Visualisation d'informations
Analyse visuelle de données d'expression génique

Benetti Julien, Maunier Tristan
Tuteur : Bourqui Romain
Master 2 Bioinformatique
Année universitaire 2017/2018

25 janvier 2018

Introduction

L'analyse des données est une famille de méthodes statistiques dont les principales caractéristiques sont d'être multidimensionnelles et descriptives. Certaines méthodes, pour la plupart géométriques, aident à faire ressortir les relations pouvant exister entre les différentes données et à en tirer une information statistique qui permet de décrire de façon plus succincte les principales informations contenues dans ces données. D'autres techniques permettent de regrouper les données de façon à faire apparaître clairement ce qui les rend homogènes, et ainsi mieux les connaître.

L'analyse des données permet de traiter un nombre très important de données et de dégager les aspects les plus intéressants de la structure de celles-ci. Le succès de cette discipline dans les dernières années est dû, dans une large mesure, aux représentations graphiques fournies. Ces graphiques peuvent mettre en évidence des relations difficilement saisies par l'analyse directe des données ; mais surtout, ces représentations ne sont pas liées à une opinion "a priori" sur les lois des phénomènes analysés contrairement aux méthodes de la statistique classique.

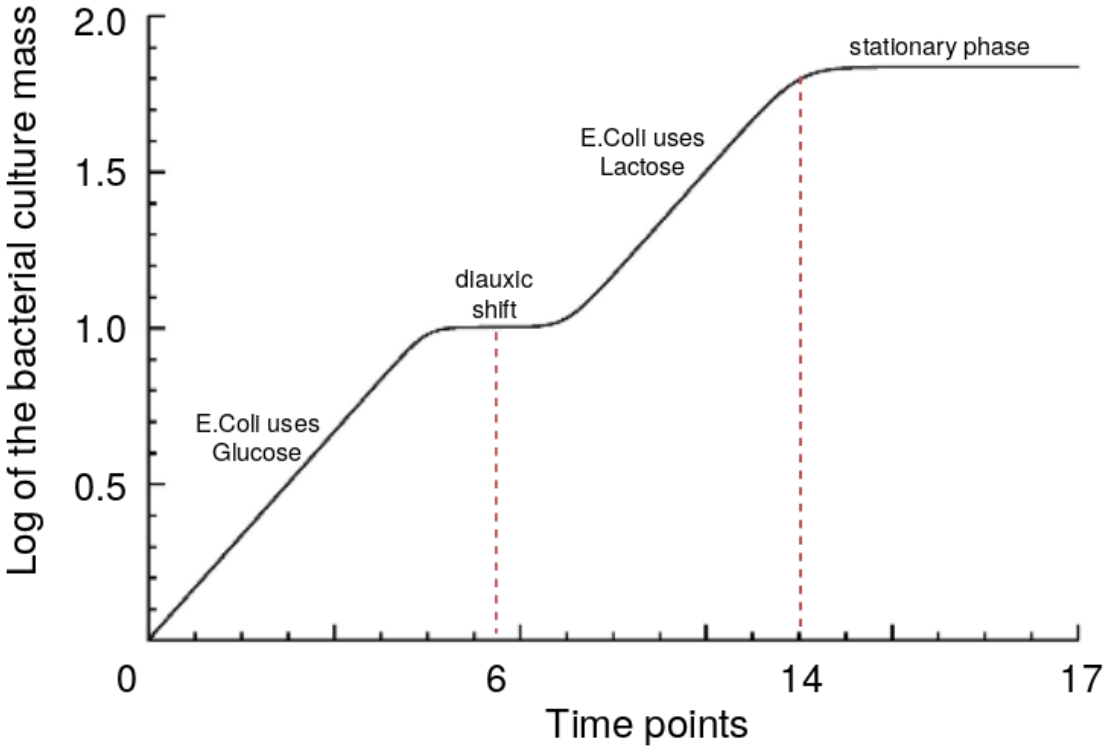
Les fondements mathématiques de l'analyse des données ont commencé à se développer au début du xxe siècle, mais ce sont les ordinateurs qui ont rendu cette discipline opérationnelle, et qui en ont permis une utilisation très étendue. Mathématiques et informatique sont ici intimement liées.

L'objectif de ce projet est d'implémenter un script permettant la visualisation d'informations à partir d'un jeu de données biologiques. Pour ce faire nous avons utilisé Tulip [1], un logiciel dédié à l'analyse et à la visualisation de données relatives.

Jeu de données

Le jeu de données utilisé au cours de ce projet contient des valeurs d'expression génique prélevées à 17 temps expérimentaux au cours d'une croissance d'*E.Coli*. La figure 1 ci-dessous représente cette croissance bactérienne dans un milieu composé de glucose et de lactose, au cours du temps.

FIGURE 1 – Croissance d'*E.Coli* en présence de Glucose et de Lactose



Lorsqu'*E.Coli* est cultivée dans un milieu contenant deux sucres comme le glucose et le lactose, la bactérie consomme en priorité le glucose jusqu'à l'épuisement de ce dernier dans le milieu. Cet épuisement énergétique se traduit par un arrêt de croissance au cours duquel la bactérie va changer de source énergétique pour se concentrer sur la dégradation du lactose, cette étape est appelée diauxie (diauxic shift, t6).
Une fois les ressources énergétiques du milieu épuisées, la croissance bactérienne s'arrête et l'organisme entre dans une phase dite "stationnaire" (t14). Au cours de cette phase, de nombreux processus biologiques ne sont plus opérationnels afin de permettre à l'organisme d'économiser de l'énergie.

Pré-traitement & première visualisation

Les données brutes ne permettent pas une visualisation claire. Il faut donc effectuer une opération préliminaire avant le traitement à proprement dit, afin de rendre les informations plus digestes.

