UNIVERSITY OF Cincinnati

julia Workshop 2019

Jeff Mills, Lindner College of Business

- **Thanks to:**
- **The Carl H. Lindner College of Business Faculty Development Committee**
- **The Yung Family Foundation**
- **The Economics Department**
- **The UC Center for Business Analytics**
- **The Kautz-Uible Economics Institute**

Workshop Materials on Github:

- https://github.com/tszanalytics/Cincinnati_Julia_Workshop_2019

Install Julia

- Download the binary from https://julialang.org/downloads/.

- For Windows, download the Windows Self-Extracting Archive (.exe). It is recommended to select the 64-bit version.

- Add a Julia shortcut to your desktop and/or add Julia to your system path (see Julia_Intro_Workshop_2019.pdf, p. 4)

- Open the REPL (Read Evaluate Print Loop) aka the console and try things.
- https://pkg.julialang.org/docs/julia/THl1k/1.1.1/stdlib/REPL.html

- Julia> prompt = enter Julia commands
- ; = shell mode – issue operating system commands (ls, cd, etc.)
- ? = help mode – get help/info on Julia commands/functions
-] = package manager for adding, updating and removing packages. (<backspace> to exit this!)

Speeding up Julia for first package loading and first plot.

- Create **startup.jl** file to compile packages on startup,
- **or**
- **PackageCompiler.jl** to create precompile image for fast startup – follow instructions carefully
  - – (see Julia_Intro_Workshop_2019.pdf, p. 1).

IDEs/Editors

- Install package **IJulia** for notebooks, then notebook()
- `]add Ijulia; using Ijulia; notebook()`

- Try some things: Unicode, tab completion, plotting, …
- dice rolls: `rand(1:6)`   swap two numbers: `a, b = b, a`

- Atom/Juno: **Download and install Atom** from https://atom.io/.
- Add package uber-juno in atom, UNCHECK automatic update!

- VSCode is an alternative: https://github.com/julia-vscode/julia-vscode

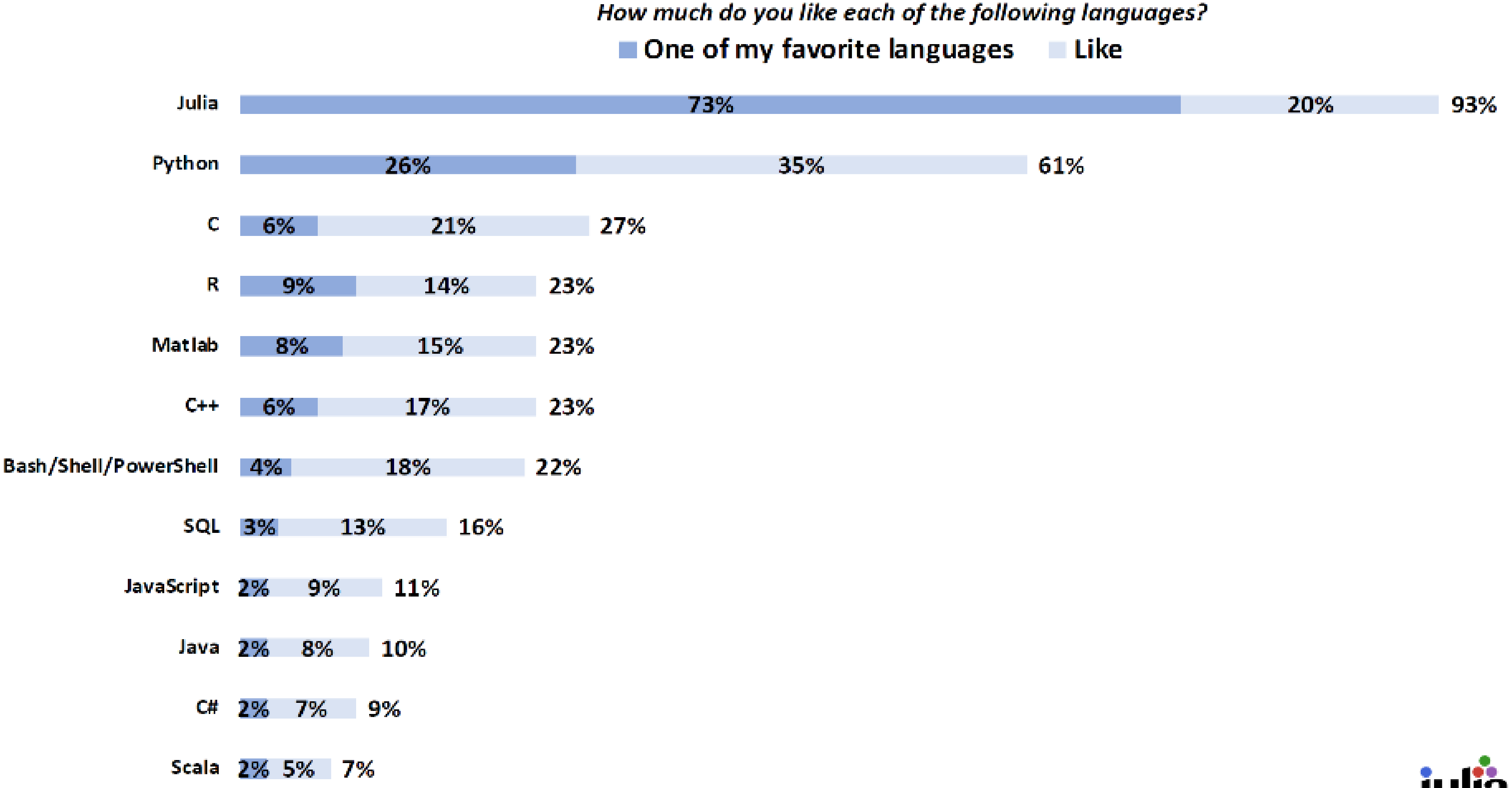
- You can use the REPL (console window) with Notepad, Notepad++ or any other editor of your choice.

