Python for Data-Science and Machine Learning: Where are things heading?

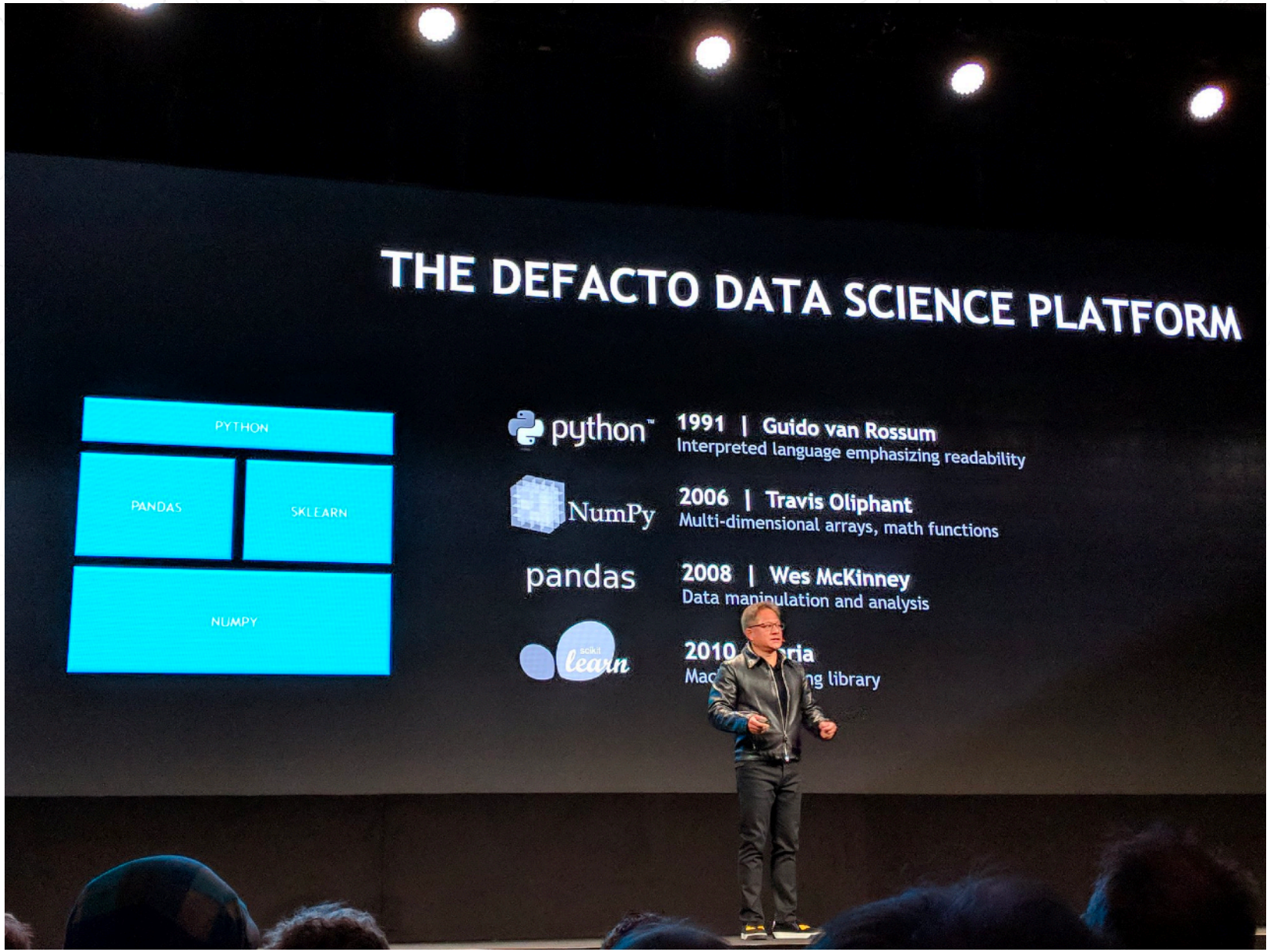
Travis E. Oliphant, PhD

GTC Europe

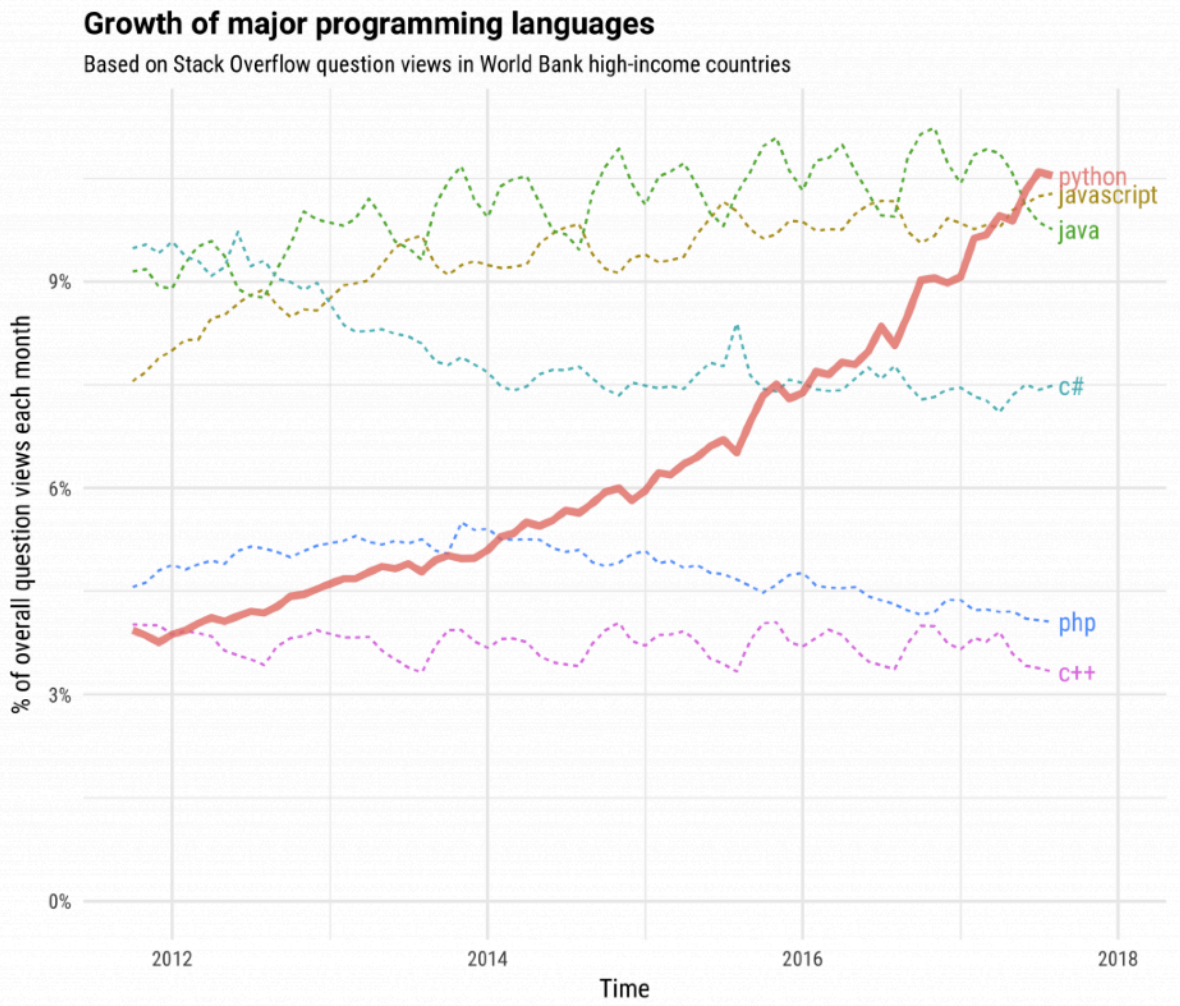
Python is the *de facto* Data Science Platform

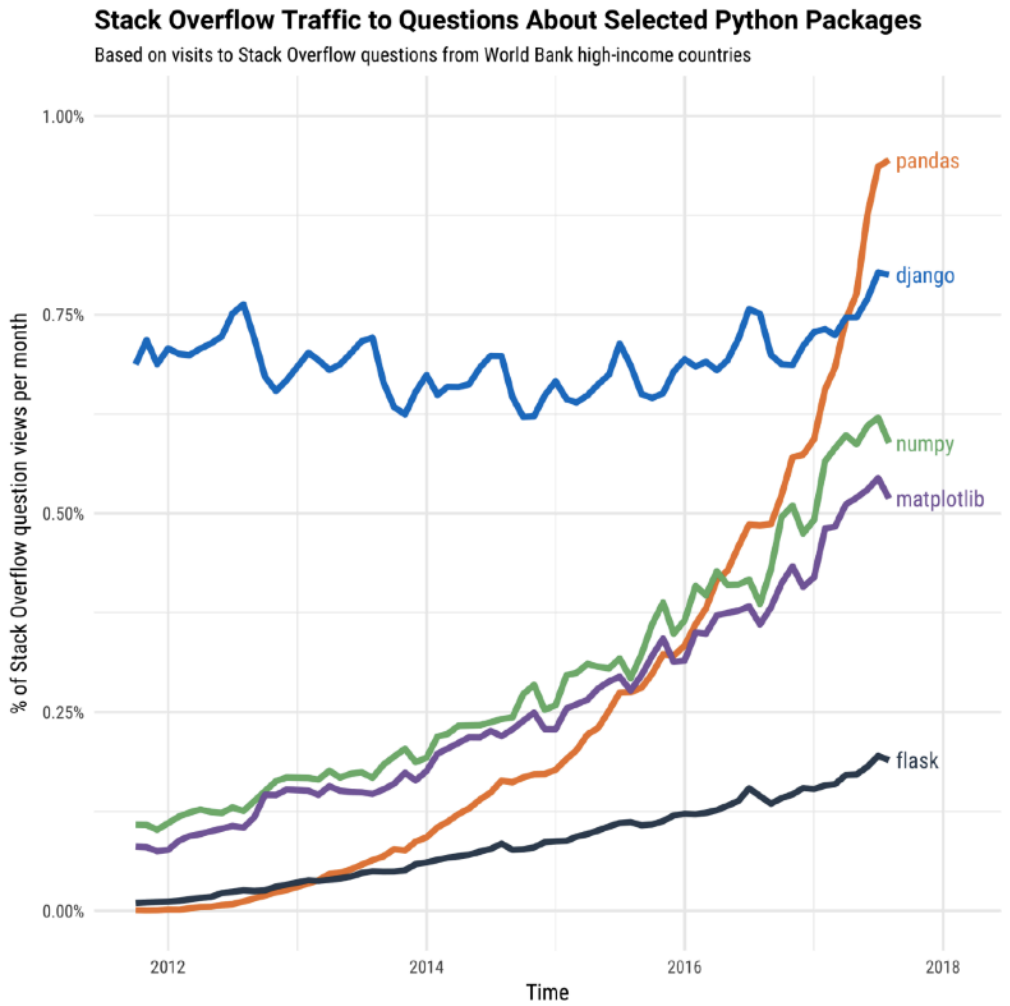
Jensen Huang
NVIDIA CEO

GPU support is now coming.



Python and in particular PyData is Growing

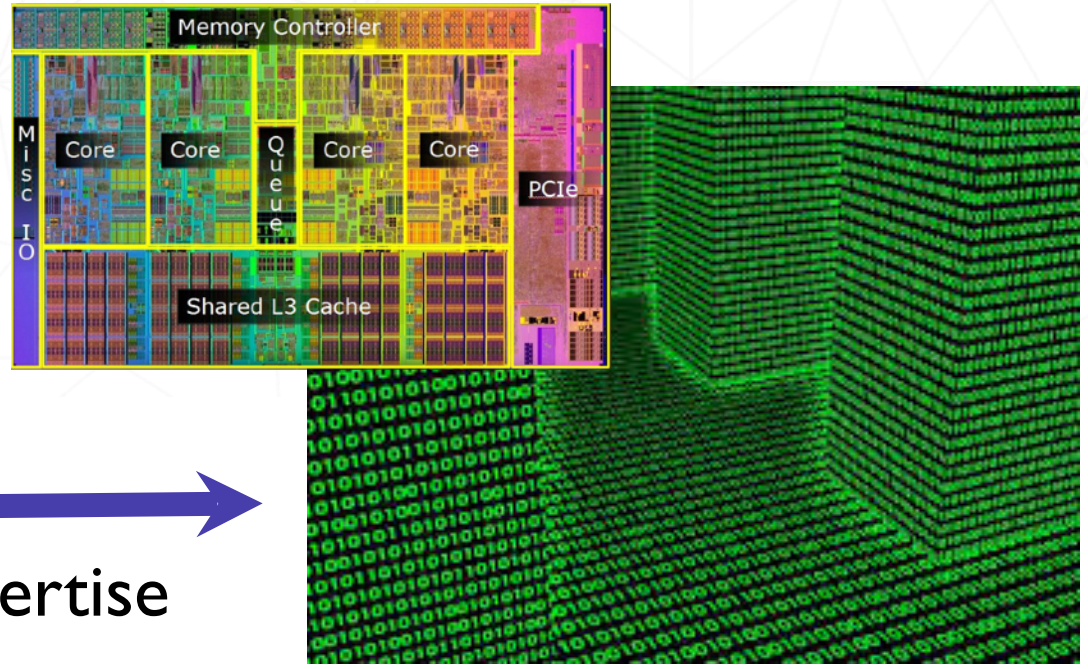




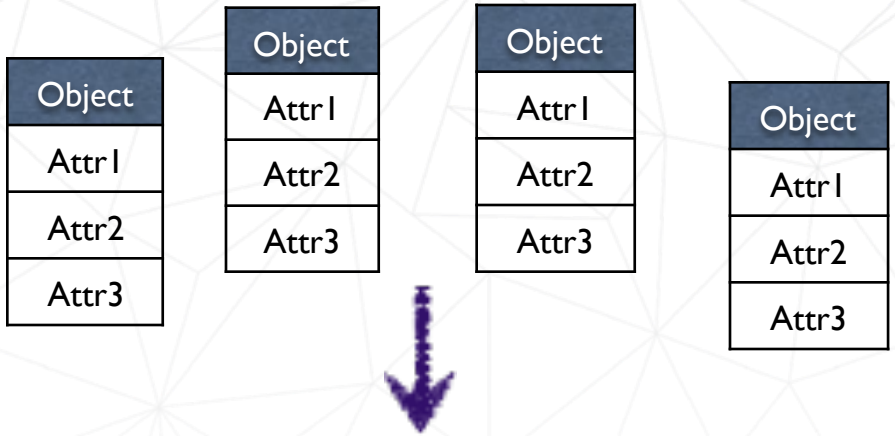
Why? — Array Oriented Computing

Python has been empowering Domain experts to use "vectorized" expressions enable parallelism for 25 years

expertise



Array-oriented computing



- Express domain knowledge directly in arrays (tensors, matrices, vectors) --- easier to teach programming in domain
- Can take advantage of parallelism and accelerators
- Array expressions

