DS-GA 1014 - Homework 8

Eric Niblock

November 2nd, 2020

1. (2 points). For any two matrices $A, B \in \mathbb{R}^{n\times m}$ we define

$$\langle A, B\rangle_F = Tr(A^T B)$$

(a) Show that $\langle \cdot, \cdot\rangle_F$ is an inner-product on $\mathbb{R}^{n\times m}$, i.e. that it verifies the points of the definition 2.1 of Lecture 4. $\langle \cdot, \cdot\rangle_F$ is called the Frobenius inner-product.

First, we introduce the necessary definition regarding inner products.

Let V be a vector space. An inner product on V is a function $\langle \cdot, \cdot\rangle$ from $V \times V$ to $\mathbb{R}$ that verifies the following points:

(1) Symmetry: $\langle \vec{v}, \vec{u}\rangle = \langle \vec{u}, \vec{v}\rangle$ for all $\vec{u}, \vec{v} \in V$

(2) Linearity: $\langle \vec{u} + \vec{v}, \vec{w}\rangle = \langle \vec{u}, \vec{w}\rangle + \langle \vec{v}, \vec{w}\rangle$ and $\langle \alpha\vec{v}, \vec{w}\rangle = \alpha\langle \vec{v}, \vec{w}\rangle$ for all $\vec{u}, \vec{v}, \vec{w} \in V$ and $\alpha \in \mathbb{R}$

(3) Positive definiteness: $\langle \vec{v}, \vec{v}\rangle \geq 0$ with equality if and only if $\vec{v} = \vec{0}$

[Def. 1]

Now we begin to show that these conditions are satisfied. Concerning (1), it first must be clear that $Tr(M) = Tr(M^T)$ for any $M \in \mathbb{R}^{k\times k}$. This is true since,

$$Tr(M) = \sum_{i=1}^{k} M_{i,i} \tag{1}$$

And the act of transposing a square matrix does not change the diagonal terms. Since $A^T B \in \mathbb{R}^{m\times m}$, we can write the following,

$$\langle A, B\rangle_F = Tr(A^T B) = Tr((A^T B)^T) = Tr(B^T A) = \langle B, A\rangle_F \tag{2}$$

This satisfies condition (1). Concerning (2), if we have $A, B, C \in \mathbb{R}^{\mathrm{n}\times\mathrm{m}}$, then we have,

$$\begin{aligned}\langle A + B, C\rangle_F &= Tr((A + B)^T C) = Tr((A^T + B^T)C) \\ &= Tr(A^T C + B^T C) = Tr(A^T C) + Tr(B^T C) \\ &= \langle A, C\rangle_F + \langle B, C\rangle_F\end{aligned} \tag{3}$$

Where the second line follows from the fact that the trace is a linear mapping. We employ this property again, concerning scalar multiplication. We have that,

$$\langle \alpha A, B\rangle_F = Tr(\alpha A^T B) = \alpha Tr(A^T B) = \alpha\langle A, B\rangle_F \tag{4}$$

This suffices to show (2). Concerning (3), we know that,

$$\begin{aligned}\langle A, A\rangle_F = Tr(A^T A) = \sum_{i=1}^{m}(A^T A)_{i,i} = \sum_{i=1}^{m}\sum_{j=1}^{n} A^T_{i,j}A_{j,i} \\ = \sum_{i=1}^{m}\sum_{j=1}^{n} A^2_{i,j} \geq 0\end{aligned} \tag{5}$$

And furthermore, since all $A^2_{i,j} \geq 0$, we have that the sum of all $A^2_{i,j} = 0$ only when all $A_{i,j} = 0$, meaning would have to be zero $A = 0$. This satisfies (3). Therefore the Frobenius inner-product is a valid inner product.

(b) The induced norm $||A||_F = \sqrt{Tr(A^T A)}$ is called the Frobenius norm. Show that

$$||A||_F = \sqrt{\sum_{i=1}^{min(n,m)} \sigma_i^2}$$

where $\sigma_1, ..., \sigma_{min(n,m)}$ are the singular values of A.

