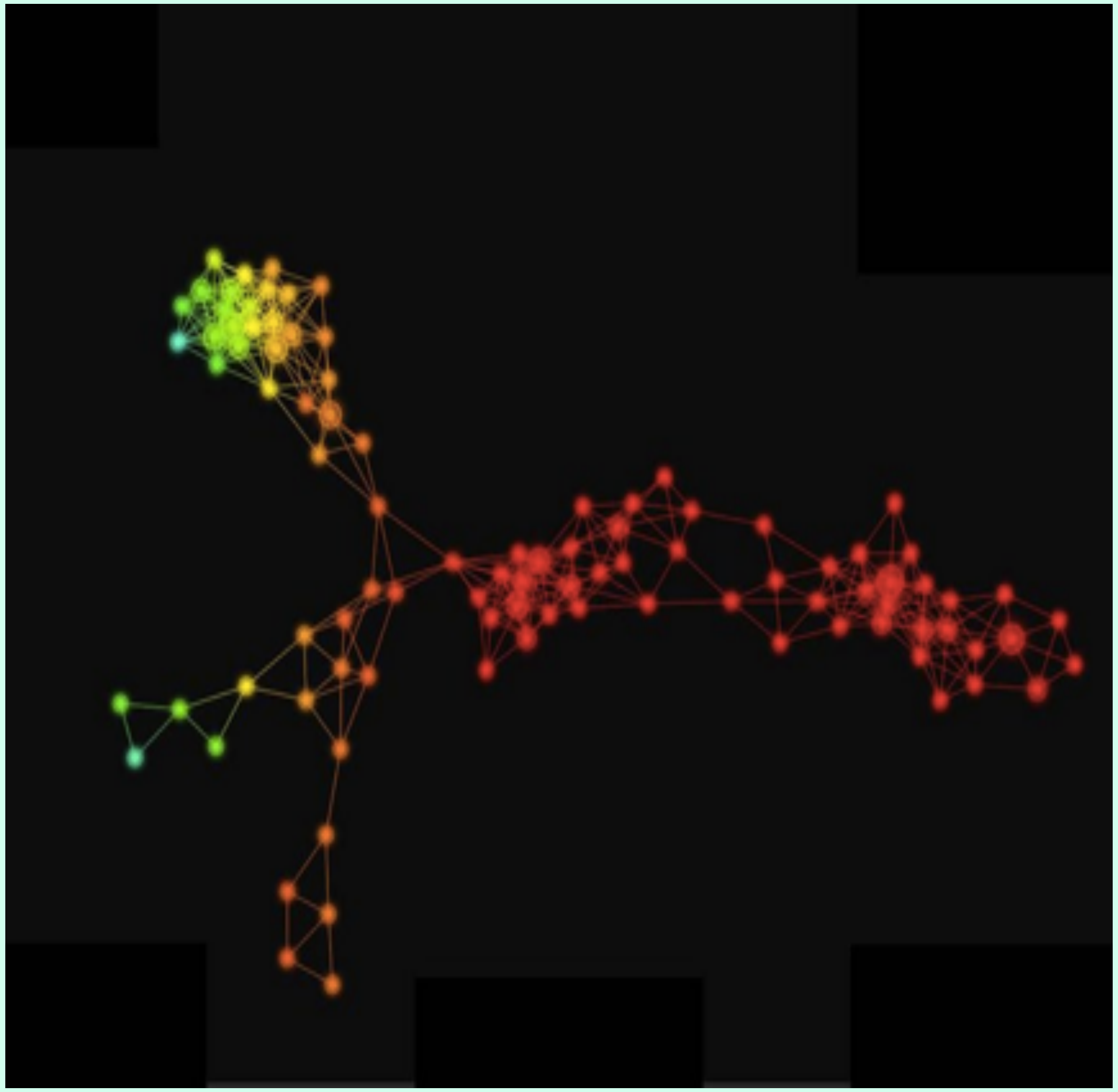
Analysis of high dimensional data via Topology

Louis Xiang, Fernando Schwartz, Kwai Wong

Overview

In this study we will focus on computing the topological invariant of high dimensional data set. By this kind of topological analysis, we are able to indicate some qualitative result about the high dimensional data. We will use the ICU medical data set as our object to show how the method describes the shape of the data set.



Overview

- ***MPI*** on forming the high-density data set
- Build the Simplicial Complex and compute the topological invariant
- ***Eden* for Large-scale parallel** numerical simulation
- Statistical analysis about the output

How to form a high-density dataset

- First, we try to reduce the dimension of the data set by selecting the most relevant 8 factors of the patients and do interpolation to fill in the missing data.

299264x43 double

	1	2	3	4	5	6	7
210	2477	-1	-1	-1	-1	-1	56.7000
211	2537	-1	-1	-1	-1	-1	56.7000
212	2597	-1	-1	-1	-1	-1	56.7000
213	2657	-1	-1	-1	-1	-1	56.7000
214	2717	-1	-1	-1	-1	-1	56.7000
215	2777	-1	-1	-1	-1	-1	56.7000
216	2837	-1	-1	-1	-1	-1	56.7000
217	0	132543	68	1	180.3000	3	84.6000
218	11	-1	-1	-1	-1	-1	-1
219	21	-1	-1	-1	-1	-1	-1
220	36	-1	-1	-1	-1	-1	84.6000
221	51	-1	-1	-1	-1	-1	84.6000
222	81	-1	-1	-1	-1	-1	84.6000

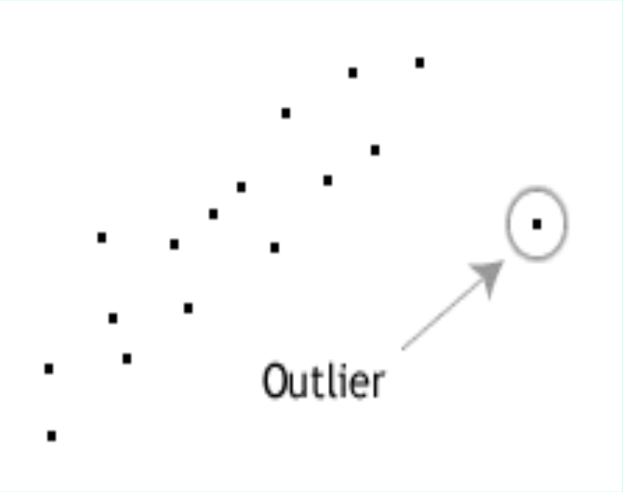
299264x8 double

	2	3	4	5	6	7	8	9
208	103	53	77	0.3000	38.1563	5	19	
209	124	65	89	1.0167	38.1500	5	19	
210	119	60	83	1.0167	38.1250	5	19	
211	121	63	87	1.6667	38.1000	5	19	
212	103	52	72	1.1667	37.8750	5	19	
213	118	59	83	0.5833	37.6500	5	19	
214	121	65	86	0.2500	37.4250	5	19	
215	124	69	91	0.5833	37.2000	5	19	
216	126	70	92	0.5833	37.2000	5	19	
217	134	63	86.6700	600	36.3000	15	19	
218	134	63	86.6700	600	36.3000	15	19	
219	134	63	86.6700	600	36.3000	15	19	
220	134	63	86.6700	600	36.3077	15	19	

Original dataset: 299264 points with dimension 43. It forms a ~300,000*43 matriX.