User & Developer Survey 2019

Viral B. Shah
Andrew Claster
Abhijith Chandraprabhu

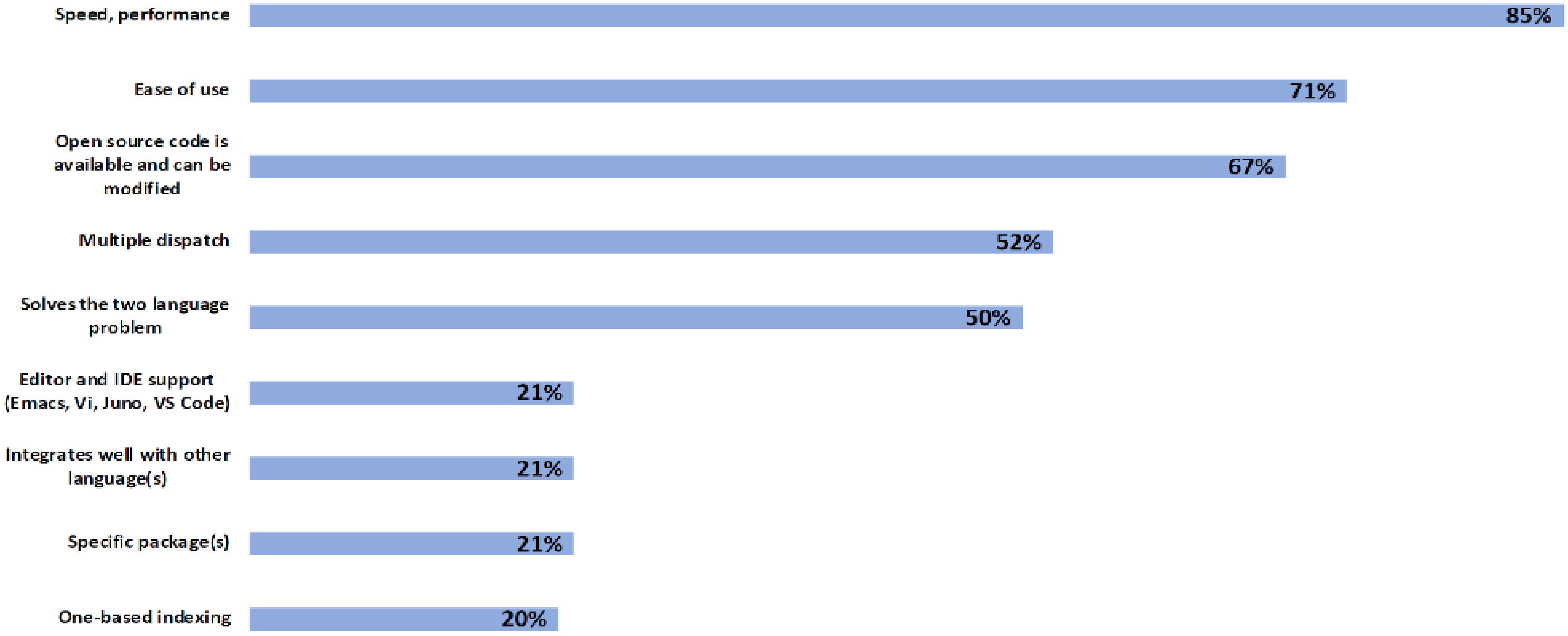
93% of Respondents Like Julia or Say Julia Is One of Their Favorite Languages

Python Comes Second Among Julia Users and Developers



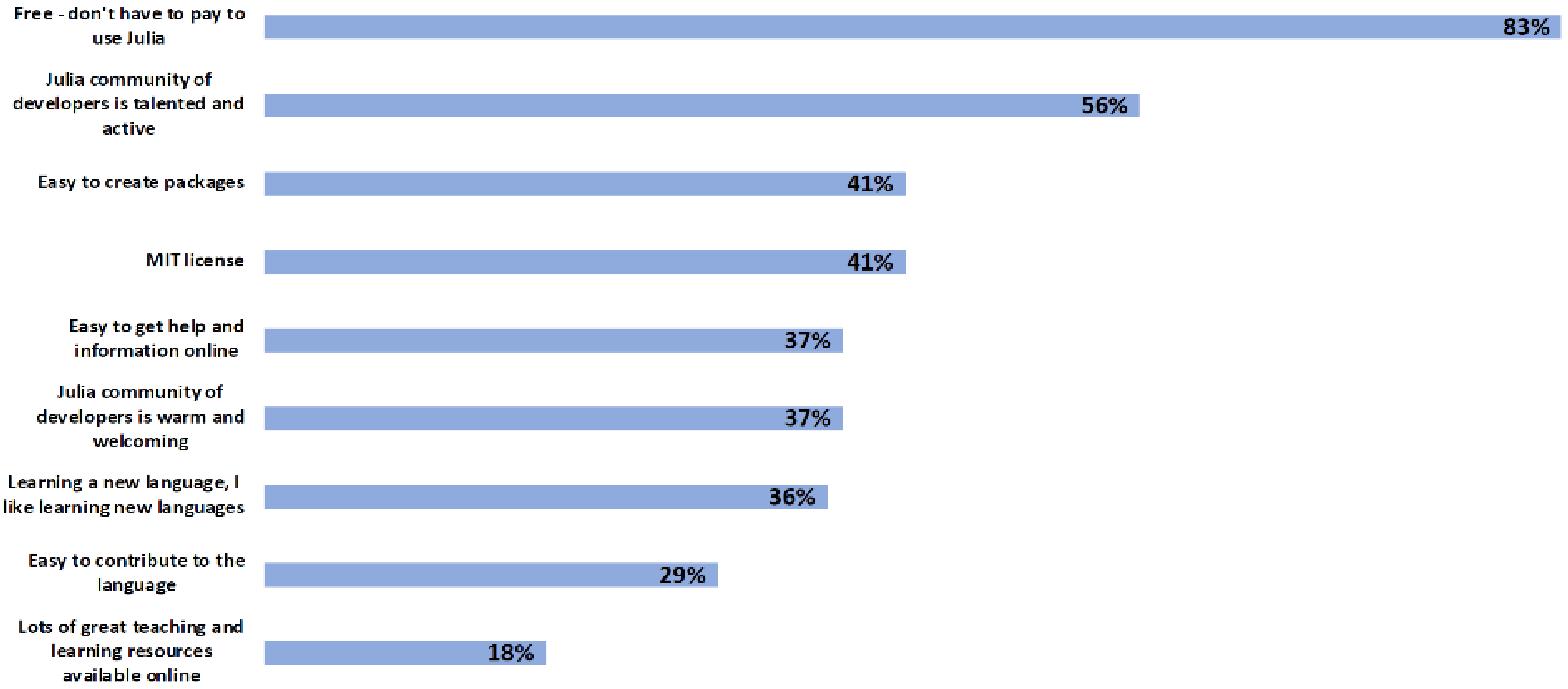
The MOST Popular TECHNICAL Features of Julia Are Speed/Performance, Ease of Use, Open Source, Multiple Dispatch and Solving the Two Language Problem

