Recent Trends in Scientific Computing

John Stachurski

November 2023

Topics

- Trends in scientific computing
- Likely future directions
- Python and Julia as MATLAB replacements

A (very) short history of scientific computing

General purpose scientific computing environments:

1. Fortran & C / C++
2. MATLAB & (Python + NumPy)
3. Julia & (Python + Numba)
4. Python + Google JAX

Fortran & C — static types and AOT compilers

```c
#include <stdio.h>
int main() {
    int x = 1 + 1;
    printf("1 + 1 = %d\n", x);
    return 0;
}
```

```
PROGRAM ONE_PLUS_ONE
INTEGER :: X = 1 + 1
PRINT *, '1 + 1 = ', X
END PROGRAM ONE_PLUS_ONE
```

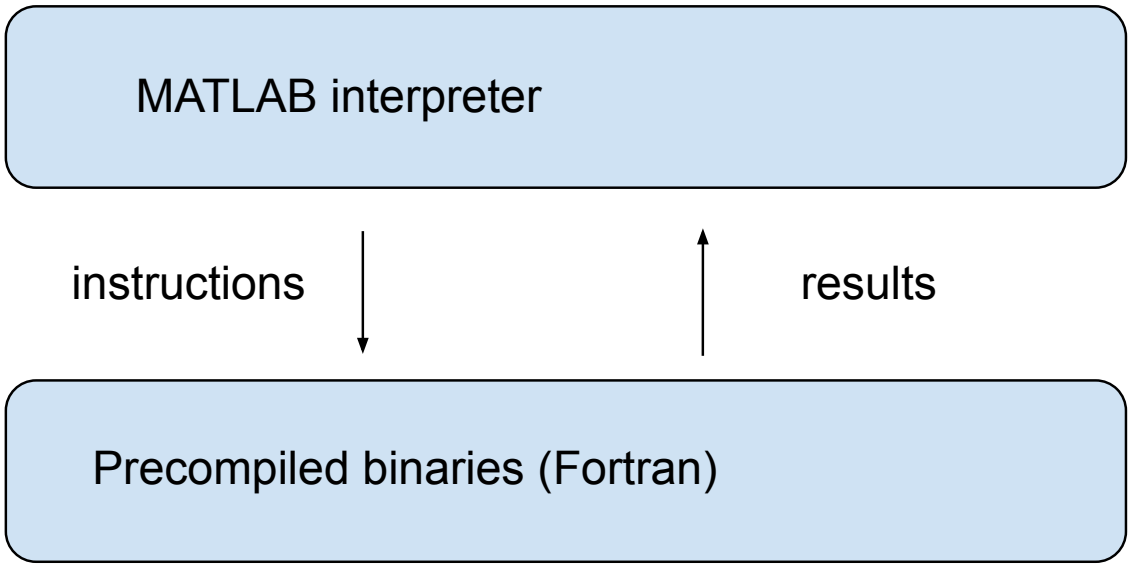
Pros

- fast — on a single thread

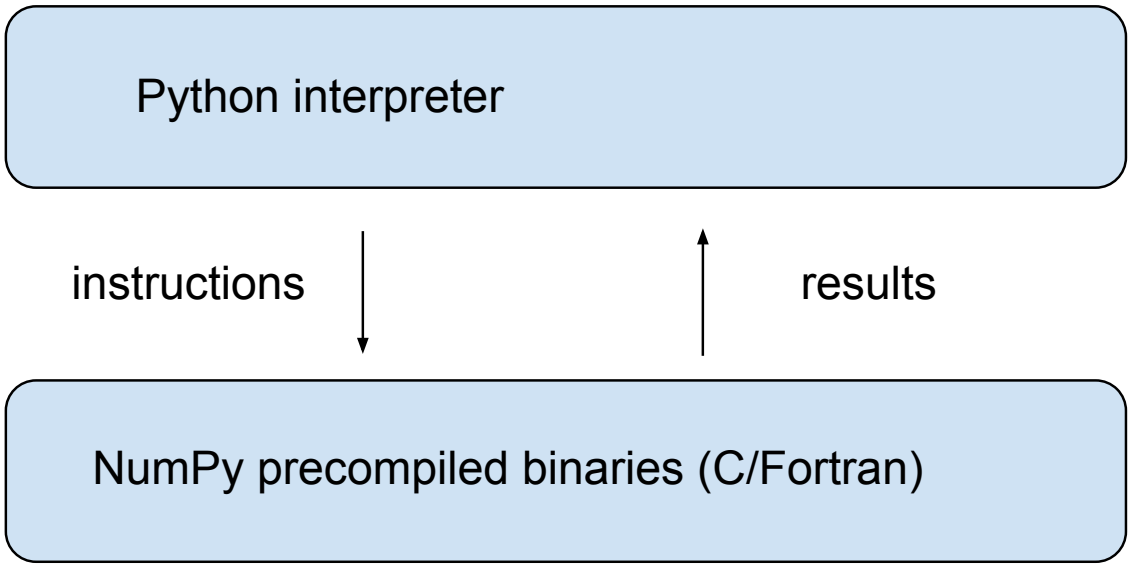
Cons

- tedious to write
- lack of portability
- hard to debug
- hard to parallelize
- low interactivity

Phase 2: MATLAB



Phase 2A: Python + NumPy



Phase 3: Julia — rise of the JIT compilers

```
function quad(x0, α, n)
    x = x0
    for i in 1:(n-1)
        x = α * x * (1 - x)
    end
    return x
