Introduction to Julia

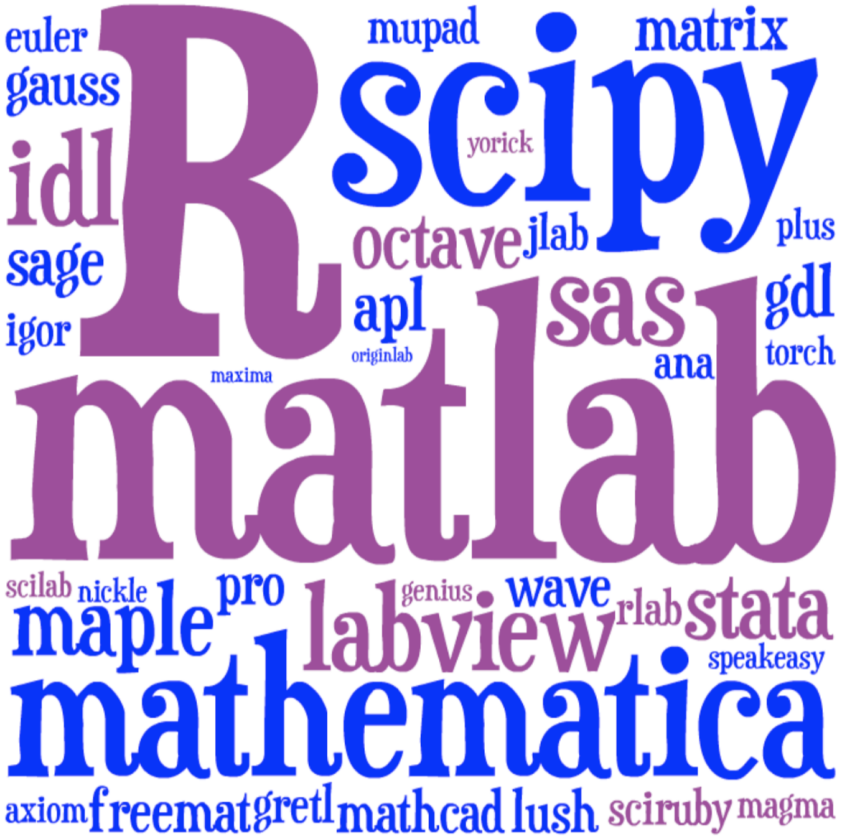
Ranjan Anantharaman

CSAIL

29th January, 2019

Why (yet) another programming language?

Lots of choices for interactive math…



[image: Viral Shah]

julia

Why (yet) another programming language?

- Why not use C/C++/Fortran for performance?

julia

MIT

Why (yet) another programming language?

- Why not use Python for general purpose computing?

julia

MIT

Why (yet) another programming language?

- Why not use R for statistical computing?

Why (yet) another programming language?

- Why not use MATLAB for scientific computing?

julia

MIT

Why (yet) another programming language?

- By the same token, why not use numpy/scipy or numba?