New dataset: 299264 points with dimension 8. It forms a ~300,000*8 matrix with missing entries filled by linear interpolation.

How to form a high-density dataset



- It must be pointed out that direct application of simplicial complex approximation to these 300,000 data points with dimension 8 will lead to a **wrong detection** because of the outliers distributed far away from main region. To obtain a high-density subset, we use the simple ***density function***

  $\rho_K(x) = |x - x_K|$ where x_K is the K-th nearest point of x.

- The crucial step is to find the ***distance matrix*** where

  $d_{ij} = d(x_i, x_j)$, x_i and x_j is the i, j rows in the matrix.

  # of points: 300,000. So the distance matrix is 300,000 by 300,000. Since this matrix is of big magnitude, we may use ***Darter*** as our supercomputer to do parellel computing.

Algorithm for MPI

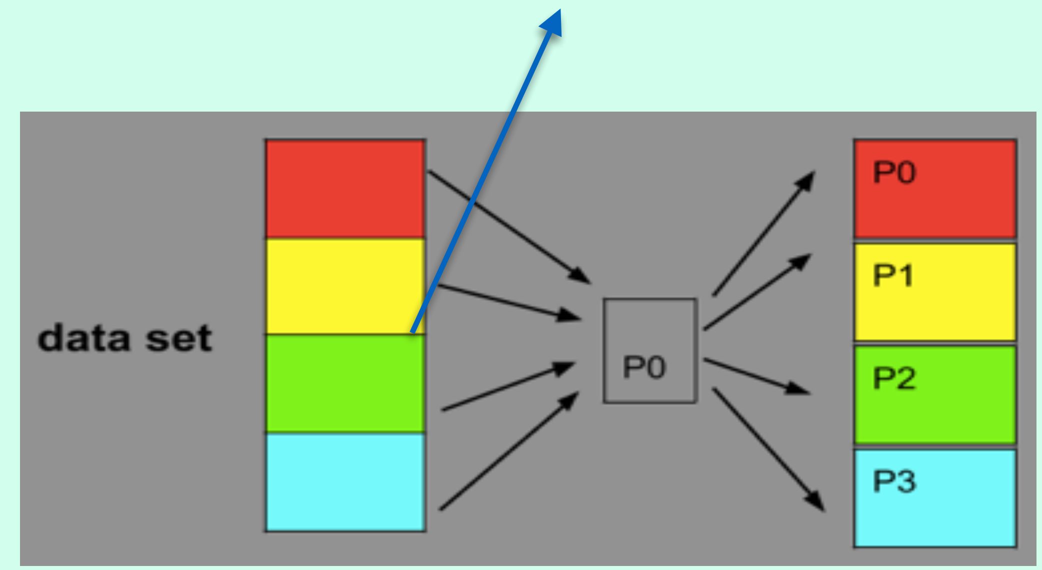
We will use 4 processors as an example.

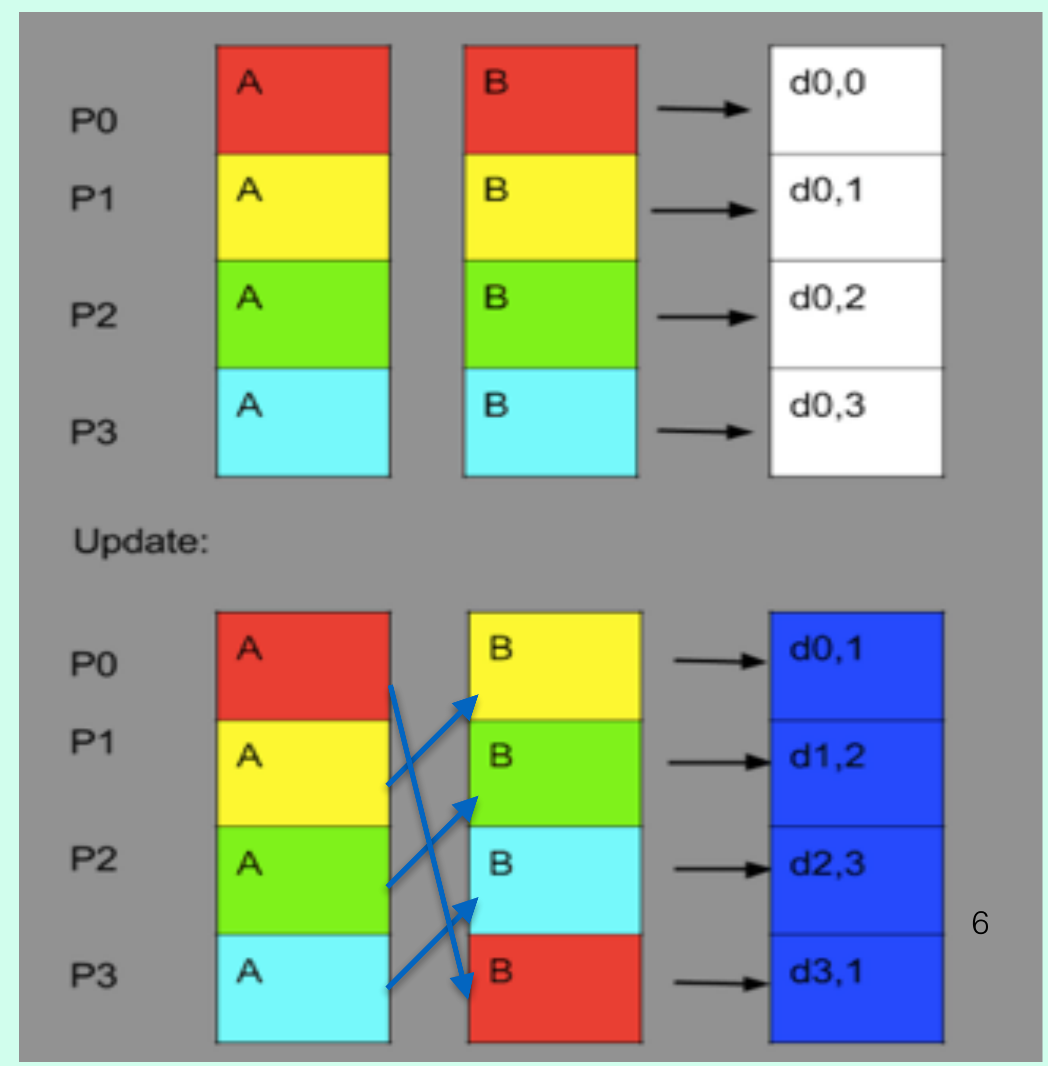
Step1: Let A and B be 2 collection of points in each processor. P0 read and send each part of the matrix to other processors and store in A. Let B copies A in each processor.