Thinking only about the TECHNICAL aspects or features of Julia, what are the TECHNICAL aspects or features you like MOST about Julia?



The MOST Popular NON-TECHNICAL Features of Julia Are Free (Don't Have to Pay) and Active and Talented Community of Julia Developers

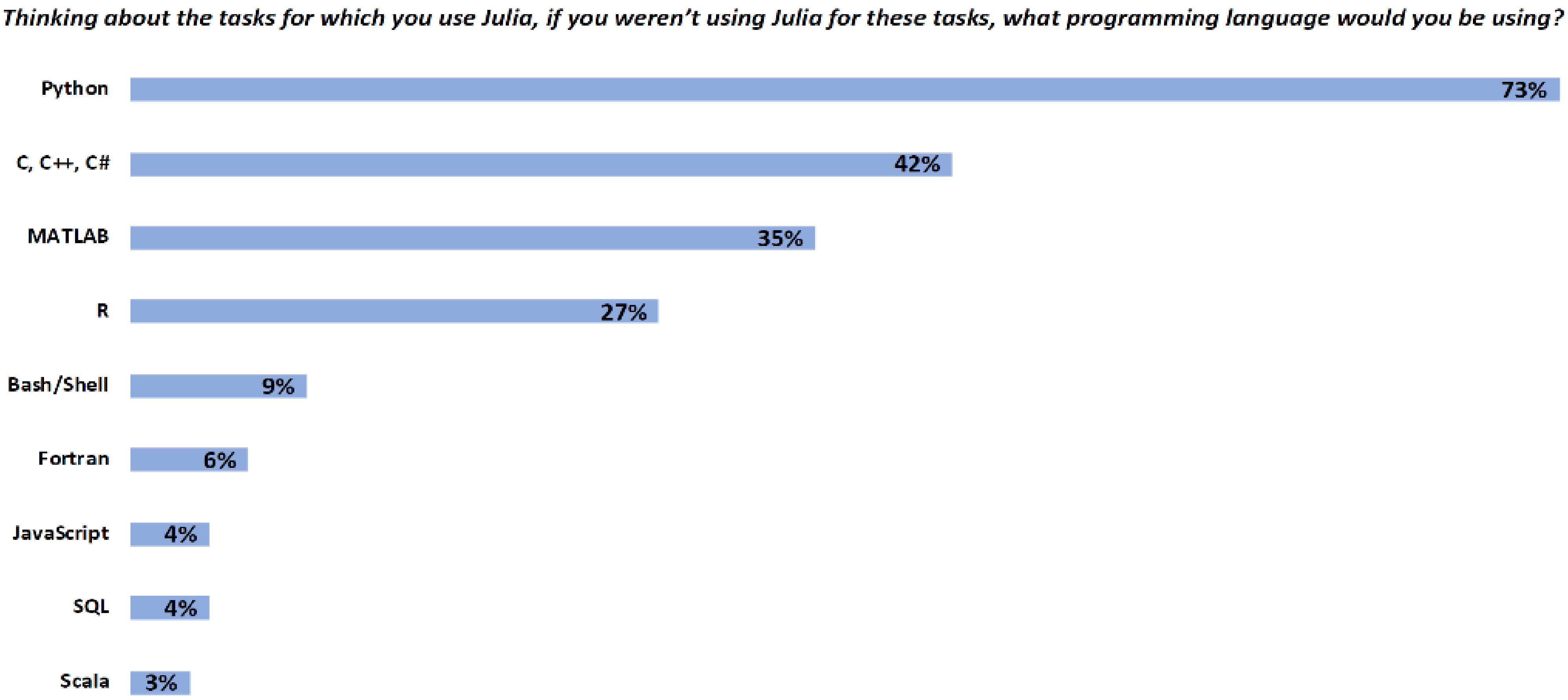
Thinking only about the NON-TECHNICAL aspects or features of Julia, what are the NON-TECHNICAL aspects or features you like MOST about Julia?



