A clustering package for nucleotide sequences using Laplacian Eigenmaps and Gaussian Mixture Model

Marine Bruneau, Thierry Mottet, Serge Moulin, Maël Kerbiriou, Franz Chouly, Stéphane Chretien, Christophe Guyeux

A clustering tool for nucleotide sequences using Laplacian Eigenmaps and Gaussian Mixture Models

Marine BRUNEAU[(1)] Thierry MOTTET[(2)]

Serge MOULIN[(2)] * Maël KERBIRIOU[(1)]

Franz CHOULY[(1)] Stéphane CHRETIEN[(3)]

Christophe GUYEUX[(2)]

(1) Laboratoire de Mathématiques de Besançon, UMR 6623 CNRS, Université de Bourgogne Franche-Comté, 16 route de Gray, 25030 Besançon, France.
(2) Computer Science Department, FEMTO-ST Institute, UMR 6174 CNRS, Université de Bourgogne Franche-Comté, 15 Bis Avenue des Montboucons, 25030 Besançon, France.

(3) National Physical Laboratory, Hampton Road, Teddington, United Kingdom.

Abstract

We propose a new procedure for clustering nucleotide sequences based on the "Laplacian Eigenmaps" and Gaussian Mixture modelling. This proposal is then applied to a set of 100 DNA sequences from the mitochondrially encoded NADH dehydrogenase 3 (ND3) gene of a collection of *Platyhelminthes* and *Nematoda* species. The resulting clusters are then shown to be consistent with the gene phylogenetic tree computed using a maximum likelihood approach. This comparison shows in particular that the clustering produced by the methodology combining Laplacian Eigenmaps with Gaussian Mixture models is coherent with the phylogeny as well as with the NCBI taxonomy. We also developed a Python package for this procedure which is available online.

Keywords: DNA Clustering, Genomics, Laplacian Eigenmaps, Gaussian mixture model.

*Corresponding author. Email `serge.moulin@univ-fcomte.fr`

1 Introduction

As the amount of available genetic sequences increases drastically, the need for scalable methods becomes urgent. A very important tool for dealing with large amounts of data is clustering. Clustering was successfully used for a large number of important applications recently. Clustering methods have thus demonstrated irrefutable efficiency for real-life applications to genetics and have provided extremely powerful tools to the community. In particular, the huge amount of molecular data available nowadays can help addressing new and essential questions in genomics if appropriate scalable methods are used. Sequence clustering is a key element among these tools (Valot *et al.*, 2015). Among the various benefits of sequence clustering are

- interpretability as for the classification of 16S ARN into OTU (Hao *et al.*, 2011),
- potential reduction of very large databases, *i.e.*, too large to be processed in a relatively short time (Rousk *et al.*, 2010), or
- creation of non-redundant protein groups (Suzek *et al.*, 2007)

for instance. Moreover, clustering is also crucial for the discovery of new hidden variables underpinning the phenomena that are analysed.

However, as noted by a number of researchers from the community of Machine Learning (ML), clustering may not be very reliable without an appropriate preliminary embedding of the data samples. The motto often quoted in ML is that high dimensional data are often too scattered for an off-the-shelf method to work properly. Several embeddings have been proposed in the literature. Some of them are better suited for supervised learning, such as the methods based on neural networks (*e.g.*, auto-encoders) and some other methods are more suitable for unsupervised learning as is the case in the present work. Among many non-linear embedding methods, the Laplacian Eigenmap (Belkin and Niyogi, 2001) approach has been extensively studied from both the theoretical side and the application one (Spielman, 2009). Such methods have received a lot of attention and can be also constrained towards better postprocessing with a view towards improved clustering as shown in, *e.g.*, (Chrétien *et al.*, 2016).

In the case study, our goal is to show the practical efficiency of a combination of the plain Laplacian Eigenmap approach with Gaussian Mixture based clustering. Our choice of this combination is mainly motivated by scalability constraints

and the fact that both methods are state of the art in the Machine Learning community. The procedure is extremely natural and can be described as follows

1. Compute a similarity matrix between each couple of sequences. That is to say, a matrix W of size $n \times n$ (where n is the number of studied sequences), such that $W_{i,j}$ increases with the similarity of sequences i and j.

2. Apply the Laplacian Eigenmap (Belkin and Niyogi, 2001) method to the matrix W. This method transforms the sequences in elements leaving in a given vectorial space, and whose positions in the space reflects well and illustrates in an richer way the sequence similarities.