Step2:Calculate the distance matrix between A and B in each processor. Then, at the k-th iteration, Pi ***sends*** A to P(i-k) if i-k>=0 (P(k-i) sends to Pi if i-k<0), P(i-k)(or Pi) ***receives*** and update B matrix and calculate the distance matrix between A and B in each processor again.

✦Note: di,j is the submatrix of the whole distance matrix produced by i-th iteration in the processor j.

300,000*8 matrix divided into 4 pieces

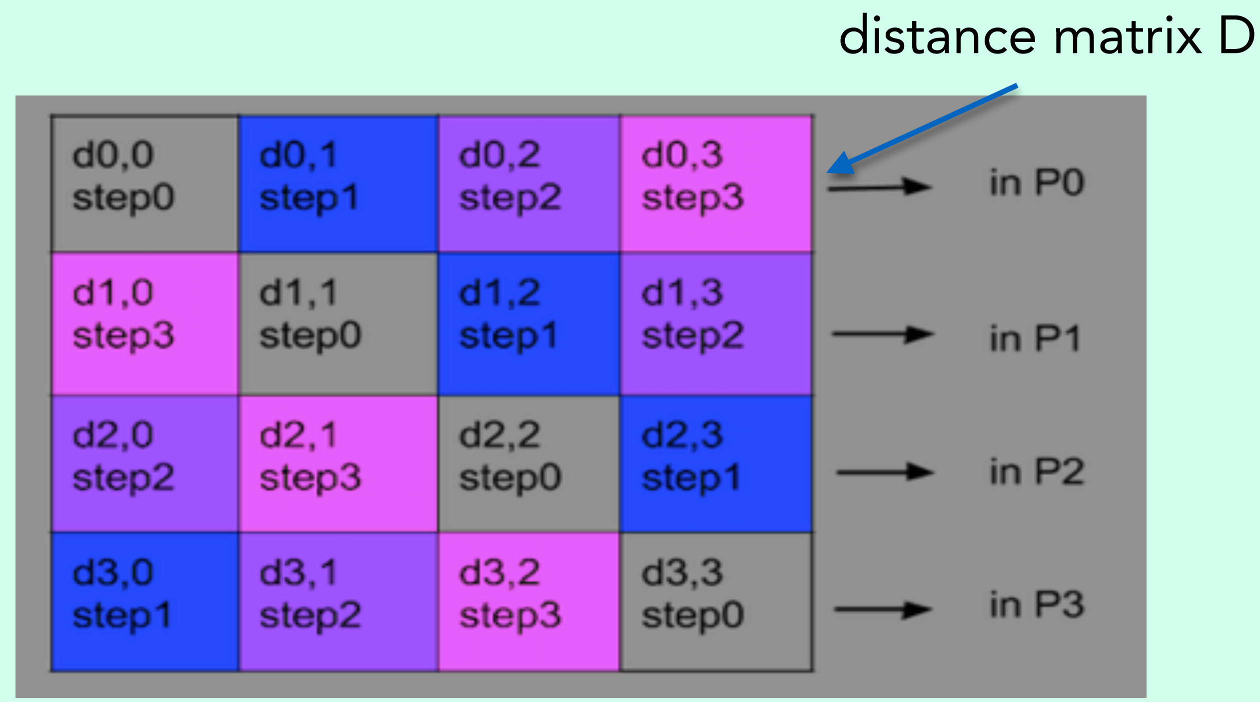




Algorithm for MPI

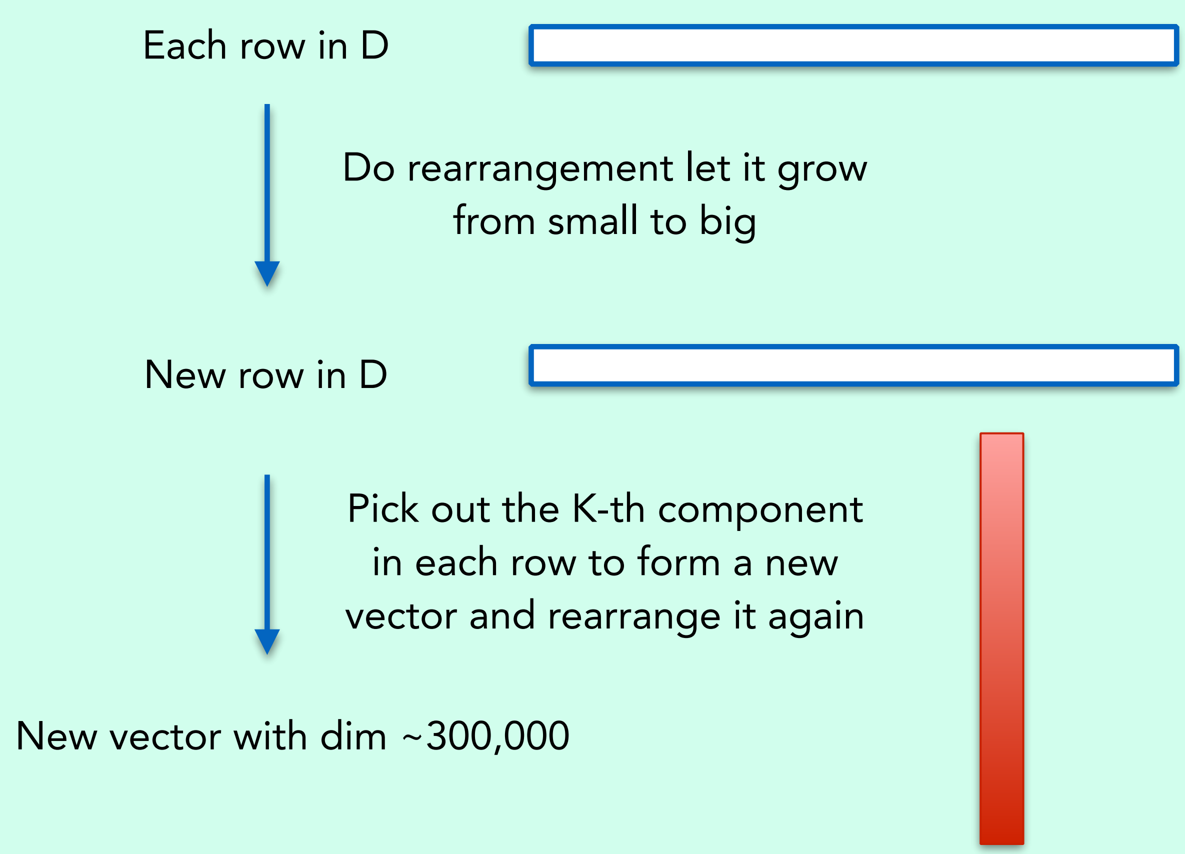
Step3:Continuing in this way until we get the distance matrix.