First we note that $Tr(AB) = Tr(BA)$ for $A \in \mathbb{R}^{\mathrm{n}\times\mathrm{m}}$ and $B \in \mathbb{R}^{\mathrm{m}\times\mathrm{n}}$. Furthermore, we know that we can express $A = U\Sigma V^T$, where U and V are orthonormal and Σ is diagonal. So,

$$\begin{aligned} ||A||_F^2 &= Tr((U\Sigma V^T)^T(U\Sigma V^T)) = Tr(V\Sigma^T U^T U\Sigma V^T) \\ &= Tr(V\Sigma^T\Sigma V^T) = Tr((V\Sigma^T)(\Sigma V^T)) \end{aligned} \tag{6}$$

Then we can use the property that $Tr(AB) = Tr(BA)$ so that,

$$Tr((V\Sigma^T)(\Sigma V^T)) = Tr(\Sigma V^T V\Sigma^T) = Tr(\Sigma\Sigma^T) \tag{7}$$

And since Σ is diagonal, the result of $\Sigma\Sigma^T$ is simply a diagonal matrix with all of the terms along the main diagonal being the square of the terms on the main diagonal of Σ, which are simply the singular values of Σ. In summary,

$$||A||_F^2 = Tr(\Sigma\Sigma^T) = \sum_{i=1}^{min(n,m)} \sigma_i^2 \tag{8}$$

So it is then clear that,

$$||A||_F = \sqrt{\sum_{i=1}^{min(n,m)} \sigma_i^2} \tag{9}$$

2. (2 points). Let A be a $n \times n$ matrix.

(a) Show that A is invertible if and only if all the singular values of A are non-zero.

We must show the implication in both directions. First, we show that if all the singular values of A are non-zero, then A is invertible.

The relationship between the singular values and eigenvalues is given by, $\sigma_i = \sqrt{\lambda_i}$, where the eigenvalues correspond to $A^T A$ and the singular values correspond to A. Any $n \times n$ matrix A can be decomposed via SVD, yielding, $A = U \Sigma V^T$, where in this case, $\Sigma_{1,1} \geq ... \geq \Sigma_{n,n} > 0$. Furthermore, we know that,

$$\vec{\mathbf{u}}_i = \frac{A\vec{\mathbf{v}}_i}{\sigma_i} \tag{10}$$

And that $u_1, ..., u_r$ will form a basis for the $Im(A)$. However, since $\sigma_1 \geq ... \geq \sigma_n > 0$, we have that $u_1, ..., u_r = u_1, ..., u_n$, and $r = n$. This implies that matrix A is of full rank, and any square matrix of full rank is invertible.

Now we must show that if A is invertible, then the singular values of A are non-zero. Since A is invertible, we have $rank(A) = n$. That means $rank(U\Sigma V^T) = n$. We know that $rank(U) = rank(V^T) = n$, since U and V^T are invertible and square, of size $n \times n$. Therefore, we know that $rank(\Sigma) = n$, otherwise $rank(U\Sigma V^T) < n$. So, we conclude that Σ must be composed of all non-zero singular values, otherwise we would have $rank(\Sigma) < n$. Thus, if A is invertible, then the singular values of A are non-zero.

Having shown the implication in both directions, we have shown that A is invertible if and only if all the singular values of A are non-zero.

(b) We assume here that A is invertible. Show that

$$\sigma_1(A)\sigma_1(A^{-1}) \geq 1$$

where $\sigma_1(A)$ and $\sigma_1(A^{-1})$ denote the largest singular value of respectively A and A^{-1}.

Given that A is invertible, all of the singular values of A are non-zero. Since A is invertible, we have $A = U\Sigma V^T$, but also $A^{-1} = (U\Sigma V^T)^{-1} = (V^T)^{-1}\Sigma^{-1}U^{-1}$. Since V and U are orthonormal, we have the further simplification that $A^{-1} = V\Sigma^{-1}U^T$. Now the inverse of a diagonal matrix is simply the original diagonal matrix, with each entry along the main diagonal being replaced by its reciprocal.

This is evident by the fact that then $\Sigma_{i,i}\Sigma_{i,i}^{-1} = 1$ and hence $\Sigma\Sigma^{-1} = Id$. Therefore, if A is invertible, all of the singular values of A^{-1} are simply the reciprocals of the singular values of A.

As was just explained, the singular values of A^{-1} are the reciprocals of the singular values of A. Call $\sigma_1 \geq ... \geq \sigma_n > 0$ the singular values of A. Then, $0 < \frac{1}{\sigma_1} \leq ... \leq \frac{1}{\sigma_n}$ are the singular values of A^{-1}. So, we have,

$$\sigma_1(A)\sigma_1(A^{-1}) = \sigma_1\left(\frac{1}{\sigma_n}\right) = \frac{\sigma_1}{\sigma_n} \geq 1 \tag{11}$$

Which is obviously true, since we know that $\sigma_1 \geq \sigma_n > 0$.

