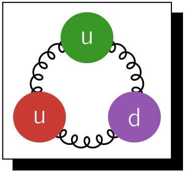
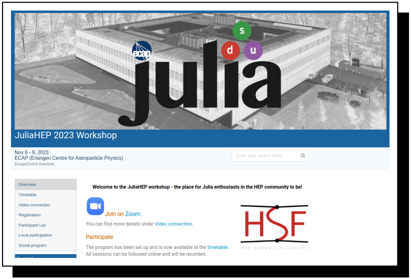
A Mini-Intro to Julia

Alexander Moreno Briceño
Universidad Antonio Nariño



Deep Learning Train-the-Trainer Workshop, September 15-19
Potsdam, Germany

JuliaHEP at the HSF



JuliaHEP 2023 Workshop

Nov 6 – 9, 2023
ECAP (Erlangen Centre for Astroparticle Physics)
Europe/Zurich timezone

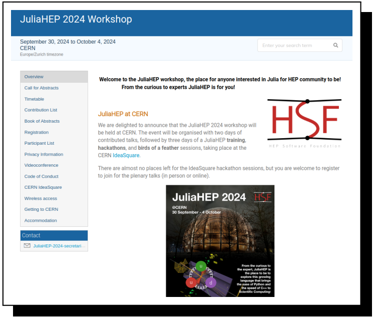
Welcome to the JuliaHEP workshop - the place for Julia enthusiasts in the HEP community to be!

Join on Zoom

You can find more details under Video connection.

Participate

The program has been set up and is now available at the timetable.
All sessions can be followed online and will be recorded.



JuliaHEP 2024 Workshop

September 30, 2024 to October 4, 2024
CERN
Europe/Zurich timezone

Welcome to the JuliaHEP workshop, the place for anyone interested in Julia for HEP community to be!
From the curious to experts JuliaHEP is for you!

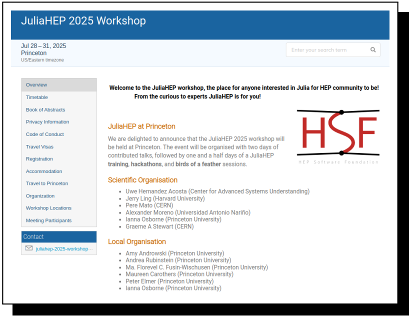
JuliaHEP at CERN

We are delighted to announce that the JuliaHEP 2024 workshop will be held at CERN. The event will be organised with two days of contributed talks, followed by three days of a JuliaHEP **training**, **hackathons**, and **birds of a feather** sessions, taking place at the CERN IdeaSquare.

There are almost no places left for the IdeaSquare hackathon sessions, but you are welcome to register to join for the plenary talks (in person or online).

JuliaHEP 2024
@CERN
30 September - 4 October

From the curious to the expert, JuliaHEP is the place to be to explore this growing language that brings the ease of Python and the speed of C++ to Scientific Computing!



JuliaHEP 2025 Workshop

Jul 28 – 31, 2025
Princeton
US/Eastern timezone

Welcome to the JuliaHEP workshop, the place for anyone interested in Julia for HEP community to be!
From the curious to experts JuliaHEP is for you!

JuliaHEP at Princeton

We are delighted to announce that the JuliaHEP 2025 workshop will be held at Princeton. The event will be organised with two days of contributed talks, followed by one and a half days of a JuliaHEP **training**, **hackathons**, and **birds of a feather** sessions.

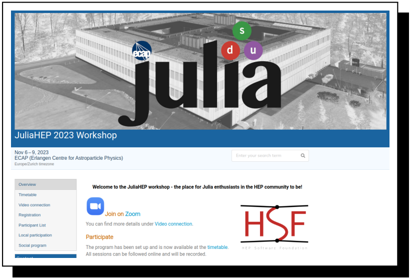
Scientific Organisation

- Uwe Hernandez Acosta (Center for Advanced Systems Understanding)
- Jerry Ling (Harvard University)
- Pere Mato (CERN)
- Alexander Moreno (Universidad Antonio Nariño)
- Ianna Osborne (Princeton University)
- Graeme A Stewart (CERN)

Local Organisation

- Amy Androwski (Princeton University)
- Andrea Rubinstein (Princeton University)
- Ma. Florevel C. Fusin-Wischusen (Princeton University)
- Maureen Carothers (Princeton University)
- Peter Elmer (Princeton University)
- Ianna Osborne (Princeton University)

JuliaHEP at the HSF



JuliaHEP 2023 Workshop

Nov 6 – 9, 2023
ECAP (Erlangen Centre for Astroparticle Physics)
Europe/Zurich timezone

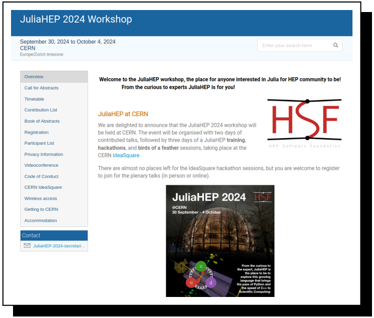
Welcome to the JuliaHEP workshop - the place for Julia enthusiasts in the HEP community to be!

