Updates in Scientific Python

- Python 3.7
- Visualization
- Data Science
- Performance

Python 3.7 Updates

dataclasses

In [11]:
```
from dataclasses import dataclass

@dataclass
class DataClassCard:
    rank: str
    suit: str

class RegularCard:
    def __init__(self, rank, suit):
        self.rank = rank
        self.suit = suit
```

In [17]:
```
queen_of_hearts = RegularCard("Queen", "Hearts")
queen_of_hearts_again = RegularCard("Queen", "Hearts")
print(queen_of_hearts)
print(queen_of_hearts == queen_of_hearts_again)
```

```
<__main__.RegularCard object at 0x12ca12c50>
False
```

In [18]:
```
queen_of_hearts = DataClassCard("Queen", "Hearts")
queen_of_hearts_again = DataClassCard("Queen", "Hearts")
print(queen_of_hearts)
print(queen_of_hearts == queen_of_hearts_again)
```

```
DataClassCard(rank='Queen', suit='Hearts')
True
```

In [5]:

```
class RegularCard:
    def __init__(self, rank, suit):
        self.rank = rank
        self.suit = suit

    def __repr__(self):
        return (f'{self.__class__.__name__}'
                f'(rank={self.rank!r}, suit={self.suit!r})')

    def __eq__(self, other):
        if other.__class__ is not self.__class__:
            return NotImplemented
        return (self.rank, self.suit) == (other.rank, other.suit)
```

In [6]:
```
from dataclasses import dataclass, field

@dataclass(order=True)
class Country:
    name: str
    population: int
    area: float = field(repr=False, compare=False)
    coastline: float = 0
```

Breakpoints

In [20]:
```
def divide(e, f):
    breakpoint()
    return f / e
```

In [21]:
```
a, b = 0, 1
print(divide(a, b))
```

```
> <ipython-input-20-4fa30e2346d3>(3)divide()
-> return f / e
(Pdb) display e
display e: 0
(Pdb) n
ZeroDivisionError: division by zero
> <ipython-input-20-4fa30e2346d3>(3)divide()
-> return f / e
(Pdb) q

---------------------------------------------------------------------------
BdbQuit                                   Traceback (most recent call last)
<ipython-input-21-72499869c711> in <module>
      1 a, b = 0, 1
----> 2 print(divide(a, b))

<ipython-input-20-4fa30e2346d3> in divide(e, f)
      1 def divide(e, f):
      2     breakpoint()
----> 3     return f / e

/anaconda/envs/presentation/lib/python3.7/bdb.py in trace_dispatch(self, fram
e, event, arg)
     92             return self.dispatch_return(frame, arg)
     93         if event == 'exception':
---> 94             return self.dispatch_exception(frame, arg)
     95         if event == 'c_call':
     96             return self.trace_dispatch

/anaconda/envs/presentation/lib/python3.7/bdb.py in dispatch_exception(self, f
rame, arg)
    172                     and arg[0] is StopIteration and arg[2] is None):
    173                 self.user_exception(frame, arg)
--> 174                 if self.quitting: raise BdbQuit
    175         # Stop at the StopIteration or GeneratorExit exception when th
e user
    176         # has set stopframe in a generator by issuing a return comman
d, or a

BdbQuit:
```

Better resource importing

```
data/
|
├── alice_in_wonderland.txt
└── __init__.py
```

In [11]:
```
from importlib import resources
with resources.open_text("data", "alice_in_wonderland.txt") as fid:
    alice = fid.readlines()
```

Other

- `python -X importtime script.py`
- Typing enhancements
- Optimization - less overhead for method calls, startup time reduced
- asyc/await, asyncio, context variables, timing, module attributes, ...

Visualization

- Dash
- Dash-Bio
- Voilà

Dash: A web application framework for Python

In [3]:
```
app
```

Out[3]: 404: Not Found

Open in new window (/app/endpoints/586e2ff4693c48cd90d8a0a45afa5589/) for /app/endpoints/586e2ff4693c48cd90d8a0a45afa5589/

Dash-Bio

A free, open-source Python library for bioinformatics and drug development applications.