3. (2 points). Let $A \in \mathbb{R}^{n \times n}$ be the adjacency matrix of a graph G. We define a path from a node i_1 to a node i_k as a succession of nodes $i_1, i_2, ..., i_k$ such that

$$i_1 \sim i_2 \sim ... \sim i_{k-1} \sim i_k, \qquad \textbf{i.e.} \qquad A_{i_1,i_2} = A_{i_2,i_3} = ... = A_{i_{k-1},i_k} = 1$$

The nodes ij of the path do not need to be distinct. We say that the path $i_1, i_2, ..., i_k$ has length $k-1$ which is the number of edges in this path. The goal of this exercise is to prove that for all $k \geq 1$

$\mathcal{H}(k)$: For all $i, j \in \{1, ..., n\}$ the number of paths of length k from i to j is $(A^k)_{i,j}$

We will prove that $\mathcal{H}(k)$ holds for all k by induction, that is, we will first prove that $\mathcal{H}(1)$ is true. Then we will prove that if $\mathcal{H}(k)$ is true for some k, then $\mathcal{H}(k+1)$ is true. Combining these two things, we get that $\mathcal{H}(2)$ holds, hence $\mathcal{H}(3)$ holds, hence $\mathcal{H}(4)$ holds... and therefore $\mathcal{H}(k)$ will be true for all $k \geq 1$.

(a) Show that $\mathcal{H}(1)$ is true.

$\mathcal{H}(1)$ is clearly true. The number of paths of length 1 from node i to node j is given by $(A^1)_{i,j} = (A)_{i,j}$ which is simply the i,j entry of the adjacency matrix A. If the i,j entry of the adjacency matrix A is 1, this means the node i and node j are connected directly by an edge - the length between them is therefore 1. Otherwise entry i,j is zero. There cannot exist a direct connection between nodes i and j that has a length other than 1. Therefore, $\mathcal{H}(1)$ is clearly true.

(b) Show that if $\mathcal{H}(k)$ is true for some k, then $\mathcal{H}(k+1)$ is also true.

We make the assumption that $\mathcal{H}(k)$ is true for some k. The problem we face is determining the length between node i and node j. If we consider any vertex m such that m is connected directly to j by an edge, and indirectly to node i (or directly), then we know the length between node i and node m is k. Then, if we consider all of the nodes m where this is setup is possible, we find the total number of possible paths of length $k+1$ from i to j. Mathematically, we can write this as,

$$\sum_{m \in \{1,...,n\}} (A^k)_{i,m}(A)_{m,j} = (A^{k+1})_{i,j} \qquad (12)$$

However, this is precisely the result we hoped to find. Thus the induction is complete.

4. **(4 points). The goal of this problem is to use spectral clustering techniques on real data. The file *adjacency.txt* contains the adjacency matrix of a graph taken from a social network. This graphs has $n = 328$ nodes (that corresponds to users). An edge between user i and user j means that i and j are "friends" in the social network. The notebook *friends.ipynb* contains functions to read the adjacency matrix as well as instructions/questions.**

   While we focused in the lectures (and in the notes) on the graph Laplacian

$$L = D - A$$

   where A is the adjacency matrix of the graph, and $D = Diag(deg(1), ..., deg(n))$ is the degree matrix, we will use here the "normalized Laplacian" (instead of L)

$$L_{norm} = D^{-1/2} L D^{-1/2} = Id - D^{-1/2} A D^{-1/2}$$

   where $D^{-1/2} = Diag(deg(1)^{-1/2}, ..., deg(n)^{-1/2})$. The reason for using a different Laplacian is that then "unnormalized Laplacian" L does not perform well when the degrees in the graph are very broadly distributed, i.e. very heterogeneous. In such situations, the normalized Laplacian L_{norm} is supposed to lead to a more consistent clustering.

   Below, you will find the attached python file, containing all of the work for this question.