Join on Zoom

You can find more details under Video connection.

Participate

The program has been set up and is now available at the timetable.
All sessions can be followed online and will be recorded.



JuliaHEP 2024 Workshop

September 30, 2024 to October 4, 2024
CERN
Europe/Zurich timezone

Welcome to the JuliaHEP workshop, the place for anyone interested in Julia for HEP community to be!
From the curious to experts JuliaHEP is for you!

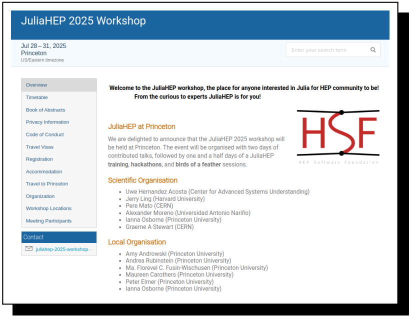
JuliaHEP at CERN

We are delighted to announce that the JuliaHEP 2024 workshop will be held at CERN. The event will be organised with two days of contributed talks, followed by three days of a JuliaHEP **training**, **hackathons**, and **birds of a feather** sessions, taking place at the CERN IdeaSquare.

There are almost no places left for the IdeaSquare hackathon sessions, but you are welcome to register to join for the plenary talks (in person or online).

JuliaHEP 2024
@CERN
30 September - 4 October

From the curious to the expert, JuliaHEP is the place to be to explore this growing language that brings the ease of Python and the speed of C++ to Scientific Computing!



JuliaHEP 2025 Workshop

Jul 28 – 31, 2025
Princeton
US/Eastern timezone

Welcome to the JuliaHEP workshop, the place for anyone interested in Julia for HEP community to be!
From the curious to experts JuliaHEP is for you!

JuliaHEP at Princeton

We are delighted to announce that the JuliaHEP 2025 workshop will be held at Princeton. The event will be organised with two days of contributed talks, followed by one and a half days of a JuliaHEP **training**, **hackathons**, and **birds of a feather** sessions.

Scientific Organisation

- Uwe Hernandez Acosta (Center for Advanced Systems Understanding)
- Jerry Ling (Harvard University)
- Pere Mato (CERN)
- Alexander Moreno (Universidad Antonio Nariño)
- Ianna Osborne (Princeton University)
- Graeme A Stewart (CERN)

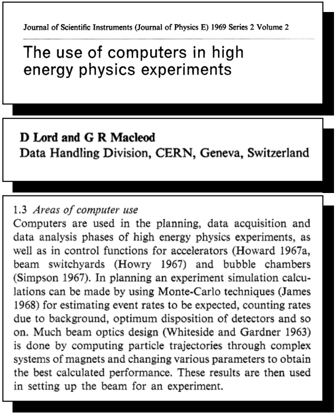
Local Organisation

- Amy Androwski (Princeton University)
- Andrea Rubinstein (Princeton University)
- Ma. Florevel C. Fusin-Wischusen (Princeton University)
- Maureen Carothers (Princeton University)
- Peter Elmer (Princeton University)
- Ianna Osborne (Princeton University)

JuliaHEP 2025 - Germany?

Why do we need software?

Software is everywhere!
More than 50 M lines of C++, Python and Fortran code!



Journal of Scientific Instruments (Journal of Physics E) 1969 Series 2 Volume 2

The use of computers in high energy physics experiments

D Lord and G R Macleod
Data Handling Division, CERN, Geneva, Switzerland

1.3 *Areas of computer use*
Computers are used in the planning, data acquisition and data analysis phases of high energy physics experiments, as well as in control functions for accelerators (Howard 1967a, beam switchyards (Howry 1967) and bubble chambers (Simpson 1967). In planning an experiment simulation calculations can be made by using Monte-Carlo techniques (James 1968) for estimating event rates to be expected, counting rates due to background, optimum disposition of detectors and so on. Much beam optics design (Whiteside and Gardner 1963) is done by computing particle trajectories through complex systems of magnets and changing various parameters to obtain the best calculated performance. These results are then used in setting up the beam for an experiment.

