Julia tutorial

- Introduction
- Some useful pointers
- Getting started
- Julia syntax
- Plots in Julia
- Learning JuMP
- Submitting a notebook

Laurent Lessard (www.laurentlessard.com)

Why this tutorial?

To give you the resources and tools necessary to learn Julia, IJulia, and JuMP quickly and efficiently.

- Most of the learning will happen on your own as you work on homework assignments and the project
- The goal of this tutorial is to make that learning easy
- This tutorial was written on 1/24/2017. Current versions: Julia: 0.5.0, IJulia: 1.4.1, JuMP: 0.15.0.
  Things change quickly!

What is Julia?

- A modern programming language developed for scientific computing. Created in 2012 by a group of MIT students.
- **features**
  - designed for scientific computing but with the functionality of a modern object-oriented programming languages
  - simple efficient syntax similar to Matlab
  - dynamic language with speed comparable to statically compiled languages (e.g. C)
  - designed for parallelism and distributed computation
  - can call Python and C functions directly
  - IJulia notebooks
  - free and open-source

What is Julia?

- **disadvantages**
  - use is not widespread. Engineering community uses mostly Matlab, data science community uses mostly Python or R.
  - popular in classrooms at many schools, unclear whether it will become more broadly adopted. Time will tell.
  - does not have the same level of support as other languages. Julia community is small (but very active!)
  - weak toolboxes and packages compared to other languages. If you want something, you might have to code it yourself!
  - no good IDE (Juno is buggy and still crashes a lot), which is essential for data science applications.
  - plotting still has a long way to go.

Some useful pointers

Installation instructions for Julia + IJulia + JuMP along with a short tutorial: check out this Google Doc.

Help with Julia

- Noteworthy differences between Julia and other languages: http://docs.julialang.org/en/stable/manual/noteworthy-differences/
- Useful tutorial "learn X in Y minutes": https://learnxinyminutes.com/docs/julia/
- Julia manual: http://docs.julialang.org/en/stable/
- ? (help mode) in Julia
- Piazza + GIYF

Some useful pointers

Help with IJulia notebooks

- IJulia command list: in command mode (`ESC`), press `h`.
- Markdown reference:
  https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet
- LaTeX (pronounced lay-tech) cheat sheet:
  http://users.dickinson.edu/~richesod/latex/latexcheatsheet.pdf
  detexify: http://detexify.kirelabs.org/classify.html

Help with JuMP

- JuMP manual: https://jump.readthedocs.org/en/latest/
- more JuMP examples: check the class website!

Managing modules

- `Pkg.add("...")`: download and install a new package
- `Pkg.status()`: list of installed packages
- `Pkg.update()`: self-explanatory

Some extra tips:

- error messages involving packages are often informative. Read them! e.g. "you must run `Pkg.build("JuMP")`".
- any time a package is updated, the first time you try `using` it, Julia will update its cache, which may take 10–30 sec. This only happens the first time.
- if a successfully installed package produces an error when you try `using` it, try restarting the kernel.

Getting started

To start a new notebook, at the Julia prompt type:

```
using IJulia
notebook()
```

or, run `jupiter notebook` from a command prompt.

The basics

- `ESC` and `ENTER` switches between command/edit modes.
- to execute a block, select it and hit `SHIFT+ENTER`.
- for a list of commands, hit `h` in command mode.
- to open/edit a notebook, browse to it in the IJulia interface. To view a notebook, you can use http://nbviewer.jupyter.org/

Julia basics

Follow along with: https://learnxinyminutes.com/docs/julia/
Key items (sections 1–4 in the document above):

- Comments: `#` and `#= ... =#`.
- Types (Int, Float, String, Symbol), the `typeof` command
- Unicode, e.g. `\pi[TAB]` (I'll probably regret this...)
  http://docs.julialang.org/en/stable/manual/unicode-input/
- Ranges, arrays, and indexing. Stack operations.
- Tuples, dictionaries, and sets.
- Flow control (loops)
- Functions. `fun(...)` vs `fun!(...)` vs `@fun(...)`

Plots in Julia

Three methods currently available that I recommend:

1. **The `PyPlot` package**

- Identical to Python's `matplotlib` library.
- Plotting syntax similar to matlab.
- Static plot or dynamic windowed plot via `pygui(true)`.
- Static plot uses png or svg via `PyPlot.svg(true)`.
- Resize your figure! `figure(figsize=(x,y))`

Basic usage: https://github.com/JuliaPy/PyPlot.jl
Examples: https://gist.github.com/gizmaa/7214002
matplotlib manual: http://matplotlib.org/

Plots in Julia

2. The `PlotlyJS` package

- Do not use the ordinarly `Plotly` package; it requires an account, user credentials, an internet connection... This package runs entirely locally and for free.
- Allows for dynamic plots inside your notebook!
- Syntax is annoying; learn by example!

Usage and examples: http://spencerlyon.com/PlotlyJS.jl/
More examples: https://plot.ly/julia/

Plots in Julia

3. The `Plots` package

- Will likely become the defacto plotting package for Julia.
- Can create animations, save as animated gif!
- It's a front end that can call other plotting libraries using a unified syntax. Of course, has its own syntax...
- Still quite buggy at this point.

Usage and examples: https://juliaplots.github.io/

Learning JuMP

- There are only a few JuMP commands. Use the manual: https://jump.readthedocs.org/en/latest/
- Most aspects of JuMP will be covered in class through examples and posted on the website. Note: files from last year that haven't been updated yet may return errors!
- If you're struggling with an error or with syntax, Piazza is probably your best bet.

Submitting a notebook

- For homeworks, you will submit a PDF file. Two ways:
  - In Jupyter, use File → Download as → "PDF via LaTeX". May require installing pandoc/latex. Disadvantages: it doesn't always handle graphics correctly and it won't render HTML properly. e.g. if you use HTML code in a text block.
  - Print the webpage directly to PDF. Results can vary wildly depending on browser and OS. A common error: graphics don't print properly or there are large white spaces in your final PDF. One fairly reliable approach: use Chrome browser, use the browser's print function, choose "Save as PDF" as the Destination, and pick a large paper size (e.g. A3). This will mitigate the effect of the large white spaces.
- For the final project, you will turn in your `.ipynb` notebook file along with any supplemental attachments called by the code (data files, images, etc.). Instructions to come later.

Final comments

- Go through the "learn X in Y minutes" tutorial yourself (sections 1–4). It's definitely worth it.
- For more complicated aspects we discussed, it's ok to "learn as you go". Just make sure to start your assignments early to give yourself time to debug, learn syntax, deal with random printing bugs, etc.
- Julia is rapidly evolving. If you find a better way to do something in Julia/IJulia/JuMP than how I've been teaching it, please let me know!