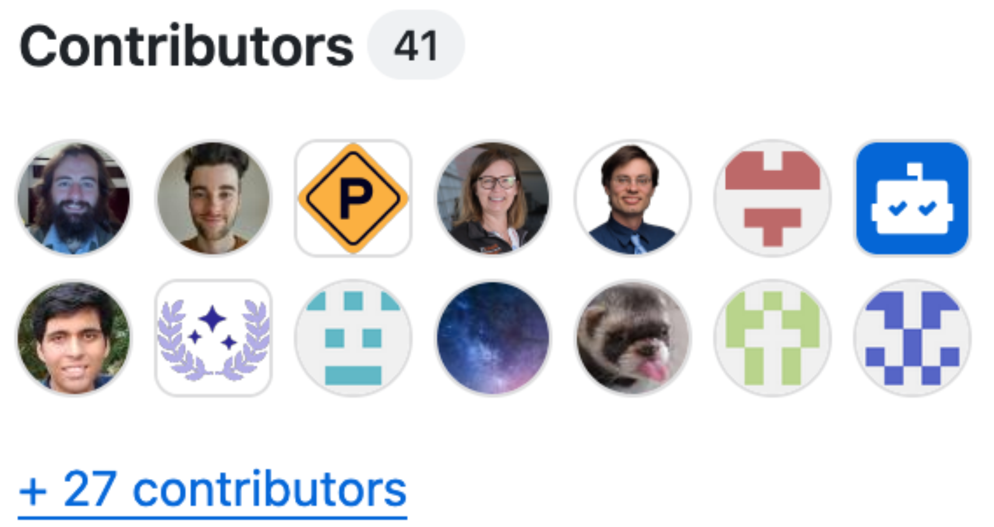
Power of Python and Julia

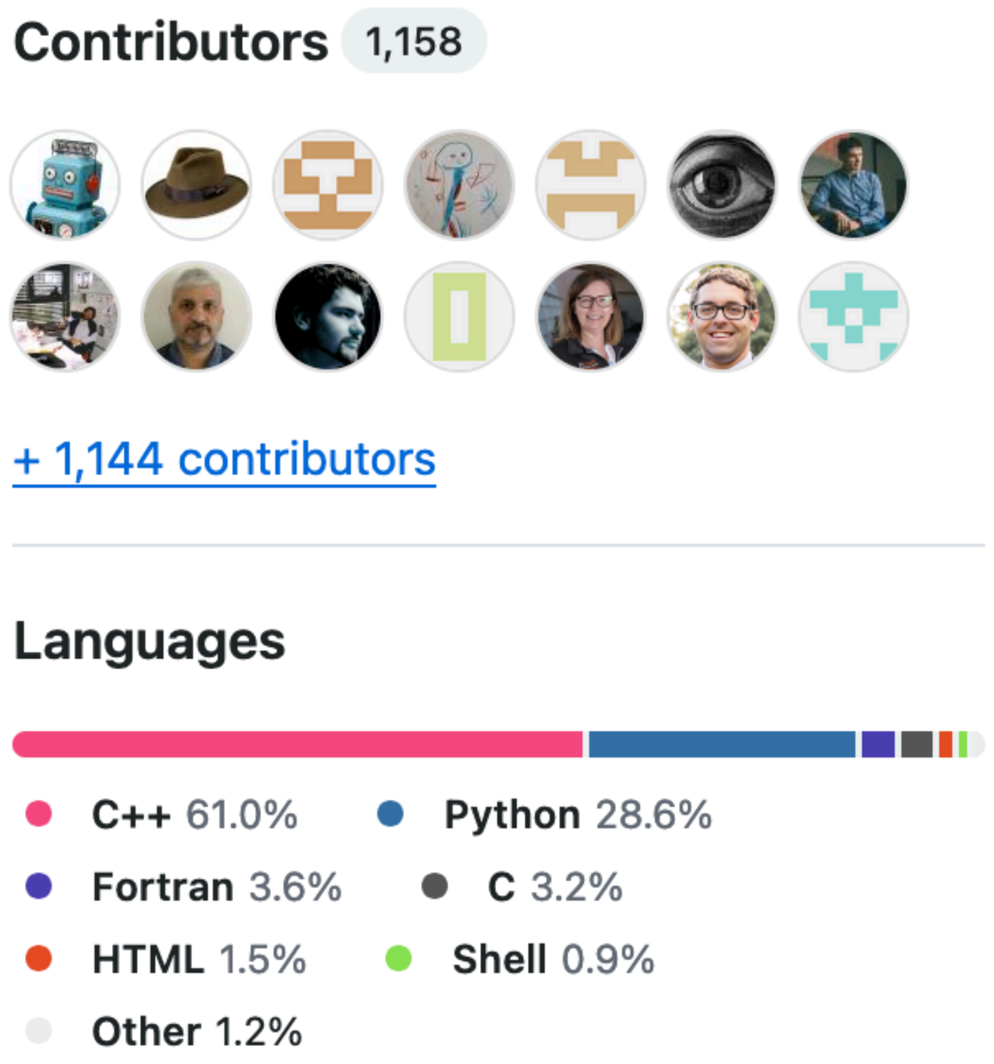
for Advanced Data Analysis

Ianna Osborne

Who I am

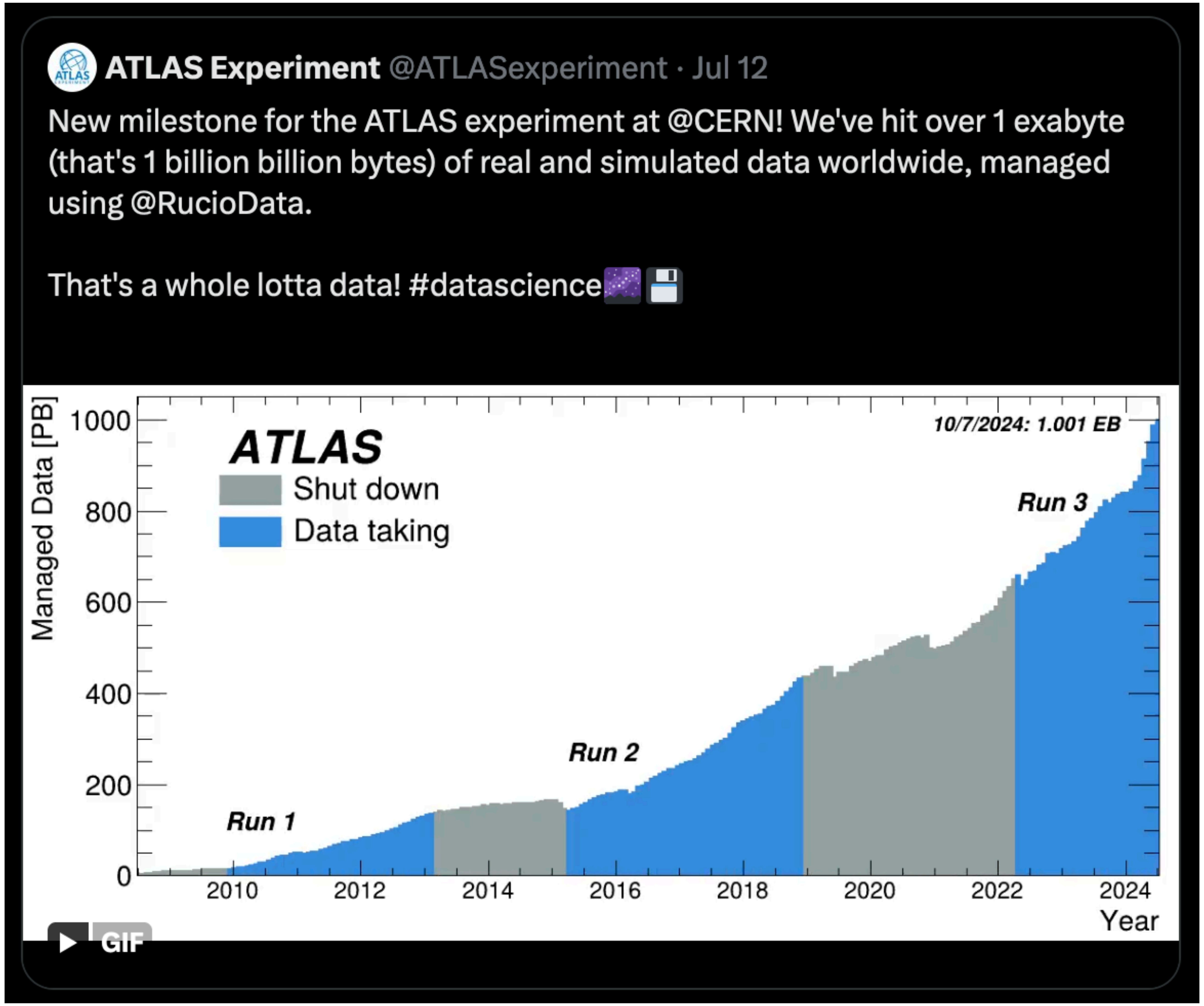
- Ianna Osborne (“ee-AN-uh”)
  - Research Software Engineer
  - Princeton and IRIS-HEP based at CERN
    - #3 contributor to AwkwardArray
- member of CMS experiment
  - #12 contributor to CMSSW (core software, geometry description, event display, simulation, etc.)
- background in Physics and Computer Science
- C++, Python, Julia
- Contact