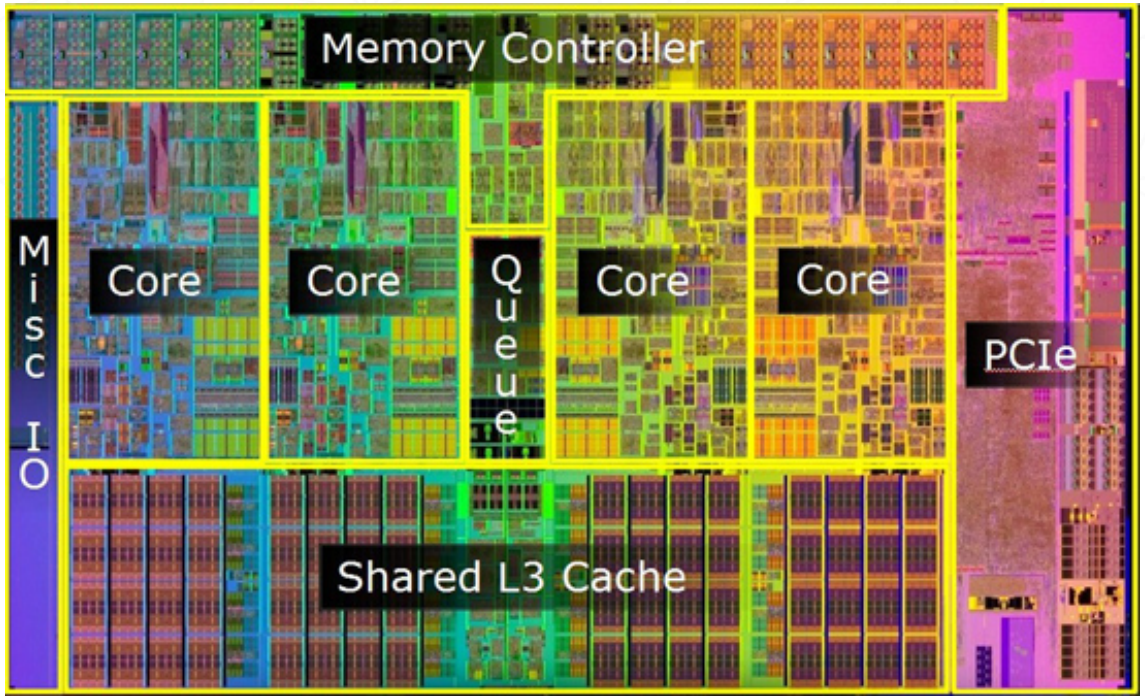
np.max(prices - np.minimum.accumulate(prices))

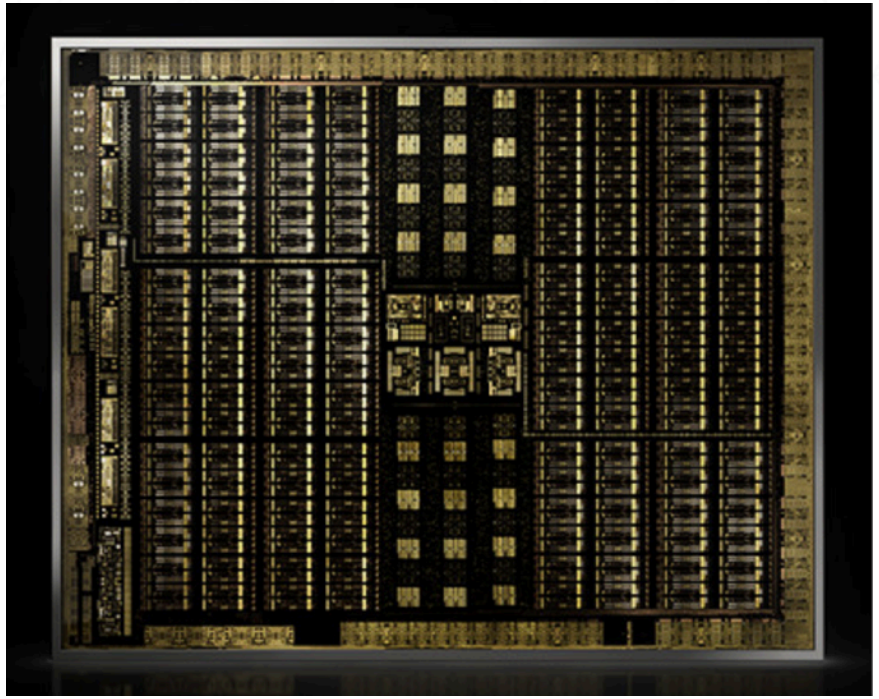
	Attr1	Attr2	Attr3
Object1			
Object2			
Object3			
Object4			
Object5			
Object6			

https://realpython.com/numpy-array-programming/

Benefits of array-oriented

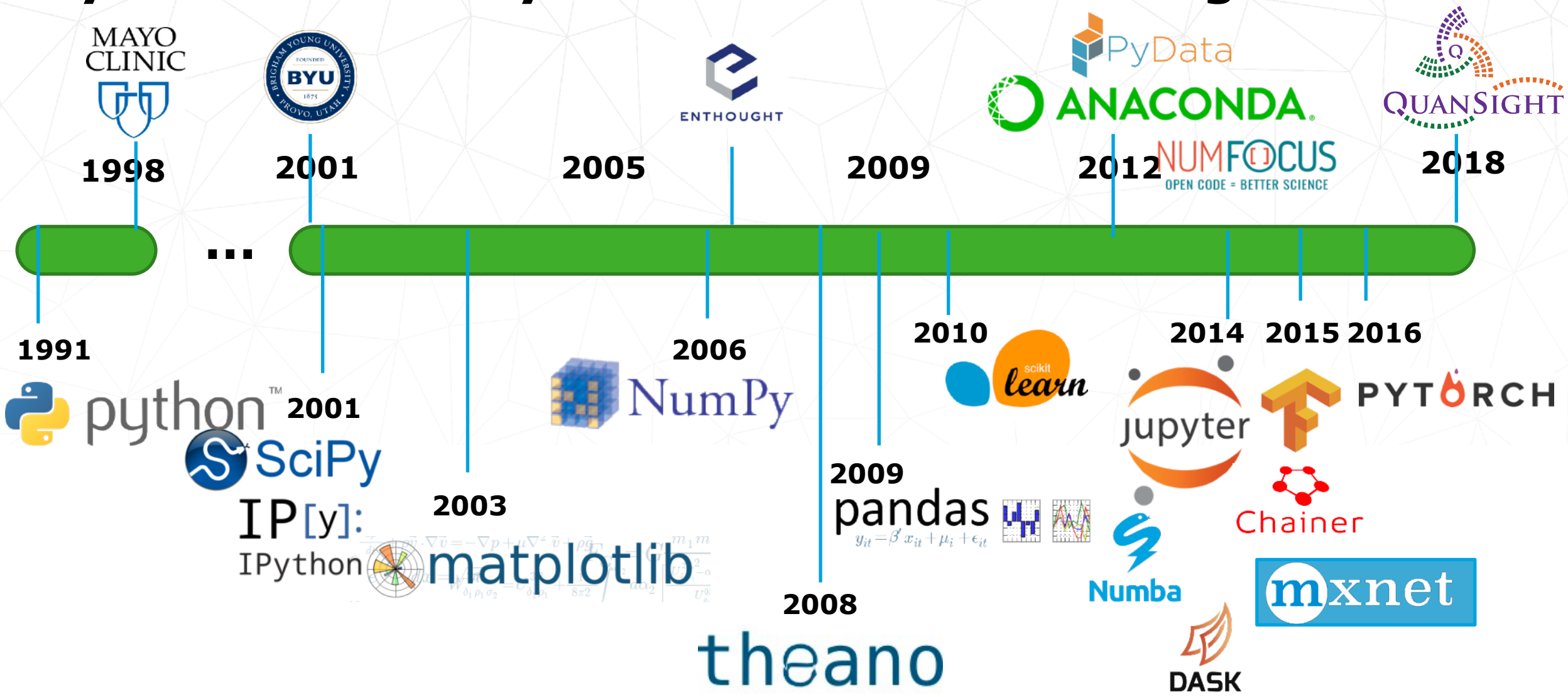
- Today's vector machines (and vector co-processors, or GPUS) were **made** for array-oriented computing.
- The software stack has just not caught up --- starting to with "Tensor" Programming
- There is a reason Fortran remains popular among High Performance groups.





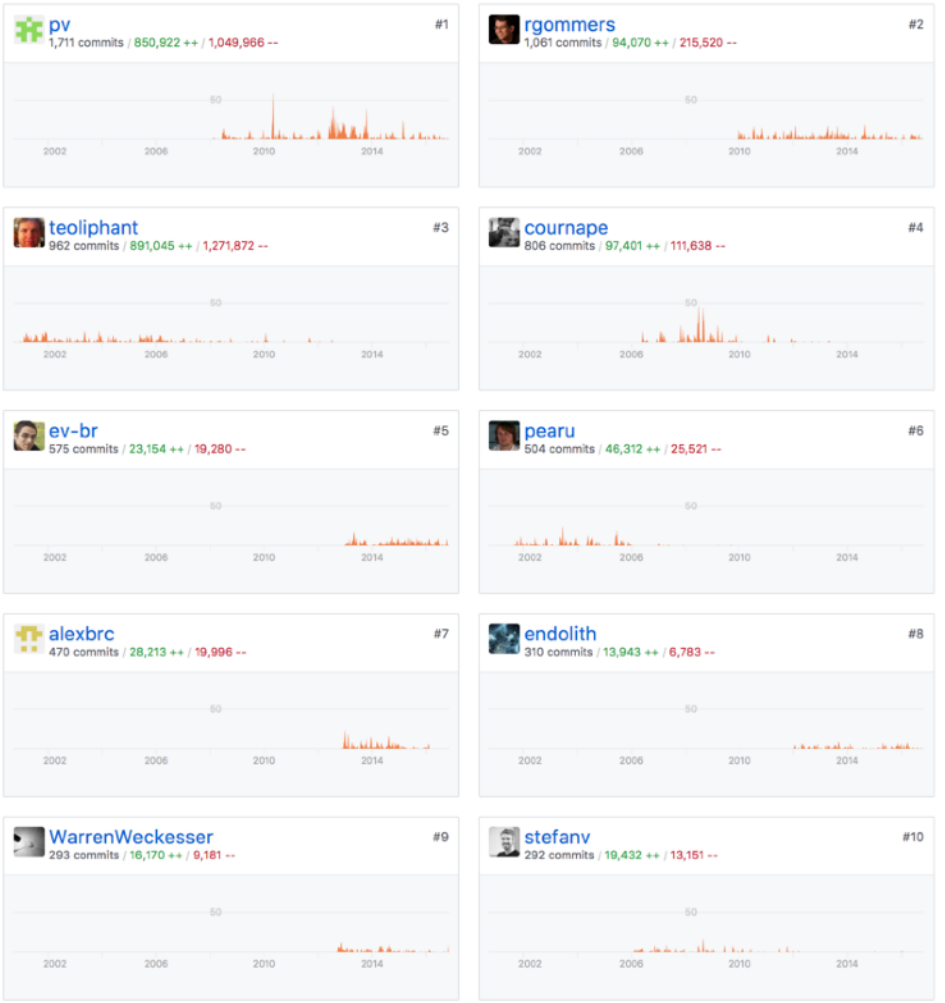
NVIDIA Turing™ architecture

Python Data Analysis and Machine Learning Time-Line



Where it started

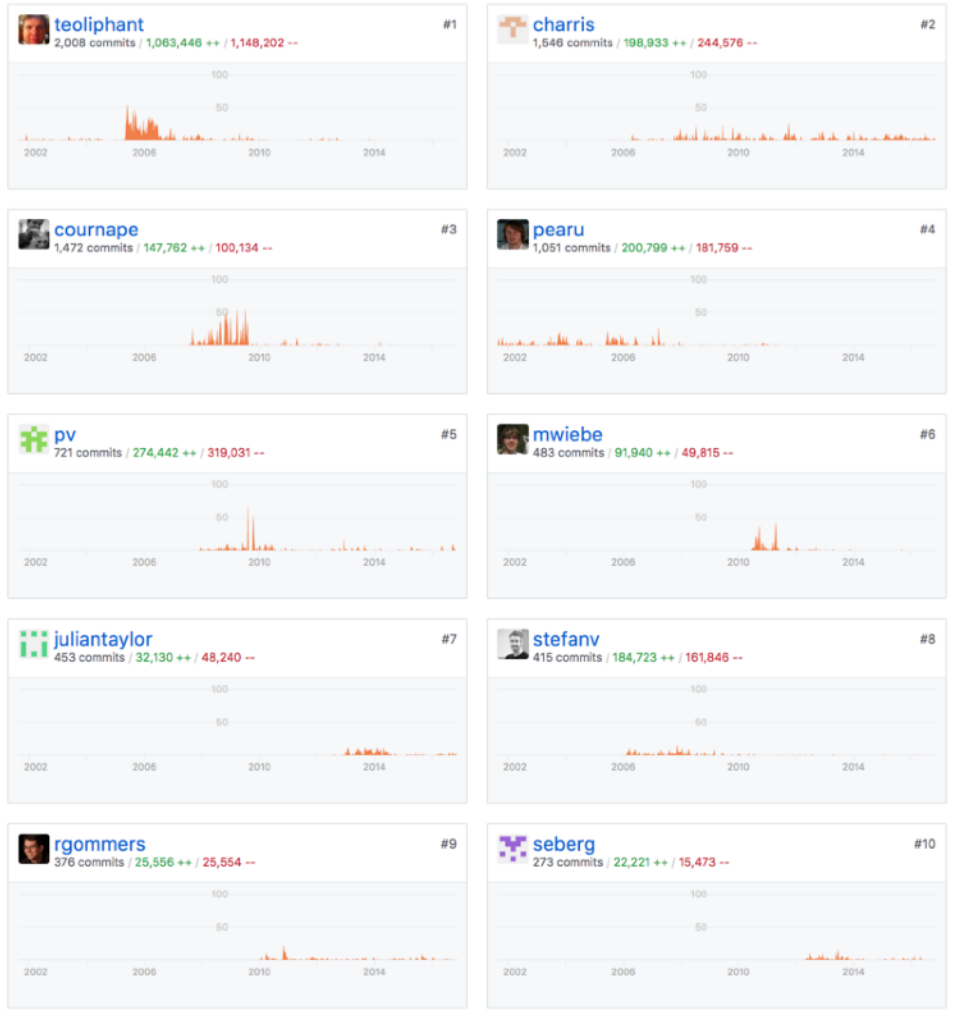
99 releases, 653 contributors



Started as graduate student "procrastination project" (as Multipack) in 1998 and became SciPy in 2001 with the help of colleagues.

