A brief introduction of Julia Programming Language

Chuhao Li

Institute of Physics, Chinese Academy of Science

October 11, 2019

Why Julia?

- It is fast!

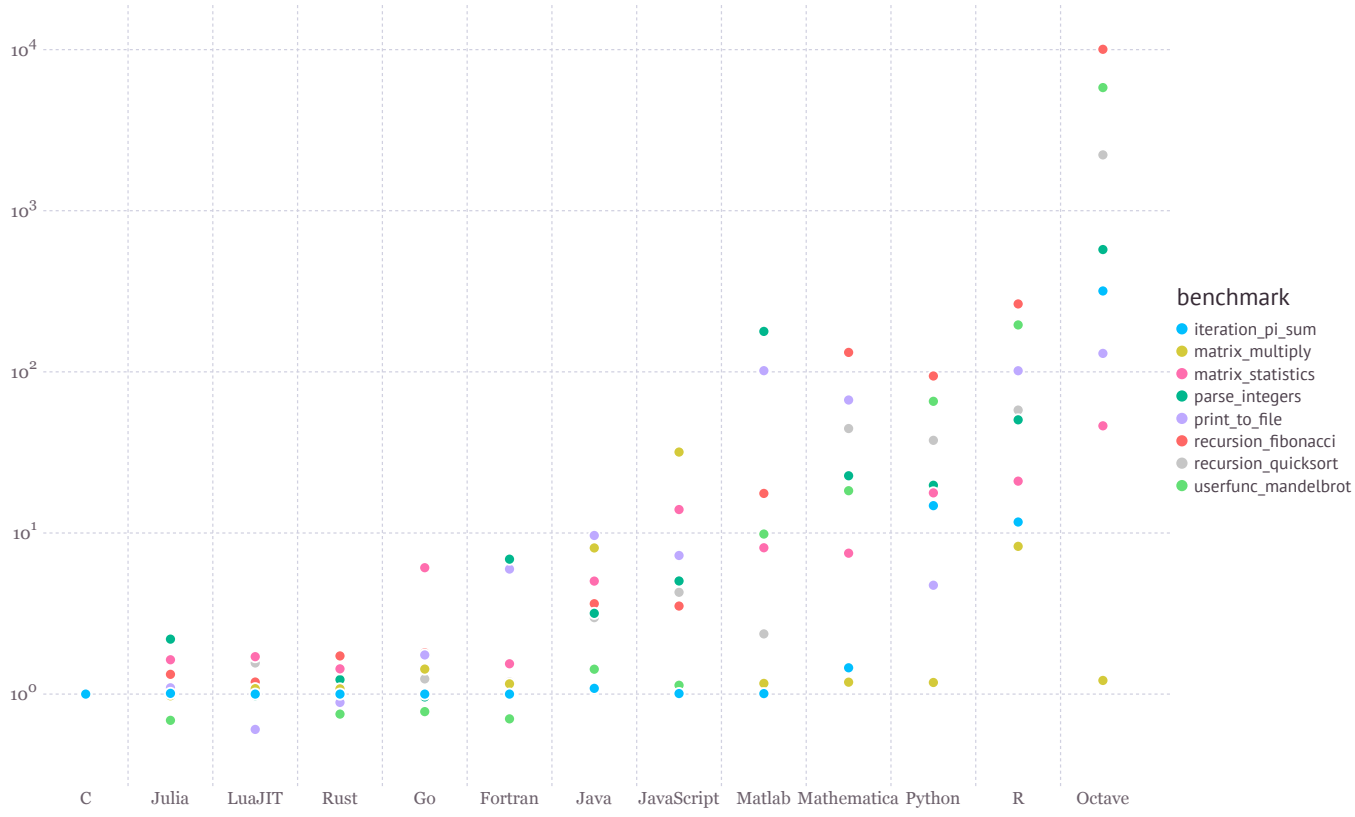

Figure: Source: `https://julialang.org/benchmarks/`

Why Julia?