  - linkedin.com/in/ianna-osborne-9982a342
  - ianna.osborne@cern.ch



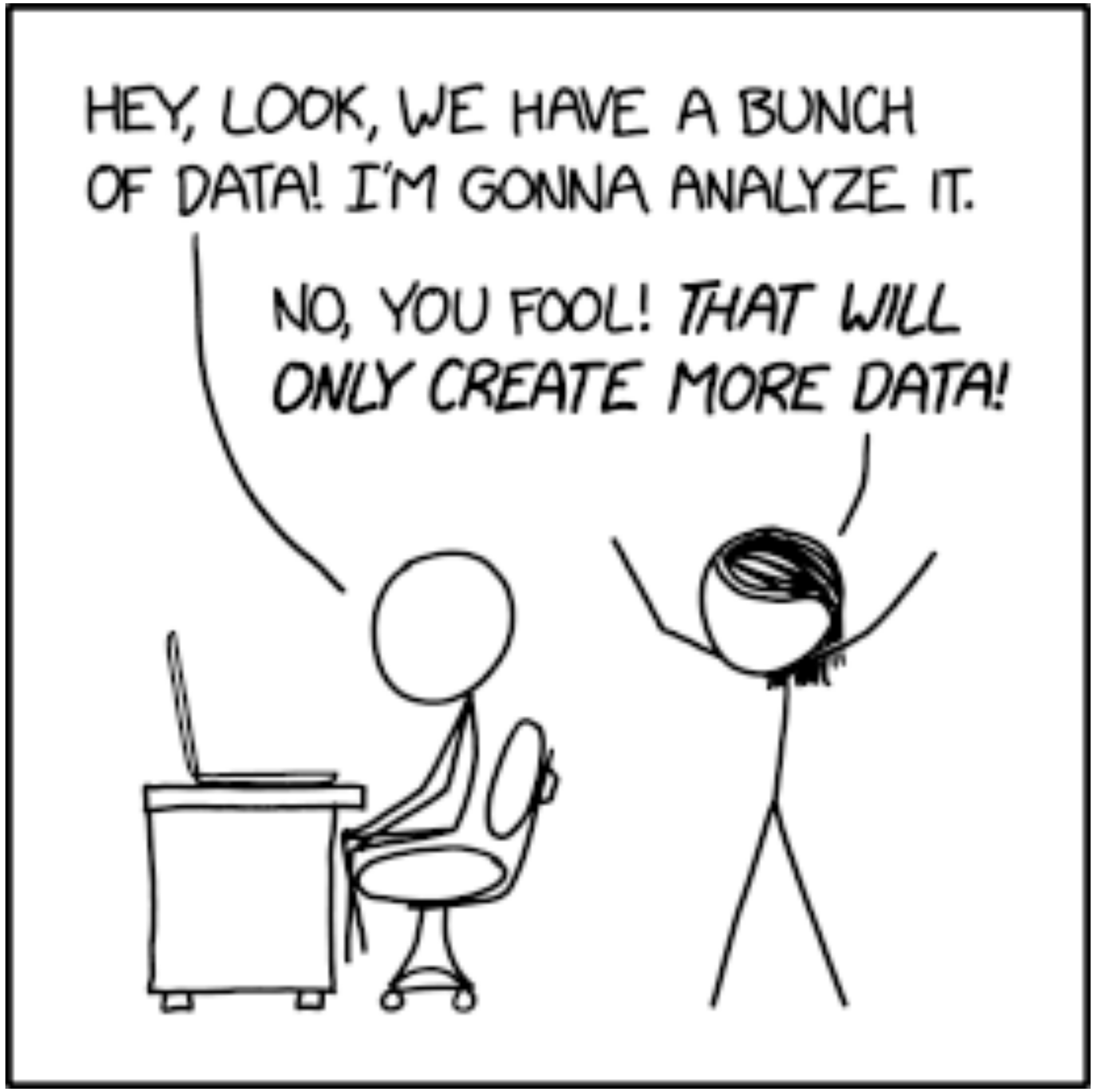


PRINCETON UNIVERSITY

Motivation


ATLAS Experiment @ATLASexperiment · Jul 12
New milestone for the ATLAS experiment at @CERN! We've hit over 1 exabyte (that's 1 billion billion bytes) of real and simulated data worldwide, managed using @RucioData.
That's a whole lotta data! #datascience
ATLAS
Shut down
Data taking
10/7/2024: 1.001 EB
Run 1
Run 2
Run 3
Managed Data [PB]
0
200
400
600
800
1000
2010
2012
2014
2016
2018
2020
2022
2024
Year
GIF

