

# Interop

October 24, 2016

## 1 Interop Project

Please choose a project and a form of interop that appeals to you and your background.

### 1.1 Project 1: Interfacing with your own scripts

Choose a script from your own work. Modify it so that way it requires and input, and spits out a sensible output. Use Julia's interop functionalities to call this script and retrieve the answer.

### 1.2 Project 2: Wrapped Least Squares

Use Julia's interop capabilities to solve the least squares problem using your language of choice and retrieve the answer back to Julia.

### 1.3 Tools

These are mature interop libraries provided by Julia:

**ccall** The documentation page for `ccall` is: <http://docs.julialang.org/en/release-0.4/manual/calling-c-and-fortran-code/>. Since I found that slightly opaque on my first read, I wrote a tutorial for using `ccall` to call compile your C-code for usage with `ccall` and writing the wrapper code: <http://www.stochasticlifestyle.com/using-julias-c-interface-utilize-c-libraries/>

**Xeon Phi** While not technically supported, note that you can use `ccall` to control Xeon Phi acceleration cards: <http://www.stochasticlifestyle.com/interfacing-xeon-phi-via-julia/>

**Cxx.jl** <https://github.com/Keno/Cxx.jl>

**RCall** <https://github.com/JuliaInterop/RCall.jl>  
[https://github.com/joshday/Talks/blob/master/SLG2016\\_IntroToJulia/Slides.ipynb](https://github.com/joshday/Talks/blob/master/SLG2016_IntroToJulia/Slides.ipynb)

**PyCall** <https://github.com/JuliaPy/PyCall.jl>  
[https://github.com/joshday/Talks/blob/master/SLG2016\\_IntroToJulia/Slides.ipynb](https://github.com/joshday/Talks/blob/master/SLG2016_IntroToJulia/Slides.ipynb)

**MATLAB.jl** <https://github.com/JuliaInterop/MATLAB.jl>

**CUDArt.jl** <http://www.stochasticlifestyle.com/julia-on-the-hpc-with-gpus/>  
<http://www.stochasticlifestyle.com/multiple-gpu-on-the-hpc-with-julia/>