end

quad(0.2, 4.0, 10_000_000)
```

Phase 3 continued: Python + Numba copy Julia

```
from numba import jit

@jit
def quad(x0, α, n):
    x = x0
    for i in range(n-1):
        x = α * x * (1 - x)
    return x

quad(0.2, 4.0, 10_000_000)
```

Phase 4: AI-driven scientific computing

Key players

- TensorFlow, PyTorch
- Google JAX
- Mojo?

Examples.

- OpenAI uses PyTorch
- Google Bard, Apple Ajax uses Google JAX

Lightning introduction to deep learning

Supervised deep learning: find a good approximation to an unknown functional relationship

$$y = f(x)$$

- x is the input and y is the output

Examples.

- $x =$ weather sensor data, $y =$ max temp tomorrow
- $x =$ current distribution of income, $y =$ tax revenue
- $x =$ unfinished sentence, $y =$ next word

Training

Take data set $(x_i, y_i)_{i=1}^n$ and solve

$$\min_\theta \ell(\theta) = \sum_{i=1}^n (y_i - \psi_\theta(x_i))^2 \qquad \text{s.t.} \qquad \theta \in \Theta$$

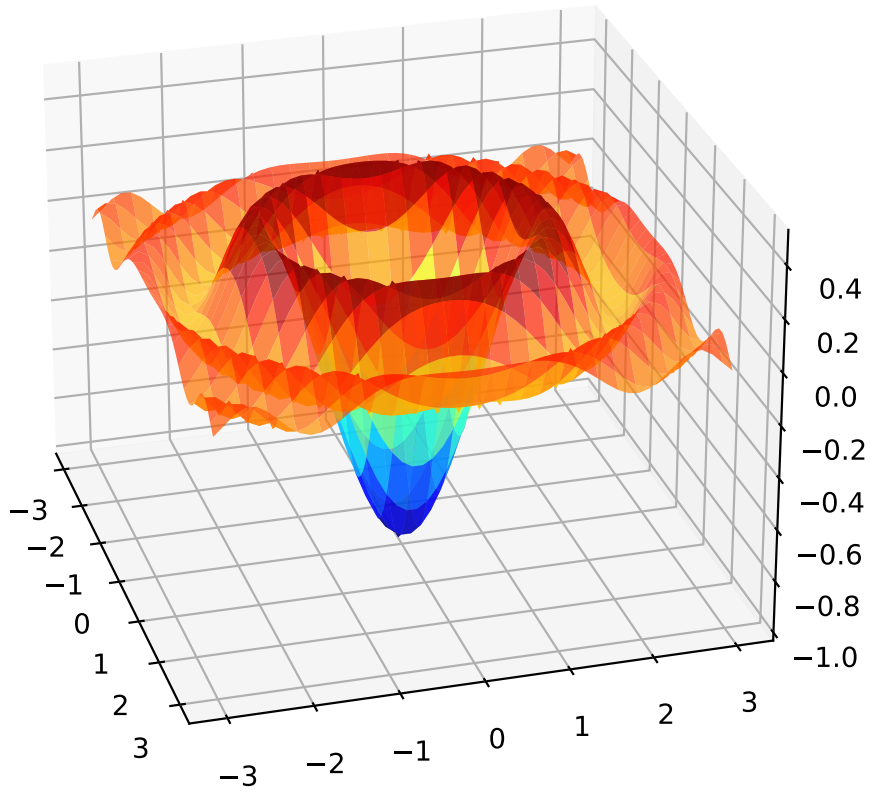
In the case of ANNs, the function class is all ψ_θ having the form