3. Apply a clustering method to the points of the yielding space using a Gaussian mixture model (Day, 1969).

Our main contribution is twofold: we propose a ready to use Python package available online and we demonstrate the efficiency of the approach for the problem of clustering nucleotide sequences.

In particular, we tested the proposed methodology on a sample of one hundred ND3 genes (DNA sequences) from *Platyhelminthes* and *Nematoda* species that have been downloaded on the NCBI website[1]. The classification produced by the proposed method has been compared with the phylogenetic tree of these species obtained by a likelihood maximization method using PhyML (Guindon *et al.*, 2005). According to this comparison, the clustering is consistent with both clades appearing in the phylogenetic tree and the NCBI taxonomy. More specifically, our clustering method perfectly separates the *Nematoda* and *Platyhelminthes* phyla. Among the *Nematoda*, we obtained a cluster for *Trichocephalida* order, another one for most of the species of the *Spirurida* order, and a last cluster for the remainder.

The plan of the paper is the following. The three stages of the proposed method are detailed in the next section, while an application example is provided in Section 3. The whole proposal is discussed in Section 4. Finally, this article ends with a conclusion section, in which the contribution is summarized and intended future work is outlined.

[1] http://www.ncbi.nlm.nih.gov/

2 The Clustering Method

2.1 Design of the similarity matrix

The first step in the construction of a good embedding is the creation of a similarity matrix W. This matrix measures the similarity between every two sequences by providing a number between 0 and 1, the greater this number is, the most similar the sequences are.

We introduce here a way to build similarity matrices using the Needleman Wunsch "distance" (Needleman and Wunsch, 1970). In order to do so, a multiple global alignment of the DNA sequences is run, for instance using MUSCLE (Multiple Sequence Comparison by Log-Expectation (Edgar, 2004)) software. Based on the results of this alignment, we can then define a first matrix M satisfying the following requirements: for all $i, j \in [\![1, n]\!]$, $M_{i,j}$ is the "Needleman-Wunsch distance" between sequence i and sequence j. More precisely, M can be computed, by evaluating the Needleman-Wunsch edit distance between each couple of aligned sequences, which is the reference distance when considering nucleotidic sequences. In practice, this can be done using the `needle` command from EMBOSS package (Rice *et al.*, 2000).

The matrix M is then divided by the largest distance value, so that all its coefficients are between 0 and 1. W can finally be obtained as follows:

$$\forall\, i, j \in [\![1, n]\!],\ W_{i,j} = 1 - M_{i,j},$$

in such a way that $W_{i,j}$ represents the similarity between species i and j.

2.2 Laplacian Eigenmaps on W

The so-called Laplacian Eigenmaps (Belkin and Niyogi, 2001) is an original method of representation, in a k-dimension vector space, of a given matrix of similarity between complex objects (here, words on the $\{$A,C,G,T$\}$ alphabet). In this space, points with real coordinates are close when their similarity is large. Indeed, increasing the dimension leads to a richer view of the similarities between different objects, onto which standard clustering techniques can be applied.

The first step is to create the normalized Laplacian matrix (Chen and Zhang, 2007):

$$L = D^{-1/2}(D - W)D^{-1/2},$$

where W is the similarity matrix defined previously and D is the degree matrix of W. That is to say, D is the diagonal matrix defined by:

$$\forall i \in [\![1,n]\!], D_{i,i} = \sum_{j=1}^{n} W_{i,j} \ .$$

L being symmetric and real, it is diagonalisable in a basis of pairwise orthogonal eigenvectors $\{\phi_1, ..., \phi_n\}$ associated with eigenvalues $0 = \lambda_1 \leqslant \lambda_2 \leqslant ... \leqslant \lambda_n$.

The Laplacian Eigenmaps consists in considering the following embedding function:

$$c_{k_1}(i) = \begin{pmatrix} \phi_2(i) \\ \phi_3(i) \\ \vdots \\ \phi_{k_1+1}(i) \end{pmatrix} \in \mathbb{R}^{k_1},$$

where $c_{k_1}(i)$ is the coordinate vector of the point corresponding to the i^{th} sequence. In other words, the coordinate vector of the point corresponding to the i^{th} sequence is constituted by the i^{th} coordinate of each of the k_1 first eigenvectors, ordered according to the size of their eigenvalues. As stated previously, this cloud of points is a model of the DNA sequences, such that the proximity between two points increases (considering the Euclidean distance) with the sequence similarity.