Where it led

144 releases, 698 contributors

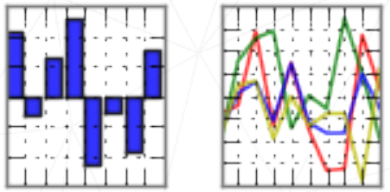


Gave up my chance at tenured academic position in 2005-2006 to bring together the diverging array community in Python and unify Numeric and Numarray.

What amplified data science

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



Created by Wes McKinney. Also, AQR agreed to release this data-frame he started at AQR (while dozens of other data-frames in hedge-funds and investment banks did not get open-sourced)

97 releases, 1292 contributors

Why Python for ML?

Created by David Cournapeau as Google Summer of Code Project and then quickly added to by 100s of researchers around the world. Supported by INRIA.

89 releases, 1187 contributors

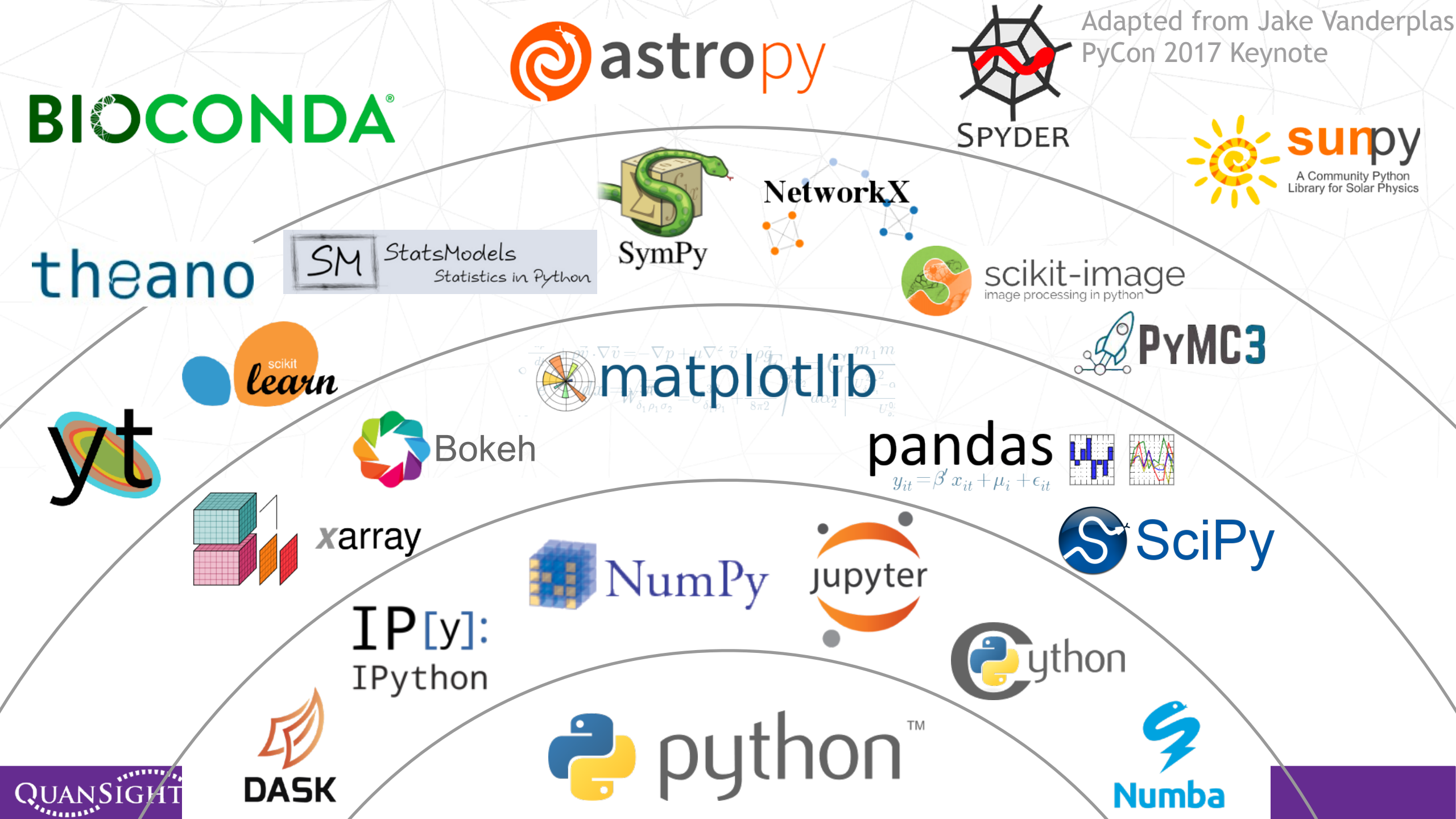
First DL Framework in Python

32 releases, 329 contributors

theano

Built at Université de Montréal by Frédéric Bastien and his students. Many contributors. Forms foundation for PyMC3 and other libraries.

Adapted from Jake Vanderplas
PyCon 2017 Keynote



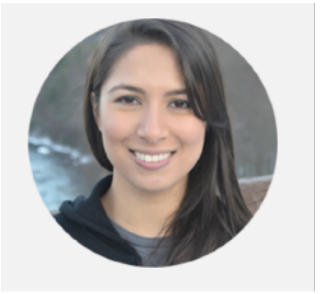
Community-centric organizations are critical to the past and future

NUMFOCUS

OPEN CODE = BETTER SCIENCE

PyData

Founded in 2012 with industry leaders of NumPy, Jupyter, SciPy, and Matplotlib



Board Members selected for 2018

Develop with community
Deploy easily

Scott Collison
CEO

~7 million Anaconda users

CONTINUUM ANALYTICS

renamed

ANACONDA®

Travis Oliphant

Founders

Peter Wang

CONDA® PACKAGES

- Anaconda Repository Curated by Anaconda
- Anaconda Cloud Uploaded by users & organizations
- Anaconda Enterprise Curated by your organization
- conda-forge Curated by the community

conda install <package>

Key advances from Continuum / Anaconda

Better Packaging:

- User-mode
- Cross Language
- Cross Platform
- Variants on the same platform

Allows cleaning up NumPy/SciPy

Compiler for a subset of Python (NPython):

- NumPy Arrays
- Numerical Computing
- Parallel Acceleration
- Generalized Universal Functions
- GPU support

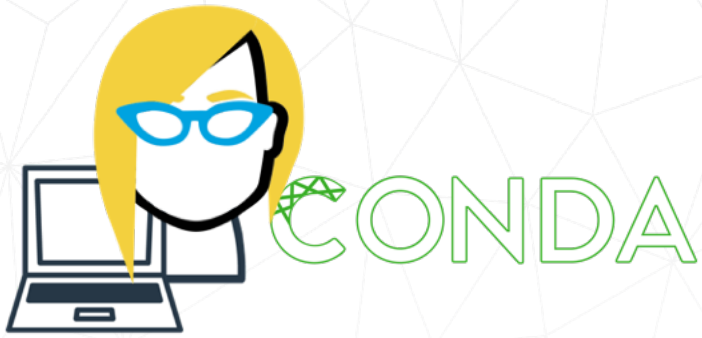
Parallel Scientific Python at Scale

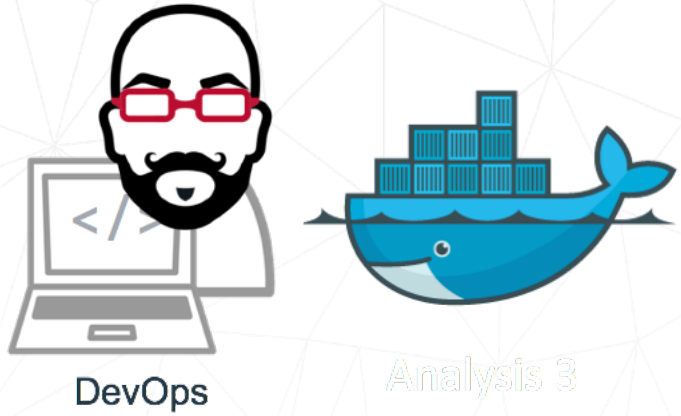
- Resilient and Scalable to 1000s of machines
- Pythonic API
- Dask Array — NumPy
- Dask Dataframe — Pandas
- Dask Delayed — any

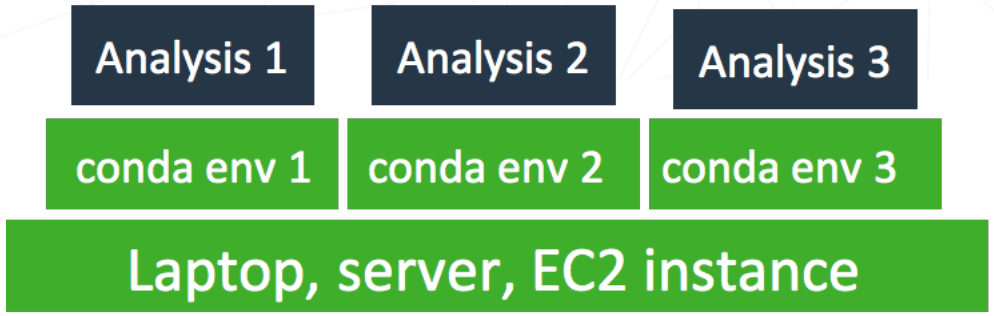
Also see Dask-ML

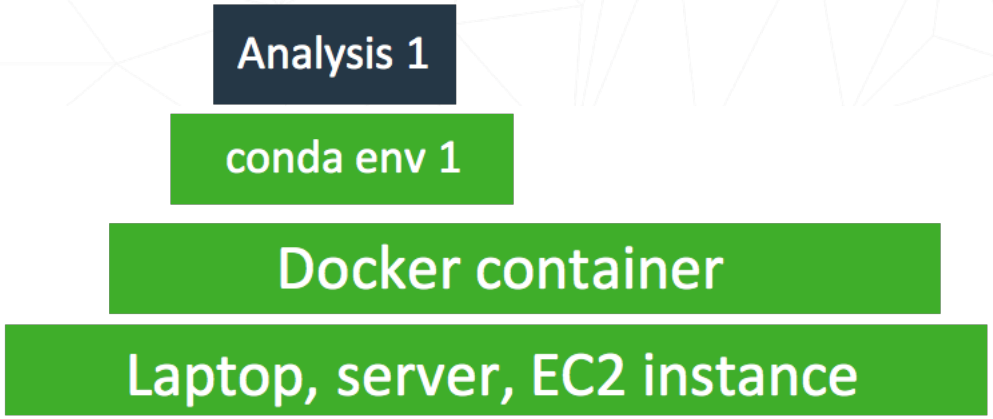
Conda	A cross-platform and language agnostic package and environment manager
Conda Forge	A community-led collection of recipes, build infrastructure, and packages for conda.
Conda Environments	Custom isolated software sandboxes to allow easy reproducibility and sharing of data-science work.
Anaconda.org	Web-site for freely hosting public packages and environments. Example of conda repository.