julia

MIT

A new programming language

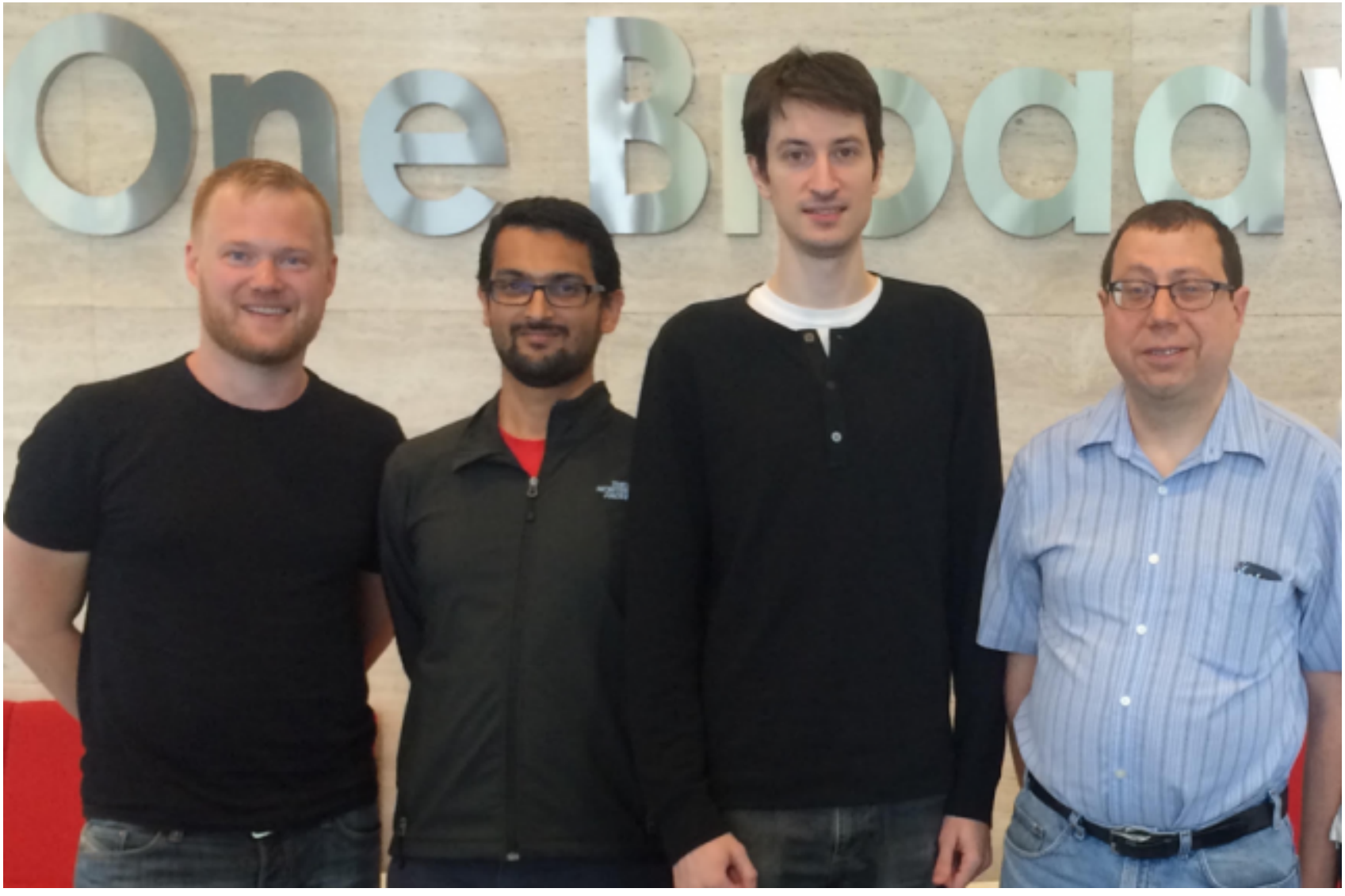


Figure 1: (left to right) Stefan, Jeff, Viral, Alan

A New Programming Language

Being Greedy

All these languages are very good for specialised tasks, but
We need a language as **interactive** as MATLAB or Python, as **general purpose** as Python, while being as **fast** as C.

Read the blog post.

Did they succeed?

julia

MIT

A New Programming Language

Being Greedy

All these languages are very good for specialised tasks, but
We need a language as **interactive** as MATLAB or Python, as **general purpose** as Python, while being as **fast** as C.

Read the blog post.

Did they succeed?

- January 2019: Julia team wins Wilkinson Prize. (MIT News)

julia

MIT

A Simple Example

Let's compare the Julia vs Numpy code generate Vandermonde matrices:

$$\begin{pmatrix} 1 & \alpha_1 & \alpha_1^2 & \dots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \alpha_2^2 & \dots & \alpha_2^{n-1} \\ 1 & \alpha_3 & \alpha_3^2 & \dots & \alpha_3^{n-1} \\ \vdots & \vdots & \ddots & \vdots & \\ 1 & \alpha_m & \alpha_m^2 & \dots & \alpha_m^{n-1} \end{pmatrix}$$