In [1]:
```
%matplotlib inline
import matplotlib
import numpy as np
import matplotlib.pyplot as plot
from sklearn.cluster import KMeans
```

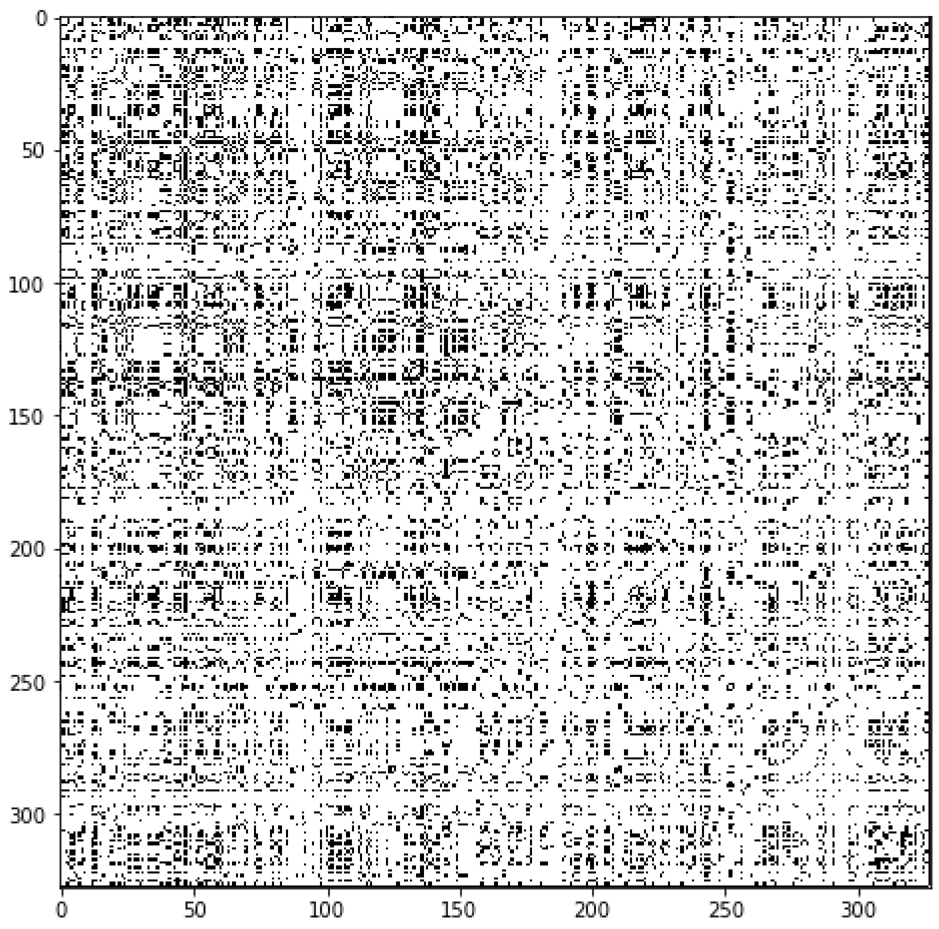
In [2]:
```
# Reads the adjacency matrix from file
A=np.loadtxt('adjacency.txt')
print(f'There are {A.shape[0]} nodes in the graph.')
```

```
There are 328 nodes in the graph.
```

As you can see above, the adjacency matrix is relatively large (328x328): there are 328 persons in the graph. In order to visualize this adjacency matrix, it is convenient to use the 'imshow' function. This plots the 328x328 image where the pixel (i,j) is black if and only if A[i,j]=1.

In [3]:
```
plot.figure(figsize=(8,8))
plot.imshow(A,aspect='equal',cmap='Greys',  interpolation='none')
```

Out[3]:
```
<matplotlib.image.AxesImage at 0x2871389bc88>
```



(a) Construct in the cell below the degree matrix:

$$D_{i,i} = \deg(i) \qquad \text{and} \qquad D_{i,j} = 0 \ \text{ if } i \neq j,$$

the Laplacian matrix:

$$L = D - A$$

and the normalized Laplacian matrix:

$$L_{\text{norm}} = D^{-1/2} L D^{-1/2}.$$

In [84]:
```
diags = np.array([sum(A[:,i]) for i in range(len(A))])
D = np.diag(diags)

L = D-A

sqrt_diags = 1/diags**0.5
Dsq = np.diag(sqrt_diags)

L_norm = np.matmul(L,Dsq)
L_norm = np.matmul(Dsq,L_norm)
```

(b) Using the command 'linalg.eigh' from numpy, compute the eigenvalues and the eigenvectors of L_{norm}.