- ✦Note: The different colour means the output in different iterations.
- ✦Entries in the same row are produced by the same processor.



Step4:Do the rearrangement on each row and take out the k-th column in each processor and combine them together to form a long vector.

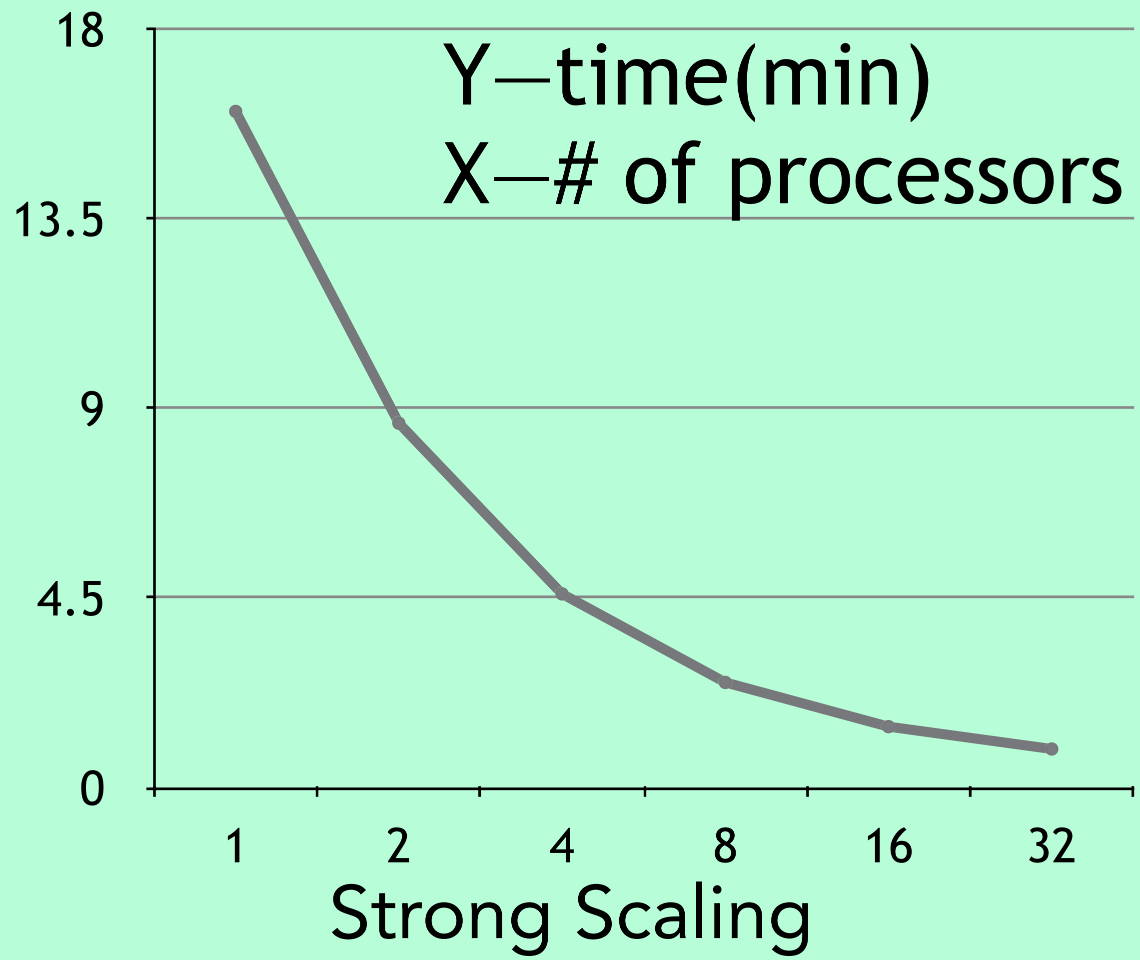
Step5:Do the rearrangement again on k-th column and record the points which is on the top p% in the rearrangement vector.