In [5]:
```
manhattan_app
```

Out[5]: 404: Not Found

Open in new window (/app/endpoints/0a8fded8caf548b89d4f2cb25beab2cf/) for /app/endpoints/0a8fded8caf548b89d4f2cb25beab2cf/

In [7]:
```
molecule_2d_app
```

Out[7]: 404: Not Found

Open in new window (/app/endpoints/dfaa19aa2eee4d0b84ee56fa50aba87c/) for /app/endpoints/dfaa19aa2eee4d0b84ee56fa50aba87c/

In [9]:
```
molecule_3d_app
```

Out[9]: 404: Not Found

Open in new window (/app/endpoints/32299216fc4e4517a292f0ad591f4687/) for /app/endpoints/32299216fc4e4517a292f0ad591f4687/

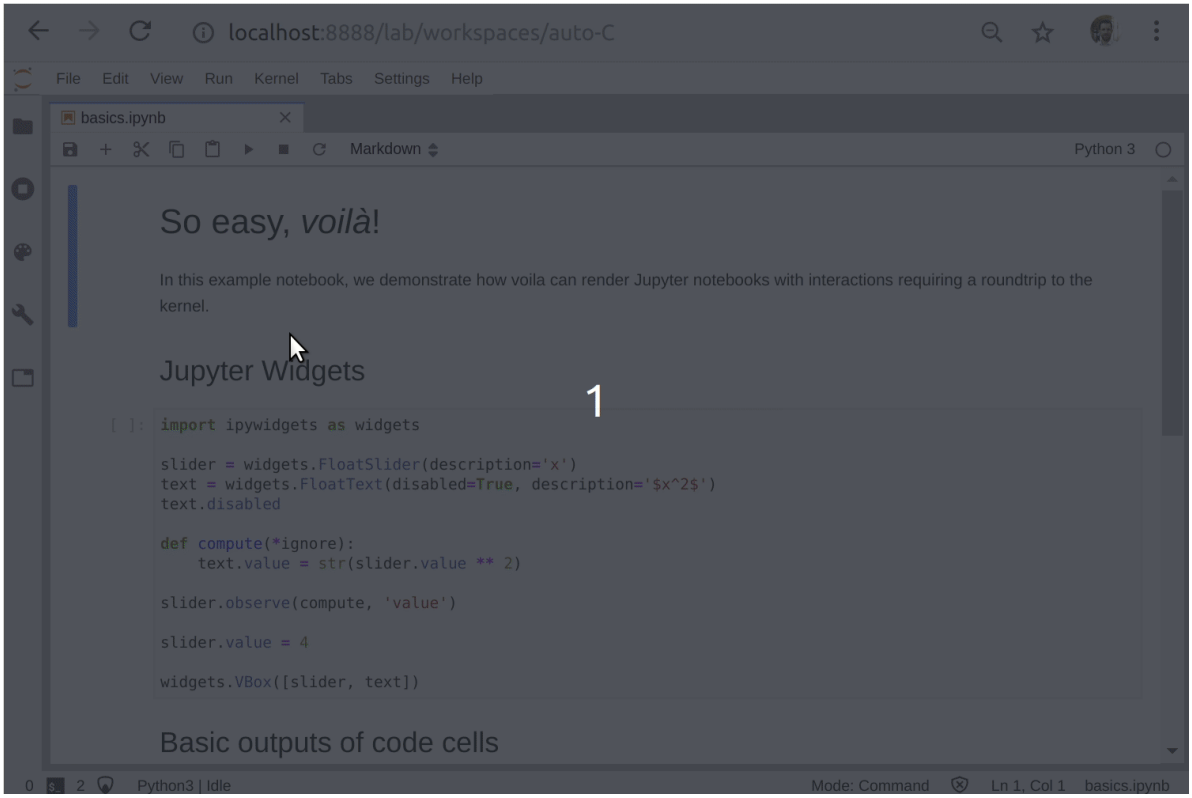
Combine and interlink multiple components.

- inDelphi (https://indelphi.giffordlab.mit.edu/single)
- Drug Discovery (https://dash-gallery.plotly.host/dash-drug-discovery/)
- *Workshop!*

Voilà

Voilà turns Jupyter notebooks into standalone web applications.