Conda eases rapid deployment



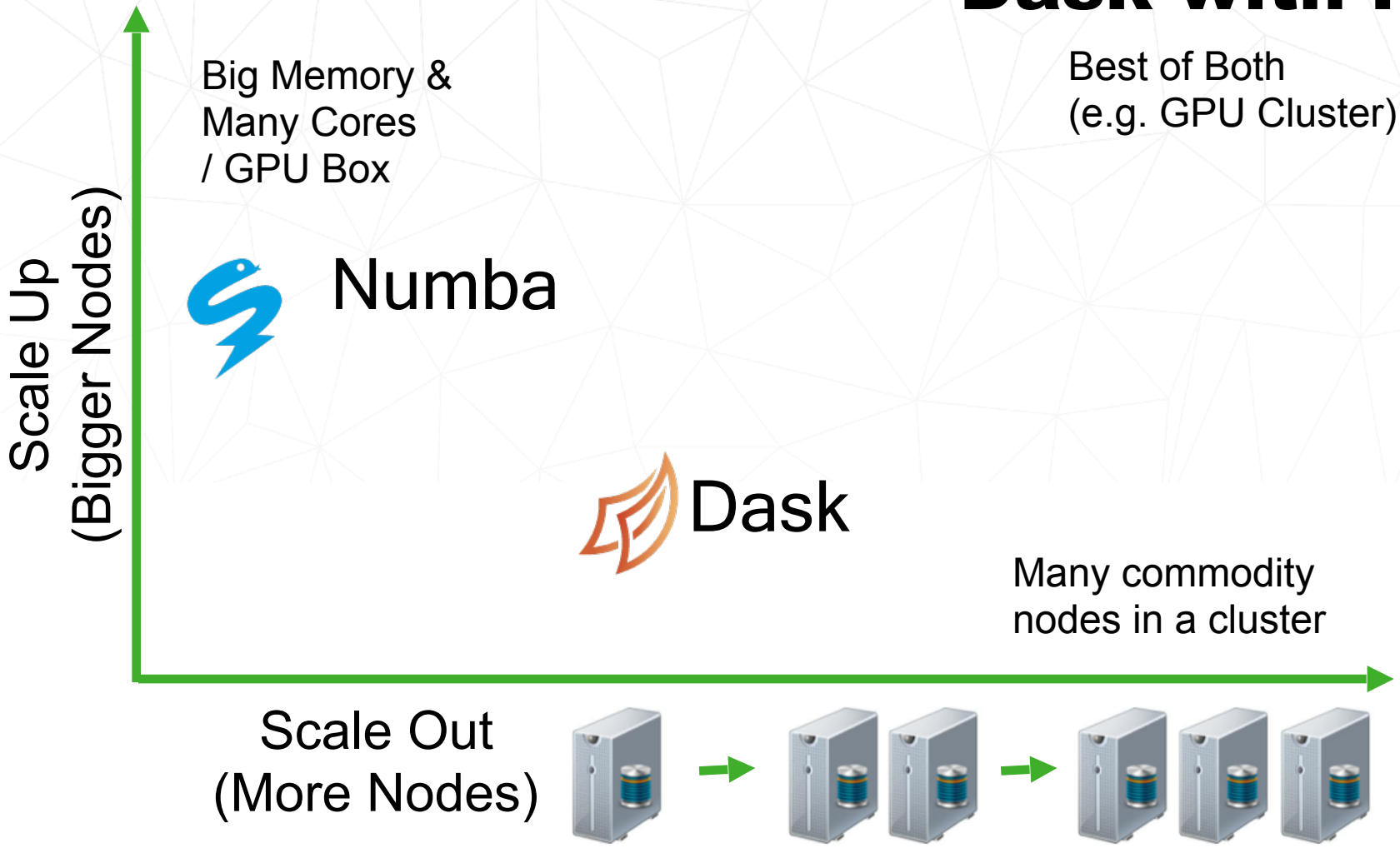






Development → Deployment

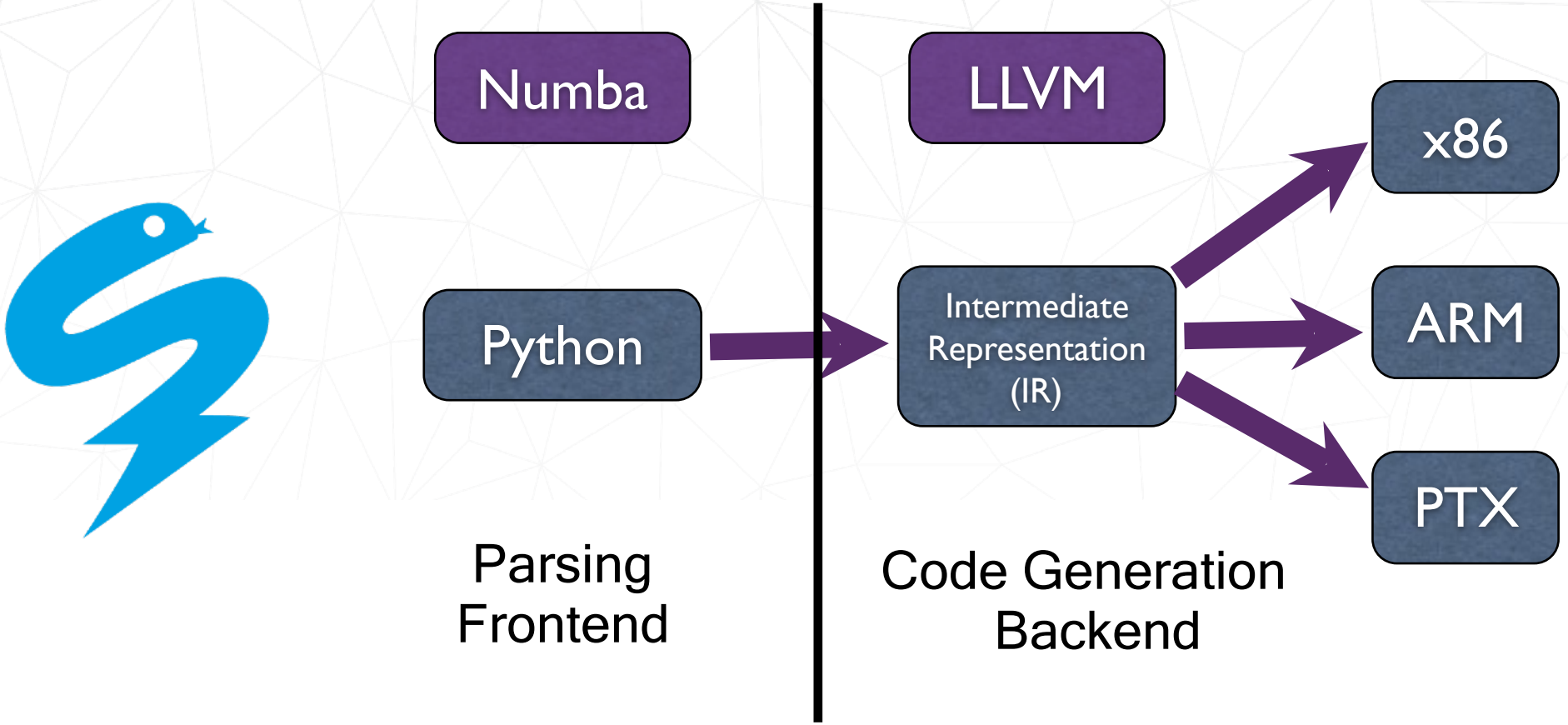
Scale Up vs Scale Out



Dask with Numba

Best of Both
(e.g. GPU Cluster)

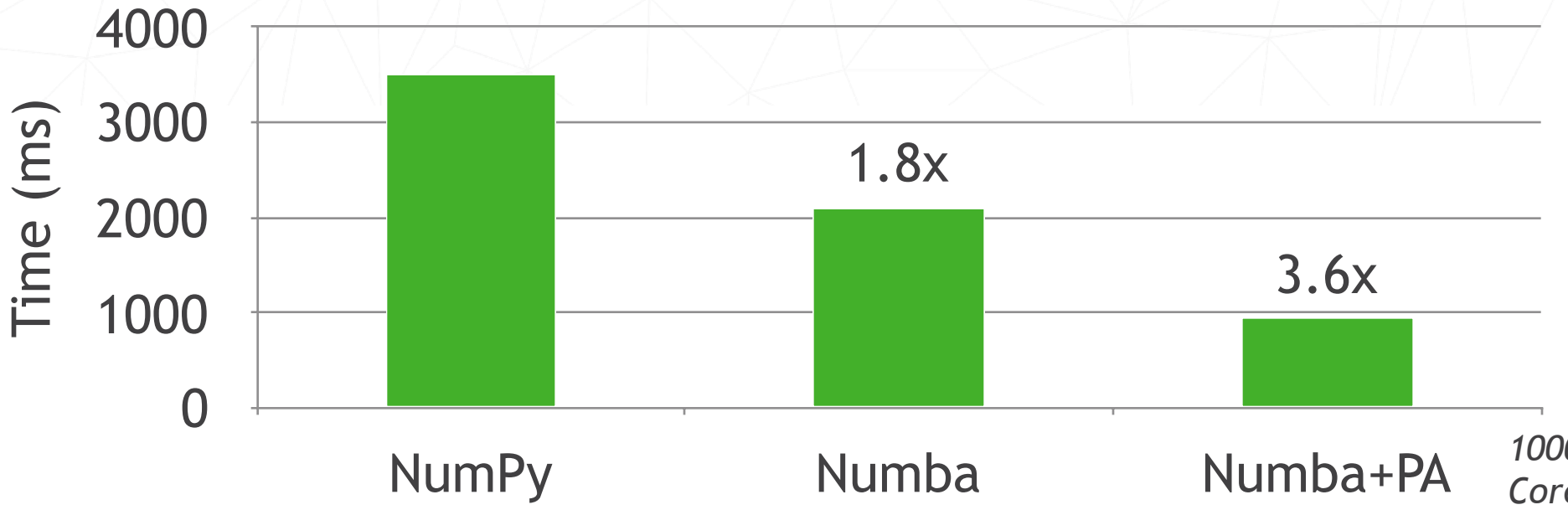
Numba (compile Python to CPUs and GPUs)




```
conda install numba
```

ParallelAccelerator: Example #1

[3]:
```
@numba.jit(nopython=True, parallel=True)
def logistic_regression(Y, X, w, iterations):
    for i in range(iterations):
        w -= np.dot(((1.0 / (1.0 + np.exp(-Y * np.dot(X, w))) - 1.0) * Y), X)
    return w
```



1000000x10 input,
Core i7 Quad Core CPU

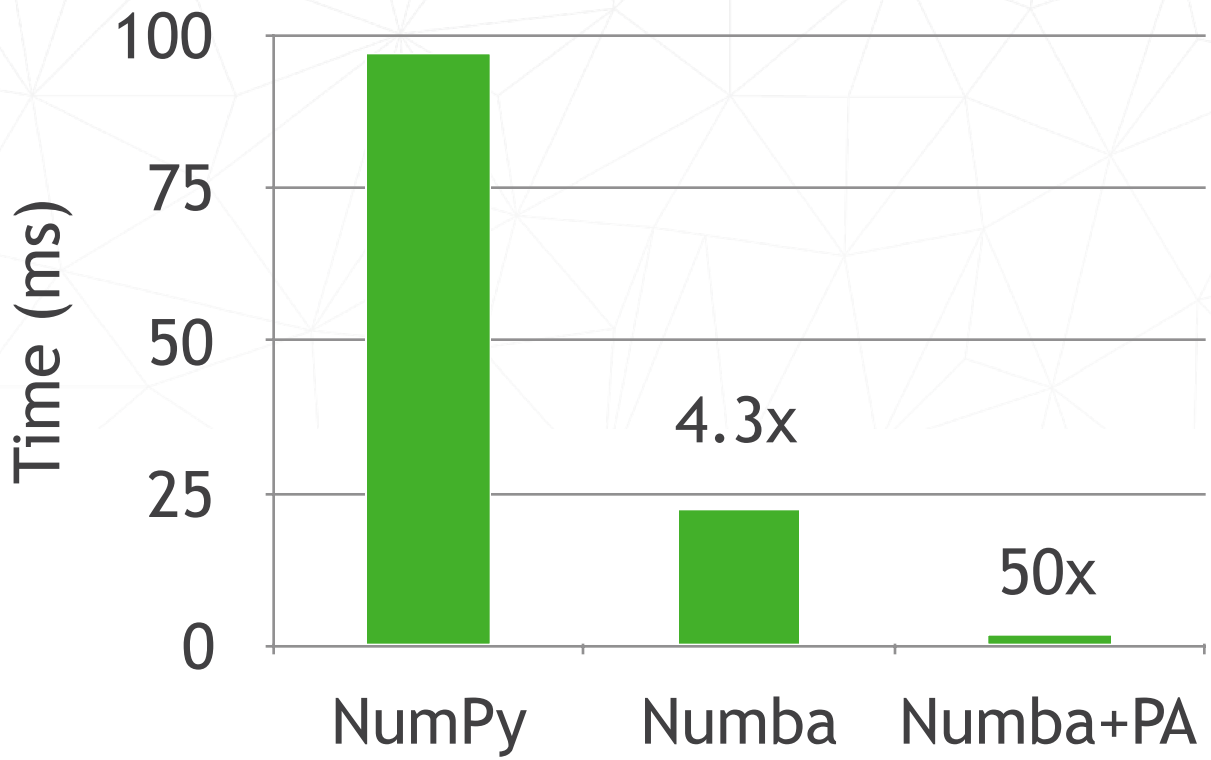
ParallelAccelerator: prange()

In [44]:
```
@numba.jit(nopython=True, parallel=True)
def normalize(x):
    ret = np.empty_like(x)

    for i in numba.prange(x.shape[0]):
        acc = 0.0
        for j in range(x.shape[1]):
            acc += x[i,j]**2

        norm = np.sqrt(acc)
        for j in range(x.shape[1]):
            ret[i,j] = x[i,j] / norm

    return ret
```