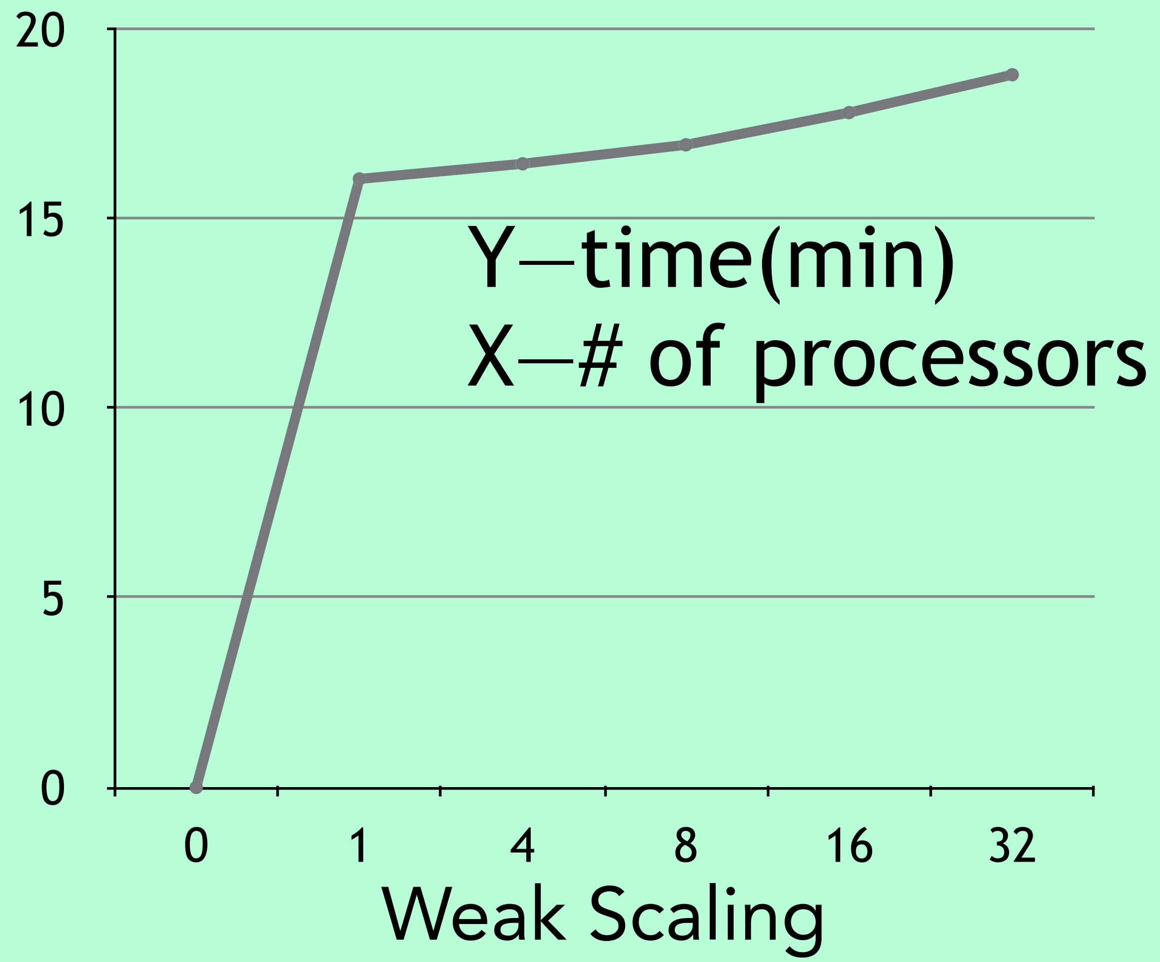


Through these 5 steps, we are able to form the X(K,p), a subset of the original data set.

Scaling

Below is the strong and weak scaling of this method for a relative small data set, say, 10,000 point cloud.





- ✦ For strong scaling, when # of processors is doubled, the time is decreased by half
- ✦ For weak scaling, we double the # of points but we need to make the # of proc. 4 times as many as before. Since all the rearrangements are done locally within each processor, so the time will have a small growth if the problem size doubled.

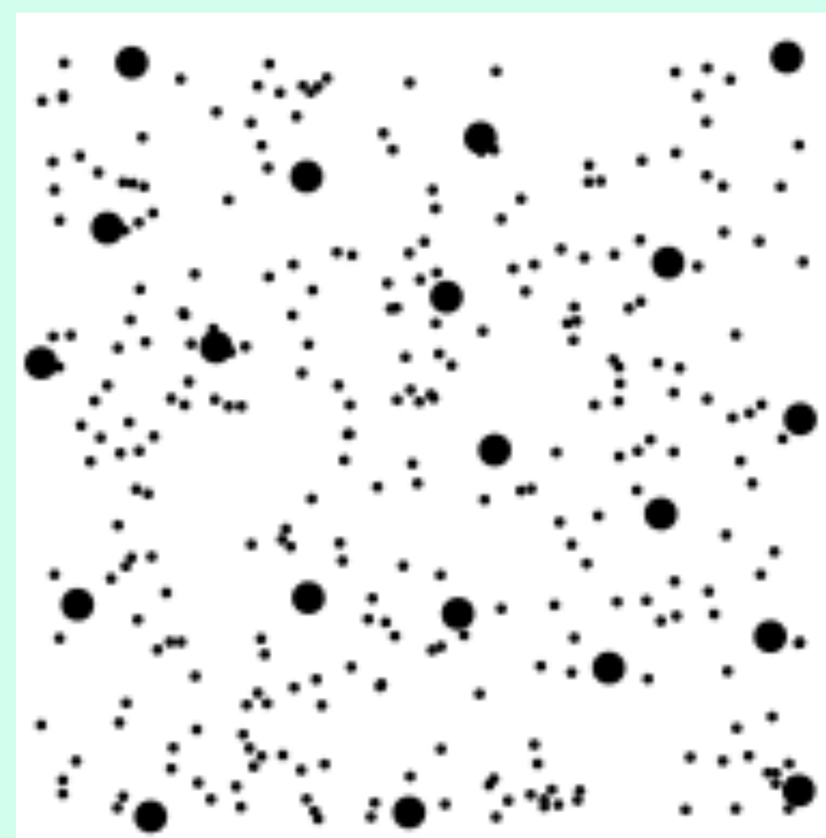
Choose the landmarks

After obtaining the X(K,p), we recommend selecting the landmark points by *maxmin* method. These landmarks are used to build the simplicial complex.

Algorithm:

- Initialise by selecting $l_1 \in Z$ randomly.
- For each $i \geq 2$, if $l_1, l_2, \ldots, l_{i-1}$ have been chosen, let $l_i \in Z \setminus l_1, l_2, \ldots, l_{i-1}$ be the data point which maximises the function $f(x) = \min_{1<=j<=i-1} D(x, j)$ where D is the normal metric.

example of landmarks from data set:



