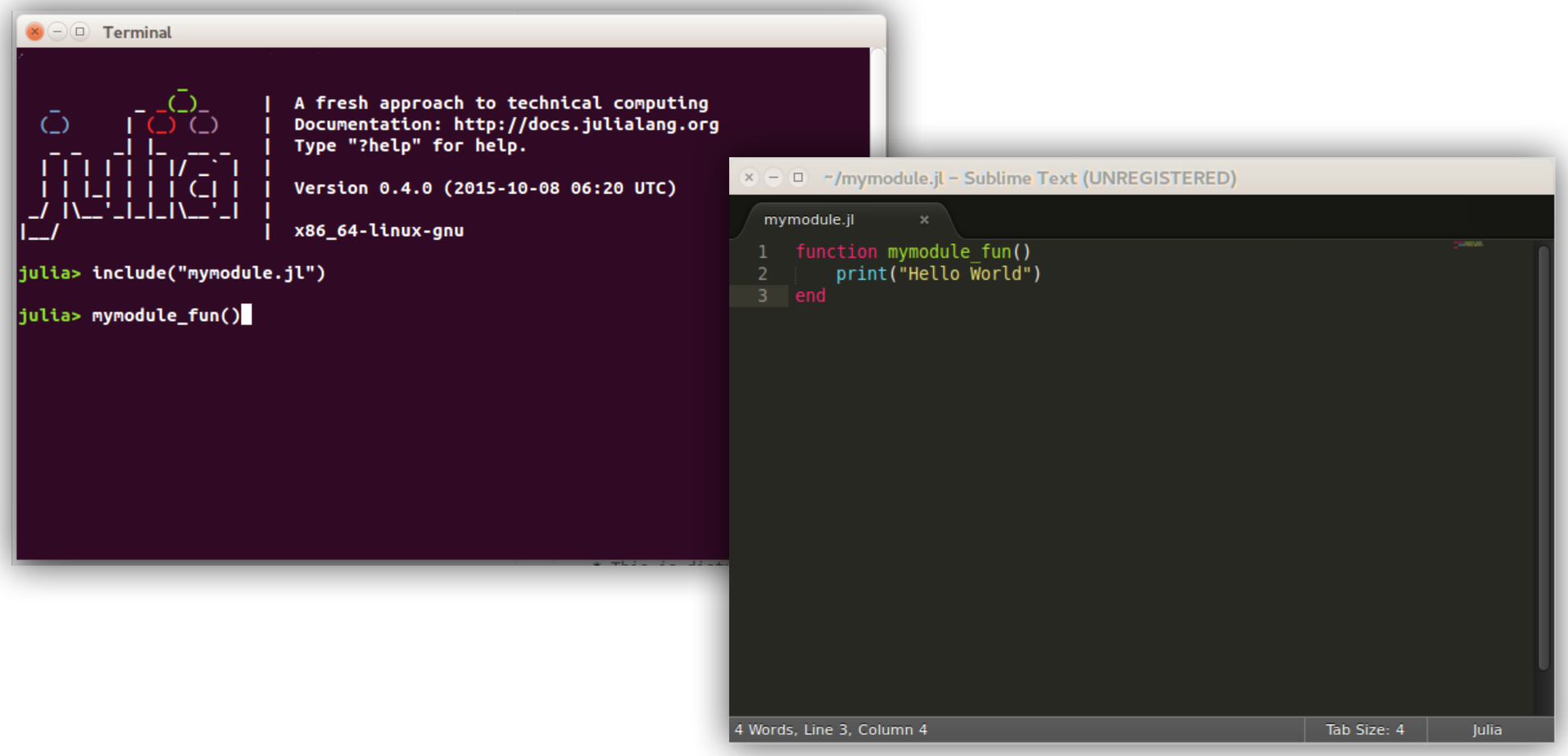
Walks like Python. Runs like C.

Juno builds on Julia's unique combination of ease-of-use and performance. Beginners and experts can build better software more quickly, and get to a result faster.



Developing with Julia

- Use terminal + text editor



Debugging Julia

TECHNISCHE UNIVERSITÄT DARMSTADT

- Blog post introducing a new debugger
  - Collection of tools (powered by single interpreter)
  - Rebugger (REPL interface ~ gdb)
  - Revise (update definitions automatically)

- Enter debugging session with
  - `Juno.@run`
  - `Juno.@enter`

- Console output
  - `@assert`
  - `@test`

Debugging Julia – Helpful links

TECHNISCHE UNIVERSITÄT DARMSTADT

- Blog post introducing a new debugger

- Installation
  - https://discourse.julialang.org/t/ann-juno-0-8/22157

- Some problems/issues are discussed here:
  - https://discourse.julialang.org/t/new-julia-debugger-in-juno/22359

- Debugging how-to
  - http://docs.junolab.org/latest/man/debugging/

Best practices

- Use explicit `return` keyword in functions
- Exploit multiple dispatch
- Avoid global code:
  - Use functions instead of scripting

Julia – Helpful links

- Julia homepage:
  - http://www.julialang.org/
- Julia documentation:
  - https://docs.julialang.org/en/v1/
- Julia package list:
  - http://pkg.julialang.org/
- Performance Tips:
  - https://docs.julialang.org/en/v1/manual/performance-tips/index.html