A Simple Example

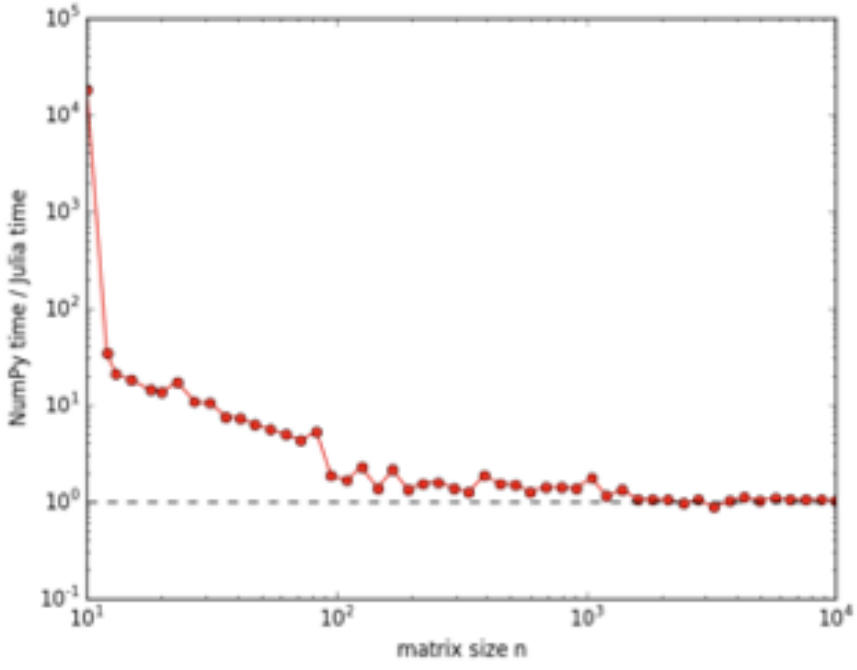


Figure 2: $\frac{NumpyTime}{JuliaTime}$. See how the ratio tapers off to 1, indicating similar level of performance

A Simple Example

Numpy : Python code wraps C code, which wraps generated C code. The following is the Julia code:

```
function vander(x, n=length(x))
    m = length(x)
    V = Array(eltype(x), m, n)
    for j = 1:m
        V[j,1] = one(x[j])
    end
    for i = 2:n
        for j = 1:m
            V[j,i] = x[j] * V[j,i-1]
        end
    end
    return V
end
```

This works for any container for any type with the $*$ operation.

julia

MIT

So why Julia?

Workflow 1

- Prototype algorithms in a high-level scripting language, like MATLAB or Python.
- Port performance critical code to C/C++/Fortran
- Flexible, but a lot of work, and two runtimes to maintain.

So why Julia?

Workflow 1

- Prototype algorithms in a high-level scripting language, like MATLAB or Python.
- Port performance critical code to C/C++/Fortran
- Flexible, but a lot of work, and two runtimes to maintain.

Workflow 2

- Find favourite Python package which runs C/C++/Fortran
- Make sure all your runtime is spent calling fast code.
- Gets the job done, but inflexible.

So why Julia?

Workflow 1

- Prototype algorithms in a high-level scripting language, like MATLAB or Python.
- Port performance critical code to C/C++/Fortran
- Flexible, but a lot of work, and two runtimes to maintain.

Workflow 2

- Find favourite Python package which runs C/C++/Fortran
- Make sure all your runtime is spent calling fast code.
- Gets the job done, but inflexible.

What if you need **performance** and **flexibility**?

julia

MIT

So why Julia?

Consider Julia instead:

- High level syntax, making code accessible & maintainable.
- Dynamically typed, feels like a scripting language.
- High performance, ideal for real work.

But you don't lose access to all your favourite libraries in other languages: easy to call C/Fortan, Python, R etc.

julia

MIT

Course Outline

This course has the following learning objectives:

1. How to write performant, generic code in Julia.
2. How to understand why it isn't performant, and to fix it, without sacrificing readability.
3. Understand how Julia works.
4. Start or contibute to a Julia project.