Julia, a programming language for **Operations Research**

Prof. Marc Sevaux

Université de Bretagne-Sud – Lab-STICC – Lorient – FRANCE

with the support of Dr. Alexandru OLTEANU, Quentin PERRACHON and Owein THUILLIER

September, 2023

Research

http://people.univ-ubs.fr/marc.sevaux/

Research tools

- Heuristics
- Metaheuristics
- Mathematical programming
- Matheuristics

Interest

- Combining Metaheuristics and Machine Learning
- Quantum Operations Research

Main goal

- Designing efficient solving methods
- Playground: Combinatorial Optimization Problems

Applications

- Routing and Scheduling problems
- WSN, Sonar networks
- Warehouse management
- ...

Teaching

Teaching programs

- Mathematical engineering
- Electronic design
- Production management
- Industrial engineering

Courses

- Intro OR/graph theory
- Mathematical programming
- Heuristics/metaheuristics
- Data analysis/management
- Large scale optimization

Level of students

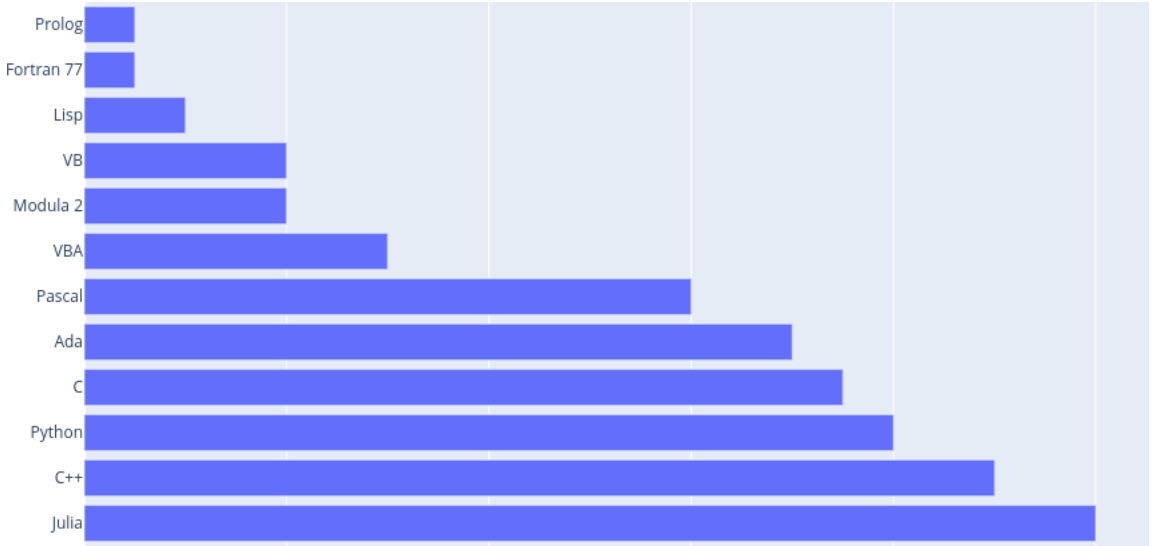
- Bachelor students (last year)
- Master students
- Engineer students

Type of students

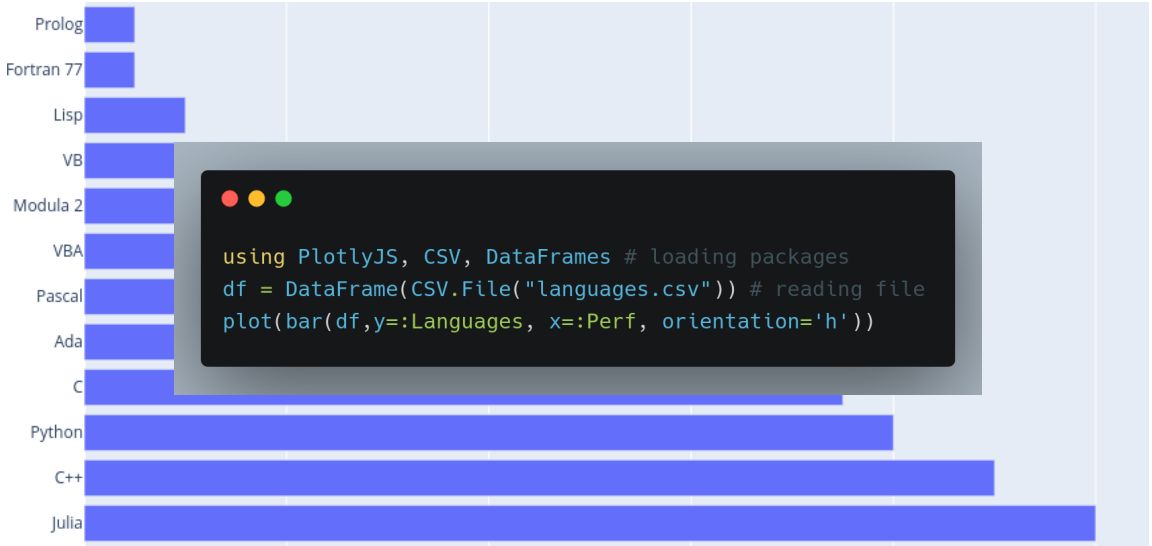
- Classical students
- Full distance learning
- Professional students

Not a specialist in Computer Science, but use programming languages in every class...