- Why not Python/Cython/Pypy/Numba?

  https://jakevdp.github.io/blog/2014/05/09/why-python-is-slow/
  https://hackernoon.com/why-is-python-so-slow-e5074b6fe55b
  http://www.stochasticlifestyle.com/why-numba-and-cython-are-not-substitutes-for-julia/
  ...

Why Julia?

- It is easy to write.

```
julia> [1 1; 0 1] * [1 0; 1 1]
2×2 Array{Int64,2}:
2  1
1  1
```

Why Julia?

- Why not Matlab®?
- Julia is free and open source.
  `https://github.com/JuliaLang/julia`

Install

The latest version is v1.2.0. All the download link can be found at `https://julialang.org/downloads/`.

For Windows user:

- 32-bit: `https://julialang-s3.julialang.org/bin/winnt/x86/1.2/julia-1.2.0-win32.exe`
- 64-bit: `https://julialang-s3.julialang.org/bin/winnt/x64/1.2/julia-1.2.0-win64.exe`

Install

For macOS user:

- Download and install directly:
  `https://julialang-s3.julialang.org/bin/mac/x64/1.2/julia-1.2.0-mac64.dmg`
- via Homebrew

```
brew cask install julia
```

Install

For GNU/Linux user:

- Ubuntu/Debian etc with APT package manager(may old version)

```
$ sudo apt install julia
```

- Fedora/CentOS/RHEL with YUM/DNF package manager(may old version)

```
$ sudo yum install julia
$ sudo dnf install julia
```

- Pre-complied Binary
  Page: `https://julialang.org/downloads/`
  Link: `https://julialang-s3.julialang.org/bin/linux/x64/1.2/julia-1.2.0-linux-x86_64.tar.gz`

```
$ tar xvf julia-1.2.0-linux-x86_64.tar.gz
```

Work with Jupyter Notebook

Jupyter = [JU]lia [Pyt]hon and [R]. However, jupyter was famous because of Python. It is necessary to install Julia backend manually, in order to use the Julia with Jupyter Notebook. These steps require network connection.

1. Install Jupyter First
   - via PIP(Python Package Index):

     ```
     $ python3 -m pip install jupyter
     ```

     or

     ```
     $ pip3 install jupyter
     ```

     in case of some distribution with Python 2 and 3 installed at the same time
   - via Anaconda
2. Install IJulia, the backend for Jupyter.

   ```
   julia> using Pkg; Pkg.add("IJulia")
   ```

Basics

1. REPL

```
julia>
```

2. Try some elementary arithmetic

```
julia> 1 + (2 - 3) * 4 / 5
```

3. Variable

```
julia> a = sqrt(2)

julia> \alpha<Tab> = 3 * pi
```

Basics

- If statement

```
if (condition1)
    something()
elseif(condition2)
    run_code()
else
    another_thing()
end
```

Basics

- for statement

```
sum = 0
for i = 1:100
    sum = sum + i
end
```

- function definition

```
function func(args1, args2)
    do_some_things()
end
```

Basics

```
julia> using LinearAlgebra
julia> A = [1. 0. 0. 0. 2.; 0. 0. 3. 0. 0.;
0. 0. 0. 0. 0.; 0. 2. 0. 0. 0.]
4×5 Array{Float64,2}:
1.0  0.0  0.0  0.0  2.0
0.0  0.0  3.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0
0.0  2.0  0.0  0.0  0.0

julia> F = LinearAlgebra.svd(A);

julia> F.U * LinearAlgebra.Diagonal(F.S) * F.Vt
4×5 Array{Float64,2}:
1.0  0.0  0.0  0.0  2.0
0.0  0.0  3.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0
0.0  2.0  0.0  0.0  0.0
```

More Information

1. Official document:
   `https://docs.julialang.org/en/v1/index.html`
2. Learning material: `https://julialang.org/learning/`