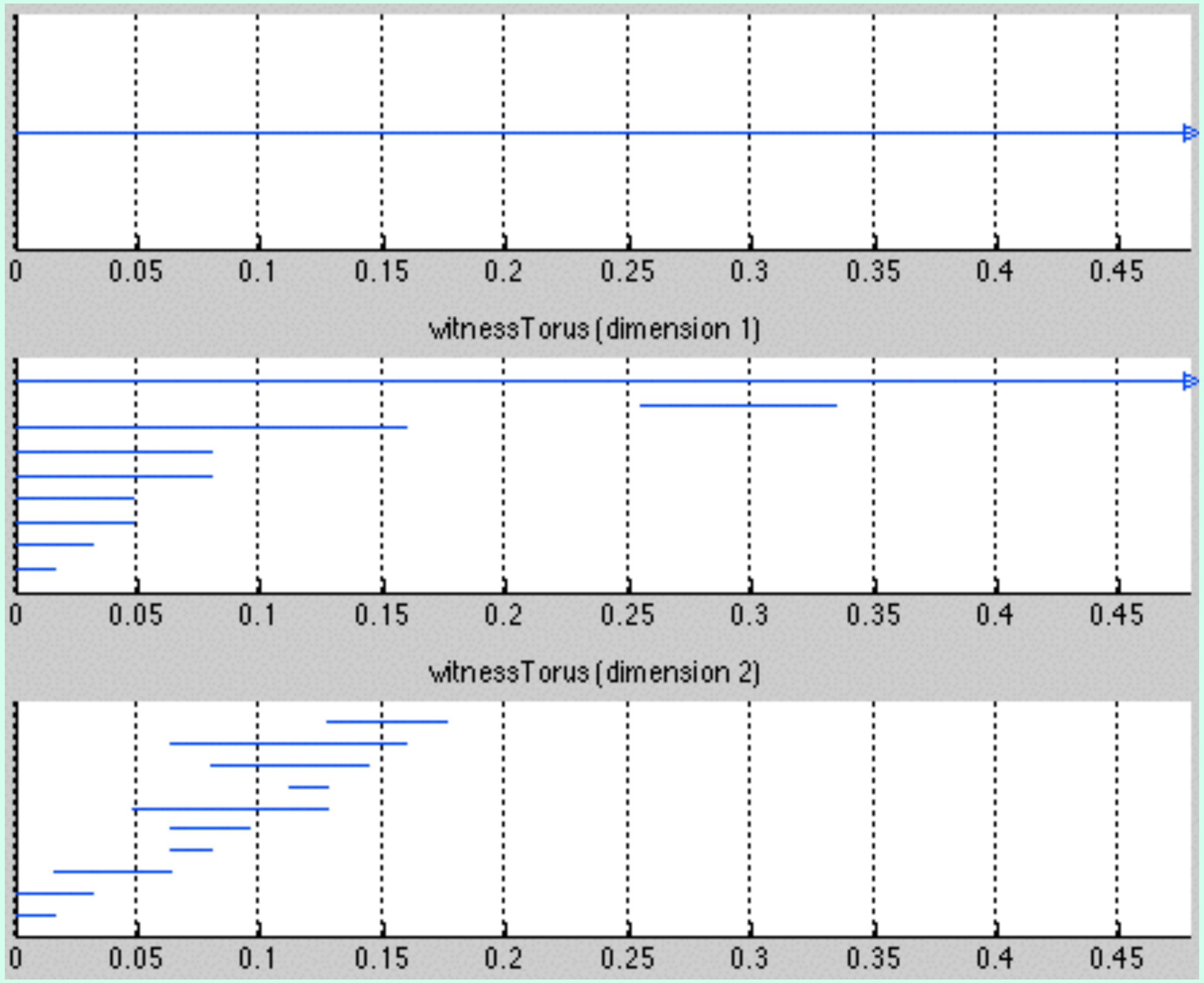
Computation of homology

- We use the landmarks to build the simplicial complex.
- Compute the homology using ***Fundamental lemma of topology.*** Then we can get the betti number which indicates *how many holes in each dimension* of the shape that the data set formed. For more details about the knowledge of topology and computing the betti number, it is suggested to look at *H.Edelsbrunner, COMPUTATIONAL TOPOLOGY: An Introduction (2008).*

Javaplex is a ***Java package*** developed by ***Stanford University*** and it can directly give us the barcode of holes in each dimension for input data.

Javaplex example

- The first barcode represents the betti0 which is the number of the connected components
- Betti1 and 2 represent the number of 1-dim and 2-dim holes in the graph which are the second and 3rd barcodes.

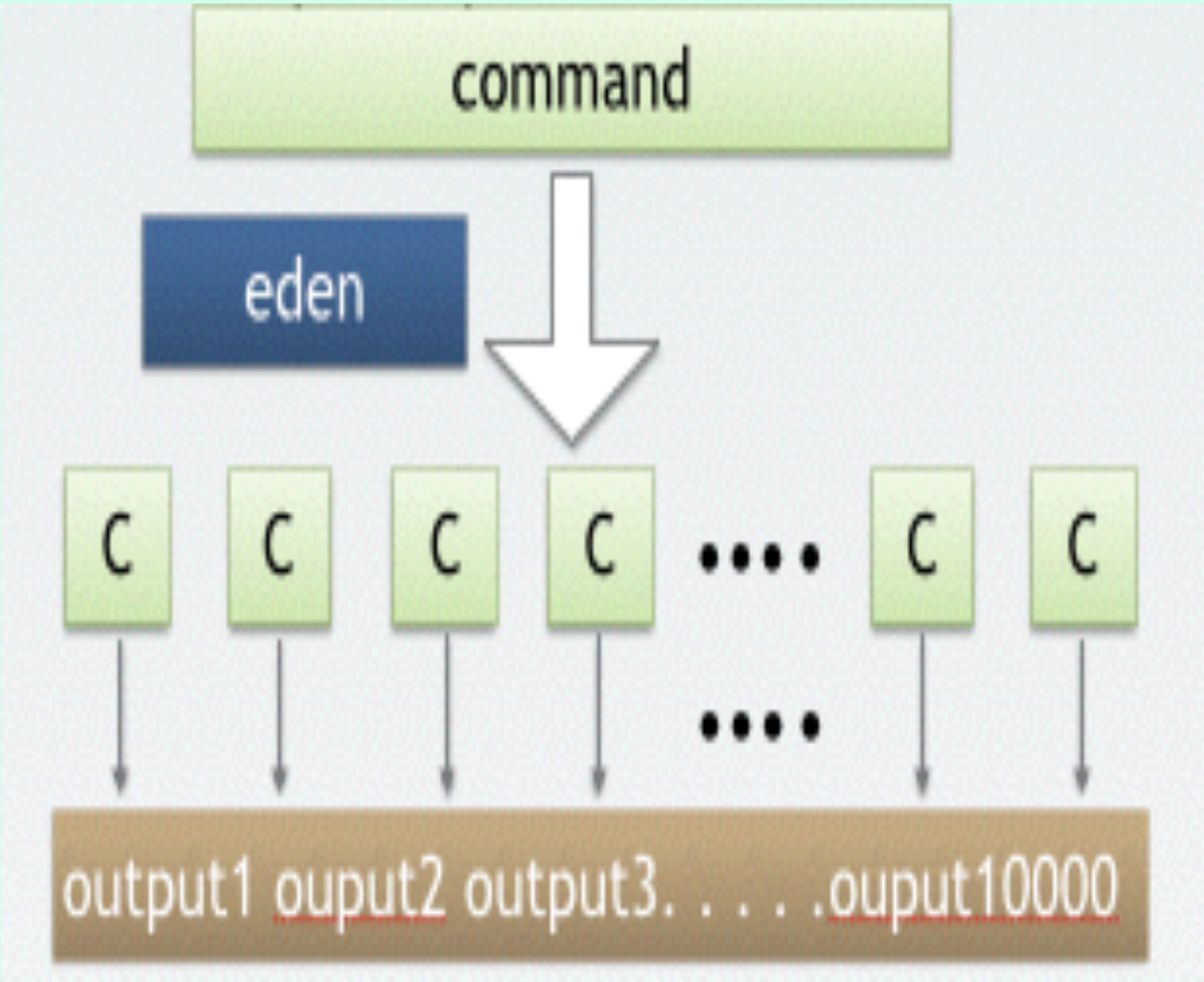


✤ Note: The blue line indicates the existence of the hole with the change of parameter. Even though there are some short lines which are the noise, we can still concentrate on the long lasting lines.

Large-Scale Parellel Numerical simulation

- Note: Betti number depends on the landmarks while the first landmark is chose *randomly* as it impacts on the choice of other points and further the whole set of landmarks.

- Because of the undetermined characteristic of landmarks, we need to run *Javaplex* for 10,000 times for each K and p to give a statistical certainty of 97% of the betti number. To implement the large-scale simulation, we use ***Eden*** on the ***Nautilus*** as it can speed up our calculation for a lot.