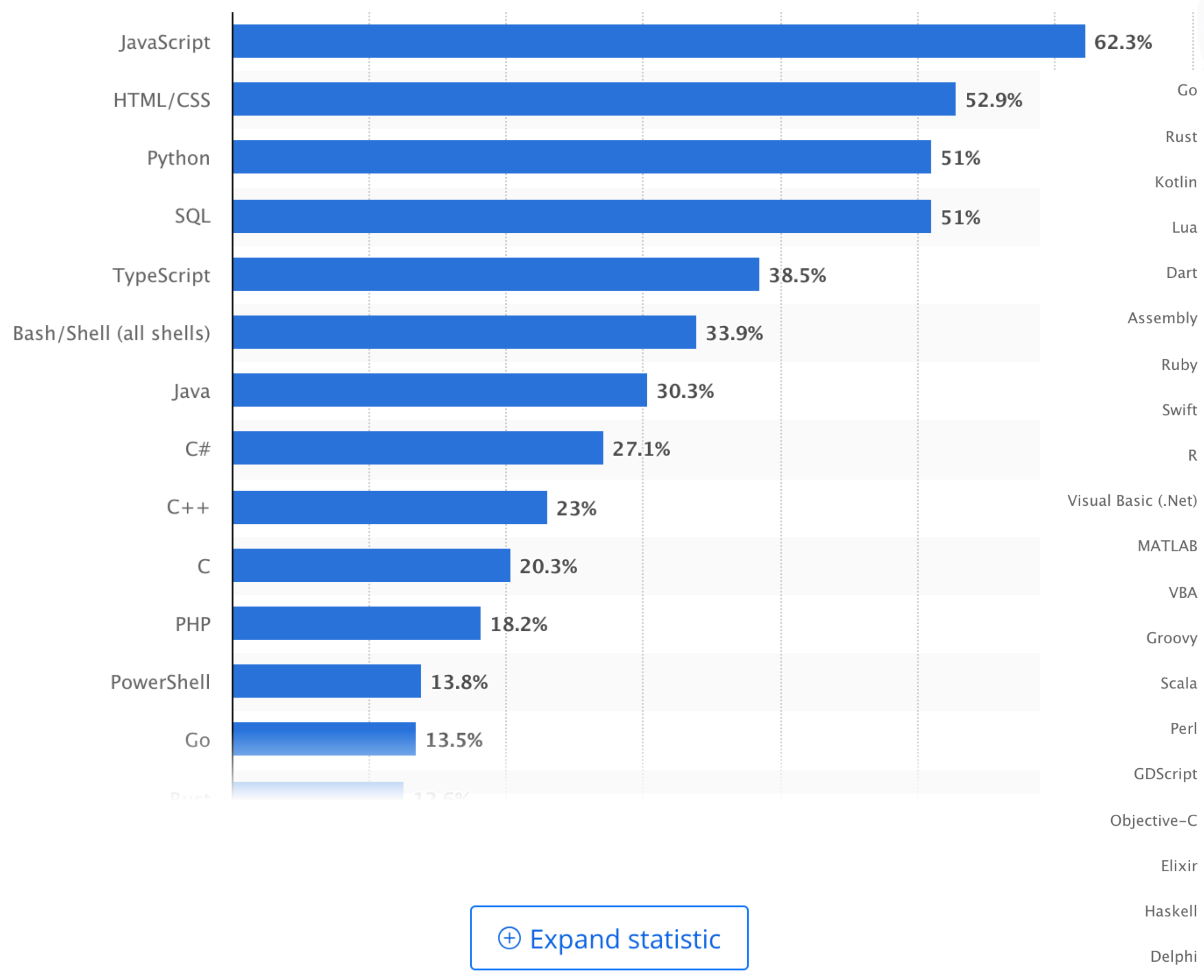
Motivation

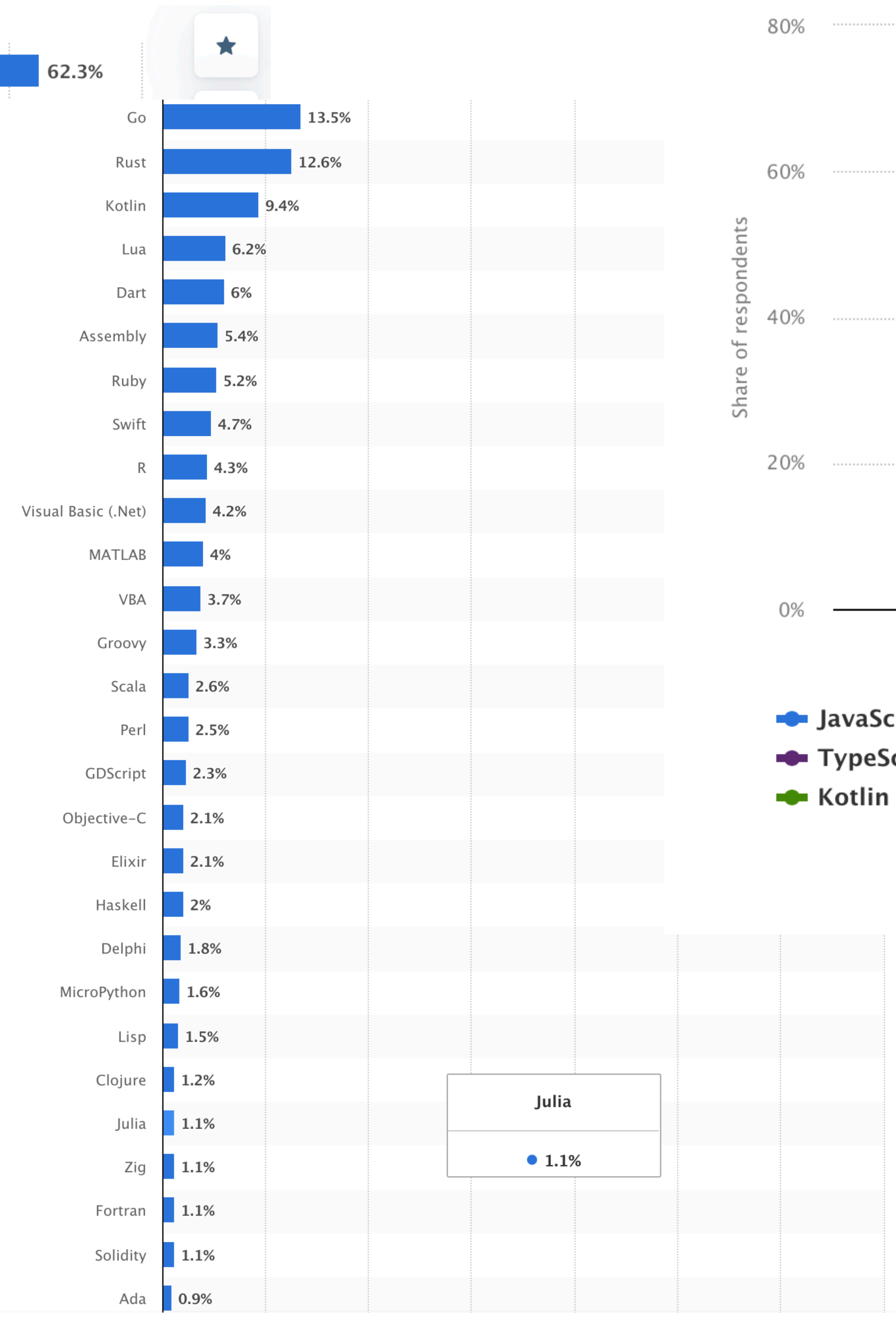


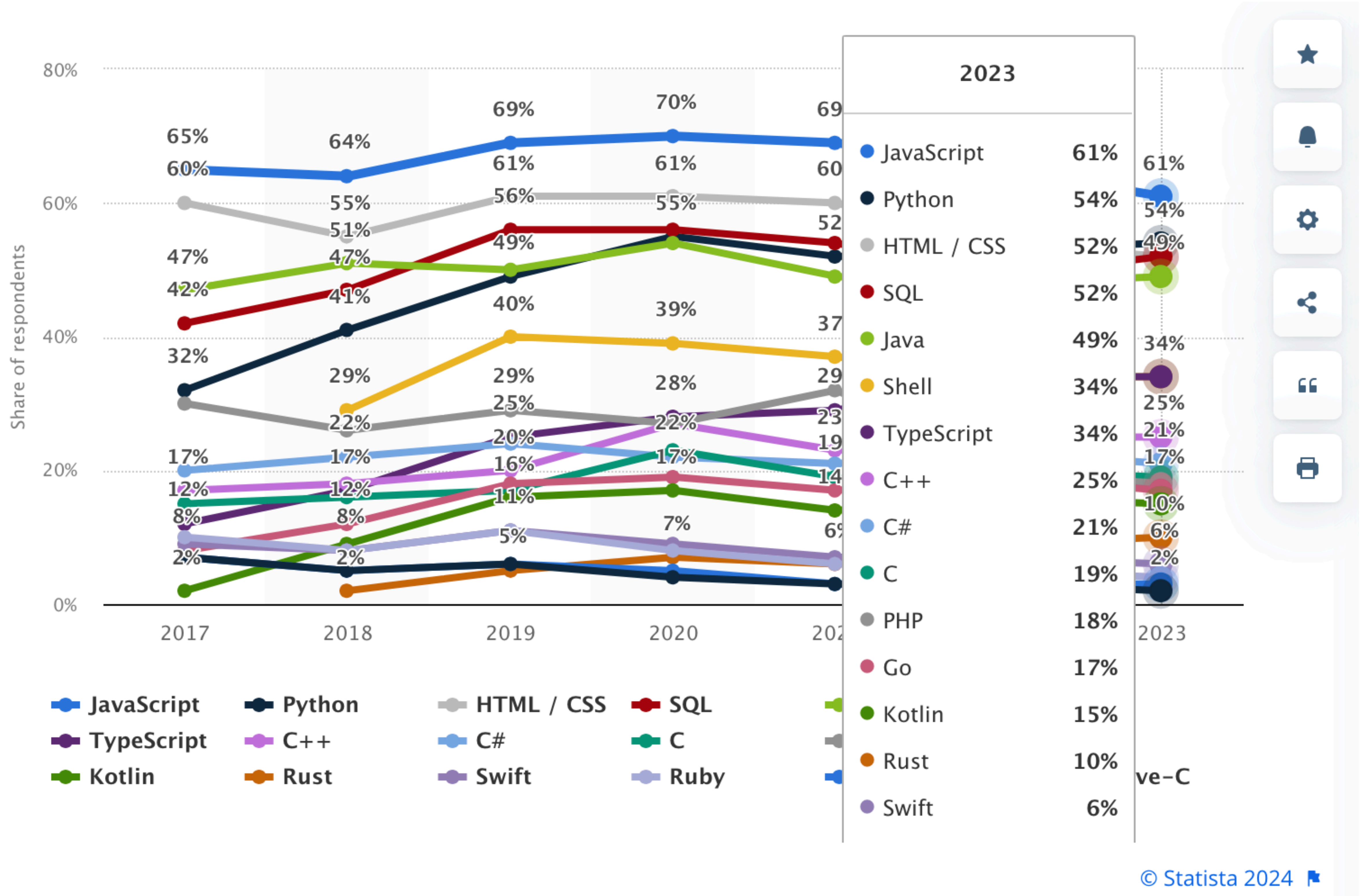
HEY, LOOK, WE HAVE A BUNCH OF DATA! I'M GONNA ANALYZE IT.
NO, YOU FOOL! THAT WILL ONLY CREATE MORE DATA!

Most Used Programming Languages

among developers worldwide as of 2024







- [TIOBE Index](#) for September 2024:
- Python #1 and Julia #31

Why Python is so popular

for data analysis

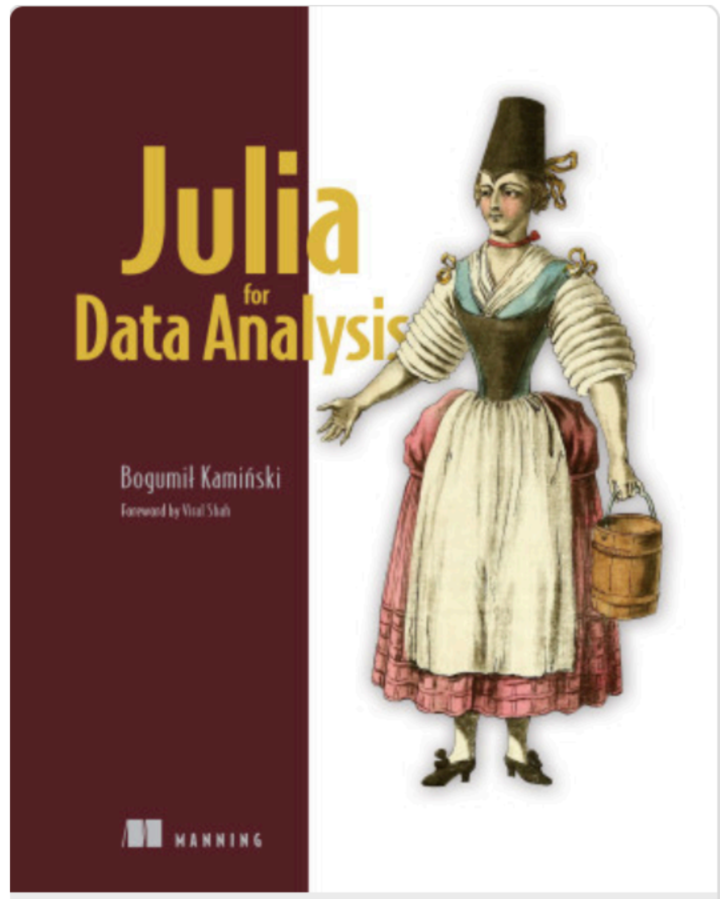
- Simple and easy to understand syntax
- Rich ecosystem of libraries tailored for data analysis
- Versatile, integrates well with other languages and tools
- Vast and active community
- Various programming paradigms support, including procedural, object-oriented, and functional programming
- Jupyter notebooks as an interactive environment where users can combine code execution, visualization, and narrative text
- Scalability and performance

CHEP2001: Python is a trend in HEP

Key features of Julia

from a data scientist's perspective

- Speed of code execution
- Designed for interactive use
- Composability, leading to highly reusable code that is easy to maintain
- Package management: built in state-of-the-art package manager
- Ease of integration with other languages



