Master de Chemoinformatique et Modélisation M1S1
Examen de mathématiques pour la chimie*

MARCOU Gilles

2013-2014

Résumé

Documents autorisés. Durée : 2h. La copie de l'étudiant sera constituée d'un support papier traditionnel et d'un ou plusieurs fichiers de calcul pour Maple. ***Toute réponse doit être motivée ou sera considérée comme nulle.***

Exercise 1

This exercise aims at testing a projection method based on the description of a dataset as a graph : the Laplacian eigenmap algorithm [Neural Computation, June 2003 ; 15 (6) :1373-1396]. The projection method consider that the dimensionality of the space embedding the dataset is too large : the dataset itself is approximately contained in a much simpler manifold. The objective is to project the initial dataset on a lower dimensionality space (in general 2D), while keeping the representation meaningful.

To start, each point of the dataset is considered a node of a graph G. Edges are connecting neighboring point : two instances i and j are connected if their distance $d_{ij} < \epsilon$. Then, each edge is weighted and the weight of the edge between the node i and j is stored in a matrix W, and the corresponding component $W_{ij} = e^{-\frac{d_{ij}}{\sigma^2}}$ while all outer diagonal elements are 0. Next, the degree matrix D of the weighted graph is computed. The degree matrix is a diagonal matrix containing as a diagonal element the sum of all components of the corresponding line in the matrix W : $D_{ii} = \sum_{j=1}^{N} W_{ij}$.The Laplacian matrix $L = D - W$ is then computed. This matrix can be interpreted as an estimate of the Laplace Beltrami differential operator on the manifold embedding the initial data. Finally, the vectors solutions to the generalized eigenvalues problem $L.v = \lambda.D.v$ with the lowest eigenvalues contains the projection of the instances on relevant subspaces, capturing information on the local curvature this manifold. The eigenvalues are interpreted as a measure of how much on average, nearest neighbors in the initial space results into neighboring points in the resulting projection.

The algorithm uses three steps :

1. build the adjacency matrix of the graph representing the data ;
2. weight the edges and compute the matrices W, D and L (respectively the weights, the degree and the Laplacian matrices) ;
3. Solve the generalized eigenvalues problem $L.v = \lambda.D.v$.

Plots of the eigenvectors of lowest eigenvalues are providing a relevant representation of the initial data.

*Enseignants: G. Marcou, *Université de Strasbourg, Faculté de Chimie, 1, rue Blaise Pascal, 67000 Strasbourg*

Question 1.1

Define the following concepts :
- diagonal matrix ;
- eigenvalue and eigenvector ;
- dimension of a vector space.

The method is applied to an artificial dataset of $N = 250$ points. The points are located on two manifolds which are selected randomly following a random variable v. The actual position of a point on its manifold depend on two random variables θ and ϕ. The random variables are sampled to generate 3 series of values for the x, y and z coordinates of each point in an orthonormal basis. The series are stored into 3 N-dimensional vectors.

Question 1.2

Comment the following lines :

```
r1 := .5; -1; N := 250;
theta := Sample(RandomVariable(Uniform(0, 2*Pi)));
phi := Sample(RandomVariable(Uniform(0, Pi)));
v := Sample(1+RandomVariable(Bernoulli(.5)));
ltheta := theta(N); lphi := phi(N); lv := v(N);
x:=Vector([seq(r1*lv[i]*cos(ltheta[i])*sin(lphi[i]),i=1..N)]);
y:=Vector([seq(r1*lv[i]*sin(ltheta[i])*sin(lphi[i]),i=1..N)]);
z:=Vector([seq(r1*lv[i]*cos(lphi[i]),i=1..N)]);
```

To help the analysis, the manifold a point belongs to is encoded by a color, red or blue, stored in the list `clr`.

```
clr := [seq(COLOR(RGB, lv[i]-1, 0., 2-lv[i]), i = 1 .. N)];
```

Question 1.3

Explain the following concepts of Maple :
- a sequence
- a list
- a set

Question 1.4

Complete the following command in order to plot the 3D cloud of points of the dataset. Describe the two manifolds.

```
plots[pointplot3d](,,,color=);
```

A few functions are provided to help realizing the algorithm. For instance the function `dist2` uses the three coordinate vectors x,y and z and the index i and j to compute the squared Euclidean distance between the points i and j. The function `dk` mimics the behavior of the Kronecker delta function.

```
dist2:=
proc(x::Vector,y::Vector,z::Vector,i::integer,j::integer)::float;
return evalf((x[i]-x[j])^2+(y[i]-y[j])^2+(z[i]-z[j])^2)
end proc;
```

```
dk:=(i,j)->eval(evalb(i=j),[true=1,false=0]);

g:=(x,epsilon)->if␣epsilon␣<␣x␣then␣x␣else␣0␣end␣if;
```

Question 1.5

Explain what are doing, in practice, the functions `g` and `dk` ?

The weight matrix W is then computed with the following parameters : the width of the Gaussian is set to (`sigma2`) and two points are connected in a graph if the squared distance between them is smaller than ϵ^2 (`epsilon2`). Example values for these parameters are $\sigma^2 = 0.1$ and $\epsilon^2 = 0.4$. The following commands fill the matrix W :

```
W := Matrix([seq([seq(g(exp(-dist2(x, y, z, i, j)/sigma2)*
  (1-dk(i, j)), exp(-epsilon2/sigma2)), i = 1 .. N)],
  j = 1 .. N)]);
```

Question 1.6

Explain how is working the line computing the matrix W.

The degree matrix D contains on the diagonal the sum of the components of the corresponding line in the matrix W : $D_{ii} = \sum_{j=1}^{N} W_{ij}$. The degree matrix is stored into the variable `De`.

Question 1.7

Complete the following command line that uses the command `sum` of the package `MTM` to compute the degree matrix.

```
De␣:=␣Matrix(MTM[sum](,),shape=)
```

Question 1.8

Compute the Laplacian matrix $L = D - W$.

Question 1.9

Solve the generalized eigenvalue problem $L.v = \lambda.D.v$. Store the eigenvalues in a vector named `evl` and the eigenvectors in a vector named `evc`.

Question 1.10

Use the command `pointplot` of the library `plots`, in order to plot the most relevant pair of eigenvector : those with the lowest eigenvalues, except the one with the zero eigenvalue.

Question 1.11

Give your conclusions.

Barème

- Question 1.1 : 2 points
- Question 1.2 : 2 points
- Question 1.3 : 2 points
- Question 1.4 : 2 points
- Question 1.5 : 2 points
- Question 1.6 : 2 points
- Question 1.7 : 2 points
- Question 1.8 : 2 points
- Question 1.9 : 2 points
- Question 1.10 : 2 points
- Question 1.11 : 2 points