MIT Course 18.S096, IAP 2018
Performance Computing in a High-Level Language

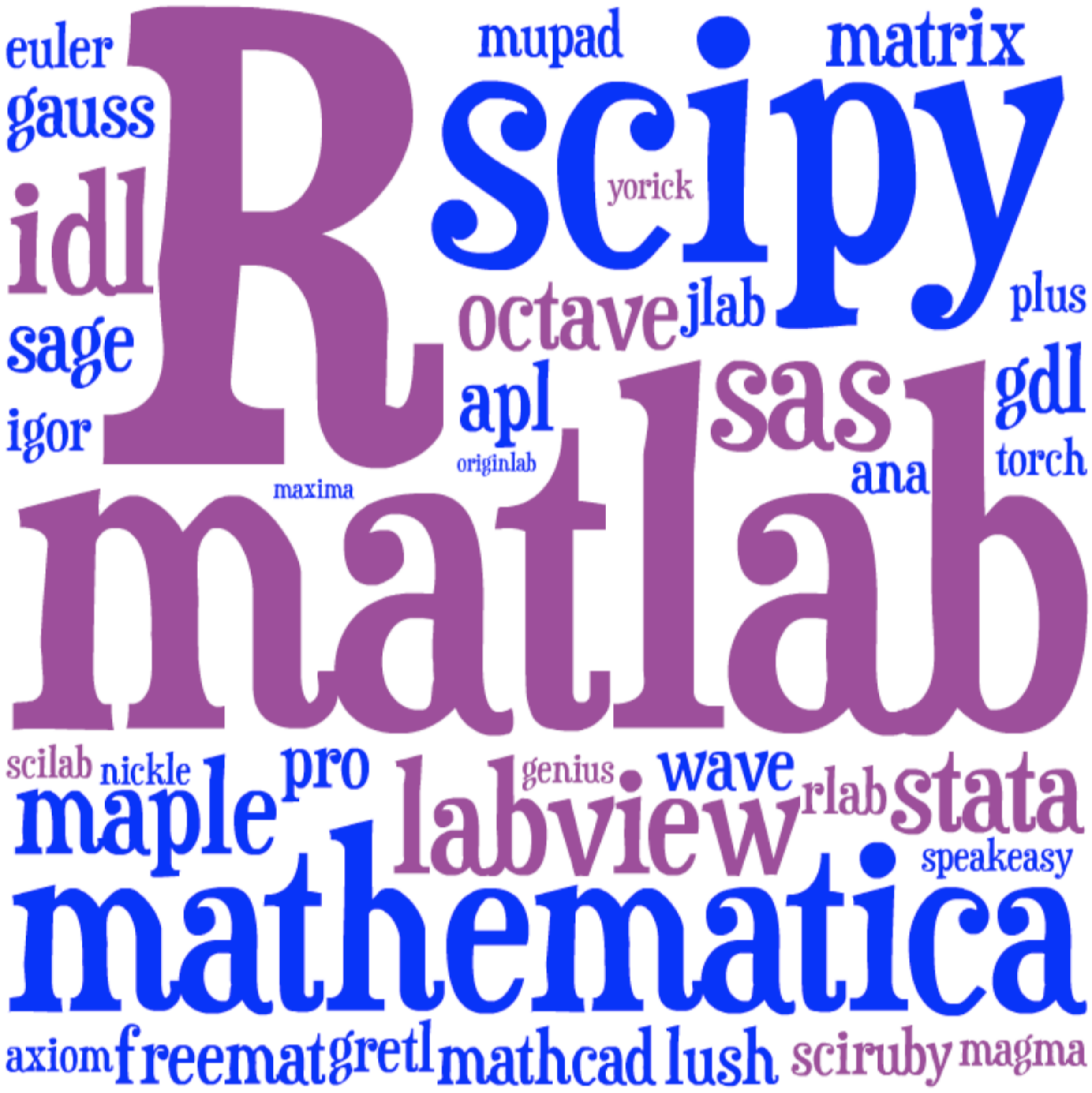
Alan Edelman & Steven G. Johnson,
MIT Applied Math

https://github.com/stevengj/18S096

Administrivia

- Lectures Tues/Wed/Fri 2–4pm, 2-135
- “Lab” Thursday 2pm: Graded attendance
- Weekly psets, due Monday
  - We give you slow code, you give us fast code
- 4 units, A–F grading

Lots of choices for interactive math…



[image: Viral Shah]

Course goals

- Understand the connection between the low-level architecture of the computer and the performance characteristics of high-level languages.
- Learn how to write and optimize your own performance-critical code
- Have fun!

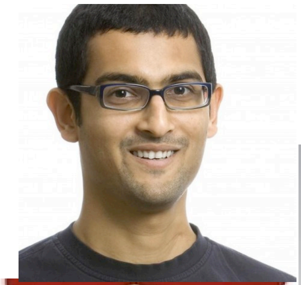
Need a language for all three

- Matlab/Python/R: Too slow
- C/C++/Fortran: Too low-level/insane
- Go/Rust/Haskell: Not interactive enough (statically typed, not dynamically typed)
- … ?

A new programming language?

Jeff Bezanson

Viral Shah



Stefan Karpinski

Alan Edelman

[MIT]

[40+ developers with 100+ commits,
1600+ external packages, 5th JuliaCon in 2018]

julia

julialang.org

[begun 2009, “0.1” in 2013, ~40k commits,
“0.6” release in June 2017]

As high-level and interactive as Matlab or Python+IPython,
as general-purpose as Python,
as productive for technical work as Matlab or Python+SciPy,
but as **fast as C**.

Installing Julia

- *Quick start:* run it “in the cloud”: juliabox.com
- Install it on your own machine:
  - Download Julia 0.6 from julialang.org
  - Launch julia
  - Install IJulia/Jupyter notebook interface

```
Pkg.add("IJulia")
```

  - Run:

```
using IJulia
notebook()
```