PRINCETON UNIVERSITY

iris hep

PythonCall and JuliaCall

Home | GitHub

PythonCall & JuliaCall

Search docs (Ctrl + /)

PythonCall & JuliaCall

Bringing **Python®** and **Julia** together in seamless harmony:

- Call Python code from Julia and Julia code from Python via a symmetric interface.
- Simple syntax, so the Python code looks like Python and the Julia code looks like Julia.
- Intuitive and flexible conversions between Julia and Python: anything can be converted, you are in control.
- Fast non-copying conversion of numeric arrays in either direction: modify Python arrays (e.g. `bytes`, `array.array`, `numpy.ndarray`) from Julia or Julia arrays from Python.
- Helpful wrappers: interpret Python sequences, dictionaries, arrays, dataframes and IO streams as their Julia counterparts, and vice versa.
- Beautiful stack-traces.
- Works anywhere: tested on Windows, MacOS and Linux, 32- and 64-bit, Julia Julia 1.6.1 upwards and Python 3.8 upwards.

Guide »

- PythonCall & JuliaCall allow to call Python code from Julia and Julia code from Python via a symmetric interface.
- Using PythonCall to bring Python functions and libraries into Julia
- Embed Julia code right into our Python scripts using JuliaCall

conda install conda-forge::pyjuliacall

JuliaPkg

What if Julia is not installed?

- JuliaCall relies on JuliaPkg to discover and install Julia
- JuliaCall uses JuliaUp (which uses the default depot) IF it is already installed - you need to install it manually. Otherwise JuliaCall will download and install Julia directly into your Python environment, but still using the default depot

In [2]:
```python
import awkward as ak
from juliacall import Main as jl

jl.seval("using AwkwardArray")
```

```
[juliapkg] Found dependencies: /usr/local/lib/python3.10/site-packages/awkward/juliapkg.jso
n
[juliapkg] Found dependencies: /usr/local/lib/python3.10/site-packages/juliacall/juliapkg.j
son
[juliapkg] Found dependencies: /usr/local/lib/python3.10/site-packages/juliapkg/juliapkg.js
on
[juliapkg] Locating Julia ~1.9, =1.10.0, ^1.10.3
[juliapkg] Querying Julia versions from https://julialang-s3.julialang.org/bin/versions.jso
n
[juliapkg] WARNING: About to install Julia 1.10.4 to /home/cms-jovyan/.julia/environments/p
yjuliapkg/pyjuliapkg/install.
[juliapkg]   If you use juliapkg in more than one environment, you are likely to
[juliapkg]   have Julia installed in multiple locations. It is recommended to
[juliapkg]   install JuliaUp (https://github.com/JuliaLang/juliaup) or Julia
[juliapkg]   (https://julialang.org/downloads) yourself.
[juliapkg] Downloading Julia from https://julialang-s3.julialang.org/bin/linux/x64/1.10/jul
ia-1.10.4-linux-x86_64.tar.gz
             download complete
[juliapkg] Verifying download
```

Best Practices for Managing Dependencies

and keeping everything running smoothly

- JuliaUp is a recommended way to install Julia

curl -fsSL https://install.julialang.org | sh

Important: pin Julia version!

Set or Pin a Julia Version as Default

Pin a Julia Version for a Specific Project

yana — -zsh — 115×36

```
Last login: Sun Aug 25 07:56:24 on ttys009
(base) yana@Iannas-MacBook-Pro-2 ~ %
```

yana — -zsh — 115×36

```
Last login: Sun Aug 25 07:56:24 on ttys009
(base) yana@Iannas-MacBook-Pro-2 ~ % curl -fsSL https://install.julialang.org | sh
```

yana — juliainstaller --juliaup-channel release — 115×36

```
Last login: Sun Aug 25 07:56:24 on ttys009
(base) yana@Iannas-MacBook-Pro-2 ~ % curl -fsSL https://install.julialang.org | sh

info: downloading installer
Welcome to Julia!

This will download and install the official Julia Language distribution
and its version manager Juliaup.

Juliaup will be installed into the Juliaup home directory, located at:

  /Users/yana/.juliaup

The julia, juliaup and other commands will be added to
Juliaup's bin directory, located at:

  /Users/yana/.juliaup/bin

This path will then be added to your PATH environment variable by
modifying the profile files located at:

  /Users/yana/.profile
  /Users/yana/.bash_profile
  /Users/yana/.zshrc

Julia will look for a new version of Juliaup itself every 1440 minutes when you start julia.

You can uninstall at any time with juliaup self uninstall and these
changes will be reverted.

? Do you want to install with these default configuration choices? ›
❯ Proceed with installation
  Customize installation
  Cancel installation
```

yana — -zsh — 115×36

```
Juliaup will be installed into the Juliaup home directory, located at:

  /Users/yana/.juliaup

The julia, juliaup and other commands will be added to
Juliaup's bin directory, located at:

  /Users/yana/.juliaup/bin

This path will then be added to your PATH environment variable by
modifying the profile files located at:

  /Users/yana/.profile
  /Users/yana/.bash_profile
  /Users/yana/.zshrc

Julia will look for a new version of Juliaup itself every 1440 minutes when you start julia.

You can uninstall at any time with juliaup self uninstall and these
changes will be reverted.

✔ Do you want to install with these default configuration choices? · Proceed with installation

Now installing Juliaup
Installing Julia 1.10.4+0.x64.apple.darwin14
Configured the default Julia version to be 'release'.
Julia was successfully installed on your system.

Depending on which shell you are using, run one of the following
commands to reload the PATH environment variable:

  . /Users/yana/.profile
  . /Users/yana/.bash_profile
  . /Users/yana/.zshrc

(base) yana@Iannas-MacBook-Pro-2 ~ %
```