- Supports Jupyter interactive widgets.
- Does not permit arbitrary code execution.
- Works with any Jupyter kernel (C++, Python, Julia).
- Includes a flexible template system.


localhost:8888/lab/workspaces/auto-C
File Edit View Run Kernel Tabs Settings Help
basics.ipynb
Markdown
Python 3
So easy, voilà!
In this example notebook, we demonstrate how voila can render Jupyter notebooks with interactions requiring a roundtrip to the kernel.
Jupyter Widgets
1
[ ]:
import ipywidgets as widgets
slider = widgets.FloatSlider(description='x')
text = widgets.FloatText(disabled=True, description='x^2')
text.disabled
def compute(*ignore):
text.value = str(slider.value ** 2)
slider.observe(compute, 'value')
slider.value = 4
widgets.VBox([slider, text])
Basic outputs of code cells
0 2 Python3 | Idle
Mode: Command
Ln 1, Col 1
basics.ipynb

Data Science

- SciPy
- PyTorch
- ELI5
- HyperTools

PyTorch reaches 1.0

AdverTorch

A toolbox for adversarial robustness research. It contains modules for generating adversarial examples and defending against attacks.

AllenNLP

AllenNLP is an open-source research library built on PyTorch for designing and evaluating deep learning models for NLP.

ELF

ELF is a platform for game research that allows developers to train and test their algorithms in various game environments.

fastai

fastai is a library that simplifies training fast and accurate neural nets using modern best practices.

Flair

Flair is a very simple framework for state-of-the-art natural language processing (NLP).

Glow

Glow is a ML compiler that accelerates the performance of deep learning frameworks on different hardware platforms.

GPyTorch

GPyTorch is a Gaussian process library implemented using PyTorch, designed for creating scalable, flexible Gaussian process models.

Horovod

Horovod is a distributed training library for deep learning frameworks. Horovod aims to make distributed DL fast and easy to use.

Ignite

Ignite is a high-level library for training neural networks in PyTorch. It helps with writing compact, but full-featured training loops.

ParlAI

ParlAI is a unified platform for sharing, training, and evaluating dialog models across many tasks.

PennyLane

PennyLane is a library for quantum ML, automatic differentiation, and optimization of hybrid quantum-classical computations.

Pyro

Pyro is a universal probabilistic programming language (PPL) written in Python and supported by PyTorch on the backend.

PySyft

PySyft is a Python library for encrypted, privacy preserving deep learning.

PyTorch Geometric

PyTorch Geometric is a library for deep learning on irregular input data such as graphs, point clouds, and manifolds.

skorch

skorch is a high-level library for PyTorch that provides full scikit-learn compatibility.

TensorLy

TensorLy is a high level API for tensor methods and deep tensorized neural networks in Python that aims to make tensor learning simple.

Translate

Translate is an open source project based on Facebook's machine translation systems.

Skorch - Scikit-Learn API for PyTorch

- Easier to swap in and out different ML models
- Use in sklearn pipelines, GridSearch, custom scoring metrics etc.

In [28]:
```
from skorch import NeuralNetClassifier
from sklearn.metrics import accuracy_score
cnn = NeuralNetClassifier(Cnn, max_epochs=10, lr=0.0002, optimizer=torch.optim.Ada
m, device=device, iterator_train__num_workers=4, iterator_valid__num_workers=4)
cnn.fit(mnist_train, y=y_train)
y_pred_cnn = cnn.predict(mnist_test)
accuracy_score(y_test, y_pred_cnn)
```

epoch	train_loss	valid_acc	valid_loss	dur
1	0.9540	0.9236	0.2618	45.0046
2	0.3280	0.9511	0.1633	45.2179
3	0.2284	0.9619	0.1239	47.3209
4	0.1810	0.9685	0.0997	42.9972
5	0.1550	0.9733	0.0860	43.4656
6	0.1399	0.9761	0.0794	46.8313
7	0.1262	0.9775	0.0730	43.2472
8	0.1180	0.9803	0.0666	42.0751
9	0.1088	0.9808	0.0648	41.7215
10	0.1042	0.9822	0.0607	42.7039