1000000x10 input,
Core i7 Quad Core CPU

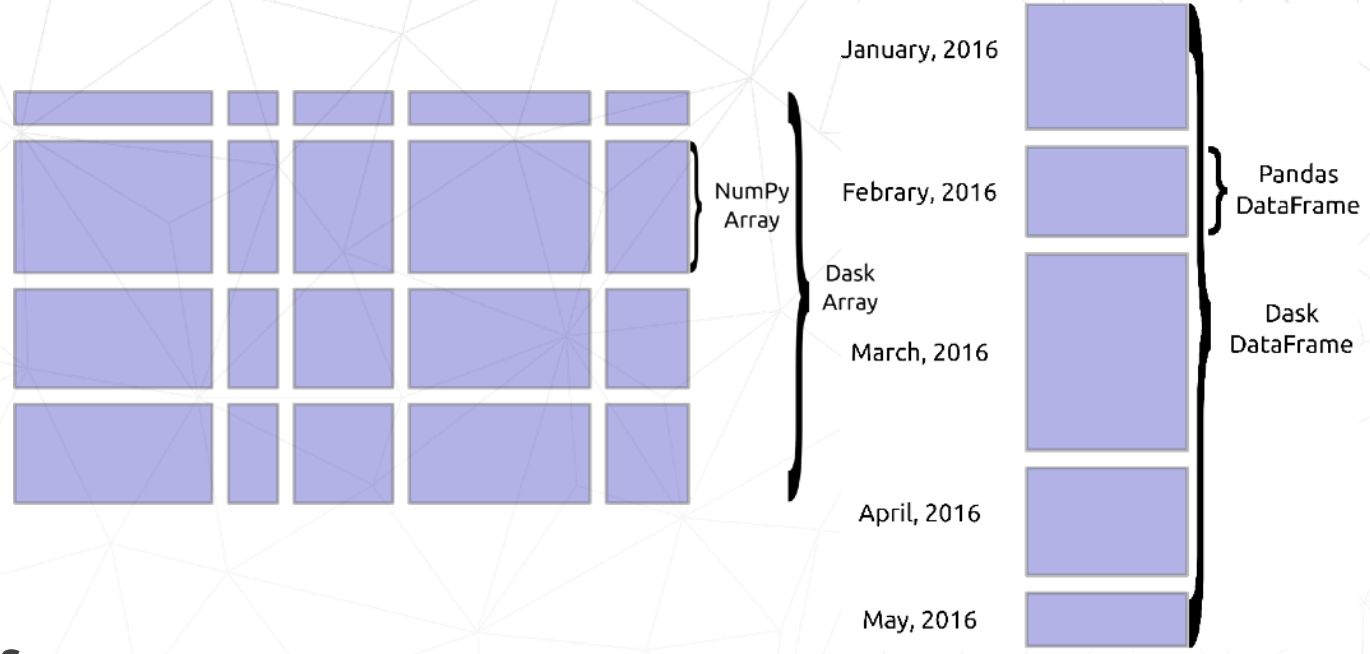
Other Numba topics

CUDA Python — write general NVIDIA GPU kernels with Python
Device Arrays — manage memory transfer from host to GPU
Streaming — manage asynchronous and parallel GPU compute streams
CUDA Simulator in Python — to help debug your kernels
ROCm — support for AMD ROCm GPUs and APUs
Pyculib — access to cuFFT, cuBLAS, cuSPARSE, cuRAND, CUDA Sorting

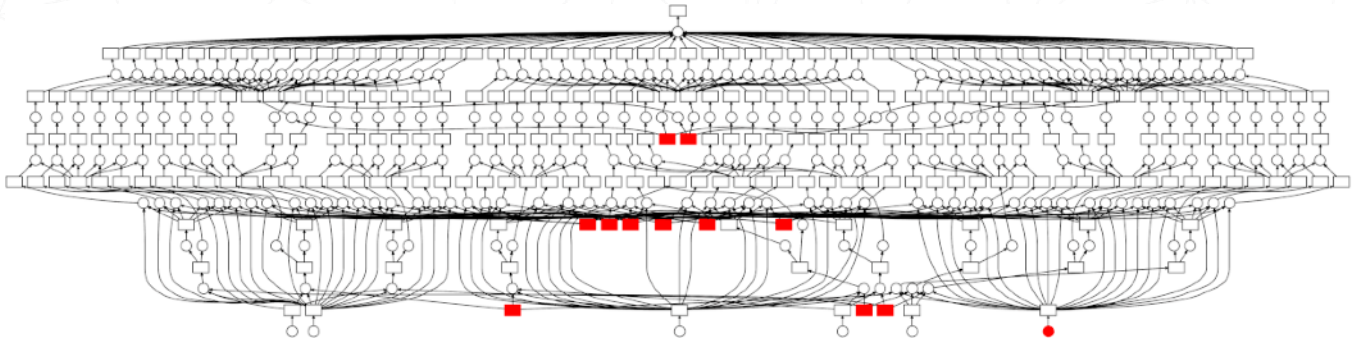
https://github.com/ContinuumIO/gtc2017-numba

DASK

- **Parallelizes NumPy, Pandas, SKLearn**
  - Satisfies subset of these APIs
  - Uses these libraries internally
  - Co-developed with these teams



- **Task scheduler supports custom algorithms**
  - Parallelize existing code
  - Build novel real-time systems
  - Arbitrary task graphs with data dependencies
  - Same scalability



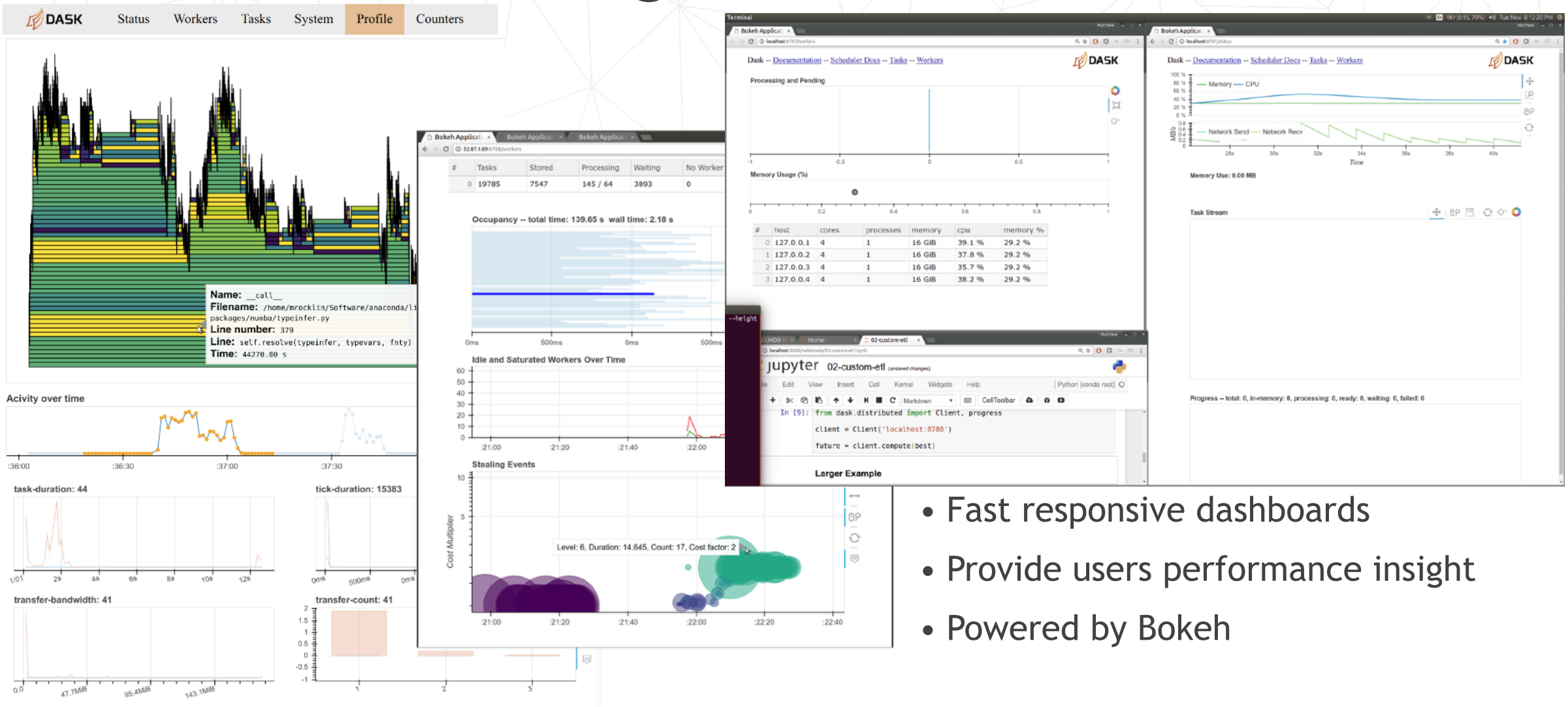
Dask Scales Up

- Thousand node clusters
  - Cloud computing
  - Super computers
- Gigabyte/s bandwidth
- 200 microsecond task overhead

Dask Scales Down (the median cluster size is one)

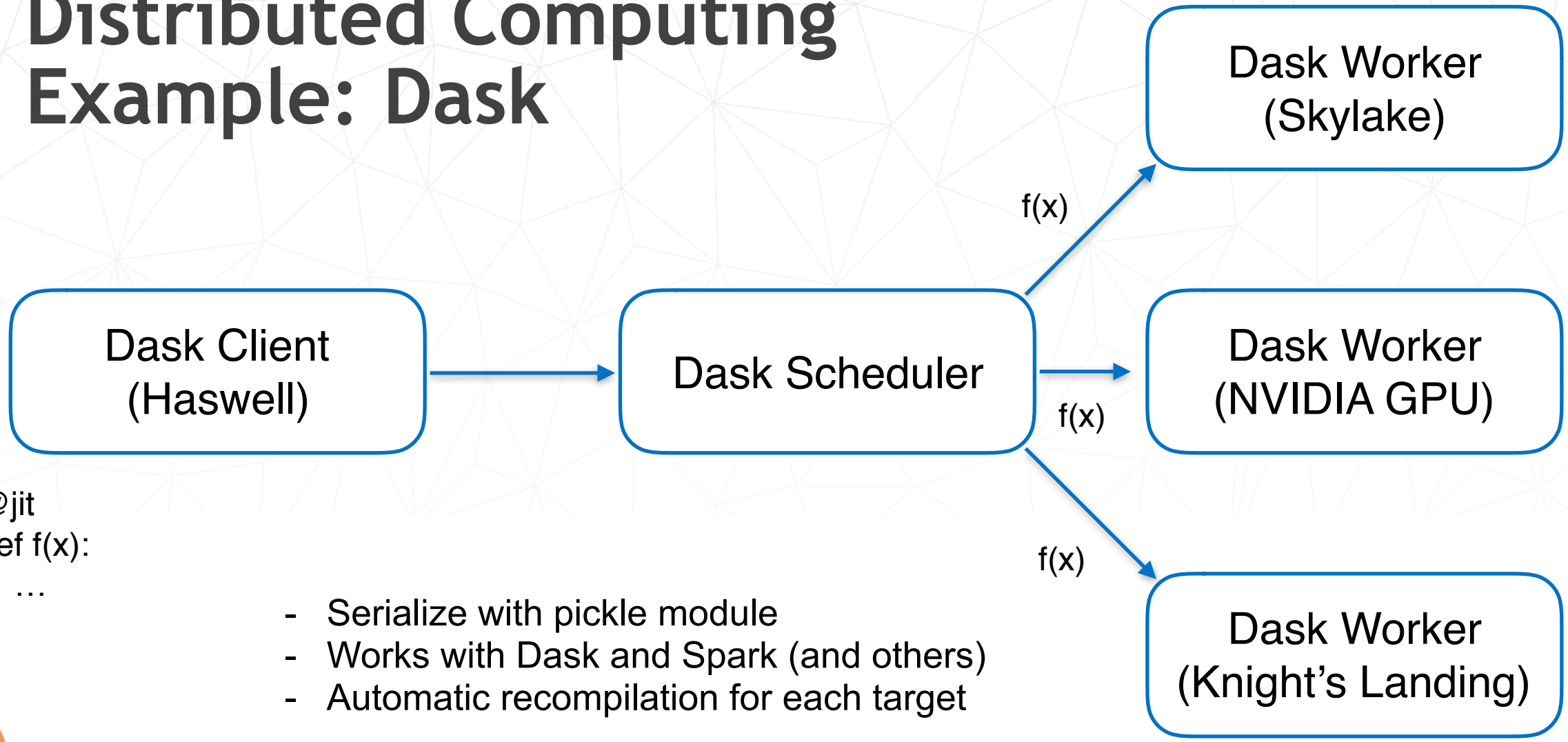
- Can run in a single Python thread pool
- Almost no performance penalty (microseconds)
- Lightweight
  - Few dependencies
  - Easy install

Beautiful Diagnostic Dashboards



- Fast responsive dashboards
- Provide users performance insight
- Powered by Bokeh

Distributed Computing Example: Dask

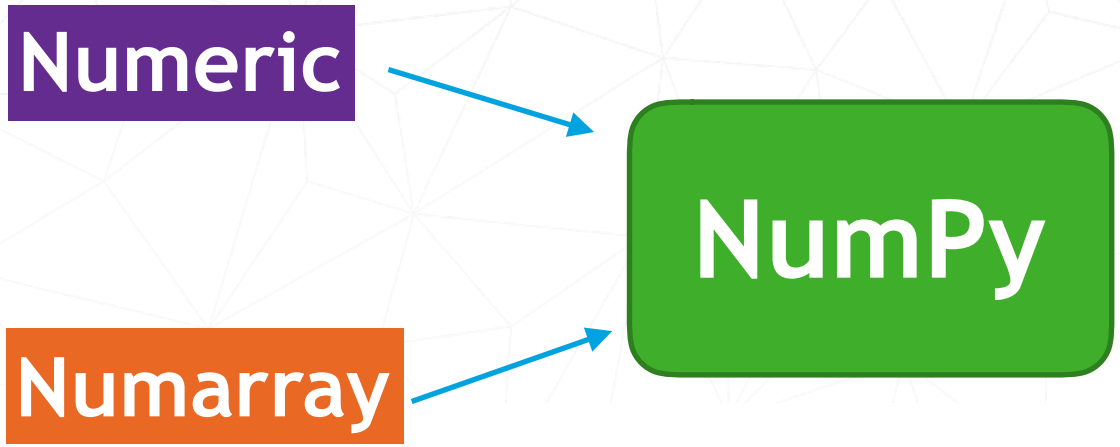



```
@jit
def f(x):
    …
```

- Serialize with pickle module
- Works with Dask and Spark (and others)
- Automatic recompilation for each target

Rise of the Machine Learning Platforms

NumPy was created to unify array objects in Python and unify PyData community



I essentially sacrificed tenure at a University to write NumPy and unify array objects.

