Accelerating Python

Accelerating scientific computing in as few lines as possible.

Ben Cardoen
Medical Image Analysis Lab

The Problem

Most Python code, with the exception of Cuda for DL, runs sequential, and is slow.

There is a clear need for fast Python code, but without spending days/weeks writing highly optimized C/C++ modules.

The takeaway of this workshop should be to give you ‘Hello World’ copy pasteable examples that allow you to speed up your code with a factor of 1-2 orders of magnitude in 15 minutes or less.*

Typical use cases:

- Hyperparameter optimization
- Augmentation, preprocessing
- Simulation, 3D point clouds processing, ...

* Add half a day if you have issues with Python dependencies

What we (me) will do

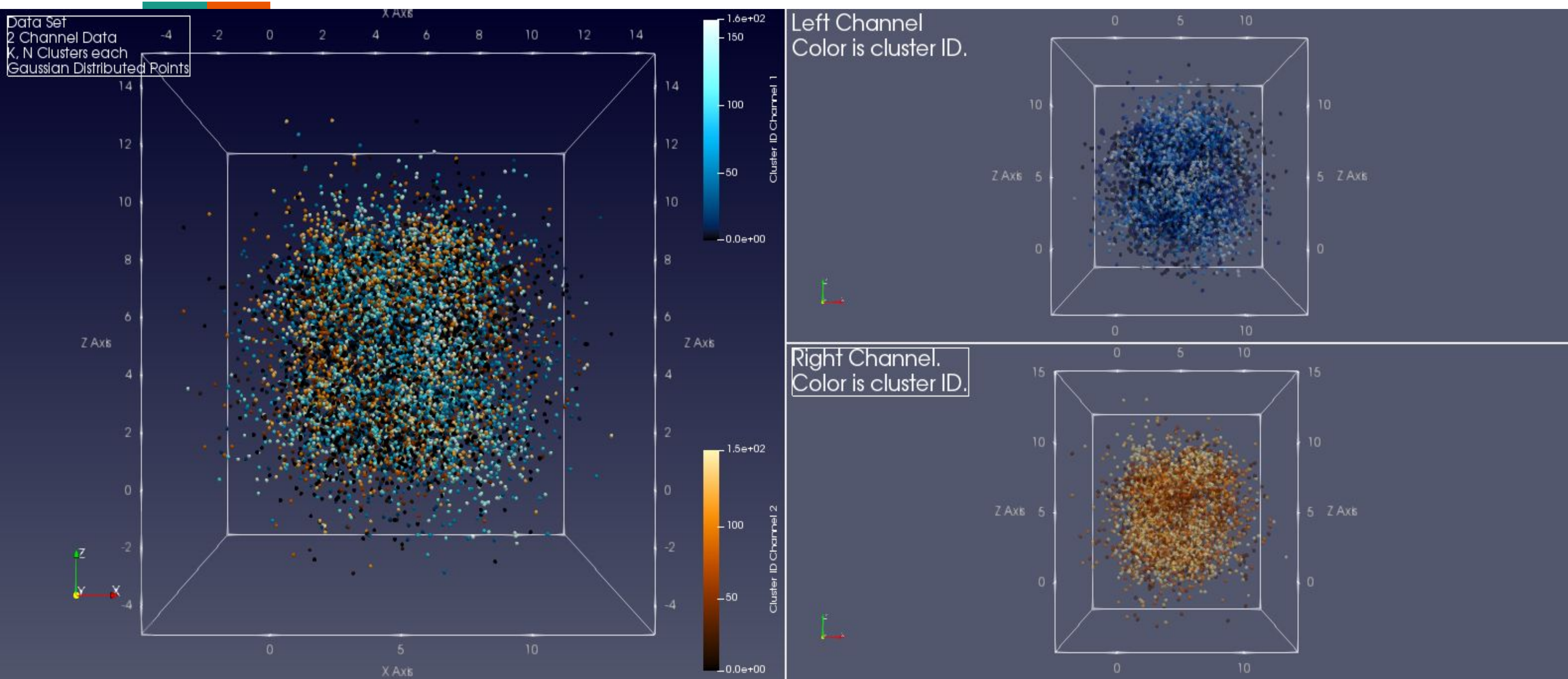
- Accelerate Sequential Code
  - Cython
  - Numba
  - Map - Reduce
- Parallelize Sequential Code
- Problems
  - Synthetic: A N^4 computation loop
  - Use Case:
    - Pairing of clusters of 3D points using Chamfer Distance
- Speedups observed:
  - Synthetic
    - Sequential speedup: 3-500
    - Parallel : linear in # threads
  - Use Case
    - Sequential speedup: 15-30
    - Parallel: (0.7 - 0.95) * # threads