Out[28]: 0.9831

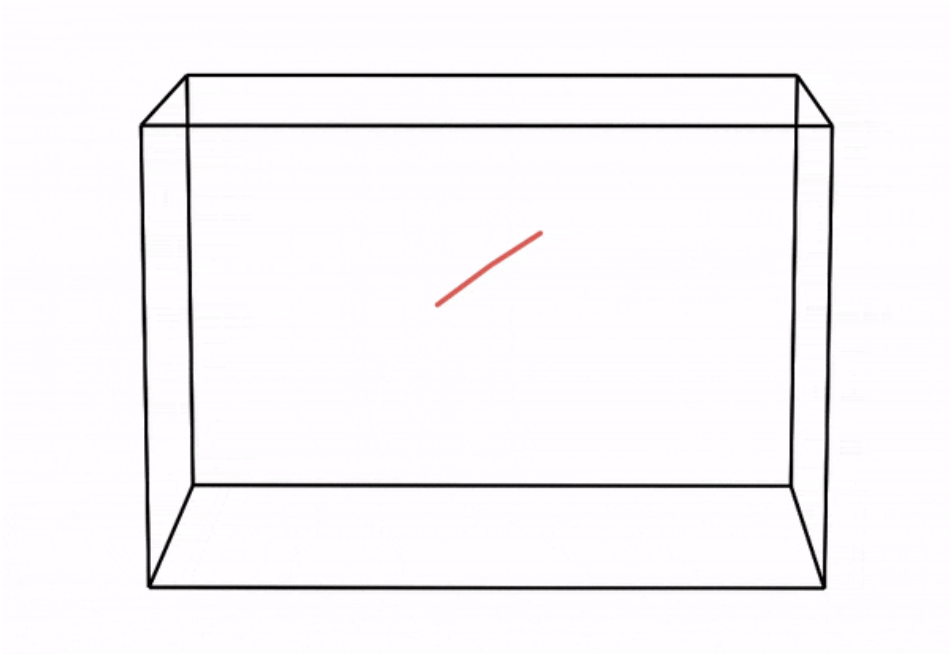
ELI5

- Debug machine learning classifiers and explain their predictions
- Supports scikit-learn, XGBoost, LightGBM etc.
- Inspect black-box models using LIME and Permutation Importance

y=0 top features		**y=1** top features		**y=2** top features	
Weight?	**Feature**	**Weight?**	**Feature**	**Weight?**	**Feature**
+0.772	keith	+1.096	graphics	+0.948	rutgers
+0.656	okcforum	+0.637	software	+0.817	christians
+0.625	mathew	+0.609	image	+0.754	church
+0.593	atheism	+0.586	host	+0.734	clh
+0.574	writes	+0.573	nntp	+0.681	christ
+0.541	psuvm	+0.529	42	+0.610	athos
+0.523	wingate	+0.510	tiff	+0.534	christian
+0.511	umd	+0.506	looking	+0.528	1993
+0.504	benedikt	+0.501	files	+0.495	petch
+0.501	islamic	+0.481	ftp	+0.482	love
+0.482	psu	+0.473	card	+0.450	bassili
… 10732 more positive …		*… 12994 more positive …*		+0.424	geneva
… 16774 more negative …		*… 14512 more negative …*		*… 12074 more positive …*	
-0.475	organization	-0.472	jesus	*… 15432 more negative …*	
-0.480	christ	-0.506	writes	-0.463	tin
-0.548	lines	-0.539	okcforum	-0.549	software
-0.554	thanks	-0.584	keith	-0.550	newsreader
-0.554	christians	-0.606	church	-0.566	article
-0.591	graphics	-0.642	christian	-0.793	posting
-0.764	rutgers	-0.674	bible	-0.904	graphics
-0.844	subject	-0.754	people	-0.960	nntp
-0.901	<BIAS>	-0.822	god	-1.013	host

HyperTools

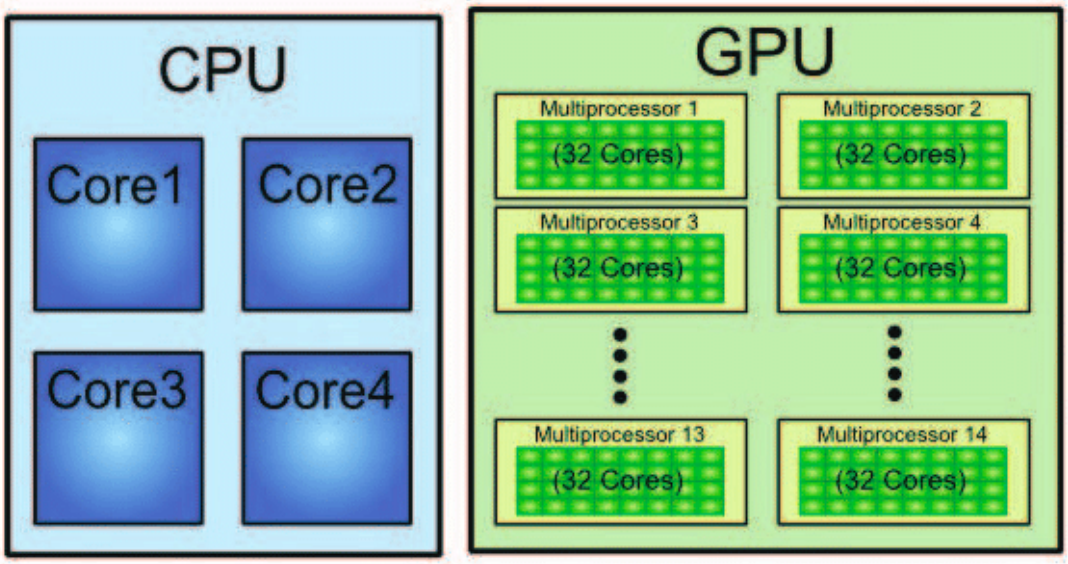
- A python toolbox for gaining geometric insights into high-dimensional data
- Plotting high-dimensional datasets in 2/3D
- Data manipulation tools - hyperalignment, k-means clustering, normalizing and more
- Support for lists of Numpy arrays, Pandas dataframes, text or (mixed) lists