julia

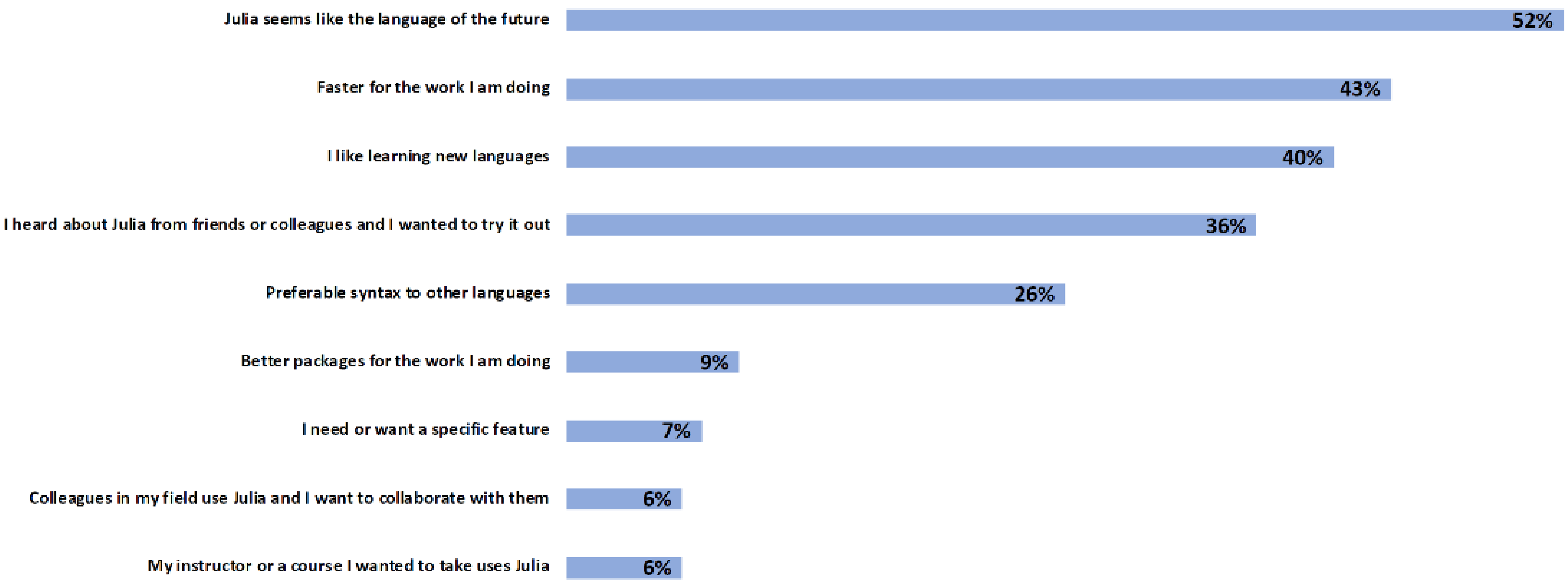
Julia computing

If Not for Julia, Most Would Be Using Python, Followed by C/C++/C#, MATLAB and R

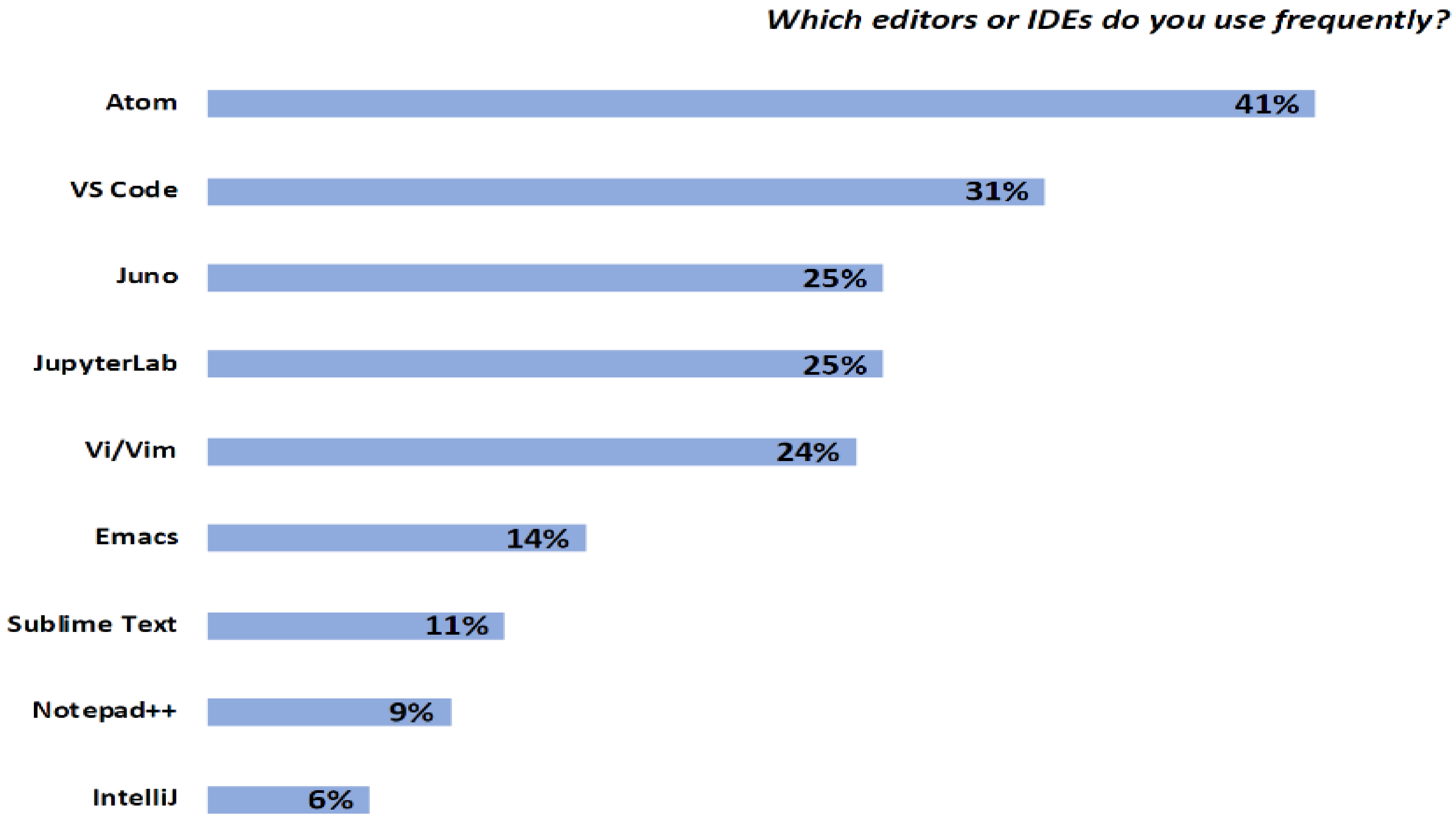


Respondents Started Using Julia Because of Speed and Because Julia Seems Like the Language of the Future

Why did you start using Julia?