The choice of the k_1 cutoff is a crucial issue and it is usually made as follows. The ordered eigenvalues are plotted, and we stop when the increase becomes negligible: the number of eigenvalues that are not discarded is k_1. For instance, in our program, we have chosen to set k_1 as the first time the difference between the k^{th} and $(k+1)^{th}$ value is lower than 0.01.

Remark that W can be seen as a weighted adjacency matrix of a graph, where nodes are the DNA sequences while edges are labeled by the degree of affinity between their adjacent nodes. In the literature, the Laplacian matrix is often described as constructed from the weighted adjacency matrix of such a graph rather than constructed from a similarity matrix. These definitions are equivalent.

2.3 Gaussian Mixture based clustering

The final step is performed by applying Gaussian Mixture based clustering (GMM, (Day, 1969)) to point cloud. Gaussian Mixture Models belong to the class of unsupervised learning schemes (Friedman *et al.*, 2001), and allows to distribute the data points into different clusters without *a priori* assumption about the clusters' interpretation. One of the very useful features of model based clustering is that the model allows to use information criteria in order to estimate the number of clusters using AIC (Akaike, 1974), BIC (Schwarz *et al.*, 1978), or ICL (Biernacki

et al., 2000). The mathematical assumption of a GMM is that the point cloud follows the distribution:

$$\sum_{i=1}^{k_2} \delta_i \, \mathcal{N}(\mu_i, \Sigma_i),$$

where k_2 is the number of clusters, δ_i is the probability for a point to be in cluster i, and $\mathcal{N}(\mu_i, \Sigma_i)$ is the normal distribution of mean μ_i and covariance matrix Σ_i. GMM parameters are computed with the Expectation-Maximization (EM) algorithm (McLachlan and Peel, 2004). Notice that the EM algorithm may converge to singular distributions exponentially fast (Biernacki and Chrétien, 2003). However, degenerate situations can be easily discarded and consistent estimators can be easly obtained in practice. Gaussian Mixture models are still a topic of current extensive research, both from the statistical perspective (Wang *et al.*, 2014) and the computational perspective (Yi and Caramanis, 2015).

We have chosen to consider the Bayesian Information Criterion (BIC, (Schwarz *et al.*, 1978)) to determine the number of clusters k_2. The BIC, which is a criterion for model selection, is defined as follows:

$$BIC = -2ln(L) + ln(n)p,$$

where L is the likelihood of the estimated model, n is the number of observations in the sample, and p is the number of model parameters. This criterion allows us to select a model whose validity is based on a compromise between the value of the model likelihood (that we want to maximize) and the number of parameters to estimate (that we want to minimize). The likelihood of the model increases with k_2 as well as the number of parameters. The selected model will be by default the one that minimizes this criterion (but, in our software, the user can set manually, if needed, the desired number of clusters).

2.4 The clustering software

The Python program corresponding to the algorithm described in this section is freely available online at https://github.com/SergeMOULIN/clustering-tool-for-nucleotide-sequences-using-Laplacian-Eigenmaps-and-Gaussian-Mixture-Models. Its main function provides a clustering either from a matrix of distances or a similarity matrix. Its prototype meets the following canvas:

```
clustering = GClust(M, refs, simil = False,
       nbClusters = 'BIC', drawgraphs = True,
       delta = 0.01)
```

where:

- `M` is the matrix of distances between the genomic sequences. `M` can also be a matrix of similarities between the genomic sequences, by setting

  `simil = TRUE` (see below).

- `refs`, is the list of species references. Note that `M` and `refs` are the only mandatory arguments.

- `simil` is an optional Boolean argument. When `simil = TRUE`, then `M` is considered as a matrix of similarities between the DNA sequences. Otherwise it is a matrix of distances (`simil = FALSE` by default).

- `nbClusters` is the number of clusters desired by the user. By default, the program applies the BIC criterium to determine it.

- `drawgraphs` is an optional Boolean value to produce some graphics. If `drawgraphs = TRUE`, a two dimensional clustering of data is plotted (*cf.* Figures 4, 5, and 6), as well as the graphical representation of similarities (as in Fig. 1).

- `delta` is the value used to select k_1 (*i.e.*, the number of eigenvectors used in the Laplacian Eigenmaps). k_1 is the lowest value such that $\lambda_{k_1+1} - \lambda_{k_1} <$ `delta`. `delta` = 0.01 by default.