Eden

- Eden is a script-based tool for easily managing runs of many small jobs on Nautilus or Darter ***without flooding the job queue***.
- Here is my run file for ***Javaplex***.

run1.sh file:

```
#PBS -S /bin/bash

cd $PBS_O_WORKDIR
eval `/usr/bin/modulecmd bash load java`
sleep 1
java -classpath .:javaplex-4.1.0.jar WitnessComplexDemo
```

✦Note: run1.sh is the command to run ***javaplex***. The tricky thing is that on eden every time you run it, you must find the directory of ***module*** and ***load*** java. Furthermore, ***sleep 1*** is needed since you can only run ***javaplex*** when you have already loaded java.

Eden

- Basically, we can put the command of running Javaplex for 10,000 times into the command file and create the head file, then Eden can do it for 10,000 times and gives out the output.

commands file:

```
./run1.sh
./run1.sh
./run1.sh
./run1.sh
./run1.sh
./run1.sh
./run1.sh
"commands" 1000L, 10000C
```

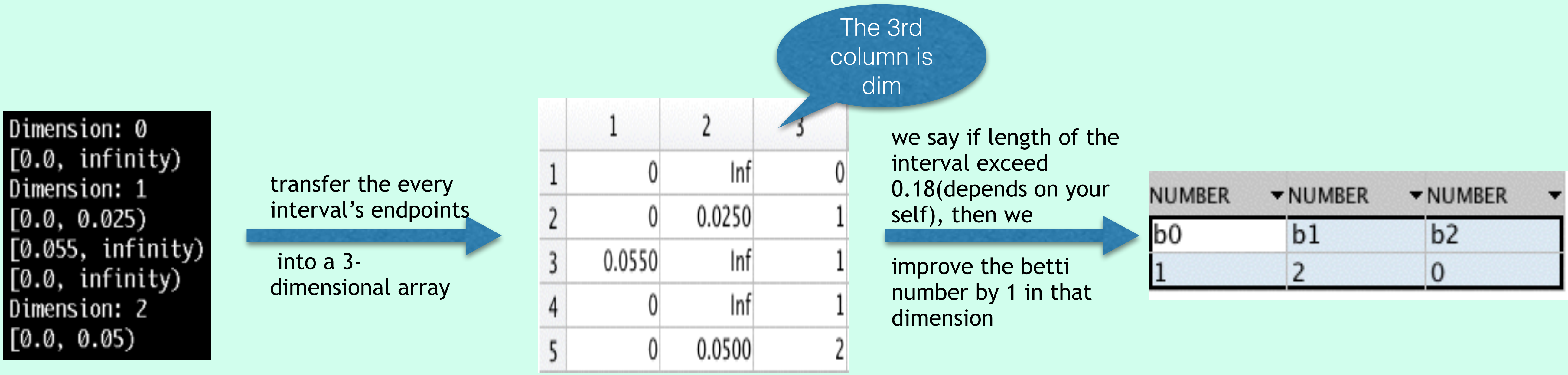
head.pbs file:

```
#PBS -l ncpus=128
#PBS -l mem=256GB
#PBS -l walltime=02:00:00
#PBS -N java_test_10000
#PBS -A UT-INTERN
~
~
~
~
```

- After that, we can do ‘cat *.out > out.txt’ to make all the output into one txt file.