In [108]:
```
vals, vect = np.linalg.eigh(L_norm)
```

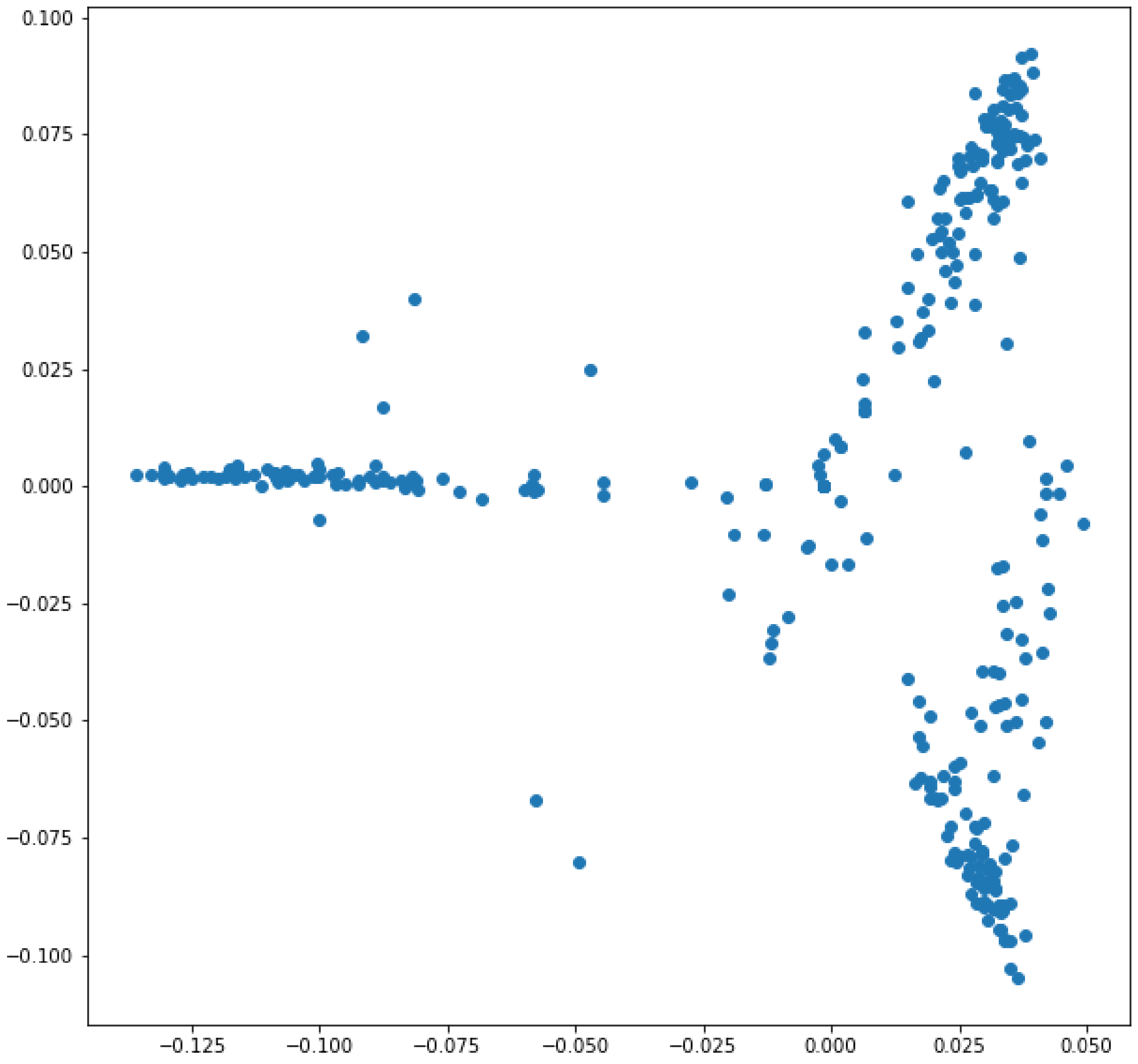
(c) We would like to cluster the nodes (i.e. the users) in 3 groups. Using the eigenvectors of L_{norm}, assign to each node a point in $\mathbb{R}^2$, exactly as in 'Algorithm 1' of the notes where you replace L by L_{norm}. Plot these points using the 'scatter' function of matplotlib.

