Homework 8

1.

```
load_javaplex
import edu.stanford.math.plex4.*;
% api.Plex4.createExplicitSimplexStream()
```

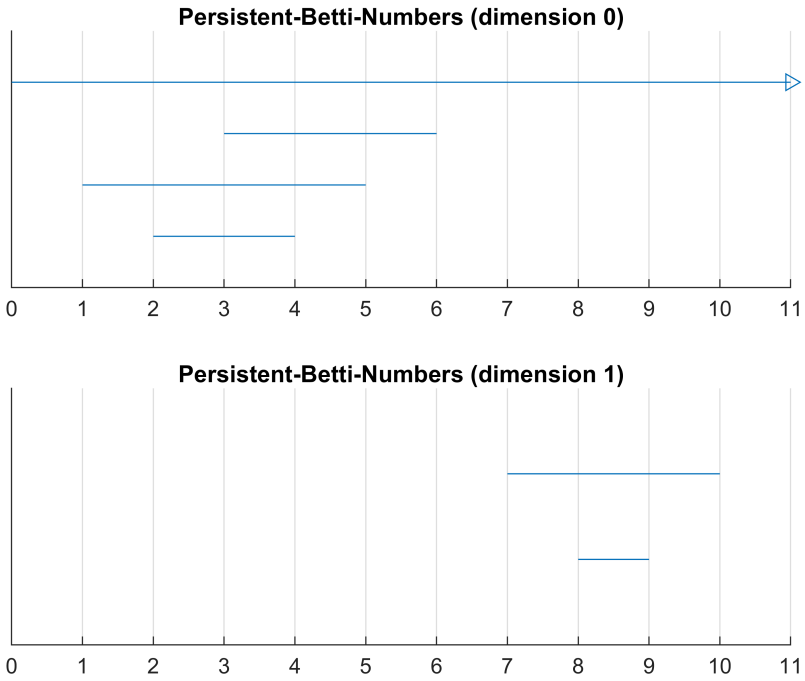
(a).

```
load_javaplex
import edu.stanford.math.plex4.*;
stream = api.Plex4.createExplicitSimplexStream();
stream.addVertex(0,0);
stream.addVertex(1,1);
stream.addVertex(2,2);
stream.addVertex(3,3);
stream.addElement([0,2],4);
stream.addElement([0,1],5);
stream.addElement([2,3],6);
stream.addElement([1,3],7);
stream.addElement([1,2],8);
stream.addElement([1,2,3],9);
stream.addElement([0,1,2],10);
stream.finalizeStream();
num_simplices = stream.getSize()
```

```
num_simplices = 11
```

(b).

```
% Compute the Z/2Z persistence homology of dimension less than 3:
import edu.stanford.math.plex4.*;
persistence = api.Plex4.getModularSimplicialAlgorithm(3, 2);
intervals = persistence.computeIntervals(stream);
options.filename = 'Persistent-Betti-Numbers';
options.max_filtration_value = 11;
% Plot the barcode of persistent Betti numbers:
plot_barcodes(intervals, options);
```