The output of `Glust` function, named "clustering" above, is a list of k_2 lists. Each list contains references to the species grouped in a particular cluster.

3 Illustration example on genomic data

The proposed method has been applied to a sample of 100 DNA sequences from the mitochondrial gene ND3 taken from various species of both *Platyhelminthes* and *Nematoda*. Figure 1 graphically displays the similarity matrix that has been obtained as described previously. As might be expected, the largest similarities are obtained on the diagonal, that is, when the sequences are compared to themselves. However, blocks seem to appear too in this representation, justifying the need to further investigate such similarities by applying our clustering method on these sequences.

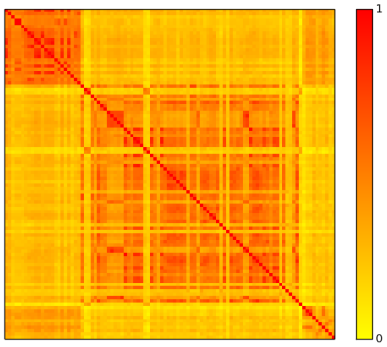

Figure 1: Similarity matrix

Figure 2 displays the 14 first ascending eigenvalues, labeled λ_i, and satisfying $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_{14}$. According to the proposed methodology, we have chosen k_1 as the minimal index k such that $\lambda_{k+1} - \lambda_k$ is lower than or equal to 0.01. In this case study, we found $k_1 = 4$.

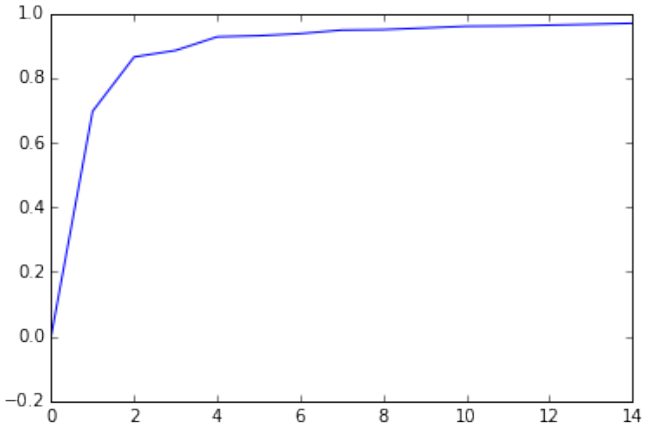

Figure 2: Curve representing the first 14 eigenvalues

Figure 3 shows the Bayesian Information Criterion of the Gaussian Mixture Models for various number of clusters. One can see that this BIC reaches its minimum for $k_2 = 4$. Thus, the program automatically achieves the sequence partition in four clusters.

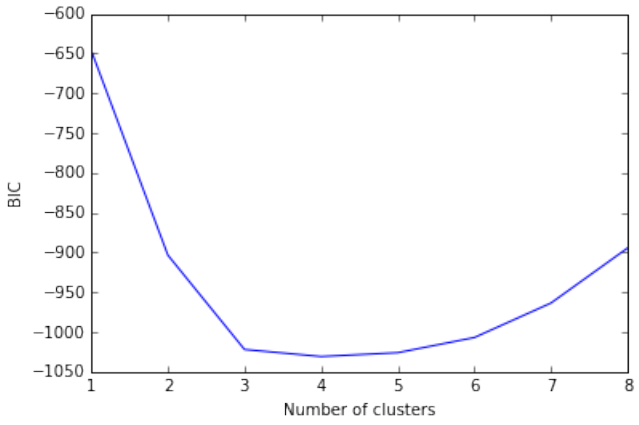

Figure 3: Bayesian Information Criterion of the Gaussian Mixture Models

Figures 4, 5, and 6 represent the point cloud divided in 4 clusters. This cloud point is projected into the plane formed by the first and second eigenvectors in Figure 4, to the one formed by the first and third eigenvectors in Figure 5, and to the plane formed by the second and the third eigenvectors in Figure 6. In these graphs, clusters 0, 1, 2, and 3 are represented in red, cyan, blue, and yellow respectively.