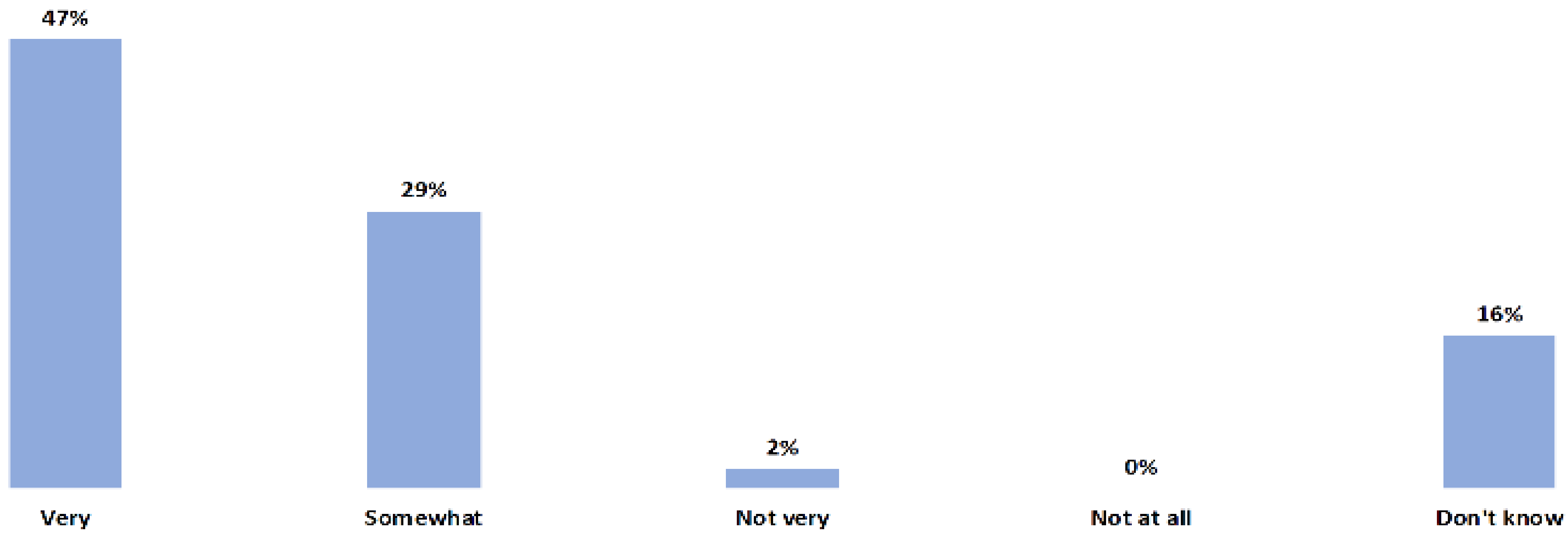


Atom and VS Code Are the Most Popular Editors or IDEs

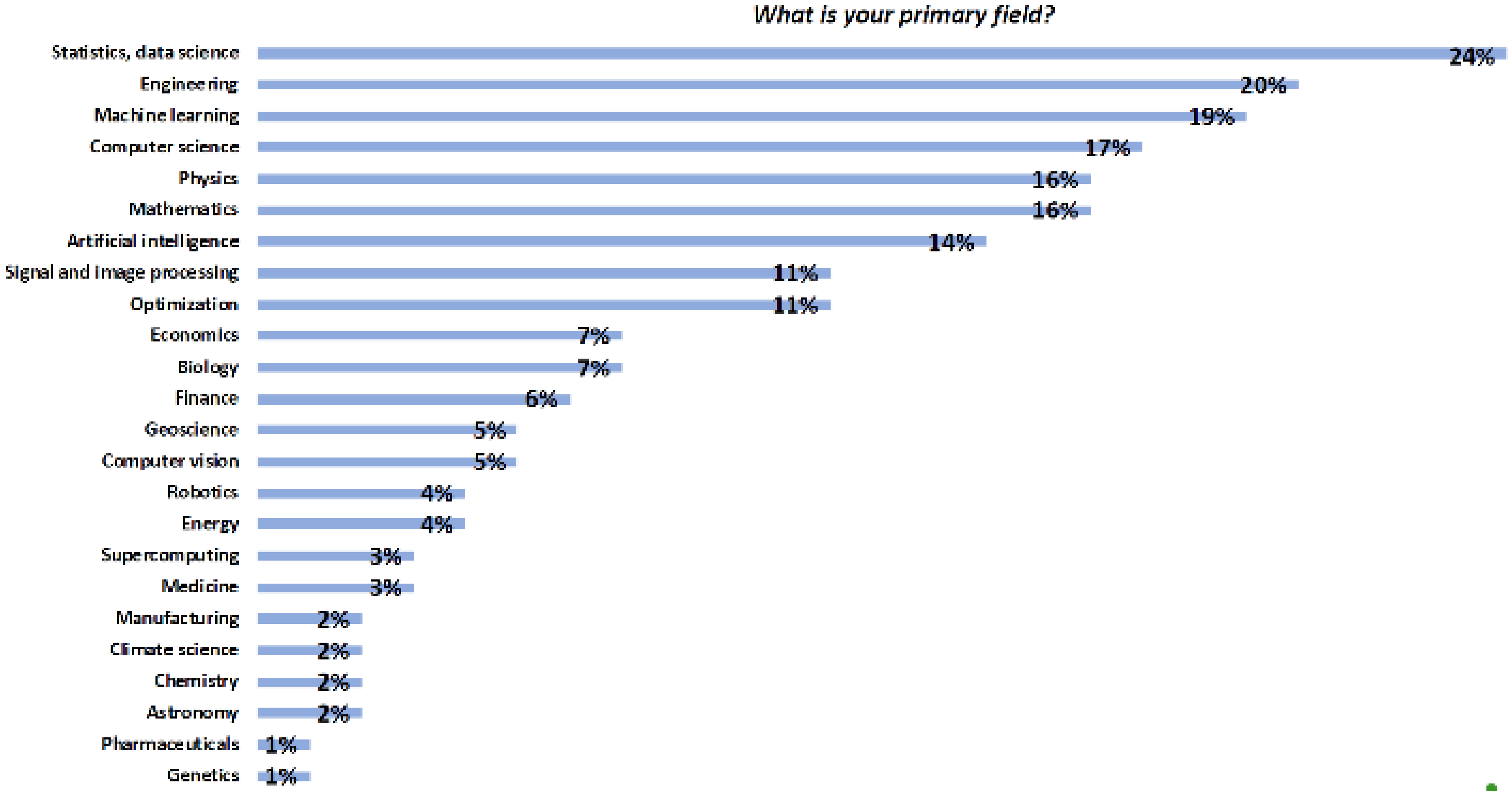


Most Say the Julia Community Is ‘Very’ or ‘Somewhat’ Helpful and Collaborative

How helpful and collaborative is the Julia community?



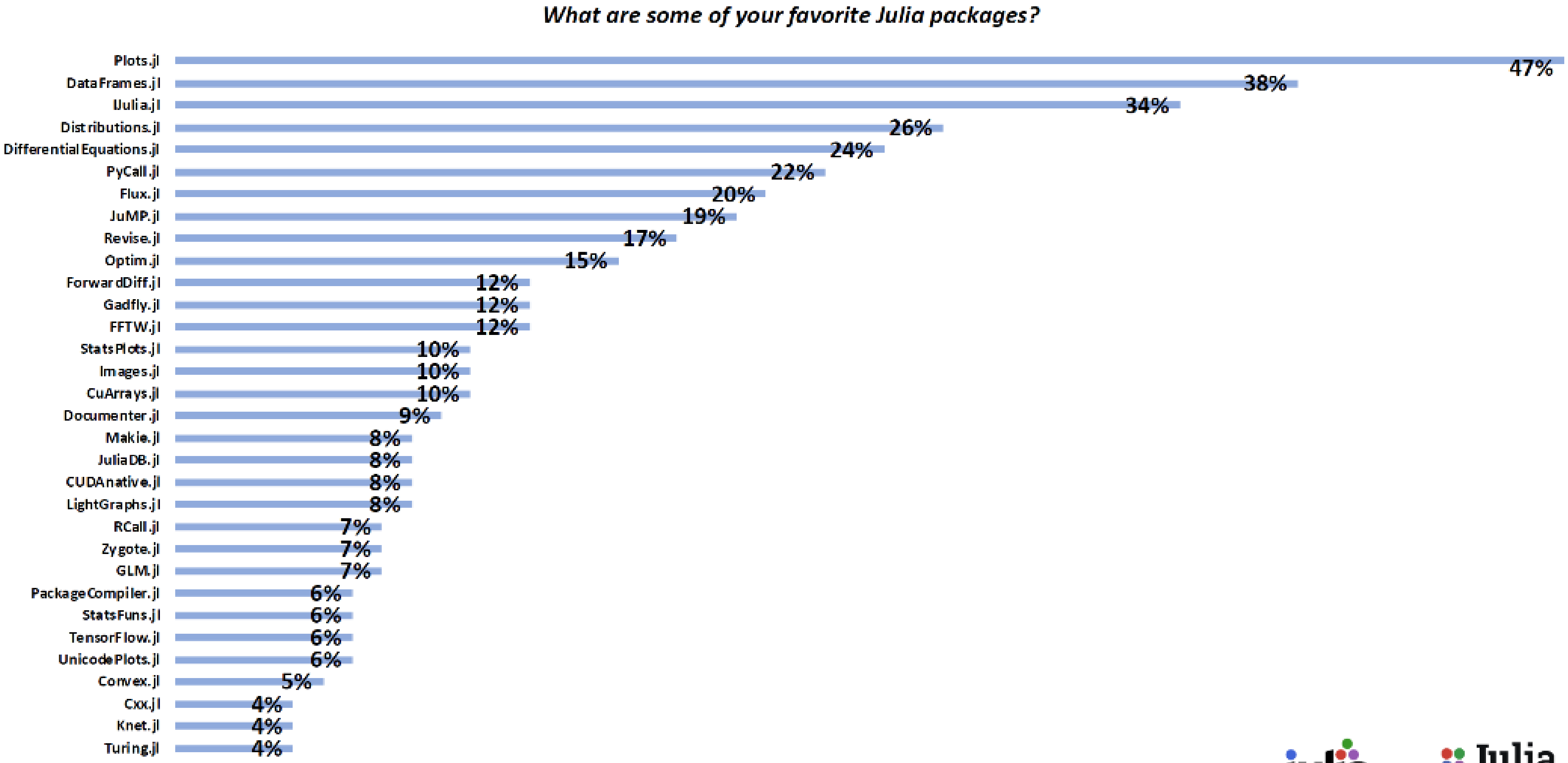
The Most Popular Fields Are Statistics, Data Science, Engineering, Machine Learning, Computer Science, Physics, Mathematics, Artificial Intelligence, Optimization and Signal and Image Processing



julia

Julia computing

The Most Popular Julia Packages Are Plots.jl, DataFrames.jl and IJulia.jl



julia

Julia computing

Some examples

```
# UNICODE characters allowed
s="abπ∑ef\n“
print(s)
ß=2π/3

Dice rolls  ∆ = rand(1:6)

# Complex floating-point numbers
x = 2.1 + 3.2im

# SWAP TWO NUMBERS: Don't need a swap macro
a,b = b,a
```

```
# checking approx. equality function:
isapprox(3.0, 3.01, rtol=0.1)

# COMPARISON OPERATORS
a = 2b = 3c = 3

# the AND operator
@show(a < c && b < c);

# the OR operator
@show(a < c || b < c);

# NOT equal
@show(a != b);
```