What we (me) will not do

- Vectorization

Use Case

Pair Clusters in 2-Channel Data



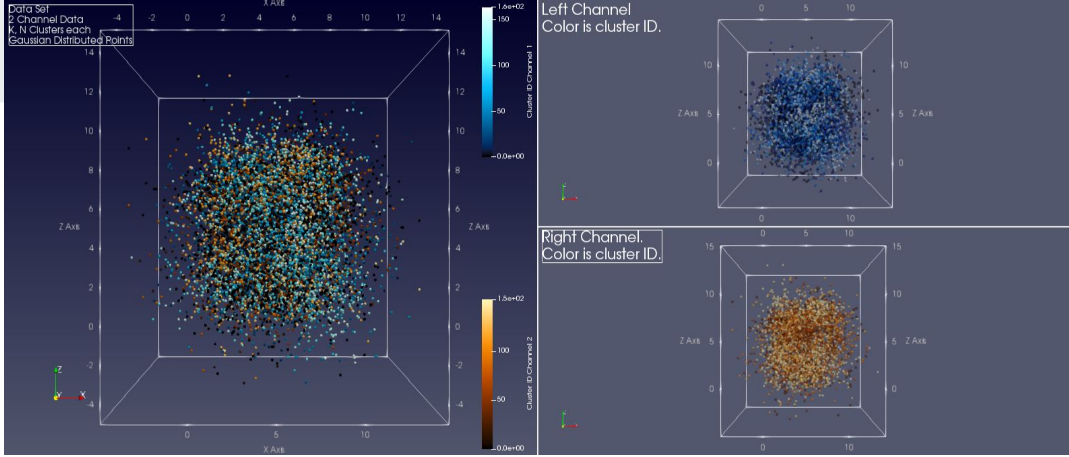
Data Set
2 Channel Data
K, N Clusters each
Gaussian Distributed Points

Left Channel
Color is cluster ID.

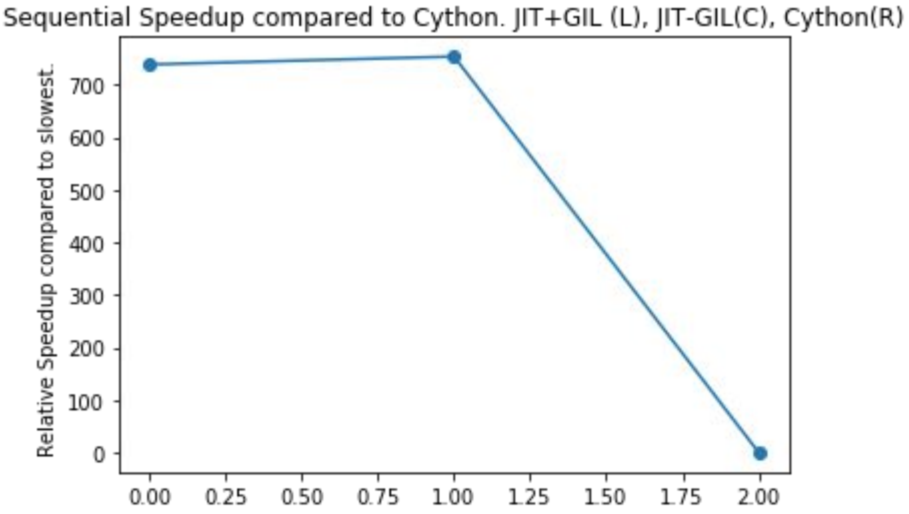
Right Channel.
Color is cluster ID.

Use Case

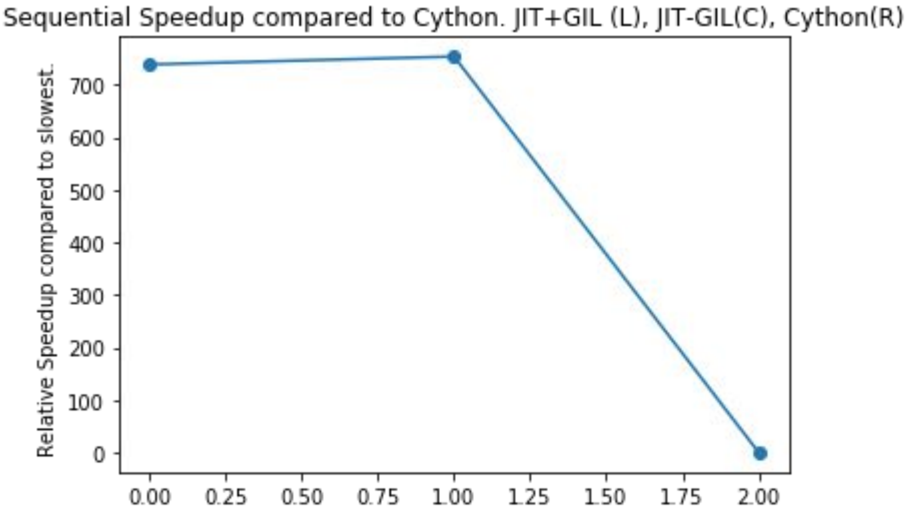
2 Channels
3D Points
M,N clusters per channel.
Pair clusters using Chamfer distance.