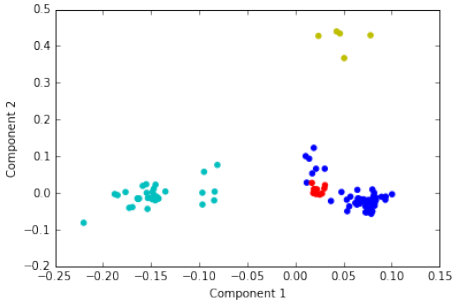

Figure 4: GMM clustering in the plane formed by the eigenvectors 1 and 2

A FASTA file has then be written, in which each nucleotide sequence has been labeled according to its taxonomy and its cluster number. The taxonomies have been found thanks to the `efetch` function (sub-package `Entrez`[2], Python pack-

[2] A package that provides code to access NCBI over the world wide web.

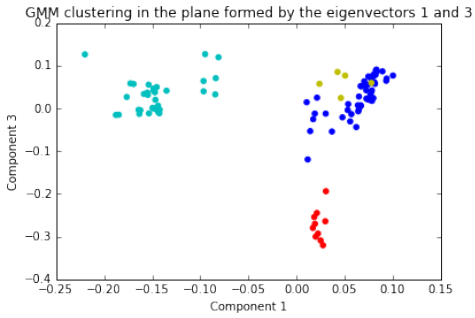

Figure 5: GMM clustering in the plane formed by the eigenvectors 1 and 3

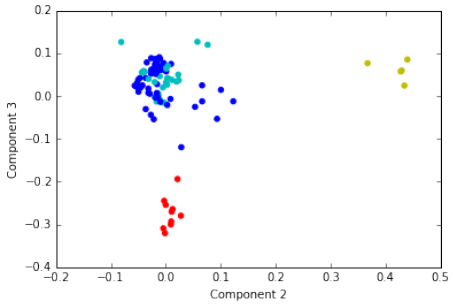

Figure 6: GMM clustering in the plane formed by the eigenvectors 2 and 3

age `Bio`). We then have used the online software *PhyML* (maximum likelihood method method for phylogenetic tree reconstruction) with default options (Guindon *et al.*, 2005), to build a tree based on the same mitochondrial ND3 gene that we have used during clustering. The tree, whose leaves contain taxa names and cluster id, has been displayed using *FigTree* software (Morariu *et al.*, 2008). This tree is depicted in Figures 7 and 8.

The `efetch` function provides 8 taxonomic levels for each species in our sample. In Figures 7 and 8, for readability reasons, we only show the taxonomy between levels 4 and 6. Note first that this clustering perfectly separates *Platyhelminthes* and *Nematoda* phyla. Indeed *Trematoda* and *Cestoda* are two classes of *Platyhelminthes*. Cluster 1 corresponds exactly to the *Platyhelminthes*, while Clusters 0, 2, and 3 represent the *Nematoda*. Cluster 0 is only composed of *Spirurida*, it contains 10 of the 12 members of this taxon. More precisely, when considering the seventh taxonomic level, we found that Cluster 0 contains nine *Filarioidea* and one *Thelazioidea*, while the *Spirurida* that are not in this cluster

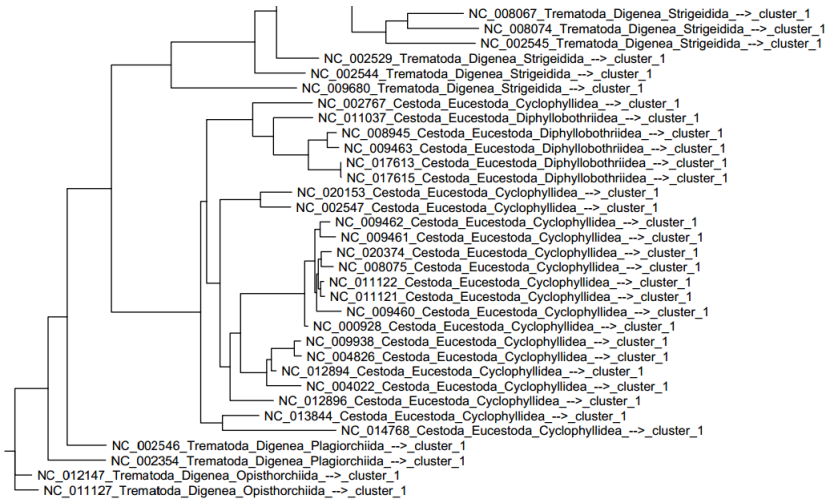

Figure 7: First part of the phylogenetic tree (*Platyhelminthes*)

are a *Dracunculoidea* and a *Physalopteroidea*. Cluster 3, for its part, corresponds exactly with the *Trichocephalida* taxon.