D Lord and G R Macleod 1969 J. Phys. E: Sci. Instrum. **2** 1

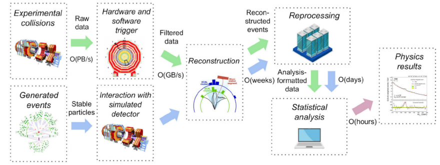
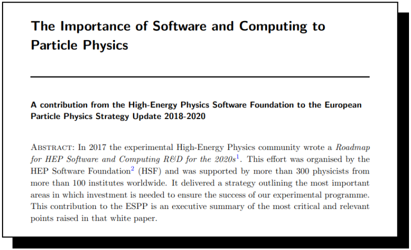


FIGURE 1 | A simplified diagram illustrating typical stages in experimental particle physics data analyses. After data acquisition that is using a multi-tiered trigger filtering step, the experimental collision data are further reduced in computing processes before they are ready for physics analyses. Events generated following theoretical models undergo a detector simulation step and are subsequently subject to the same reconstruction and processing steps as the collision data. The individual analysts then compare collision and simulated data using statistical analysis techniques. Our paper focuses mostly on the computational reproducibility challenges inherent in the last data analysis stages.

From Front. Big Data 4:661501. doi: 10.3389/fdata.2021.661501

Why do we need software?

Software is critical!

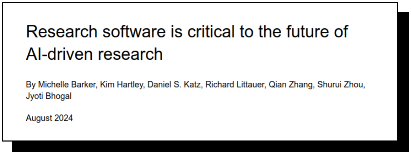


The Importance of Software and Computing to Particle Physics

A contribution from the High-Energy Physics Software Foundation to the European Particle Physics Strategy Update 2018-2020

Abstract: In 2017 the experimental High-Energy Physics community wrote a *Roadmap for HEP Software and Computing R&D for the 2020s*[1]. This effort was organised by the HEP Software Foundation[2] (HSF) and was supported by more than 300 physicists from more than 100 institutes worldwide. It delivered a strategy outlining the most important areas in which investment is needed to ensure the success of our experimental programme. This contribution to the ESPP is an executive summary of the most critical and relevant points raised in that white paper.

https://doi.org/10.5281/zenodo.2413005

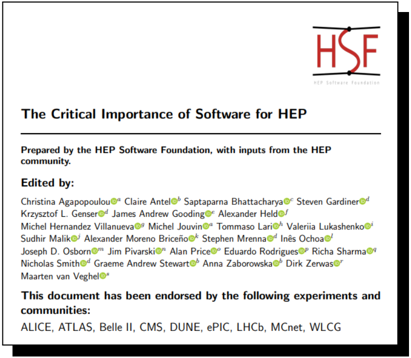


Research software is critical to the future of AI-driven research

By Michelle Barker, Kim Hartley, Daniel S. Katz, Richard Littauer, Qian Zhang, Shurui Zhou, Jyoti Bhogal

August 2024

https://doi.org/10.5281/zenodo.13993424



HSF
HEP Software Foundation

The Critical Importance of Software for HEP

Prepared by the HEP Software Foundation, with inputs from the HEP community.

Edited by:

Christina Agapopoulou[a] Claire Antel[b] Saptaparna Bhattacharya[c] Steven Gardiner[d] Krzysztof L. Genser[d] James Andrew Gooding[e] Alexander Held[f] Michel Hernandez Villanueva[g] Michel Jouvin[a] Tommaso Lari[h] Valeriia Lukashenko[i] Sudhir Malik[j] Alexander Moreno Briceño[k] Stephen Mrenna[d] Inês Ochoa[l] Joseph D. Osborn[m] Jim Pivarski[n] Alan Price[o] Eduardo Rodrigues[p] Richa Sharma[q] Nicholas Smith[d] Graeme Andrew Stewart[b] Anna Zaborowska[b] Dirk Zerwas[r] Maarten van Veghel[s]

This document has been endorsed by the following experiments and communities:

ALICE, ATLAS, Belle II, CMS, DUNE, ePIC, LHCb, MCnet, WLCG

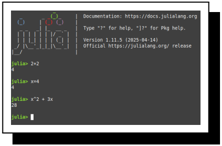
https://arxiv.org/pdf/2504.01050

Physics Event Generators, Detector Simulation, Reconstruction and Software Triggers, Data Analysis, and Training and Careers

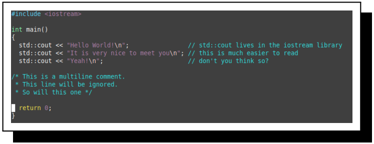
What do we need from software?