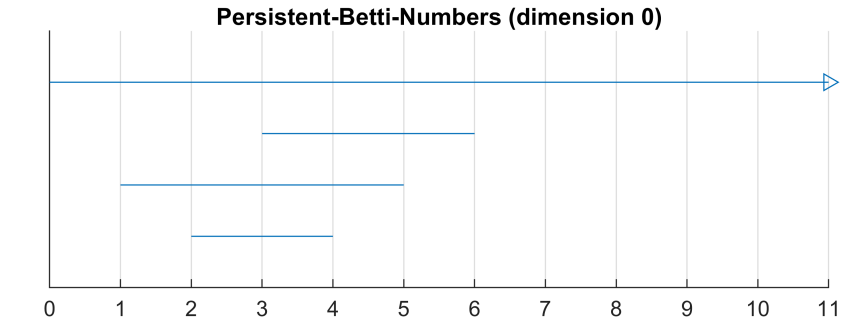


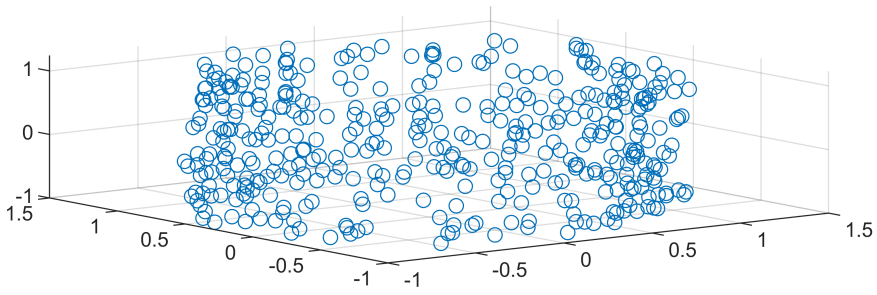
2.

```
load_javaplex
```

(a).

```
load tutorial_examples/pointsTorusGrid.mat
scatter3(pointsTorusGrid(:,1), pointsTorusGrid(:,2), pointsTorusGrid(:,3))
```





(b).

```
max_dimension = 3;
max_filtration_value = 0.9;
num_divisions = 1000;
import edu.stanford.math.plex4.*;
stream = api.Plex4.createVietorisRipsStream(pointsTorusGrid, max_dimension, max_filtration_valu
num_simplices = stream.getSize()
```

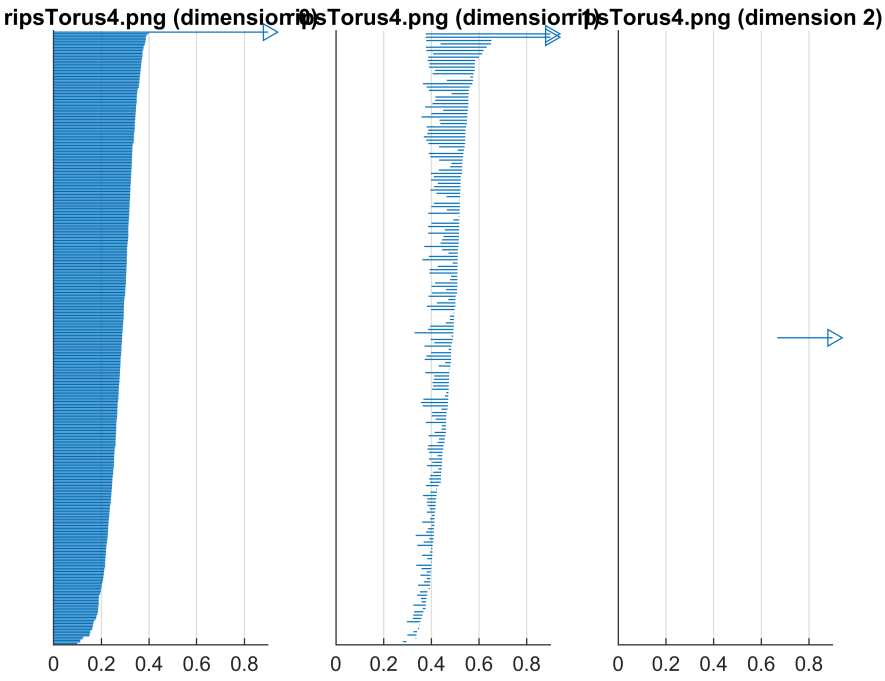
```
num_simplices = 82479
```

(c).

```
import edu.stanford.math.plex4.*;
persistence = api.Plex4.getModularSimplicialAlgorithm(max_dimension, 2);
intervals = persistence.computeIntervals(stream);
```

(d).

```
options.filename = 'ripsTorus4.png';
options.max_filtration_value = max_filtration_value;
options.max_dimension = max_dimension - 1;
options.side_by_side = true;
plot_barcodes(intervals, options);
```


ripsTorus4.png (dimension 0)
0 0.2 0.4 0.6 0.8
ripsTorus4.png (dimension 1)
0 0.2 0.4 0.6 0.8
ripsTorus4.png (dimension 2)
0 0.2 0.4 0.6 0.8