Finally, in addition to recover taxonomy, the clustering agrees very well with the tree obtained via PhyML, as shown in Figures 7 and 8. Note that the taxonomy, based on morphology and nuclear genome, fully agrees with the PhyML tree and our clustering, which are both based on the mitochondrial genome. This fact suggests that this mitochondrial gene evolves like the nuclear genome.

4 Discussion

Various options are possible to perform the analysis we have presented previously, some of them being listed hereafter.

4.1 Similarity matrix

The multiple global alignment step can be achieved using other software than MUSCLE. Among the most extensively used methods, we could have chosen MAFFT (Katoh and Standley, 2013) for instance, as well as ClustalW or ClustalX (Larkin *et al.*, 2007). It is also possible to apply a pairwise alignment method like the Needleman-Wunsch algorithm (Needleman and Wunsch, 1970). Moreover, we have chosen EDNAFULL scoring matrix, but other matrices are available to

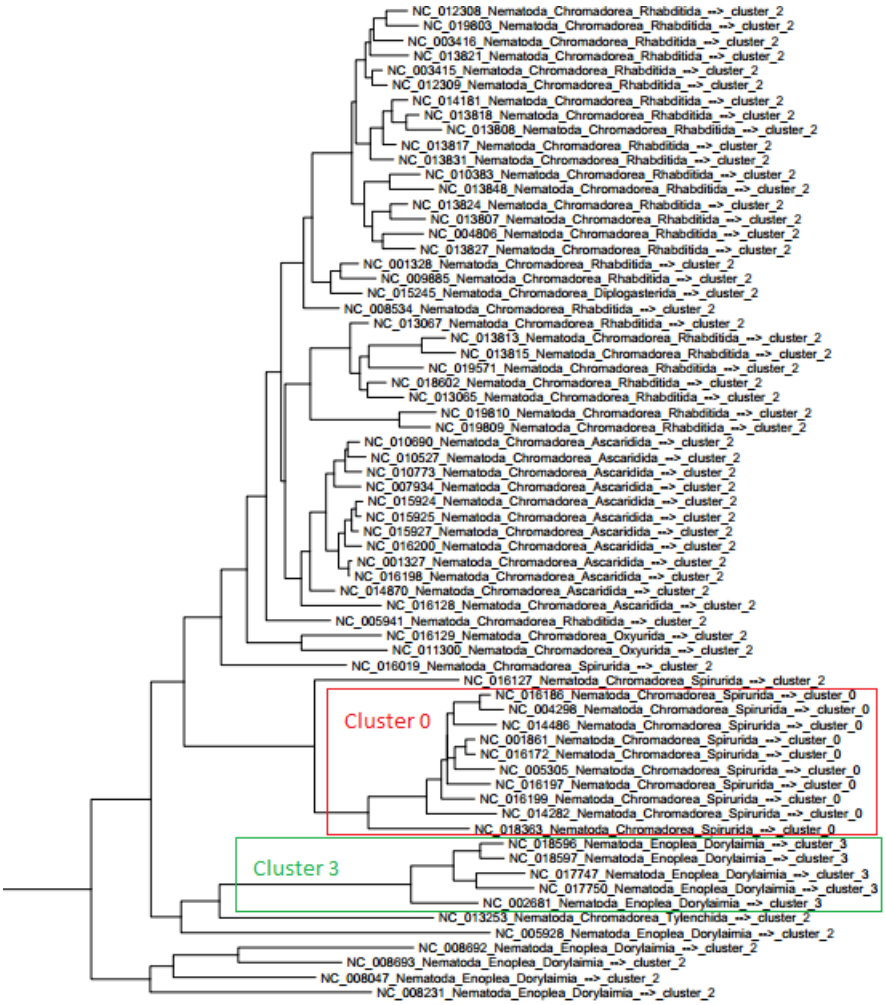

Figure 8: Second part of the phylogenetic tree (*Nematoda*)

produce a score between two aligned sequences, like PAM or BLOSUM. Finally, instead of defining the similarity matrix W as $W_{i,j} = 1 - M_{i,j}$, it could be possible to consider $W_{i,j} = \frac{1}{M_{i,j}}$ or $W_{i,j} = e^{-M_{i,j}}$.