In [109]:
```
R2_nodes = vect[:,1:3]
```

In [110]:
```
import matplotlib.pyplot as plt

plt.figure(figsize=(10,10))
plt.scatter(R2_nodes[:,0],R2_nodes[:,1])
```

Out[110]:
```
<matplotlib.collections.PathCollection at 0x28718644ac8>
```

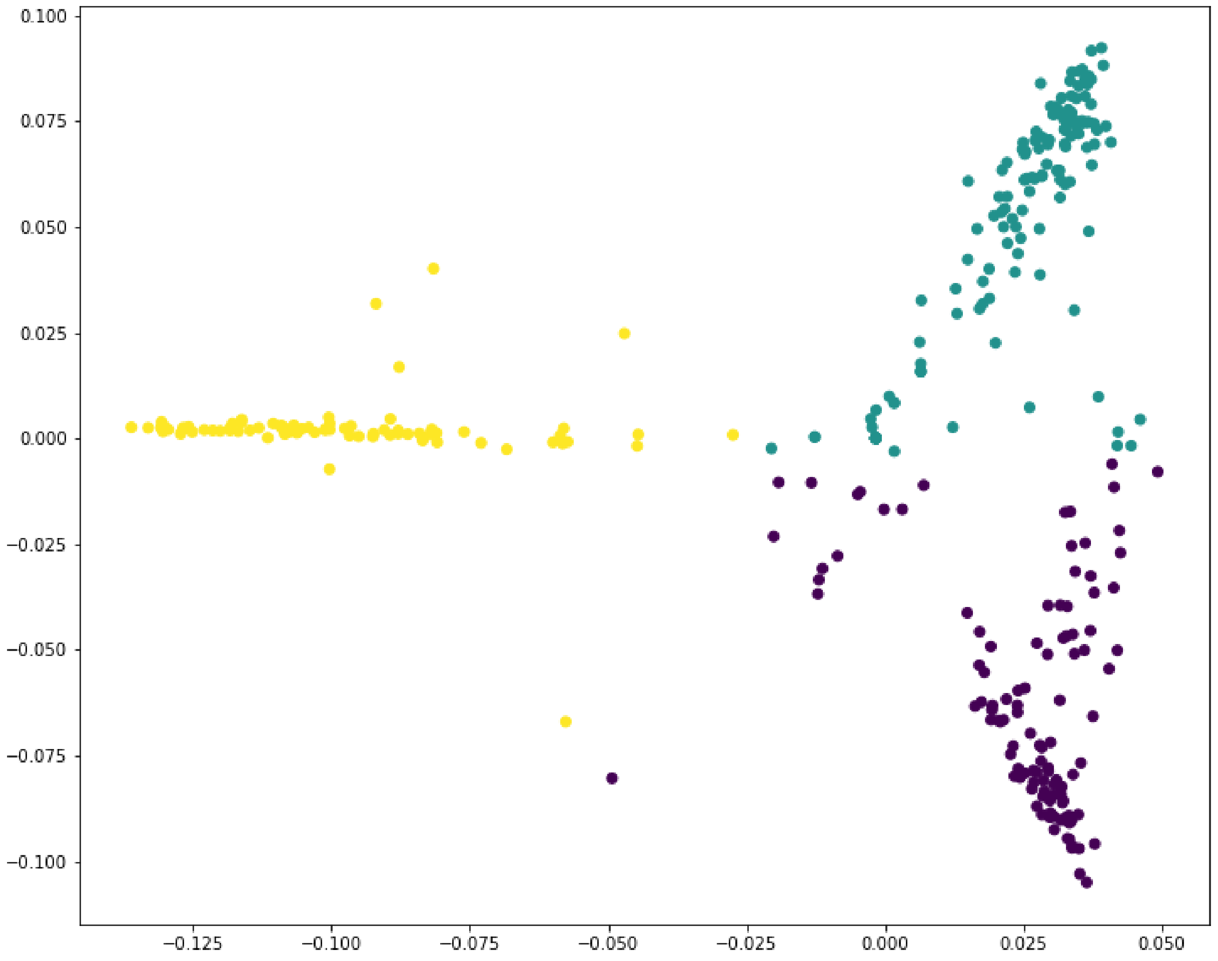


(d) Using the K-means algorithm (use the built-in function from scikit-learn), cluster the embeddings in $\mathbb{R}^2$ of the nodes in 3 groups.