yana — -zsh — 115×36

```
Juliaup will be installed into the Juliaup home directory, located at:

  /Users/yana/.juliaup

The julia, juliaup and other commands will be added to
Juliaup's bin directory, located at:

  /Users/yana/.juliaup/bin

This path will then be added to your PATH environment variable by
modifying the profile files located at:

  /Users/yana/.profile
  /Users/yana/.bash_profile
  /Users/yana/.zshrc

Julia will look for a new version of Juliaup itself every 1440 minutes when you start julia.

You can uninstall at any time with juliaup self uninstall and these
changes will be reverted.

✔ Do you want to install with these default configuration choices? · Proceed with installation

Now installing Juliaup
Installing Julia 1.10.4+0.x64.apple.darwin14
Configured the default Julia version to be 'release'.
Julia was successfully installed on your system.

Depending on which shell you are using, run one of the following
commands to reload the PATH environment variable:

  . /Users/yana/.profile
  . /Users/yana/.bash_profile
  . /Users/yana/.zshrc

(base) yana@Iannas-MacBook-Pro-2 ~ %
```

```
# >>> juliaup initialize >>>

# !! Contents within this block are managed by juliaup !!

path=('/Users/yana/.juliaup/bin' $path)
export PATH

# <<< juliaup initialize <<<
```

```
(base) yana@Iannas-MacBook-Pro-2 ~ % . /Users/yana/.zshrc
(base) yana@Iannas-MacBook-Pro-2 ~ % juliaup list
 Channel              Version
-------------------------------------------------------
 0                    0.7.0+0.x64.apple.darwin14
 0.7                  0.7.0+0.x64.apple.darwin14
 0.7.0                0.7.0+0.x64.apple.darwin14
 0.7.0~x64            0.7.0+0.x64.apple.darwin14
 0.7~x64              0.7.0+0.x64.apple.darwin14
 0~x64                0.7.0+0.x64.apple.darwin14
 1                    1.10.4+0.x64.apple.darwin14
 1.0                  1.0.5+0.x64.apple.darwin14
 1.0.0                1.0.0+0.x64.apple.darwin14
 1.0.0~x64            1.0.0+0.x64.apple.darwin14
 1.0.1                1.0.1+0.x64.apple.darwin14
 1.0.1~x64            1.0.1+0.x64.apple.darwin14
 1.0.2                1.0.2+0.x64.apple.darwin14
 1.0.2~x64            1.0.2+0.x64.apple.darwin14
 1.0.3                1.0.3+0.x64.apple.darwin14
 1.0.3~x64            1.0.3+0.x64.apple.darwin14
 1.0.4                1.0.4+0.x64.apple.darwin14
 1.0.4~x64            1.0.4+0.x64.apple.darwin14
 1.0.5                1.0.5+0.x64.apple.darwin14
 1.0.5~x64            1.0.5+0.x64.apple.darwin14
 1.0~x64              1.0.5+0.x64.apple.darwin14
 1.1                  1.1.1+0.x64.apple.darwin14
 1.1.0                1.1.0+0.x64.apple.darwin14
 1.1.0~x64            1.1.0+0.x64.apple.darwin14
 1.1.1                1.1.1+0.x64.apple.darwin14
 1.1.1~x64            1.1.1+0.x64.apple.darwin14
 1.3.1~x64            1.3.1+0.x64.apple.darwin14
 1.4.1~x64            1.4.1+0.x64.apple.darwin14
 1.4.2~x64            1.4.2+0.x64.apple.darwin14
 1.6.0                1.6.0+0.x64.apple.darwin14
 1.6.0-beta1~x64      1.6.0-beta1+0.x64.apple.darwin14
 1.6.0-rc2            1.6.0-rc2+0.x64.apple.darwin14
```

```
(base) yana@Iannas-MacBook-Pro-2 ~ % . /Users/yana/.zshrc
(base) yana@Iannas-MacBook-Pro-2 ~ % juliaup list
 Channel              Version
-------------------------------------------------------
 0                    0.7.0+0.x64.apple.darwin14
 0.7                  0.7.0+0.x64.apple.darwin14
 0.7.0                0.7.0+0.x64.apple.darwin14
 0.7.0~x64            0.7.0+0.x64.apple.darwin14
 0.7~x64              0.7.0+0.x64.apple.darwin14
 0~x64                0.7.0+0.x64.apple.darwin14
 1                    1.10.4+0.x64.apple.darwin14
 1.0                  1.0.5+0.x64.apple.darwin14
 1.0.0                1.0.0+0.x64.apple.darwin14
 1.0.0~x64            1.0.0+0.x64.apple.darwin14
 1.0.1                1.0.1+0.x64.apple.darwin14
 1.0.1~x64            1.0.1+0.x64.apple.darwin14
 1.0.2                1.0.2+0.x64.apple.darwin14
 1.0.2~x64            1.0.2+0.x64.apple.darwin14
 1.0.3                1.0.3+0.x64.apple.darwin14
 1.0.3~x64            1.0.3+0.x64.apple.darwin14
 1.0.4                1.0.4+0.x64.apple.darwin14
 1.0.4~x64            1.0.4+0.x64.apple.darwin14
 1.0.5                1.0.5+0.x64.apple.darwin14
 1.0.5~x64            1.0.5+0.x64.apple.darwin14
 1.0~x64              1.0.5+0.x64.apple.darwin14
 1.1                  1.1.1+0.x64.apple.darwin14
 1.1.0                1.1.0+0.x64.apple.darwin14
 1.1.0~x64            1.1.0+0.x64.apple.darwin14
 1.1.1                1.1.1+0.x64.apple.darwin14
 1.1.1~x64            1.1.1+0.x64.apple.darwin14
 1.3.1~x64            1.3.1+0.x64.apple.darwin14
 1.4.1~x64            1.4.1+0.x64.apple.darwin14
 1.4.2~x64            1.4.2+0.x64.apple.darwin14
 1.6.0                1.6.0+0.x64.apple.darwin14
 1.6.0-beta1~x64      1.6.0-beta1+0.x64.apple.darwin14
 1.6.0-rc2            1.6.0-rc2+0.x64.apple.darwin14
```

```
(base) yana@Iannas-MacBook-Pro-2 ~ % juliaup update
(base) yana@Iannas-MacBook-Pro-2 ~ % julia
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.10.4 (2024-06-04)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia>
```

```
(julia_hep_2024) yana@Iannas-MacBook-Pro-2 JuliaHEP-2024-Power-of-Python-and-Julia-for-Advanced-Data-Analysis % python
Python 3.12.6 | packaged by conda-forge | (main, Sep 22 2024, 14:08:13) [Clang 17.0.6 ] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import awkward as ak
>>> from juliacall import Main as jl
[juliapkg] Found dependencies: /Users/yana/anaconda3/envs/julia_hep_2024/lib/python3.12/site-packages/awkward/juliapkg.json
[juliapkg] Found dependencies: /Users/yana/anaconda3/envs/julia_hep_2024/lib/python3.12/site-packages/juliacall/juliapkg.json
[juliapkg] Found dependencies: /Users/yana/anaconda3/envs/julia_hep_2024/lib/python3.12/site-packages/juliapkg/juliapkg.json
[juliapkg] Locating Julia ~1.9, =1.10.0, ^1.10.3
[juliapkg] Installing Julia 1.10.5 using JuliaUp
[juliapkg] Using Julia 1.10.5 at /Users/yana/.julia/juliaup/julia-1.10.5+0.x64.apple.darwin14/bin/julia
[juliapkg] Using Julia project at /Users/yana/anaconda3/envs/julia_hep_2024/julia_env
[juliapkg] Installing packages:
           julia> import Pkg
           julia> Pkg.Registry.update()
           julia> Pkg.add([Pkg.PackageSpec(name="AwkwardArray", uuid="7d259134-7f60-4bf1-aa00-7452e11bde56"), Pkg.PackageSpec(name="PythonCall", uuid="6099a
3de-0909-46bc-b1f4-468b9a2dfc0d")])
           julia> Pkg.resolve()
           julia> Pkg.precompile()
    Updating registry at `~/.julia/registries/General.toml`
   Resolving package versions...
   Installed JLLWrappers ─ v1.6.0
   Installed StructTypes ─ v1.11.0
    Updating `~/anaconda3/envs/julia_hep_2024/julia_env/Project.toml`
  [7d259134] + AwkwardArray v0.1.5
  [6099a3de] + PythonCall v0.9.23
    Updating `~/anaconda3/envs/julia_hep_2024/julia_env/Manifest.toml`
  [7d259134] + AwkwardArray v0.1.5
  [8f4d0f93] + Conda v1.10.2
  [992eb4ea] + CondaPkg v0.2.23
  [9a962f9c] + DataAPI v1.16.0
  [e2d170a0] + DataValueInterfaces v1.0.0
  [82899510] + IteratorInterfaceExtensions v1.0.0
  [692b3bcd] + JLLWrappers v1.6.0
  [682c06a0] + JSON v0.21.4
  [0f8b85d8] + JSON3 v1.14.0
  [1914dd2f] + MacroTools v0.5.13
  [0b3b1443] + MicroMamba v0.1.14
  [bac558e1] + OrderedCollections v1.6.3
  [69de0a69] + Parsers v2.8.1
  [fa939f87] + Pidfile v1.3.0
  [aea7be01] + PrecompileTools v1.2.1
  [21216c6a] + Preferences v1.4.3
  [6099a3de] + PythonCall v0.9.23
  [ae029012] + Requires v1.3.0
  [6c6a2e73] + Scratch v1.2.1
  [856f2bd8] + StructTypes v1.11.0
  [3783bdb8] + TableTraits v1.0.1
  [bd369af6] + Tables v1.12.0
  [e17b2a0c] + UnsafePointers v1.0.0
  [81def892] + VersionParsing v1.3.0
```