- **Code Efficiency**
  - Fast execution
  - Scalable
  - Maintainable
  - Sustainable
  - High throughput

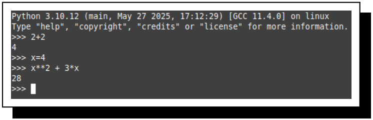
- **Human Efficiency**
  - Low barrier to entry
  - Rapid prototyping
  - Broad ecosystem
  - Excellent tooling



```
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.11.5 (2025-04-14)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> 2+2
4

julia> x=4
4

julia> x^2 + 3x
28

julia> 
```

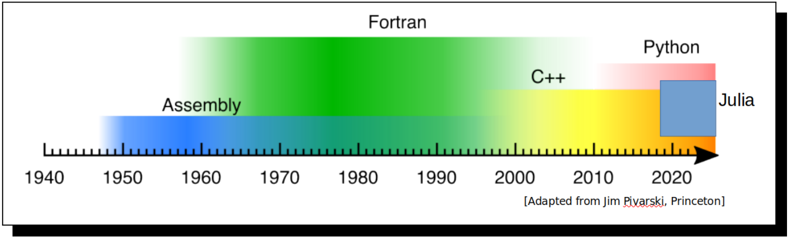


```
#include <iostream>

int main()
{
  std::cout << "Hello World!\n";                  // std::cout lives in the iostream library
  std::cout << "It is very nice to meet you\n"; // this is much easier to read
  std::cout << "Yeah!\n";                         // don't you think so?

/* This is a multiline comment.
 * This line will be ignored.
 * So will this one */

  return 0;
}
```



```
Python 3.10.12 (main, May 27 2025, 17:12:29) [GCC 11.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 2+2
4
>>> x=4
>>> x**2 + 3*x
28
>>> 
```

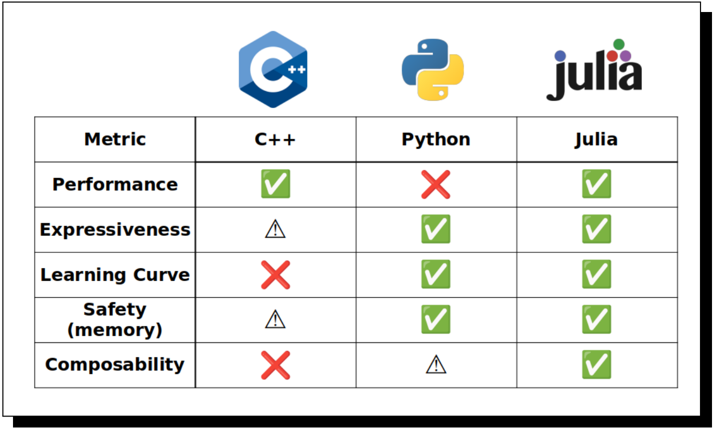
HEP Software: A little bit of History...



[Adapted from Jim Pivarski, Princeton]

- Fortran, the King for $\sim$ 40 years
- Now C++ and Python
- Julia is slowly entering in the arena

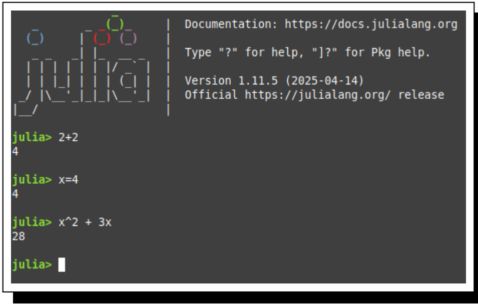
Where are we now?



Metric	C++	Python	Julia
Performance	✅	❌	✅
Expressiveness	⚠	✅	✅
Learning Curve	❌	✅	✅
Safety (memory)	⚠	✅	✅
Composability	❌	⚠	✅