Explosion of ML Frameworks and libraries

Caffe2

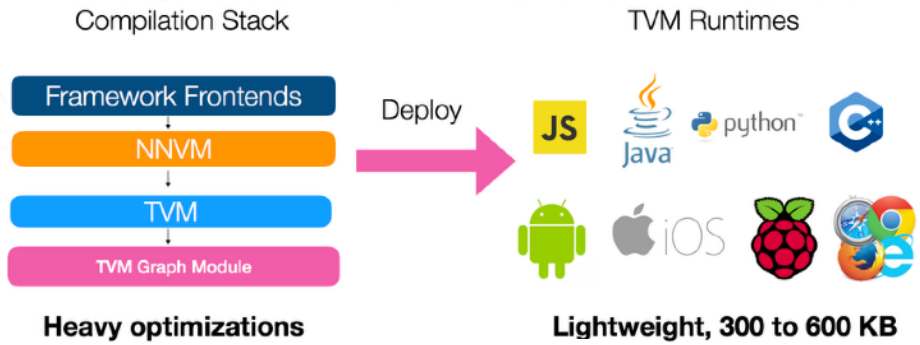
scikit learn

H_2O.ai

dmlc XGBoost

mxnet

TVM/NNVM



Chainer

PYTORCH

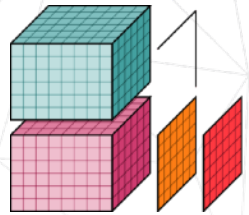
Microsoft CNTK

http://deeplearning.net/software_links/

http://scikit-learn.org/stable/related_projects.html

https://github.com/josephmisiti/awesome-machine-learning#python-general-purpose

Array-like objects everywhere



Sparse Arrays

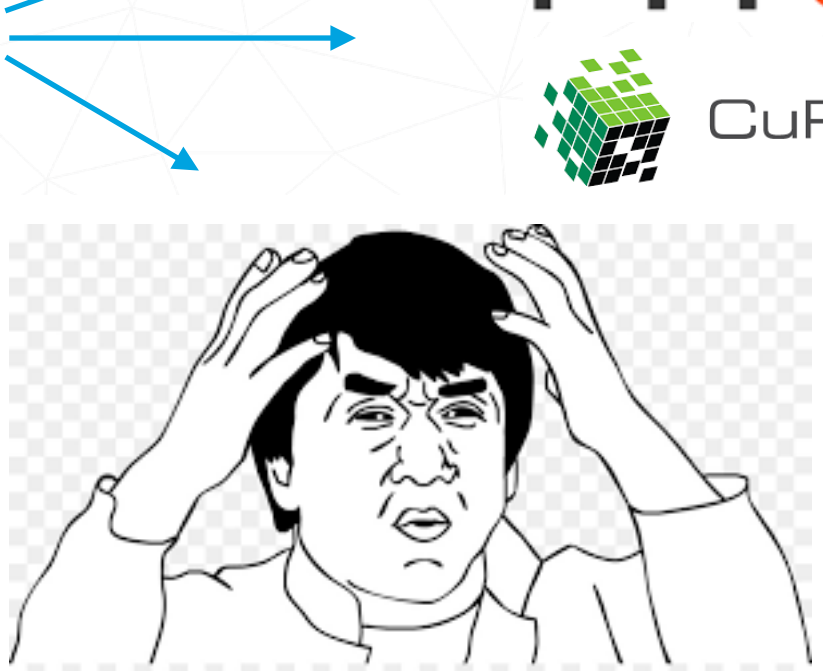
CUDArray

We have a “divided” community again!

Numeric

Numarray

NumPy



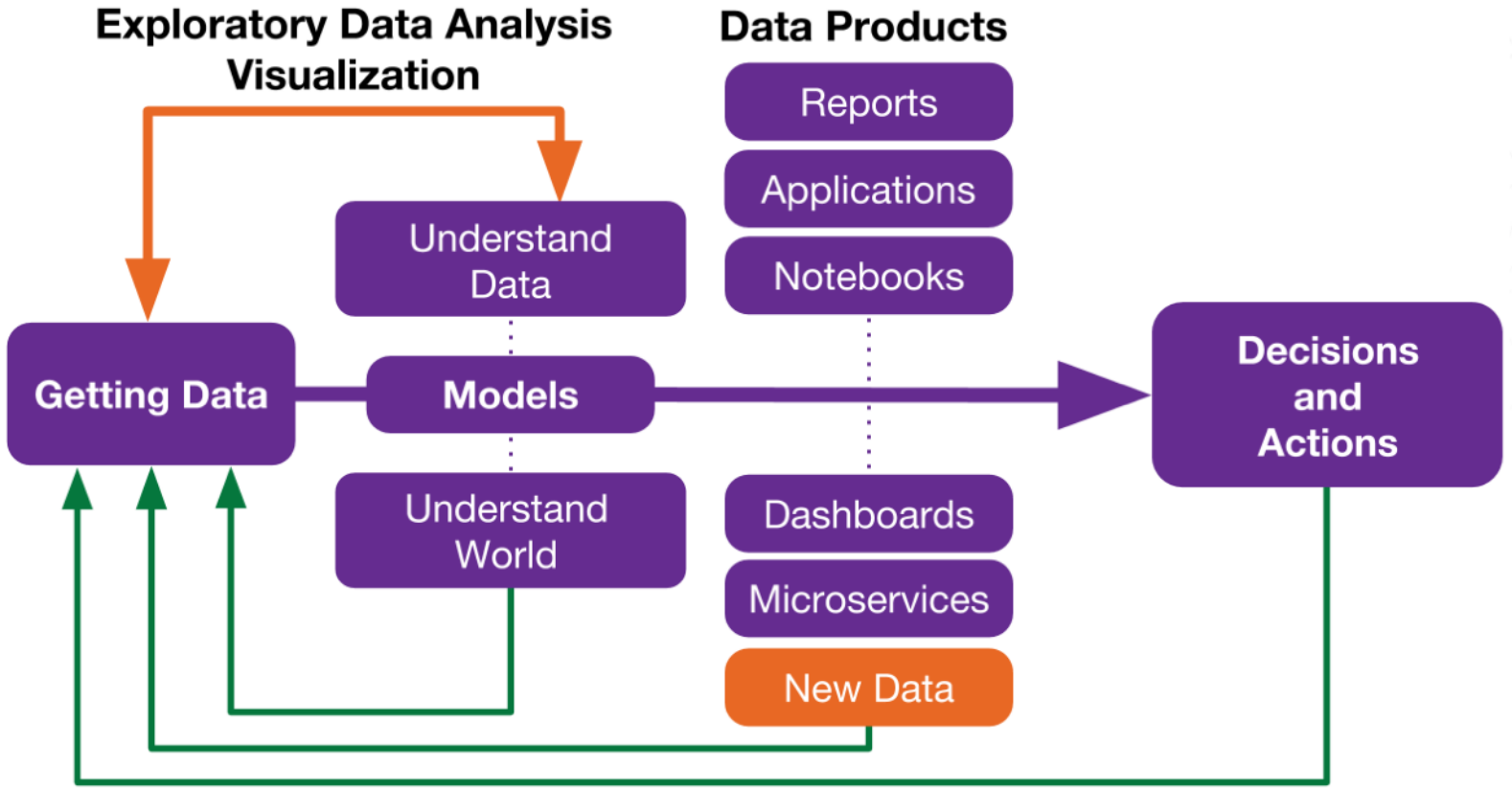
Now What?

"The best way to predict the future is to create it”

Abraham Lincoln

Peter Drucker

Build and Connect Companies and Communities to Solve Challenging Problems with Data



Continuing my quest to find more ways to pay developers to work on open source!

Open Source Partnerships

Prioritize Your Needs in Open Source

(save $$$ by leveraging open-source in a way that keeps using the OSS community instead of by-passing it or fighting it)

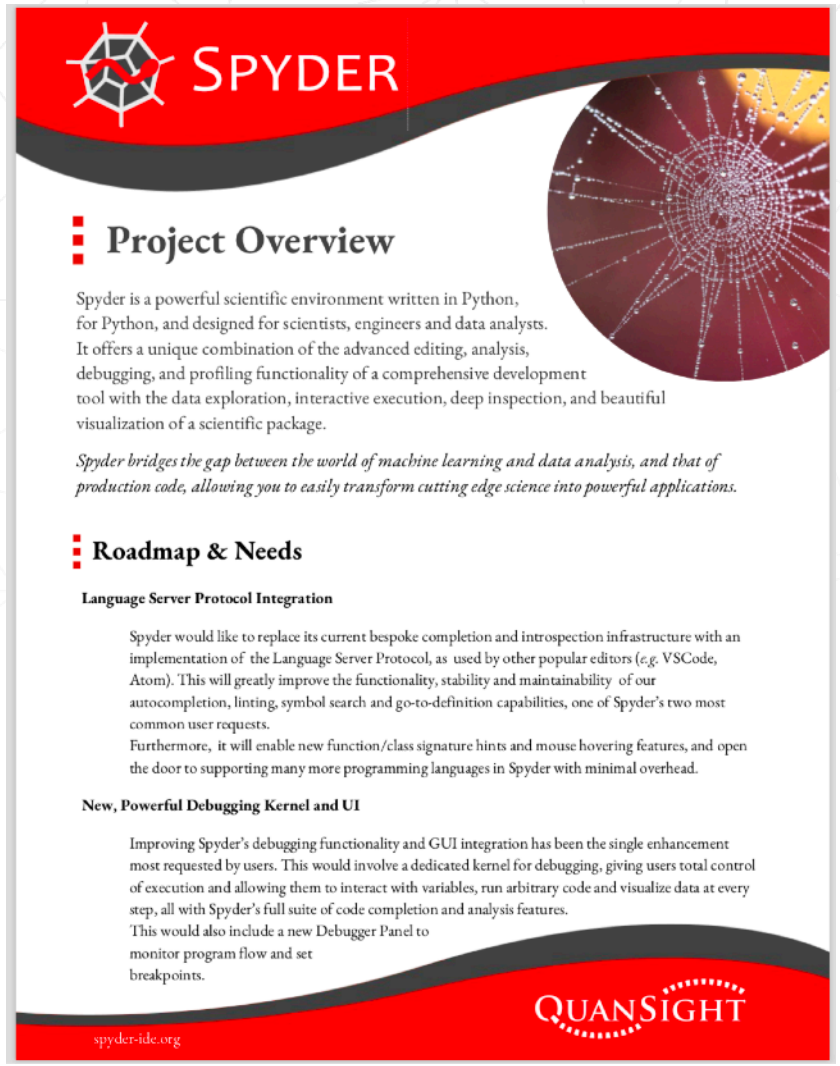
Hire from the Community

(good people flock to good projects — we help you attract and retain them)

Get Open Source Support

(Help selecting projects to depend on, SLAs for security and bug fixes, community health monitoring, expert help and support)

Open Source Directions

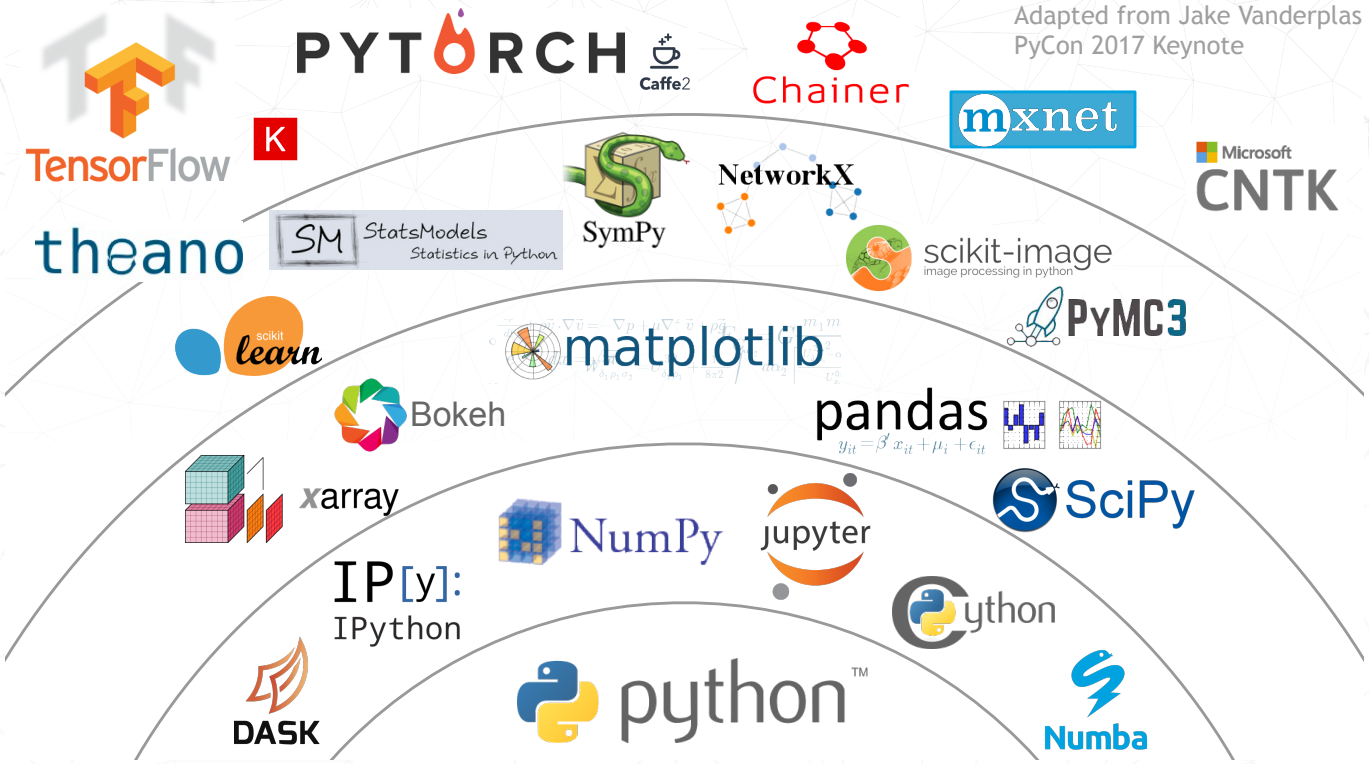


Webinar series to promote and encourage open-source roadmaps and learn where the projects you use are heading.

QUANSIGHT LABS

Sustaining the Future

Open-source innovation and maintenance around the entire data-science and AI workflow.