Takeaway: Maintaining functional bilingual environment is challenging!
Experimental feature?

PRINCETON UNIVERSITY

iris hep

Awkward Array.jl

for Julia!

Search docs

Version dev

Getting started

Let's assume that both Python and Julia are installed.

Note

If Julia is not install it is recomended to follow its official installation instructions described here.

Installation

It is recommended to use conda virtul environment.

Using Julia Awkward Arrays from Python

- To install Awkward Array Python package:

```
conda install -c conda-forge awkward
```

- To install JuliaCall:

```
conda install pyjuliacall
```

JuliaCall takes care of installing all necessary Julia packages, including this package.

```
import awkward as ak
from juliacall import Main as jl

jl.seval("using AwkwardArray")
```

Using Python Awkward Arrays from Julia

PythonCall is currently configured to use the Julia-specific Python distribution installed by the CondaPkg.jl package.

```
using CondaPkg
```

PRINCETON UNIVERSITY

Practical Examples

where mixing Python and Julia really shines

[1]:

```
import awkward as ak
import numpy as np
import matplotlib.pyplot as plt
```

PRINCETON UNIVERSITY

Practical Examples

where mixing Python and Julia really shines

[1]:
```python
import awkward as ak
import numpy as np
import matplotlib.pyplot as plt
```

[2]:
```python
from juliacall import Main as jl
from juliacall import Pkg
```

```
Detected IPython. Loading juliacall extension. See https://juliapy.github.io/PythonCall.jl/stable/compat/#IPython
```

Practical Examples

where mixing Python and Julia really shines

[3]:
```
Pkg.add('AwkwardArray')
```

```
   Updating registry at `~/.julia/registries/General.toml`
  Resolving package versions...
   Updating `~/anaconda3/julia_env/Project.toml`
  [7d259134] + AwkwardArray v0.1.5
   Updating `~/anaconda3/julia_env/Manifest.toml`
  [7d259134] + AwkwardArray v0.1.5
  [8f4d0f93] + Conda v1.10.2
  [682c06a0] + JSON v0.21.4
  [81def892] + VersionParsing v1.3.0
```

Practical Examples

where mixing Python and Julia really shines

```
[3]: Pkg.add('AwkwardArray')
```

```
    Updating registry at `~/.julia/registries/General.toml`
   Resolving package versions...
    Updating `~/anaconda3/julia_env/Project.toml`
  [7d259134] + AwkwardArray v0.1.5
    Updating `~/anaconda3/julia_env/Manifest.toml`
  [7d259134] + AwkwardArray v0.1.5
  [8f4d0f93] + Conda v1.10.2
  [682c06a0] + JSON v0.21.4
  [81def892] + VersionParsing v1.3.0
```

```
[4]: jl.seval("using AwkwardArray")
```

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

[11]:
```
jl.seval("""
function invariant_mass(tree)
  layout = AwkwardArray.PrimitiveArray{Float64}()
  for event in tree
    if event.nMuon == 2
      if event.Muon_charge[1] != event.Muon_charge[2]
        result = sqrt(2 * event.Muon_pt[1] * event.Muon_pt[2] *
                    (cosh(event.Muon_eta[1] - event.Muon_eta[2]) -
                     cos(event.Muon_phi[1] - event.Muon_phi[2])))
        if result > 70
          push!(layout, result)
        end
      end
    end
  end
  layout
end
""")
```

[11]:
```
invariant_mass (generic function with 1 method)
```

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

```
[12]: Pkg.add('UnROOT')

   Resolving package versions...
  No Changes to `~/anaconda3/julia_env/Project.toml`
  No Changes to `~/anaconda3/julia_env/Manifest.toml`

[13]: jl.seval("using UnROOT")

[14]: file = jl.Main.ROOTFile("data/SMHiggsToZZTo4L.root")

[15]: file

[15]: ROOTFile with 1 entry and 18 streamers.
      data/SMHiggsToZZTo4L.root
      └─ Events (TTree)
         ├─ "run"
         ├─ "luminosityBlock"
         ├─ "event"
         ├─ "⋮"
         ├─ "Electron_dzErr"
         ├─ "MET_pt"
         └─ "MET_phi"
```

PRINCETON UNIVERSITY

iris hep

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

```
[16]: events = jl.Main.LazyTree(file, "Events")

[17]: events

[17]:  Row │ Electron_mass       nElectron  luminosityBlock  nMuon   Electron_phi      M …
           │ SubArray{Float3     UInt32     UInt32           UInt32  SubArray{Float3   F …
      ─────┼──────────────────────────────────────────────────────────────────────────────
       1   │ []                  0          156              3       []                – …
       2   │ [0.00544,           4          156              0       [0.134, -1        2 …
       3   │ [-0.00609,          2          156              0       [2.18, 1.6        – …
       4   │ [-0.00123]          1          156              7       [-0.643]          – …
       5   │ [0.0117, 0          4          156              0       [1.01, -1.        1 …
       6   │ [-0.00183]          1          156              2       [-0.497]          – …
       7   │ [-0.00183]          1          156              1       [-1.47]           – …
       8   │ [-0.00216]          1          156              0       [-0.633]          – …
       9   │ [-0.0128,           4          156              0       [2.79, -2.        – …
       10  │ []                  0          156              0       []                – …
       11  │ [-0.00119,          2          156              1       [2.28, -2.        0 …
       12  │ [0.00608,           3          156              2       [-1.31, -1        2 …
       13  │ [-0.00765,          4          156              0       [2.52, -3.        – …
       14  │ []                  0          156              2       []                – …
       15  │ []                  0          156              0       []                – …
       ⋮   │     ⋮                   ⋮           ⋮               ⋮         ⋮              ⋱
                                                      27 columns and 299958 rows omitted
```