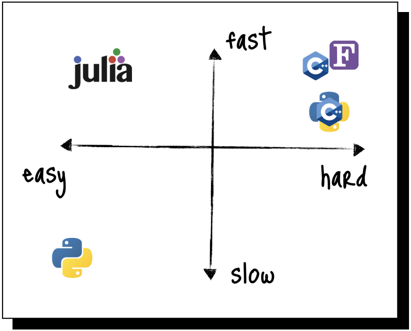
Julia Motivations

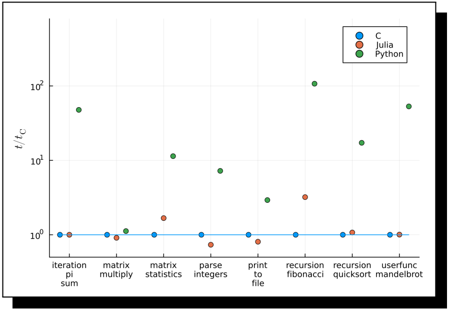
- Invented in 2012 at MIT (mostly)
  - Jeff Bezanson, Stefan Karpinski, Viral B. Shah, Alan Edelman
- Design goals and aims
  - Open source
  - Speed like C, but dynamic like Python
  - Obvious mathematical notation
  - General purpose like python
  - As easy for statistics as R
  - Powerful linear algebra like in Matlab
  - Good for gluing programs together like the shell



```
               _
   _       _ _(_)_     |  Documentation: https://docs.julialang.org
  (_)     | (_) (_)    |
   _ _   _| |_  __ _   |  Type "?" for help, "]?" for Pkg help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  |  Version 1.11.5 (2025-04-14)
 _/ |\__'_|_|_|\__'_|  |  Official https://julialang.org/ release
|__/                   |

julia> 2+2
4

julia> x=4
4

julia> x^2 + 3x
28

julia> █
```

Julia Motivations

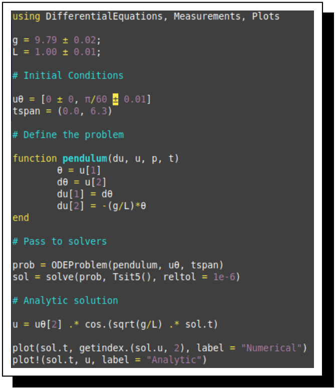




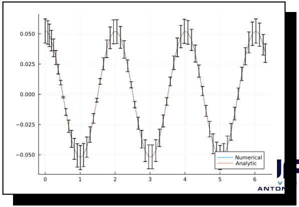
From https://julialang.org/benchmarks/

Julia in Practice - Julia is Fast

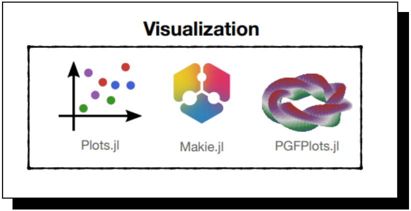
- Excellent REPL mode and notebooks
- Dinamically typed (runtime), with a powerfull type system
- Garbage collected
- Expressive maths syntax
- Extensive standard library



```julia
using DifferentialEquations, Measurements, Plots

g = 9.79 ± 0.02;
L = 1.00 ± 0.01;

# Initial Conditions

uθ = [0 ± 0, π/60 ± 0.01]
tspan = (0.0, 6.3)

# Define the problem

function pendulum(du, u, p, t)
        θ = u[1]
        dθ = u[2]
        du[1] = dθ
        du[2] = -(g/L)*θ
end

# Pass to solvers

prob = ODEProblem(pendulum, uθ, tspan)
sol = solve(prob, Tsit5(), reltol = 1e-6)

# Analytic solution

u = uθ[2] .* cos.(sqrt(g/L) .* sol.t)

plot(sol.t, getindex.(sol.u, 2), label = "Numerical")
plot!(sol.t, u, label = "Analytic")
```



Julia in Practice - Rich ecosystem

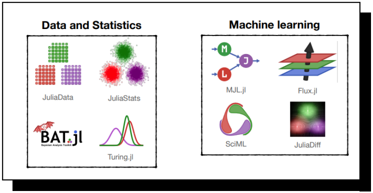


Visualization

Plots.jl

Makie.jl

PGFPlots.jl



Data and Statistics

JuliaData

JuliaStats

BAT.jl

Turing.jl

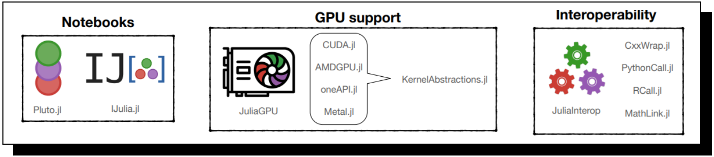
Machine learning

MJL.jl

Flux.jl

SciML

JuliaDiff



Notebooks

Pluto.jl

IJulia.jl

GPU support

JuliaGPU

CUDA.jl

AMDGPU.jl

oneAPI.jl

Metal.jl

KernelAbstractions.jl

Interoperability

JuliaInterop

CxxWrap.jl

PythonCall.jl

RCall.jl

MathLink.jl

Julia in Practice

https://juliahep.github.io/Hands-on-Julia-for-particle-physicists/00-intro.html