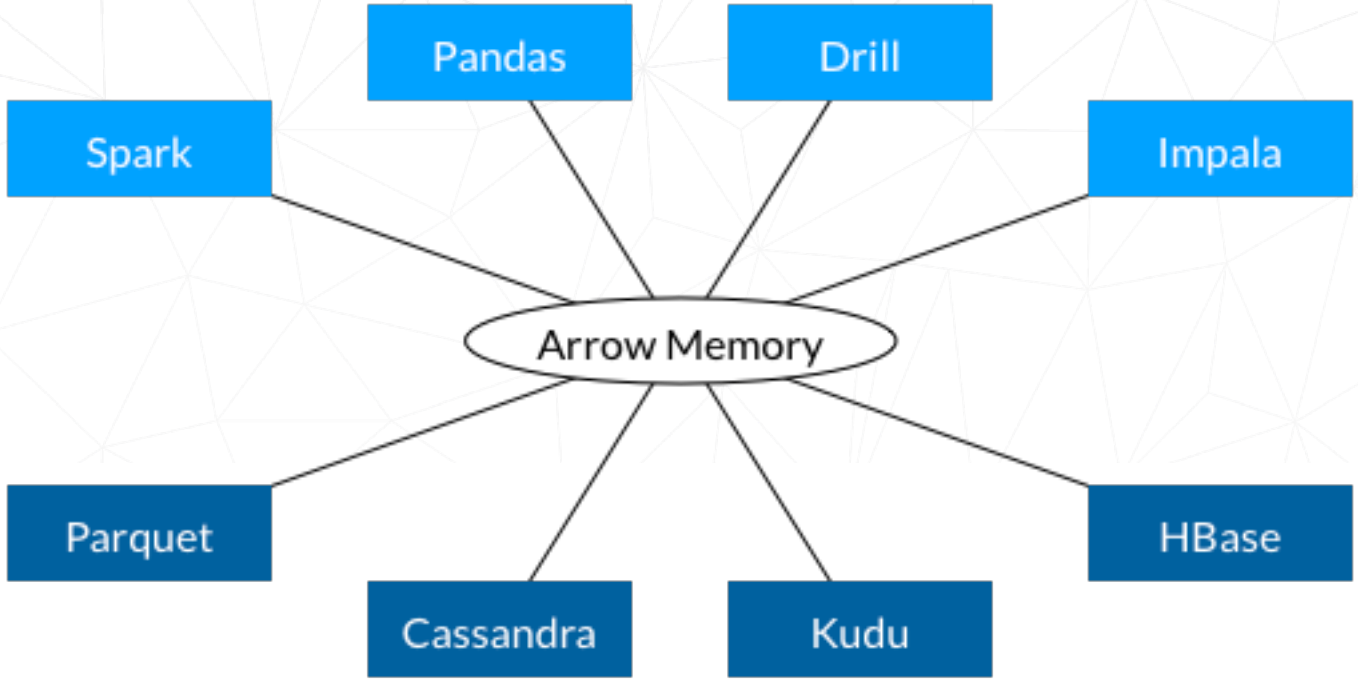


- **NumPy ecosystem maintenance (fund developers)**
- **Improve connection of NumPy to ML Frameworks**
- **GPU Support for NumPy Ecosystem**
- **Improve foundations of Array computing**
- **JupyterLab**
- **Data Catalog standards**
- **Packaging (conda-forge, PyPA, etc.)**

Partnered with NumFOCUS and Ursa Labs (supporting Apache Arrow)

uarray — unified array interface and symbolic NumPy
xnd — re-factored NumPy (low-level cross-language libraries for N-D (tensor) computing)

Apache Arrow

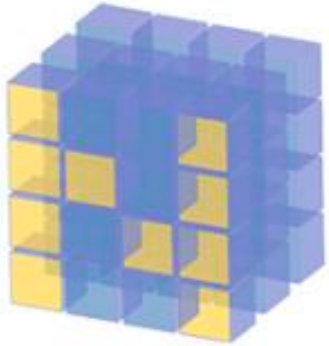


Apache Arrow is a cross-language development platform for in-memory data. It specifies a standardized language-independent columnar memory format for flat and hierarchical data, organized for efficient analytic operations on modern hardware.

XND

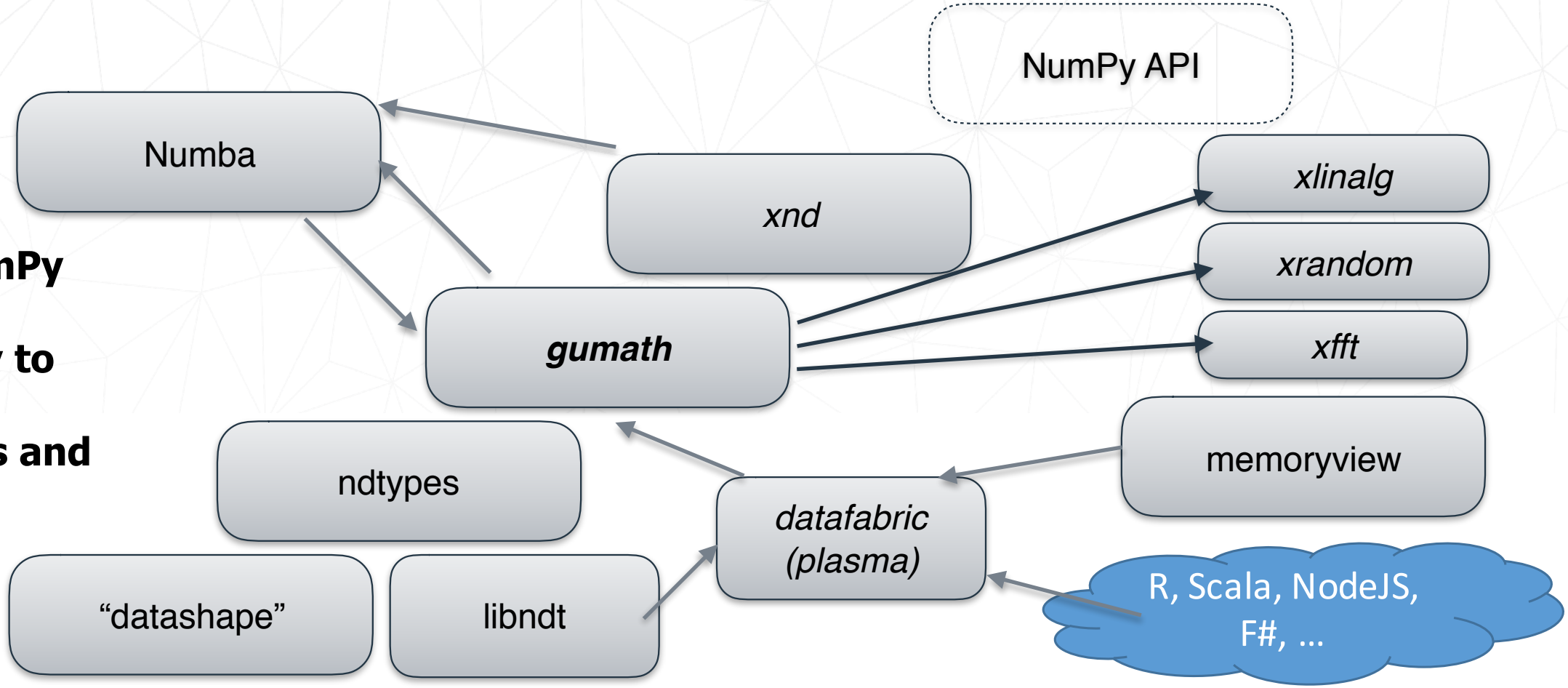
pandas

$$y_i t = \beta' x_{it} + \mu_i + \epsilon_{it}$$



NumPy

Cross language "NumPy" backend



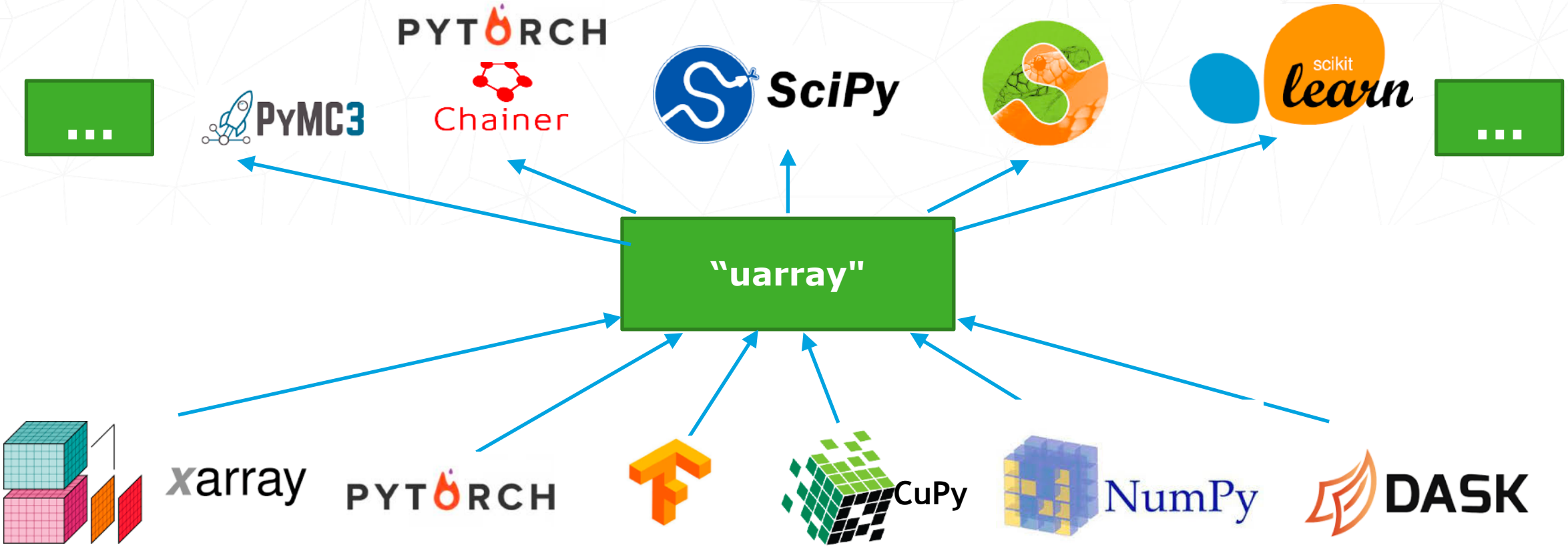
Bring NumPy and Array Capability to other languages and run-times

Unified Array Interface

Just started Project!

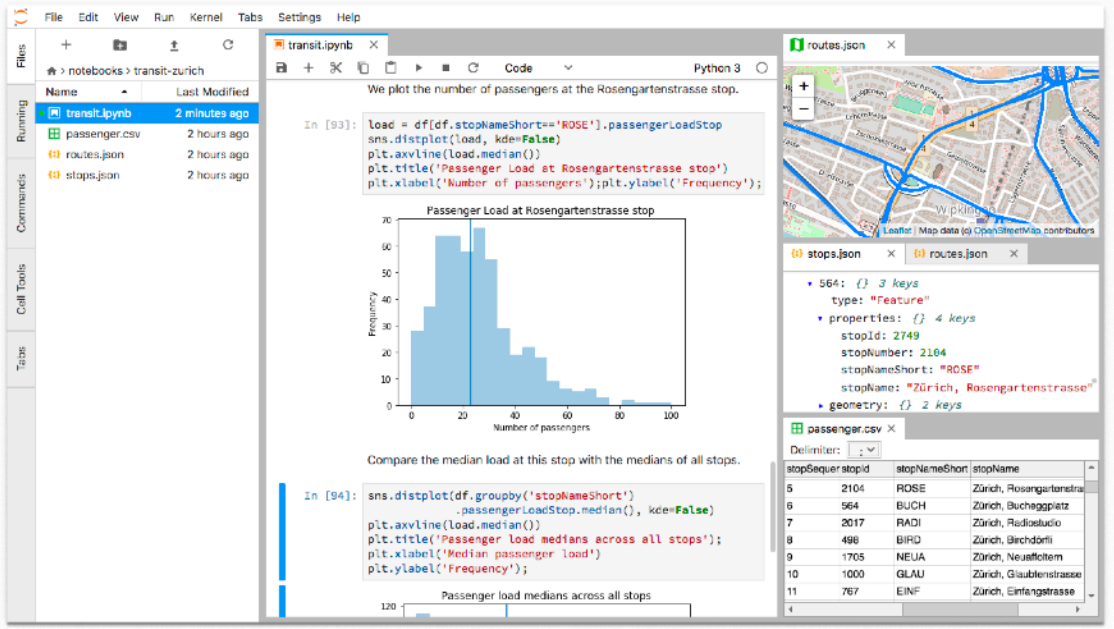
Need to fix the "string / bytes" problem of the array world!