# GETTING USER RESPONSE TO TEXT

```
println("Who are you?")
s=readline()
println("Hello $s and Hello World!")

# Works much better in a function:
function whoru()
    println("Who are you?")
    s=readline()
    println("Hello $s and Hello World!")
End
```

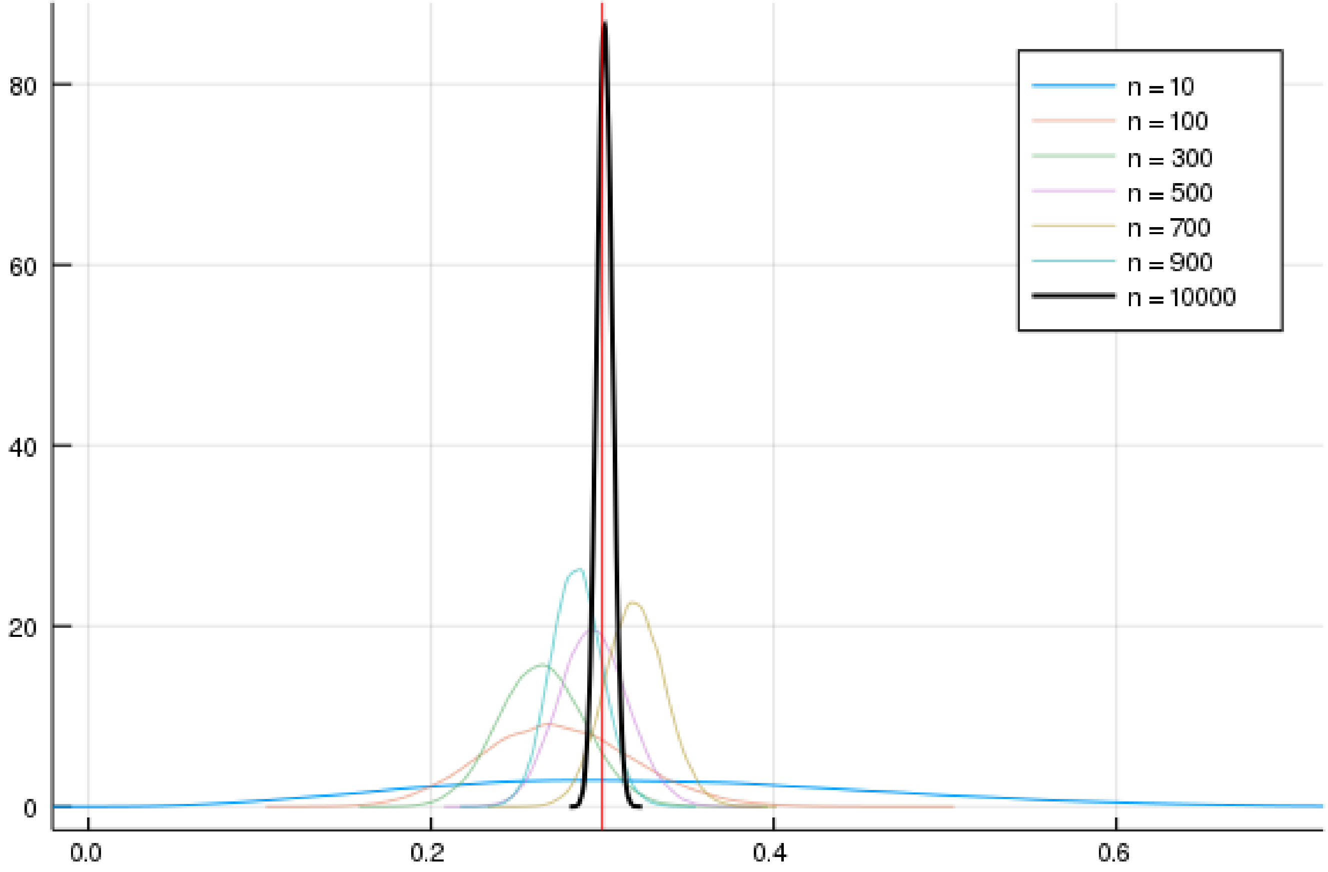
PRO TIP: put EVERYTHING in functions for greatly improved performance

An Example: Probabilistic Statistics

- Simplest case: s successes in n, with unknown probability of success, θ.
- The posterior distribution represents uncertain knowledge about the unknown constant.
- E.g., suppose true unknown θ = 0.3 and we obtain n observations.
- `θ = 0.3; Random.seed!(41); x = rand(Bernoulli(θ),10)`
- We know that p(θ|s,n) = `Beta(s+a, n−s+b), with a=b=1` for a uniform prior.
- Draw 10^6 values from the posterior density and plot.
- Compute the mean, std, 0.99 and 0.95 probability interval.
- Where is 0.5 (a “fair coin”, i.e. random chance)?
- Test the fairness hypothesis, $H_0\colon \theta = 0.5$

Uncertain knowledge about an unknown proportion

- The posterior distribution represents uncertain knowledge about the unknown constant, θ.
- Data generating process:
- `x = rand(Bernoulli(θ),n)`
- As $n \rightarrow \infty$, our knowledge about the unknown parameter approaches certainty.