In [114]:
```
# Replace ??? by the matrix of the points computed in (c)
# Each row corresponds to a data point
kmeans = KMeans(n_clusters=3, random_state=0).fit(R2_nodes)
labels=kmeans.labels_
# labels contains the membership of each node 0,1 or 2

# This colors each point of R^2 according to its label
# replace "x/y coordinates" by the coordinates you computed in (c)
plt.figure(figsize=(12,10))
plt.scatter( R2_nodes[:,0], R2_nodes[:,1], c = labels)
```

Out[114]:
```
<matplotlib.collections.PathCollection at 0x2871ad0b5f8>
```



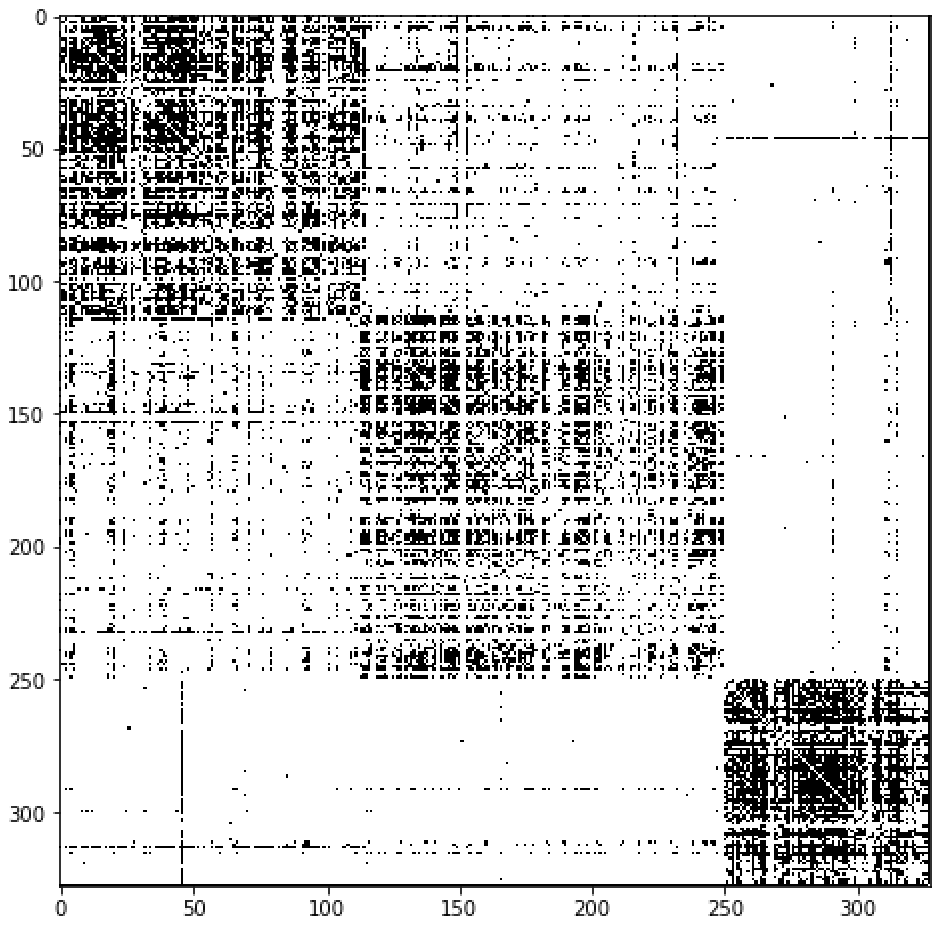
(e) Re-order the adjacency matrix according to the clusters computed in the previous question. That is, reorder the columns and rows of A to obtain a new adjacency matrix (that represents of course the same graph) such that the n_1 nodes of the first cluster correspond to the first n_1 rows/columns, the n_2 nodes of the second cluster correspond to the next n_2 rows/columns, and the n_3 nodes of the third cluster correspond to the last n_3 rows/columns. Plot the reordered adjacency matrix using 'imshow'.

In [115]:
```
ind = list(range(len(labels)))
sort_labels, sort_ind = (list(t) for t in zip(*sorted(zip(labels, ind))))
```

In [116]:
```
reorderedA = A[sort_ind,:] ## Move all rows
reorderedA = reorderedA[:,sort_ind] ## Move all columns
```

In [117]:
```
plot.figure(figsize=(8,8))
plot.imshow(reorderedA,aspect='equal',cmap='Greys',  interpolation='none')
```

Out[117]:
```
<matplotlib.image.AxesImage at 0x28719b8e780>
```


0
50
100
150
200
250
300
0
50
100
150
200
250
300