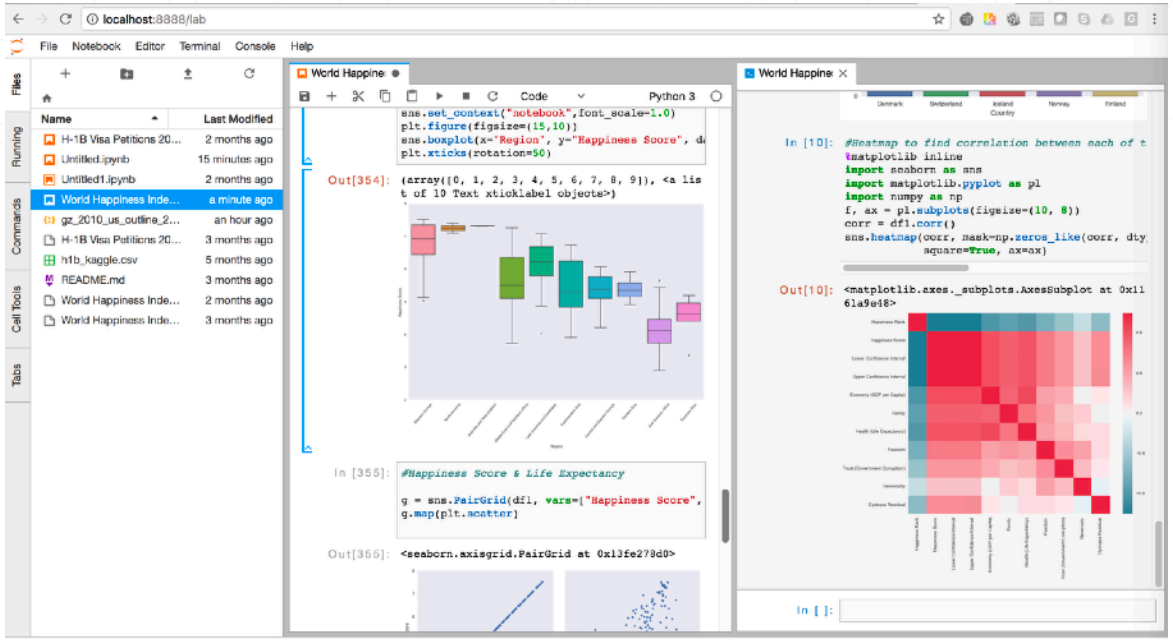
Logical array vs. strided pointer of numpy

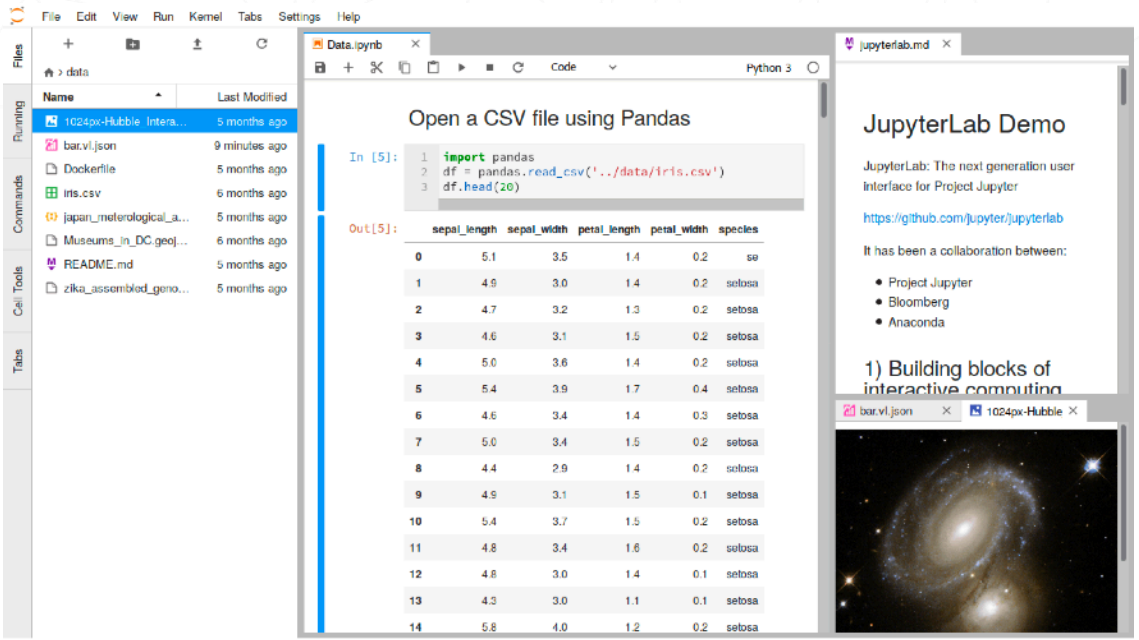


JupyterLab

- Future of Jupyter project
- More than a notebook
- Extensible data-centric app-building in the Web







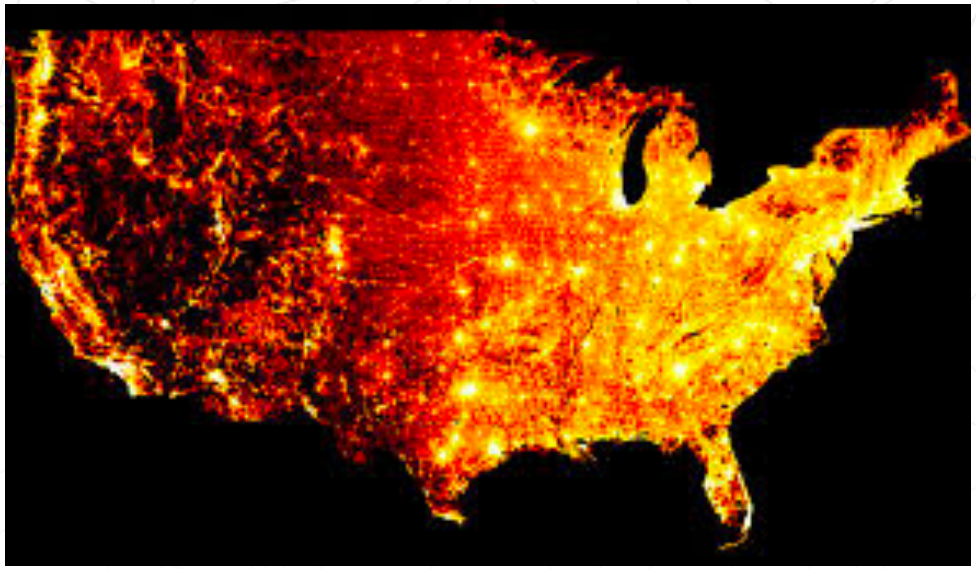
Visualization

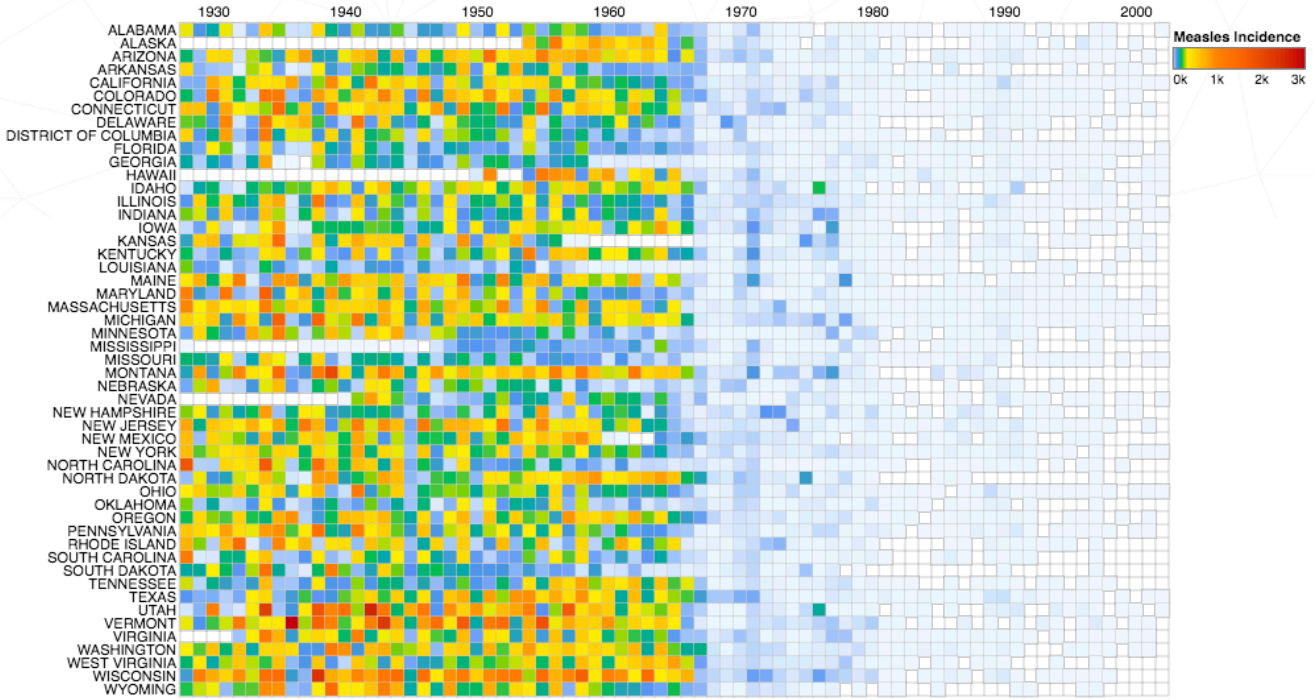
matplotlib

Altair

PyViz

Datashader





Bokeh

Taxi explorer

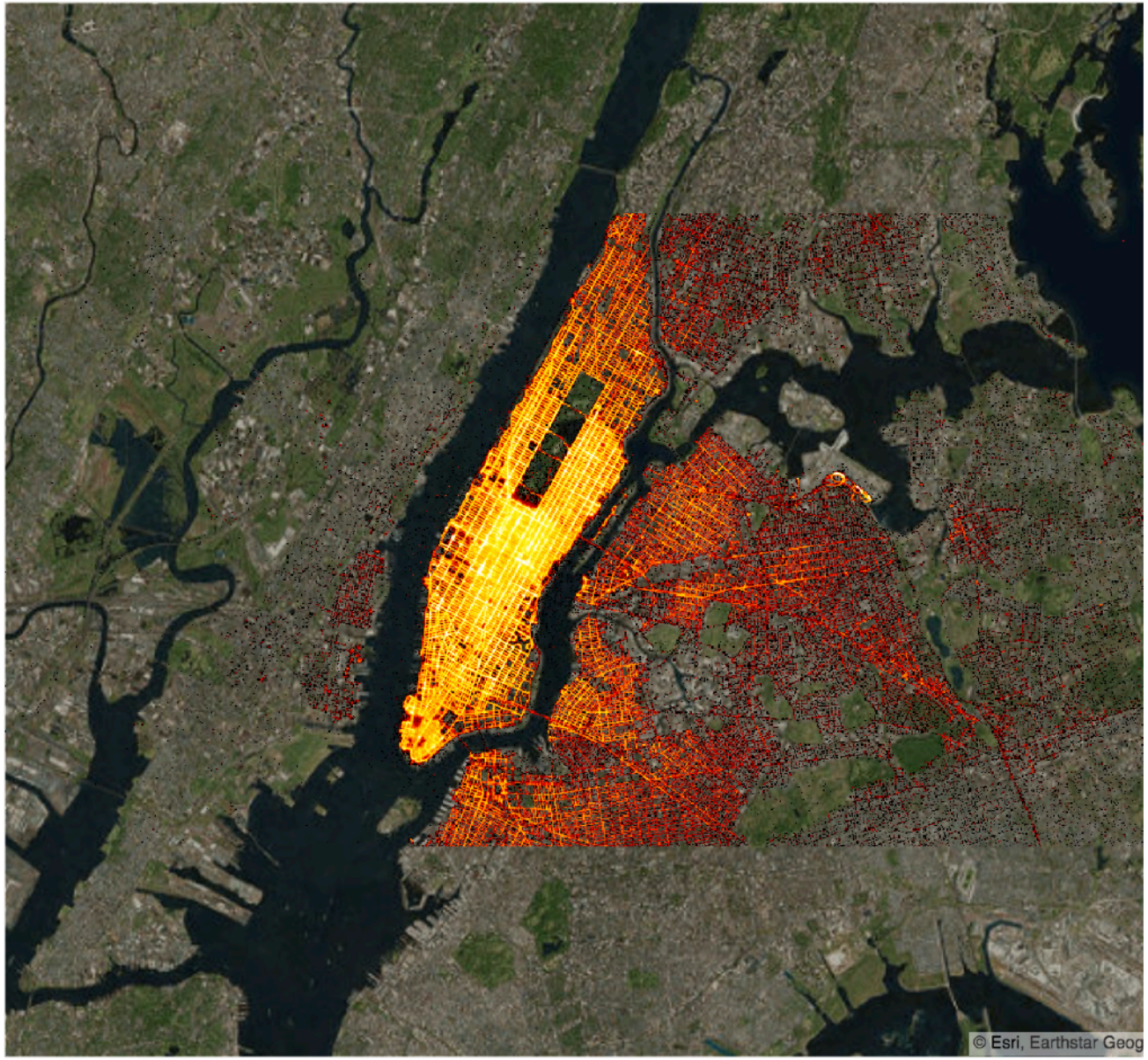
Alpha: 0.75

Cmap

fire

Location

dropoff



Easy Dashboards

http://panel.pyviz.org

What about GPUs?

http://rapids.ai

GPUs will become used by more data-scientists over the next 2-3 years!

NVIDIA's DGX platform powered by their RAPIDS initiative and the addition of GPU support across the PyData ecosystem. RAPIDS is a demo-driven initiative for now, but it will become increasingly stronger over the coming 2 years.

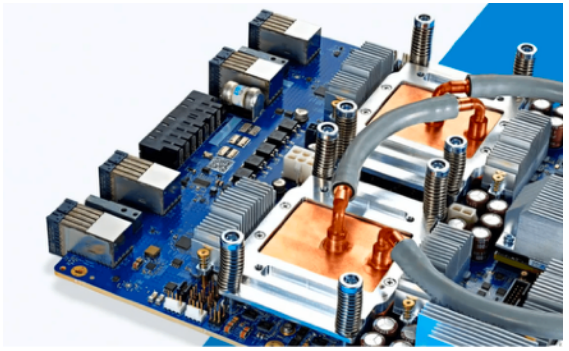
What about FPGAs?

Will have a few successes, but remain niche without significant investment

Lack of development resources will keep FPGAs from wide-spread usage.

Intel's AI chips will emerge in 2019 and start getting used.

Google's TPUs will be far more important than FPGAs.



Google

intel + nervana

ML and DL Matchup

TensorFlow

vs.

PYTORCH

Microsoft CNTK

mxnet

ONNX

Cloud TPU

intel

NVIDIA

Chainer

What about R?

- R has a stable community with good industry support that will continue to keep users.
- Domain experts don't change languages much
- Python will continue to grow and attract new users and out-pace R over the next 5 years.
- Look for more interoperability and cross-language sharing of ideas.

What about Julia?

- Julia is an excellent Research language. It will continue to grow in popularity, especially among students and hobbyists.
- Useful for research and exploring computational ideas.
- Not suitable for broad production usage — will be at least 5 years before it is. Never will be a single-language world.
- Python will learn much from Julia.
- Julia will attract R and especially more Matlab users.

Thing to Watch (over next 3 years)

WEBASSEMBLY

WebAssembly (abbreviated *Wasm*) is a binary instruction format for a stack-based virtual machine. Wasm is designed as a portable target for compilation of high-level languages like C/C++/Rust, enabling deployment on the web for client and server applications.

We will need more independent industry standards and benchmarking!

Thank you!