The julia programming language

A teaser

Gary Froyland and Bill McLean

What is julia ?

- Open-source, high-level (easy to program), high-performance language designed for mathematics, scientific computing, and data science.
- Created in 2012; see julialang.org and the wikipedia page for details.
- Has math-like syntax, including math symbols and Greek letters.
- Won the 2019 SIAM J. H. Wilkinson Prize for Numerical Software.

Today's goal: Demonstrate a variety of basic elements of Julia and provide a list of packages so that you can explore further if you wish.

Why should I learn julia ?

One can do pretty much anything in any language. Some important considerations are:

1. Time taken to write code.
2. Readability of code (e.g. can you still understand what you wrote in two years' time)?
3. Runtime of code.

- In this talk, don't think "can I do this with another language?", because you most likely can.
- Instead, pay attention to how elegantly, cleanly, and consistently things are done in julia.
- Pay attention to how math-like the syntax often is. Maybe you want your code to look almost like your paper?
- I am not giving up matlab. But every time I need to do something new, I'll ask myself "could I do this better* in julia?"

Open source and free: many of our graduates may end up working in organisations without matlab or other commercial software and need a powerful modern language that is both math-oriented and general.

A sample of packages – easily managed with julia's built-in package manager

- Included standard packages – LinearAlgebra, SparseArrays, Statistics, Random, …
- Interactive notebook - https://github.com/fonsp/Pluto.jl
- Symbolic calculus for students - https://docs.juliahub.com/CalculusWithJulia/AZHbv/0.0.13/
- Plots - http://docs.juliaplots.org/latest/, http://docs.juliaplots.org/latest/ecosystem/#ecosystem, http://docs.juliaplots.org/latest/backends/
- Differential/difference equations - https://diffeq.sciml.ai/stable/
- Optimisation - https://jump.dev/ (linear, mixed-integer, conic, semi-definite, nonlinear)
- Machine learning - https://fluxml.ai/ (various methods, all written in Julia, no hidden libraries)
- Data wrangling - https://dataframes.juliadata.org/stable/
- File input/output - https://csv.juliadata.org/stable/, https://docs.julialang.org/en/v1/stdlib/DelimitedFiles/, https://github.com/JuliaIO/MAT.jl
- Also algebra, biology, dynamics, ecology, graphs, oceanography, etc…
- **Master lists of packages** - https://julialang.org/packages/
- Differences to other languages: https://docs.julialang.org/en/v1/manual/noteworthy-differences/
- Brief unofficial page of basic code examples https://juliabyexample.helpmanual.io/