To find the best option among these possible choices, one can either test them on reference sequences on which the clusters to obtain are perfectly known, for instance by producing a laboratory-generated phylogeny. Another option can be to use a well-established phylogeny based on fossils morphology and molecular data. Finally, a third option can be to compute simulations of a set of species

having a common ancestor. These three approaches have already been implemented to compare phylogenetic tree reconstruction tools. The first approach, for instance, has been applied in (Hillis *et al.*, 1992) on the bacteriophage T7 evolved in a laboratory. The second approach, for its part, has been applied in (Cummings *et al.*, 1995), (Russo *et al.*, 1996), and (Zardoya and Meyer, 1996), who used mammalian or vertebrate phylogeny to perform these comparisons. The last one was considered a lot of time, and summarised in (Felsenstein, 1988), (Huelsenbeck, 1995), and (Nei, 1996).

4.2 Number of considered eigenvectors

The number of eigenvectors to keep is another point to investigate. As explained in Part 2.2, it is usually advised to check graphically when the increase of eigenvalues is reducing. In this article, we have chosen to consider k_1 such that $\delta = \lambda_{k_1+1} - \lambda_{k_1} < 0.01$. This criterion has led to $k = 4$ in the case study, which seems acceptable according to the considered taxonomy.

Some authors in the literature proposed to compute k_1 as the logarithm of n (Matias, 2015). It is also possible to consider a criterion related to the second derivative instead of the first one, by replacing the computation of δ by:

$$\delta = (\lambda_{k_1} - \lambda_{k_1-1}) - (\lambda_{k_1+1} - \lambda_{k_1}).$$

The latter may be a good representative of the notion of inflection. Other approach can be considered to solve this problem, which is still an open one.

4.3 Number of clusters

We have chosen to consider the BIC (Schwarz *et al.*, 1978) to determine the optimal number of clusters, which is a common choice for this type of problem. An alternative may be to use the Akaike Information Criterion (AIC, (Akaike, 1974)). The principle of calculating the AIC is the same as the BIC, since the goal is to maximize log-likelihood penalized by the number of parameters (or more precisely, to minimize the number of parameters to which the log-likelihood is subtracted). AIC formula is the following:

$$AIC = -2ln(L) + 2 \times p,$$

where L is the likelihood of the estimated model and p the number of model parameters.

Indeed, BIC is "more conservative" than AIC. That is to say, the number of clusters obtained by BIC is lower or equal to the one obtained by AIC. In other words, BIC penalization is more important, and the choice between these two criteria can be dependent on how stringent the clustering is desired.

4.4 Conclusion

In this work, we have proposed a new method of nucleotide sequence clustering. This clustering is produced by a methodology combining Laplacian Eigenmaps with Gaussian Mixture models, while the number of clusters is set according to the Bayesian Information Criterion. This methodology has been applied on 100 sequences of mitochondrially encoded NADH dehydrogenase 3. Obtained clusters are coherent with the phylogeny (gene tree obtained with PhyML) as well as with the NCBI taxonomy.

One possible extension for future work can be to investigate more deeply the impact of parameters in the obtained clusters. The effects of a modification in the similarity distance, in the manner to set the dimension of the space in Laplacian Eigenmaps, or in the number of desired clusters, will be systematically investigated. This clustering method will be applied in concrete problems related to genomics, for instance on coding sequences predicted on a set of bacteria, to determine the size of their core (AlKindy *et al.*, 2014) and pan genome.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE transactions on automatic control*, **19**(6), 716–723.

AlKindy, B., Couchot, J.-F., Guyeux, C., Mouly, A., Salomon, M., and Bahi, J. M. (2014). Finding the core-genes of chloroplasts. *Journal of Bioscience, Biochemistery, and Bioinformatics*, **4**(5), 357–364.

Belkin, M. and Niyogi, P. (2001). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591.

Biernacki, C. and Chrétien, S. (2003). Degeneracy in the maximum likelihood estimation of univariate gaussian mixtures with EM. *Statistics & probability letters*, **61**(4), 373–382.

Biernacki, C., Celeux, G., and Govaert, G. (2000). Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE transactions on pattern analysis and machine intelligence*, **22**(7), 719–725.

Chen, H. and Zhang, F. (2007). Resistance distance and the normalized laplacian spectrum. *Discrete Applied Mathematics*, **155**(5), 654–661.

Chrétien, S., Dombry, C., and Faivre, A. (2016). A semi-definite programming approach to low dimensional embedding for unsupervised clustering. *arXiv preprint arXiv:1606.09190*, *.