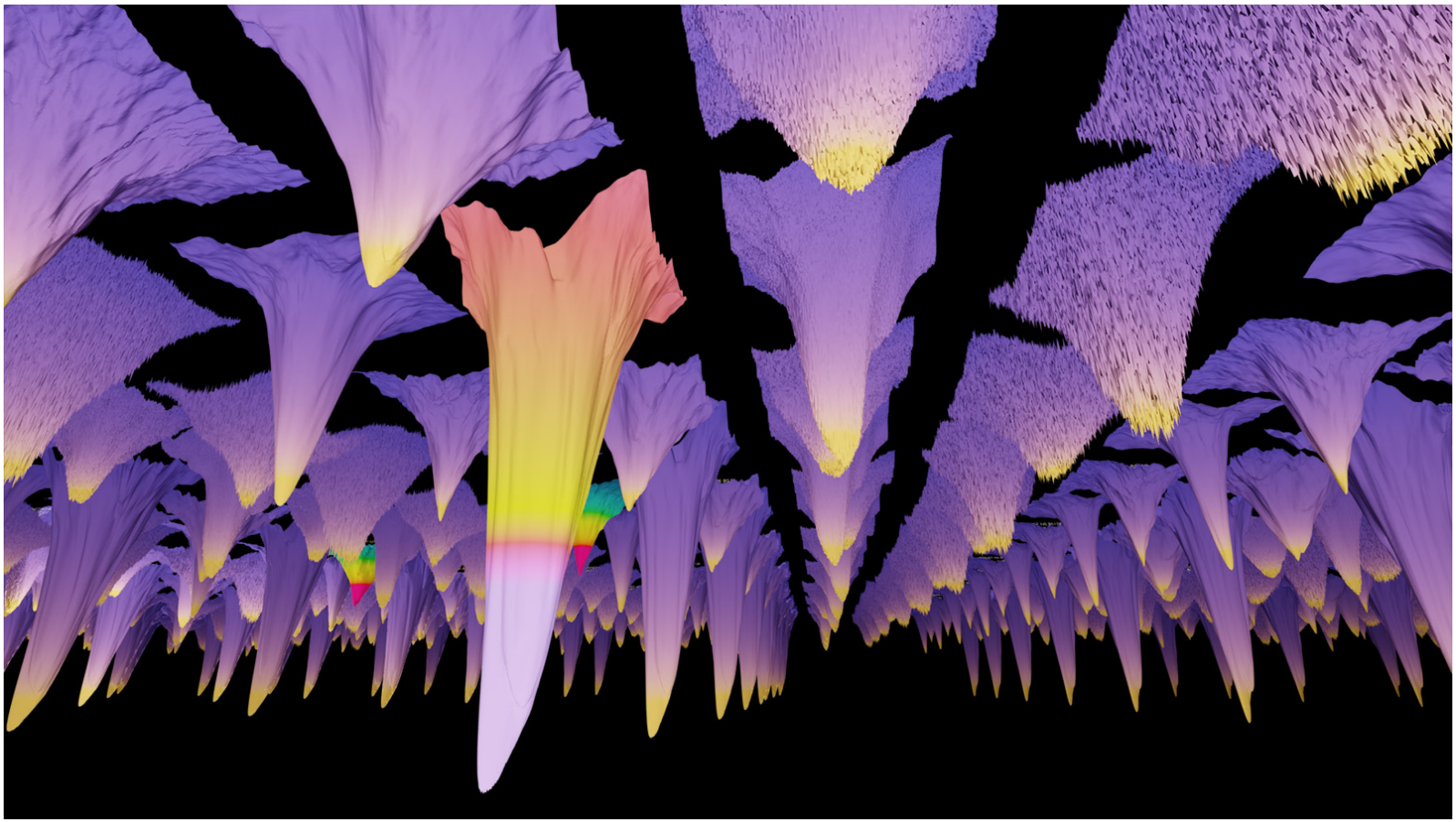
$$\psi_\theta = A_{1,\theta} \circ \alpha \circ \cdots \circ A_{k,\theta} \circ \alpha$$