PRINCETON UNIVERSITY

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

[18]:

```
jl.seval("""
using AwkwardArray

function make_record_array(events)
    array = AwkwardArray.RecordArray(
        NamedTuple{(:pt, :eta, :phi, :mass, :charge, :isolation)}((
            AwkwardArray.from_iter(events.Muon_pt),
            AwkwardArray.from_iter(events.Muon_eta),
            AwkwardArray.from_iter(events.Muon_phi),
            AwkwardArray.from_iter(events.Muon_mass),
            AwkwardArray.from_iter(events.Muon_charge),
            AwkwardArray.from_iter(events.Muon_pfRelIso03_all),
        )
    ))
    return AwkwardArray.convert(array)
end
""")
```

[18]:

```
make_record_array (generic function with 1 method)
```

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

[19]:
```
%%time
muons = jl.make_record_array(events[1])
```

```
CPU times: user 1.29 s, sys: 14.9 ms, total: 1.31 s
Wall time: 1.31 s
```

[20]:
```
%%time
muons = jl.make_record_array(events)
```

```
CPU times: user 581 ms, sys: 24.2 ms, total: 605 ms
Wall time: 603 ms
```

PRINCETON UNIVERSITY

iris hep

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

[21]:
```
muons = ak.zip({
                "pt": muons.pt,
                "eta": muons.eta,
                "phi": muons.phi,
                "mass": muons.mass,
                "charge": muons.charge,
                "isolation": muons.isolation,
            },
            with_name="PtEtaPhiMCandidate",)
```

[22]:
```
cutflow = dict()

# Sort muons by transverse momentum
muons = muons[ak.argsort(muons.pt, axis=1)]

cutflow["all events"] = ak.num(muons, axis=0)

# Quality and minimum pt cuts
muons = muons[(muons.pt > 5) & (muons.isolation < 0.2)]
cutflow["at least 4 good muons"] = ak.sum(ak.num(muons) >= 4)
```

[23]:
```
# reduce first axis: skip events without enough muons
muons = muons[ak.num(muons) >= 4]
```

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

[24]:

```
jl.seval("""
using AwkwardArray

four_muons = AwkwardArray.ListArray{
    AwkwardArray.Index64,
    AwkwardArray.ListArray{
        AwkwardArray.Index64,
        AwkwardArray.PrimitiveArray{Int64},
    },
}()

function find_4lep(events_leptons)

    for leptons in events_leptons
        nlep = length(leptons)
        for i0 in 1:nlep
            for i1 in (i0 + 1):nlep
                if leptons[i0][:charge] + leptons[i1][:charge] != 0
                    continue
                end
                for i2 in 1:nlep
                    for i3 in (i2 + 1):nlep
                        if length(Set([i0, i1, i2, i3])) < 4
                            continue
                        end
                        if leptons[i2][:charge] + leptons[i3][:charge] != 0
                            continue
                        end

                        push!(four_muons.content.content, (i0 - 1)) # Julia is 1-based, subtract 1 for 0-based indexing
                        push!(four_muons.content.content, (i1 - 1))
                        push!(four_muons.content.content, (i2 - 1))
                        push!(four_muons.content.content, (i3 - 1))
                        AwkwardArray.end_list!(four_muons.content)
                    end
                end
            end
        end
        AwkwardArray.end_list!(four_muons)
    end
    return four_muons
end""")
```

[24]:

```
find_4lep (generic function with 1 method)
```

PRINCETON UNIVERSITY

iris hep

Real-world scenarios

how this combination can speed up our data analysis and make our work more efficient

[25]:
```
fourmuon = jl.find_4lep(muons[1:10])
```

[26]:
```
%%time
fourmuon = jl.find_4lep(muons)
```

```
CPU times: user 417 ms, sys: 10.4 ms, total: 427 ms
Wall time: 426 ms
```

[27]:
```
fourmuon
```

[27]:
```
14963-element AwkwardArray.ListArray{Vector{Int64}, AwkwardArray.ListArray{Vector{Int64}, AwkwardArray.PrimitiveArray{Int64, Vector{Int64}, :default}, :default},
:default}:
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
 [[0, 1, 2, 3], [0, 2, 1, 3], [1, 3, 0, 2], [2, 3, 0, 1]]
 [[0, 2, 1, 3], [0, 3, 1, 2], [1, 2, 0, 3], [1, 3, 0, 2]]
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
 [[0, 2, 1, 3], [0, 3, 1, 2], [1, 2, 0, 3], [1, 3, 0, 2]]
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
 [[0, 2, 1, 4], [0, 2, 3, 4], [0, 4, 1, 2], ..., [3, 4, 0, 2], [3, 4, 1, 2]]
 0-element AwkwardArray.ListArray{Vector{Int64}, AwkwardArray.PrimitiveArray{Int64, Vector{Int64}, :default}, :default}
 ⋮
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
 [[0, 1, 2, 3], [0, 2, 1, 3], [1, 3, 0, 2], [2, 3, 0, 1]]
 [[0, 2, 1, 3], [0, 3, 1, 2], [1, 2, 0, 3], [1, 3, 0, 2]]
 [[0, 2, 1, 3], [0, 3, 1, 2], [1, 2, 0, 3], [1, 3, 0, 2]]
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
 0-element AwkwardArray.ListArray{Vector{Int64}, AwkwardArray.PrimitiveArray{Int64, Vector{Int64}, :default}, :default}
 [[0, 1, 2, 3], [0, 2, 1, 3], [1, 3, 0, 2], [2, 3, 0, 1]]
 [[0, 1, 2, 3], [0, 2, 1, 3], [1, 3, 0, 2], [2, 3, 0, 1]]
 [[0, 1, 2, 3], [0, 3, 1, 2], [1, 2, 0, 3], [2, 3, 0, 1]]
```

Compatibility Tools: IPython

- The line magic **%julia** code executes the given Julia code in-line
- The cell magic **%%julia** executes a cell of Julia code
- Julia's **stdout** and **stderr** are redirected to IPython
- Calling **display(x)** from Julia will display x in IPython

```
In [1]: %load_ext juliacall
```

```
Detected IPython. Loading juliacall extension. See https://juliapy.github.io/PythonCall.jl/stable/compat/#IPython
WARNING: replacing module _ipython.
```

```
In [2]: x = 2
```

```
In [3]: y = 8
```

The `%%julia` cell magic can synchronise variables between Julia and Python by listing them on the first line:

```
In [4]: %%julia x y z
            z = "$x^$y = $(x^y)";
```

```
In [5]: z
Out[5]: "2^8 = 256"
```

PRINCETON UNIVERSITY

Summary and Outlook

- Maintaining dependencies and common runtime environment is a challenge
- Python 3.13
  - The biggest changes include a new interactive interpreter, experimental support for running in a free-threaded mode (**PEP 703**), and a Just-In-Time compiler (**PEP 744**).
    - 3.13.0 candidate 3: Monday, 2024-09-30
    - 3.13.0 final: Monday, 2024-10-07
    - Subsequent bugfix releases every two months.
- Potential for deeper integration and community-driven innovation

PRINCETON UNIVERSITY

iris hep