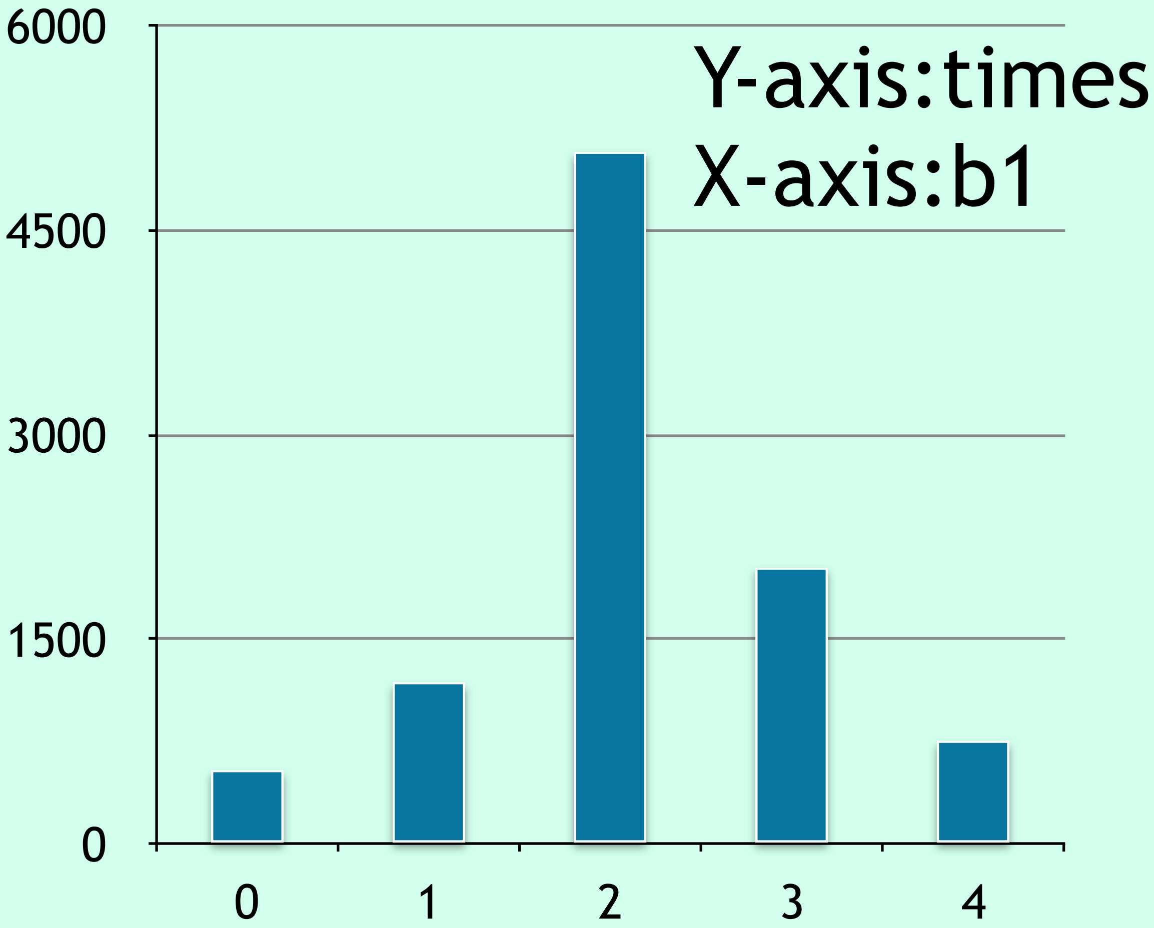
Analysis of output

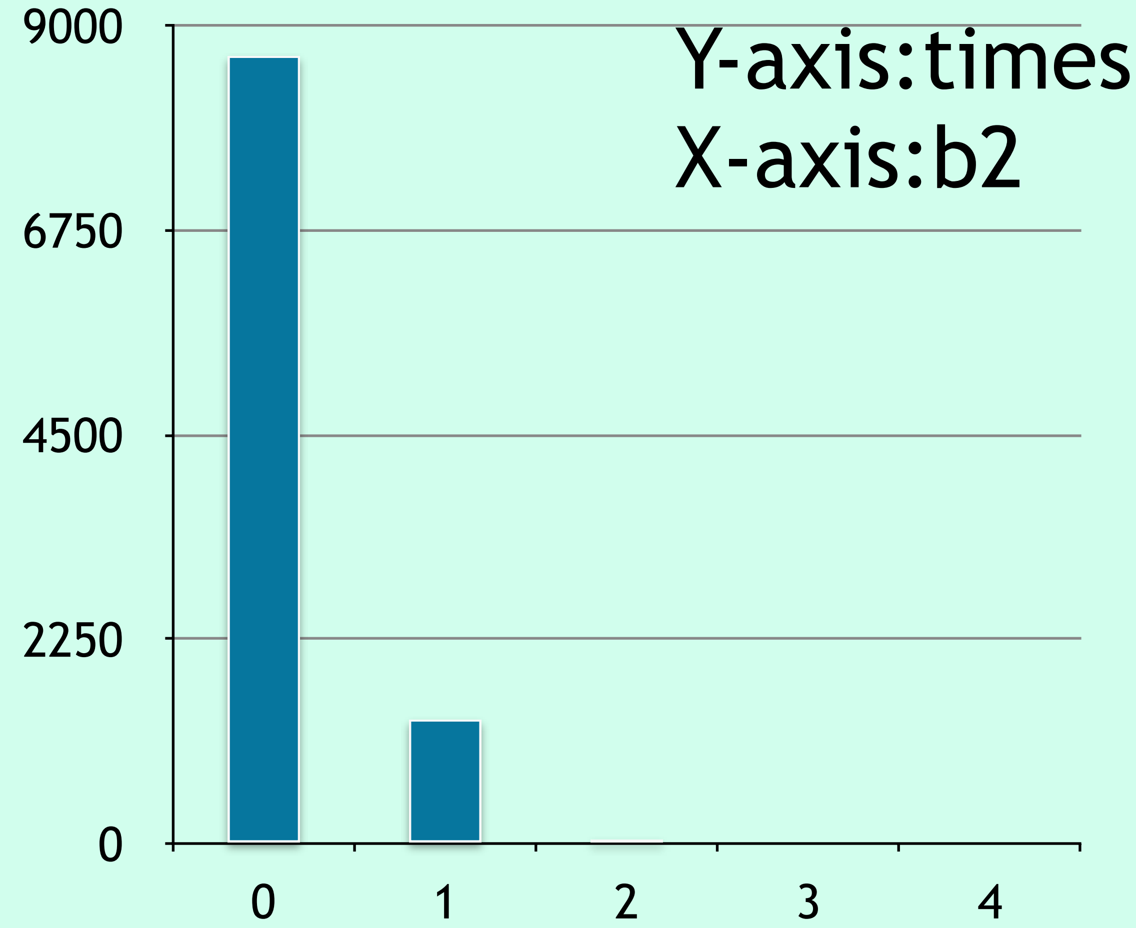
- The output is 10,000 collections of barcodes. It is not reasonable to analysis each output by hand. So, better way is to transfer the result to matrix and write some programs in Matlab to judge the number of holes in each dimension. My basic idea is below:



- Count the times of appearance of different values for each betti number and make histograms for each betti number.
- For K=50,P=50, the histograms for betti numbers of X(K,P) are in the next page.(b0 are always 1 since only 1 connected component is detected.)

Analysis of output



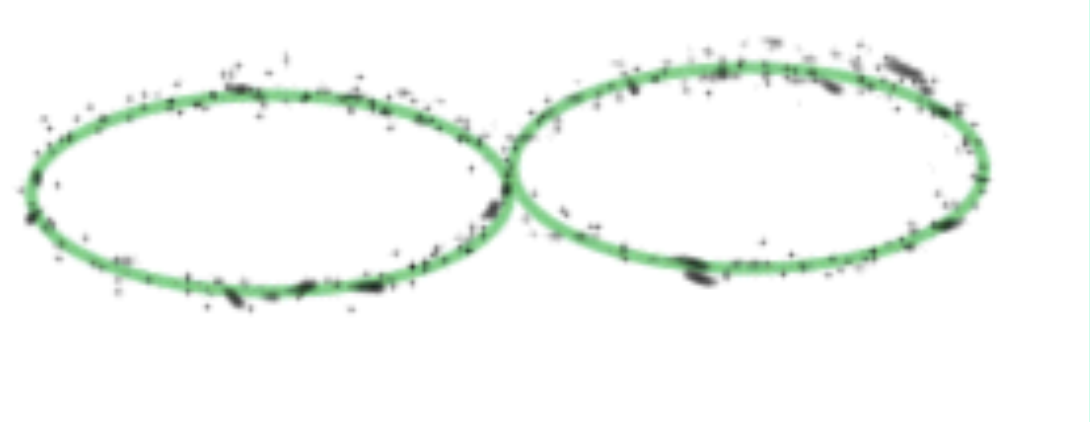


- Statistically speaking, through the distribution of the output for 10,000 iterations, we can have 97% certainty that the b1=2 and b2=1.

Conclusion and Future Study

- However, this is just the case for K=p=50, more experiments for different K and p are in the process. Through the experiments that we have conducted until now, the result that b1=2 and b2=0 is quite solid.For this property, the shape below may have much possibility to capture this dataset.

- Interesting thing is that bi=0 for any 3<=i<=7 in every iteration. The happening of this result on a 8-dim data set tells us that there may be some relationships between some of the factors of these 8. Further analysis will be expected.



References and Acknowledgement

[1] V. de Silva and G. Carlsson. *Topological estimation using witness complexes*, Eurographics Symposium on Point-Based Graphics, 2000.

[2] H.Edelsbrunner, *COMPUTATIONAL TOPOLOGY: An Introduction* (2008).

[3] H.Edelsbrunner, D.Letscher and A.Zomorodian, *Topological Persistence and Simplification*,Discrete Comput Geom*, 28:511-533,2002*.

[4] G. Carlsson, T.Ishkhanov, V. de Silva, A.Zomorodian,*On the Local Behavior of Spaces of Natural imges*, Springer, LLC2007.

The research was conducted under the Computational Science for REU project and is supported by the JICS, founded by the UTK and ORNL. The authors acknowledge ORNL for allowing access to high-performance computing resources. Also, Jacob Pollack and Pragnesh's helping in Eden part is appreciated a lot.