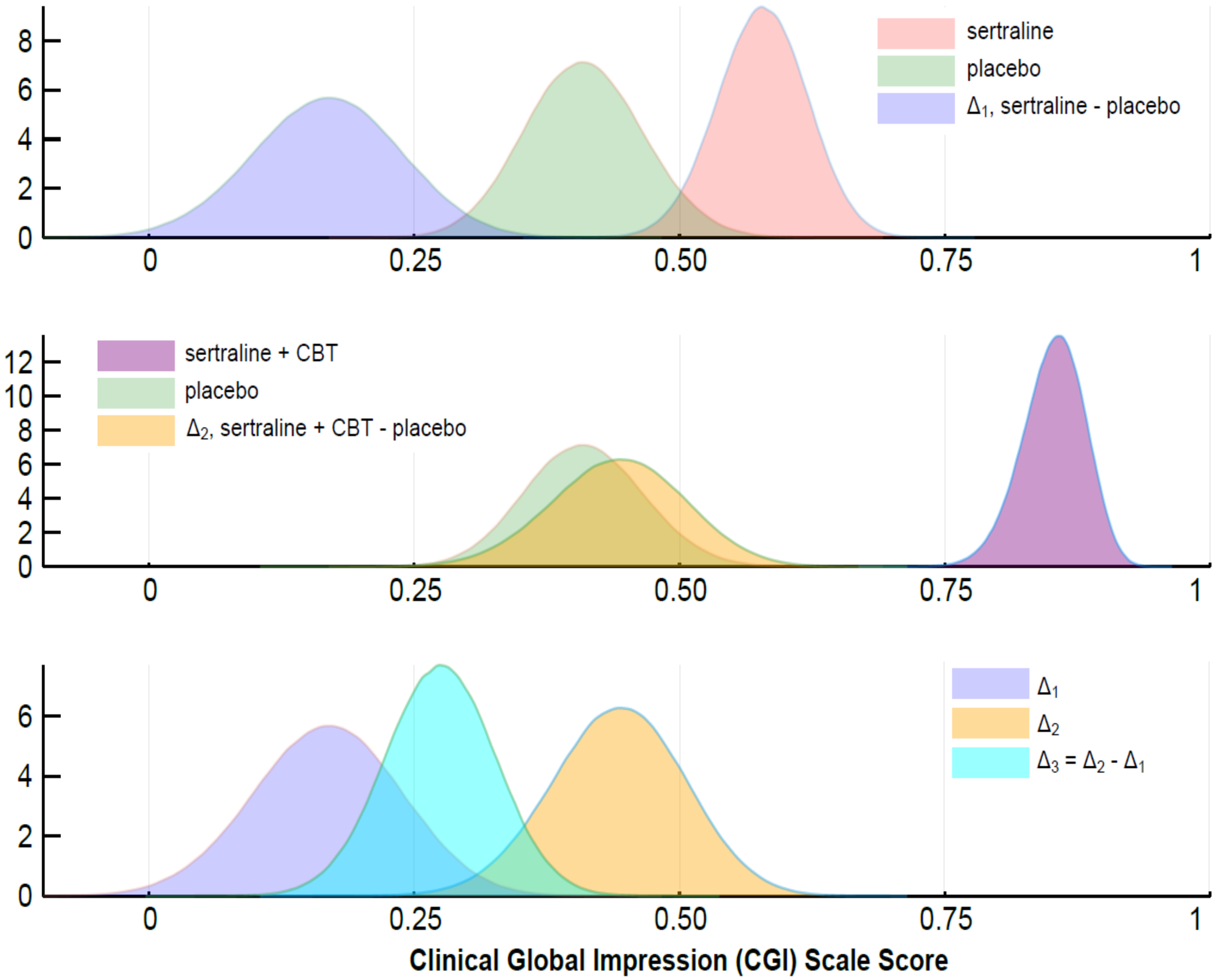


- For $n = 10$, $x =$ [0, 1, 0, 1, 0, 0, 0, 1, 0, 1], unknown $\theta = 0.3$

Pseudo-random draws from the Beta posterior given s successes in n patients for each group.

$$\Delta_1 = \theta_{T1} - \theta_{P1}$$
$$\Delta_2 = \theta_{T2} - \theta_{P2}$$
$$\Delta_3 = \Delta_2 - \Delta_1$$

An analytically intractable problem, but just a few lines of code.



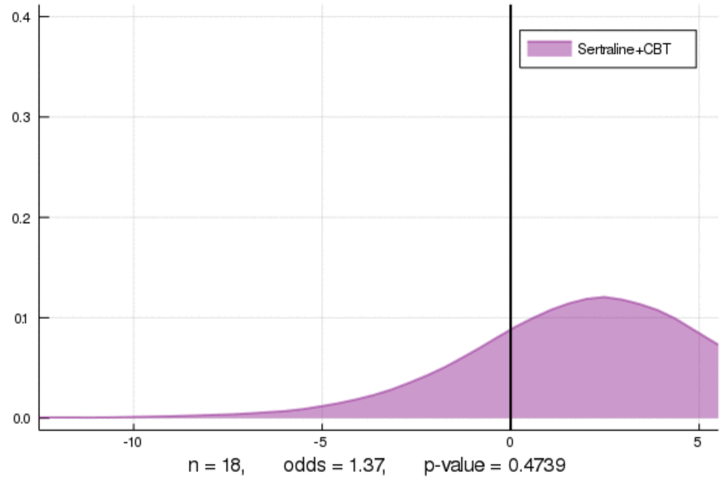
The github repository:

https://github.com/tszanalytics/Cincinnati_Julia_Workshop_2019

Some other repos you might find interesting:

https://github.com/tszanalytics/Juliacon2019

https://github.com/tszanalytics/BayesTesting.jl


Sertraline+CBT
n = 18,   odds = 1.37,   p-value = 0.4739