Sequential Synthetic -- Python vs Cython

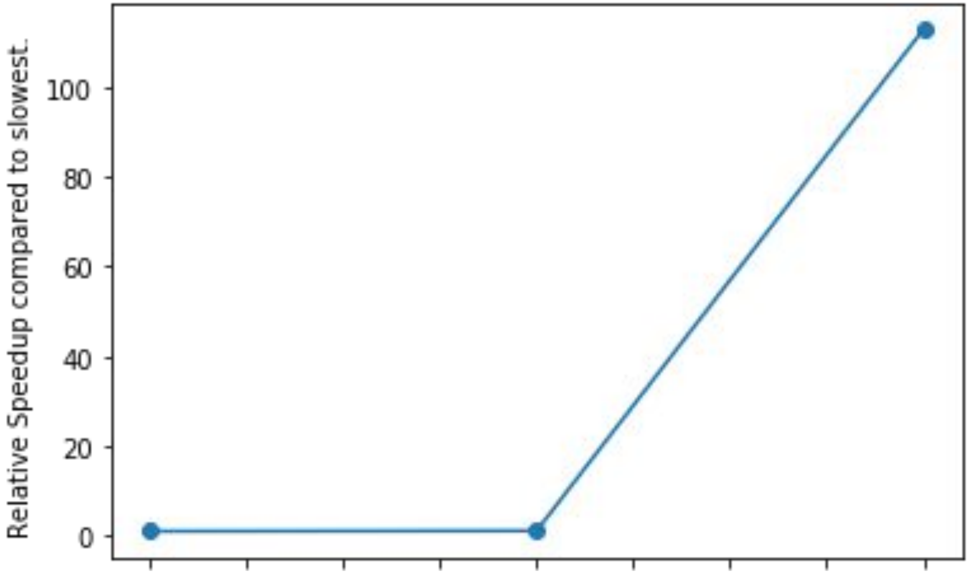


Sequential Synthetic -- JIT vs Cython

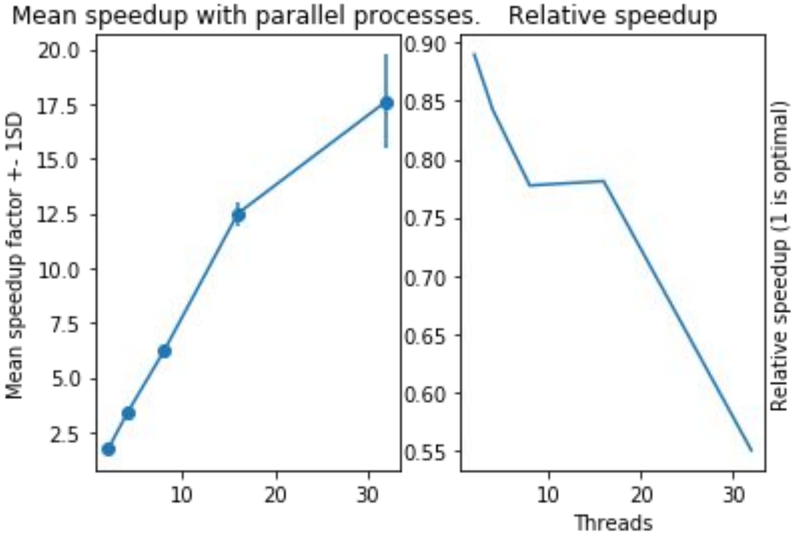


Sequential Use Case -- JIT

Sequential Speedup compared to Pure Python. Python (L), MapReduce(C), Numba(R)

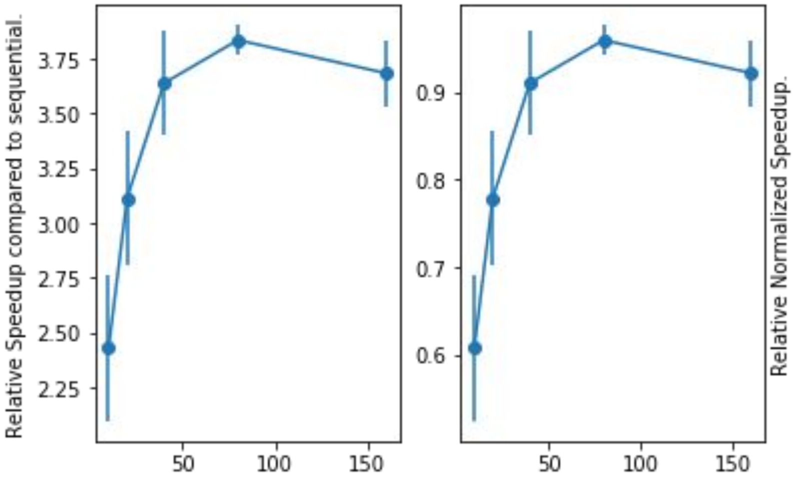


Parallel Use Case -- Weak Scaling



Parallel Use Case -- Strong Scaling

Strong scaling with doubling input size.



Conclusions

Accelerating your code:

- Add 1 decorator:
    - Speedup 1-2 orders of magnitude for numerical code
- Add 5 lines of code to parallelize (strip the benchmark code)
    - Linear parallel speedup in number of threads
- Combined:
    - Dual core hyperthreading i5 laptop (5 years old)
        - X **400** faster code for my (actual) use case

Extra

- Parallel IO (IO for the cluster)
- Numba
- Cython
- Joblib
- Cedar Summer School docs