where

- $A_{i,\theta}$ is an affine map for $i = 1, \ldots, k$ and
- α is a smooth nonlinear vector-valued function

AI / machine learning: minimizing differentiable loss functions



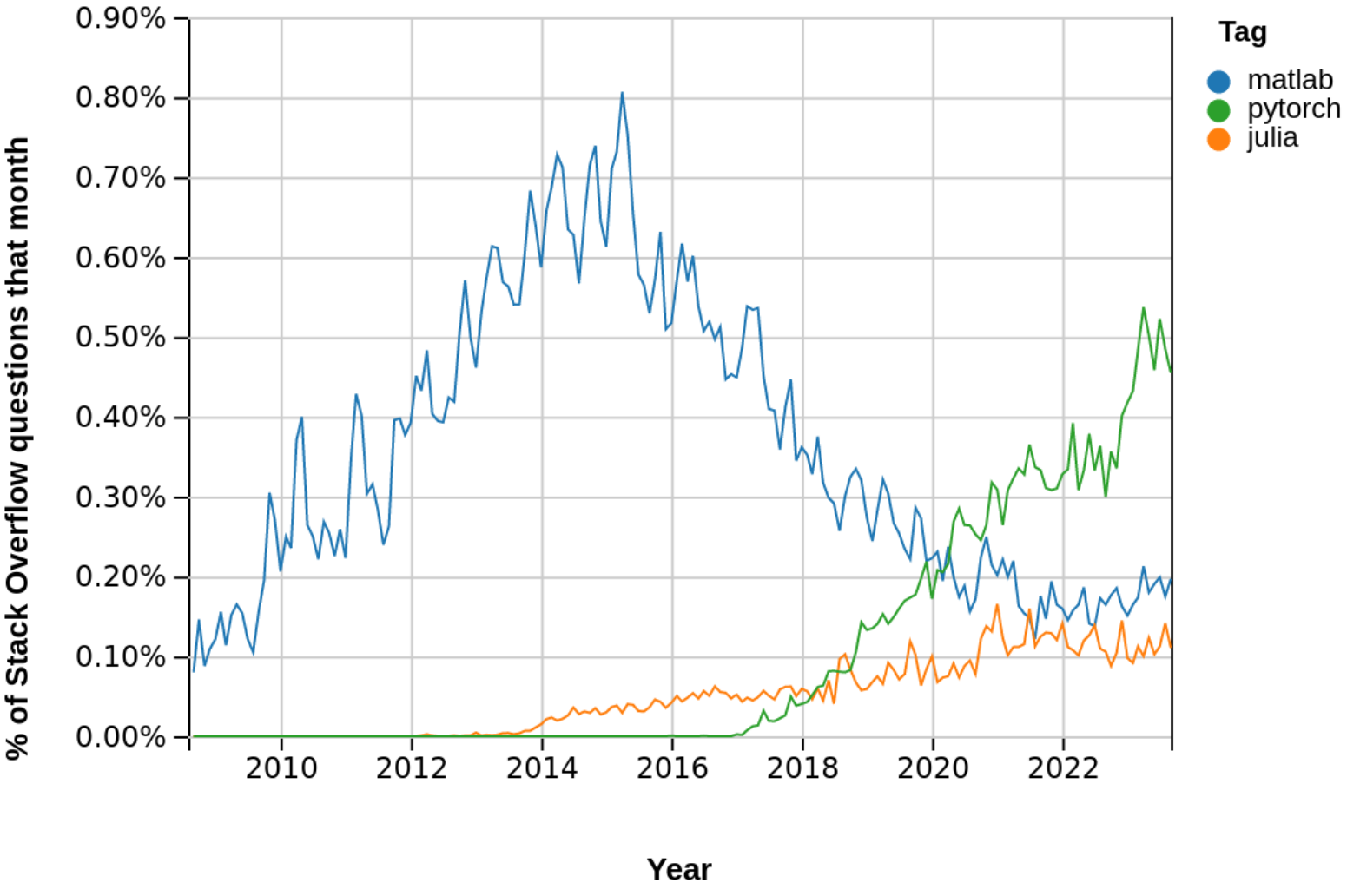


Source: https://losslandscape.com/gallery/

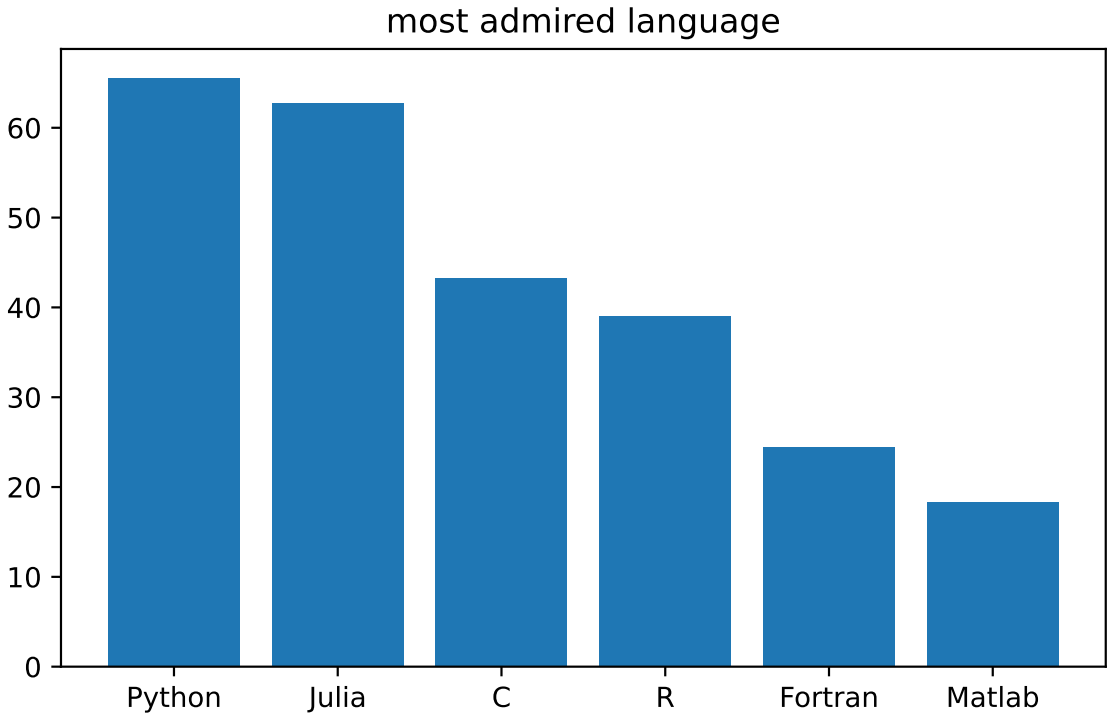
Core elements

- automatic differentiation
- parallelization (CPUs / GPUs / TPUs)
- Compilers / JIT-compilers

Stack Overflow Trends



Stack Overflow 2023 Developer Survey (50 languages)



— https://survey.stackoverflow.co/2023/

Sample code

https://github.com/QuantEcon/imf_october_2023