Cummings, M. P., Otto, S. P., and Wakeley, J. (1995). Sampling properties of DNA sequence data in phylogenetic analysis. *Molecular Biology and Evolution*, **12**(5), 814–822.

Day, N. E. (1969). Estimating the components of a mixture of normal distributions. *Biometrika*, **56**(3), 463–474.

Edgar, R. C. (2004). MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic acids research*, **32**(5), 1792–1797.

Felsenstein, J. (1988). Phylogenies from molecular sequences: inference and reliability. *Annual review of genetics*, **22**(1), 521–565.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin.

Guindon, S., Lethiec, F., Duroux, P., and Gascuel, O. (2005). PHYML online-a web server for fast maximum likelihood-based phylogenetic inference. *Nucleic acids research*, **33**(suppl 2), W557–W559.

Hao, X., Jiang, R., and Chen, T. (2011). Clustering 16s rRNA for OTU prediction: a method of unsupervised bayesian clustering. *Bioinformatics*, **27**(5), 611–618.

Hillis, D. M., Bull, J. J., *et al.* (1992). Experimental phylogenetics: Generation of a know phylogeny. *Science*, **255**(5044), 589.

Huelsenbeck, J. P. (1995). Performance of phylogenetic methods in simulation. *Systematic biology*, **44**(1), 17–48.

Katoh, K. and Standley, D. M. (2013). MAFFT multiple sequence alignment software version 7: improvements in performance and usability. *Molecular biology and evolution*, **30**(4), 772–780.

Larkin, M. A., Blackshields, G., Brown, N., Chenna, R., McGettigan, P. A., McWilliam, H., Valentin, F., Wallace, I. M., Wilm, A., Lopez, R., *et al.* (2007). Clustal w and clustal x version 2.0. *bioinformatics*, **23**(21), 2947–2948.

Matias, C. (2015). Notes de cours: Analyse statistique de graphes.

McLachlan, G. and Peel, D. (2004). *Finite mixture models*. John Wiley & Sons.

Morariu, V. I., Srinivasan, B. V., Raykar, V. C., Duraiswami, R., and Davis, L. S. (2008). Automatic online tuning for fast gaussian summation. In *Advances in Neural Information Processing Systems (NIPS)*.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, **48**(3), 443–453.

Nei, M. (1996). Phylogenetic analysis in molecular evolutionary genetics. *Annual review of genetics*, **30**(1), 371–403.

Rice, P., Longden, I., and Bleasby, A. (2000). EMBOSS: The european molecular biology open software suite. *Trends in Genetics*, **16**(6), 276–277.

Rousk, J., Bååth, E., Brookes, P. C., Lauber, C. L., Lozupone, C., Caporaso, J. G., Knight, R., and Fierer, N. (2010). Soil bacterial and fungal communities across a ph gradient in an arable soil. *The ISME journal*, **4**(10), 1340–1351.

Russo, C., Takezaki, N., and Nei, M. (1996). Efficiencies of different genes and different tree-building methods in recovering a known vertebrate phylogeny. *Molecular Biology and Evolution*, **13**(3), 525–536.

Schwarz, G. *et al.* (1978). Estimating the dimension of a model. *The annals of statistics*, **6**(2), 461–464.

Spielman, D. (2009). Spectral graph theory. *Lecture Notes, Yale University*, pages 740–0776.

Suzek, B. E., Huang, H., McGarvey, P., Mazumder, R., and Wu, C. H. (2007). Uniref: comprehensive and non-redundant uniprot reference clusters. *Bioinformatics*, **23**(10), 1282–1288.

Valot, B., Guyeux, C., Rolland, J. Y., Mazouzi, K., Bertrand, X., and Hocquet, D. (2015). What it takes to be a pseudomonas aeruginosa? the core genome of the opportunistic pathogen updated. **10**(5), e0126468.

Wang, Z., Gu, Q., Ning, Y., and Liu, H. (2014). High dimensional expectation-maximization algorithm: Statistical optimization and asymptotic normality. *arXiv preprint arXiv:1412.8729*.

Yi, X. and Caramanis, C. (2015). Regularized EM algorithms: A unified framework and statistical guarantees. In *Advances in Neural Information Processing Systems*, pages 1567–1575.

Zardoya, R. and Meyer, A. (1996). Phylogenetic performance of mitochondrial protein-coding genes in resolving relationships among vertebrates. *Molecular biology and evolution*, **13**(7), 933–942.