Performance Updates

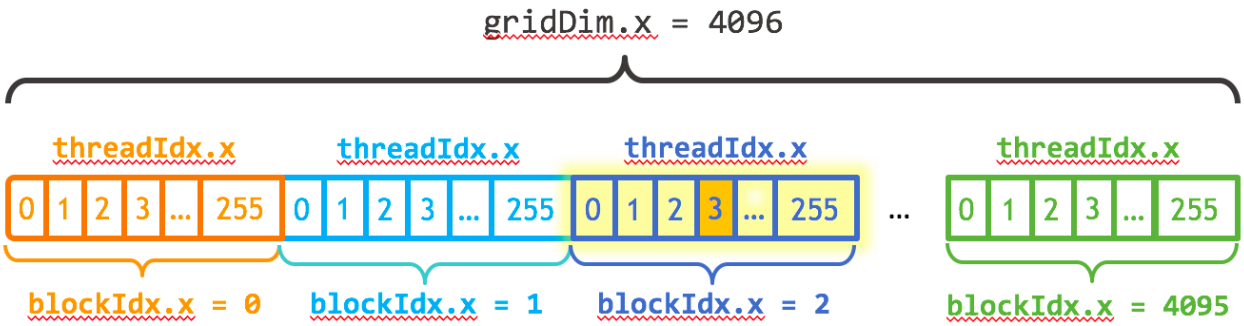
- Bad news: Core Python will always be slow
- Good news: Libraries to speed up your code
- Previously, libraries mostly focused on CPU and work distribution (cluster computing)
    - NumPy, Numba, Dask, Cython, PyPy etc.
- This year was **the year of the GPU**

Why GPUs?


CPU
Core1
Core2
Core3
Core4
GPU
Multiprocessor 1
(32 Cores)
Multiprocessor 2
(32 Cores)
Multiprocessor 3
(32 Cores)
Multiprocessor 4
(32 Cores)
Multiprocessor 13
(32 Cores)
Multiprocessor 14
(32 Cores)

CUDA

"CUDA is a parallel computing platform and application programming interface (API) model created by Nvidia."
Wikipedia



index = blockIdx.x * blockDim.x + threadIdx.x

index = (2) * (256) + (3) = 515

Python makes this easier

- We only need **NumPy arrays** or **Pandas Dataframes**
- Several libraries wrap around the CUDA API
  - Array based: CuPy, Numba
  - Dataframe based: CuDF, GPU-Dask
  - Tensor based: PyTorch

CuPy: "A NumPy-compatible matrix library accelerated by CUDA"

- Drop-in replacement for NumPy functions
- Just `import cupy` instead of `import numpy`
- Custom functions (Advanced)

In [ ]:

```
>>> import cupy as cp
>>> x = cp.arange(6).reshape(2, 3).astype('f')
>>> x
array([[ 0.,  1.,  2.],
       [ 3.,  4.,  5.]], dtype=float32)
>>> x.sum(axis=1)
array([  3.,  12.], dtype=float32)
```

CuPy custom functions:

In [ ]:

```
>>> x = cp.arange(6, dtype='f').reshape(2, 3)
>>> y = cp.arange(3, dtype='f')
>>> kernel = cp.ElementwiseKernel(
...     'float32 x, float32 y', 'float32 z',
...     '''if (x - 2 > y) {
...       z = x * y;
...     } else {
...       z = x + y;
...     }''', 'my_kernel')
>>> kernel(x, y)
array([[ 0.,  2.,  4.],
       [ 0.,  4.,  10.]], dtype=float32)
```

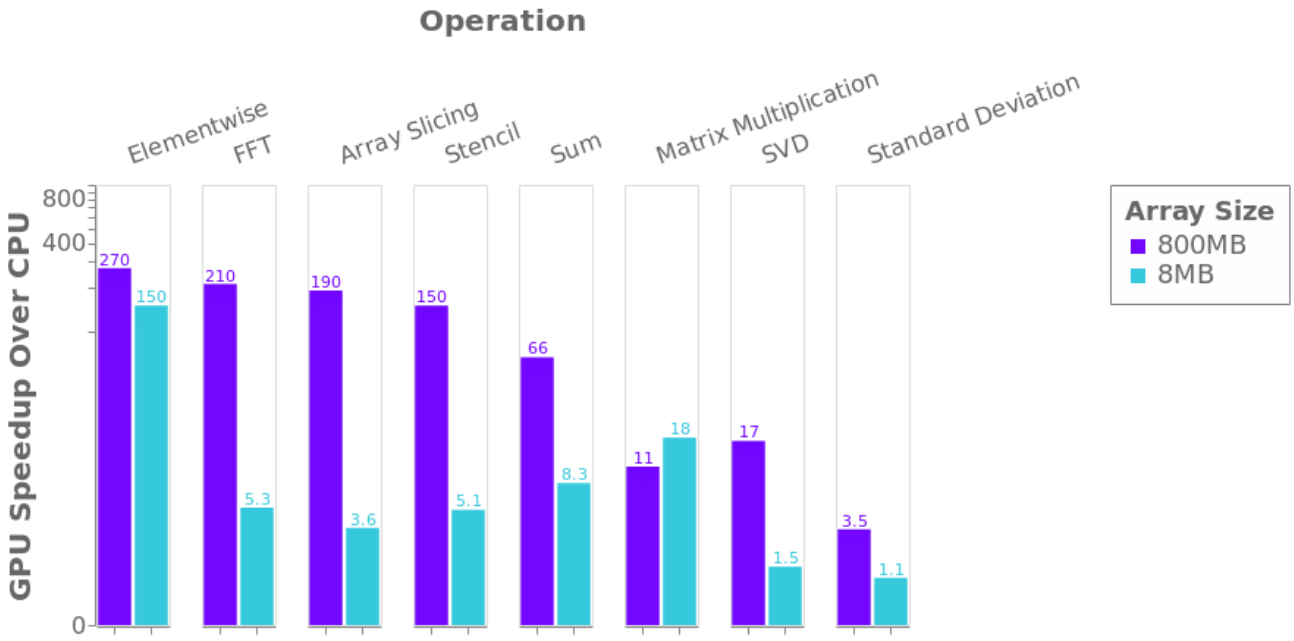
Or...use Numba!

- Numba aims to wrap the complete CUDA API and provide a Pythonic way to define custom CUDA functions
- `export NUMBA_ENABLE_CUDA_SIM=1`

In [31]:
```
from numba import cuda

@cuda.jit
def my_kernel(x, y, z):
    i = cuda.grid(1)
    if i < x.size:
        if (x[i] - 2) > y[i]:
            z[i] = x[i] * y[i]
        else:
            z[i] = x[i] + y[i]
```

CuPy/Numba - NVIDIA Tesla V100 32 GB v.s. Intel Xeon E5-2698 v4



CuDF:

- Drop-in replacement for pandas
- scikit-learn API support with CuML (based on CuDF)

Dask distributed GPU:

- Multiple GPUs can be combined using Dask with CuDF

In [ ]:

```
from dask_cuda import LocalCUDACluster
import dask_cudf
from dask.distributed import Client

cluster = LocalCUDACluster()  # runs on multiple available local GPUs
client = Client(cluster)

gdf = dask_cudf.read_csv('data/nyc/many/*.csv')  # wrap around many CSV files

>>> gdf.passenger_count.sum().compute()
```

More Updates and Developments:

- SciPy 2019 (https://www.scipy2019.scipy.org/)
- EuroScipy 2019 (https://www.euroscipy.org/2019/)

Thanks!