Programming history (proficiency)



Programming history (proficiency)




```
using PlotlyJS, CSV, DataFrames # loading packages
df = DataFrame(CSV.File("languages.csv")) # reading file
plot(bar(df,y=:Languages, x=:Perf, orientation='h'))
```

Why Julia?

5 reasons to learn julia today

Speed as general as Python, as statistics-friendly as R, but as fast as C

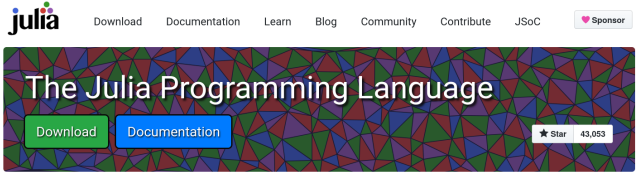
Syntax dynamically-typed, closer to R and Python, easy to learn

Memory automatic memory management

Multiple dispatch Julia functions' ability to behave differently based on the types of arguments

OR friendly Julia's environment is well adapted to OR with rapid prototyping, speed, packages, link to other languages, ...

http://julialang.org



Julia in a Nutshell

Fast

Julia was designed from the beginning for high performance. Julia programs compile to efficient native code for multiple platforms via LLVM.

Dynamic

Julia is dynamically typed, feels like a scripting language, and has good support for interactive use.

Reproducible

Reproducible environments make it possible to recreate the same Julia environment every time, across platforms, with pre-built binaries.

Composable

Julia uses multiple dispatch as a paradigm, making it easy to express many object-oriented and functional programming patterns. The talk on the Unreasonable Effectiveness of Multiple Dispatch explains why it works so well.

General

Julia provides asynchronous I/O, metaprogramming, debugging, logging, profiling, a package manager, and more. One can build entire Applications and Microservices in Julia.

Open source

Julia is an open source project with over 1,000 contributors. It is made available under the MIT license. The source code is available on GitHub.

See Julia Code Examples | Try Julia In Your Browser

Julia REPL (Read Evaluate Print Loop)

Julia is Fast

Quick version

You call a function, Julia will specialize that call all the way down to its concrete types (thanks to multiple dispatch).

In the backstage: LLVM + JIT compiler

Let's see how Julia compare to other languages...

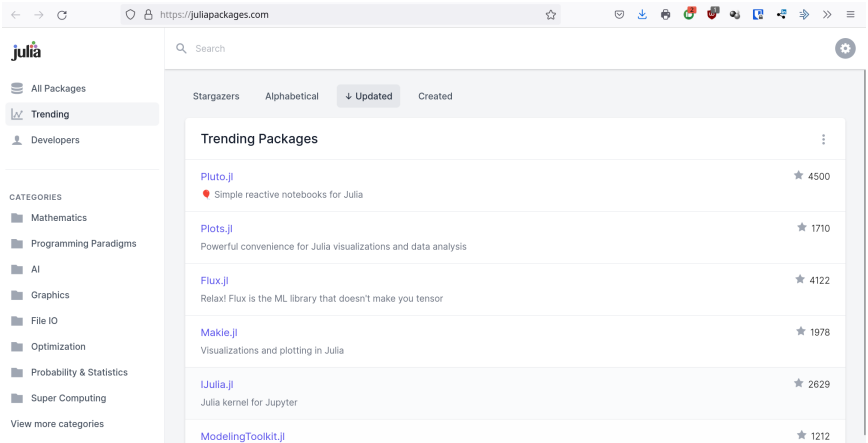
Julia is easy to understand

Intuitive and easy

- Julia is fast
- Julia is easier and cleaner than C++
- You can avoid the two-language problem

See `dataframes`, `arrays/vectors`, `functions` and `broadcast`...

Julia packages



Let's explore the packages `Graphs`, `PlotlyJS` and `JuMP`...

Conclusion

Cons.

- Julia is a young language
- Compile time latency may happen
- Ecosystem is immature
- Subtle syntax tricks

Pros.

- Julia is easy to learn
- Julia is fast
- Julia REPL is amazing
- Many ways to use it
- Subtle syntax tricks

Learn julia now and thank your younger self in the future...