Les gènes sont représentés par des sommets et leurs interactions par des arêtes. Les sommets sont labellisés avec leur locus afin de les identifier rapidement. Leur taille a aussi été changée selon leur degré, c'est-à-dire le nombre d'arêtes reliant ce sommet. Certains sommets étant reliées par des centaines d'arêtes, afin d'homogénéiser en partie la taille des nœuds, nous avons choisis d'utiliser leur logarithme. Les gènes avec le plus d'interactions seront donc représentés avec une taille plus conséquente.
Les gènes ne présentant aucune valeur d'expression ont été filtrés, ils ont été considérés en tant qu'erreur

de manipulation.

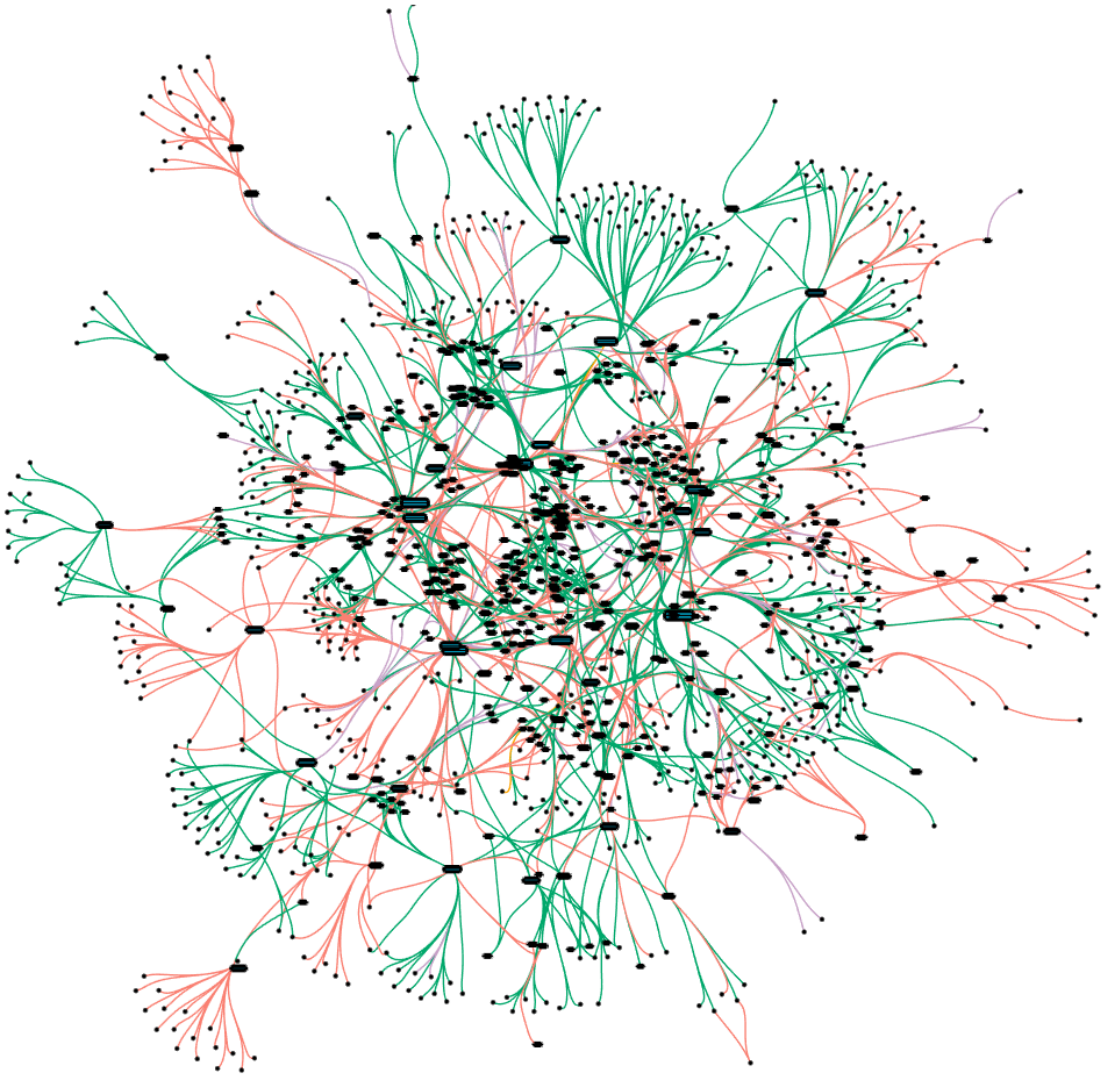
La représentation des arêtes peut aussi être prétraitée. En effet, une arête représente une interaction entre deux sommets, soit deux gènes. Cette interaction peut être positive ou négative. La couleur des arêtes a donc été modifiée en vert (Jade) lorsque l'interaction est positive et en rouge (Saumon) lorsqu'elle est négative.

Cette information est à la base stockée dans deux colonnes dans les données brutes : Negative et Positive. Une arête représentant une interaction positive aura donc "true" dans la colonne Positive et "false" dans la colonne Negative et vice-versa. Cependant certaines arêtes possède "true" dans les deux colonnes, et d'autres "false" dans les deux colonnes. Ces arêtes ont été considérées comme des faux-positifs et des faux-négatifs.

Enfin, un algorithme de dessin a été sélectionné. En effet, les arêtes ne possèdent aucun poids et ne sont donc pas représentées par une distance sur un graphe extrait uniquement des données brutes. Le graphe sans algorithme de dessin consiste donc à un empilement de sommets. Afin d'obtenir une première visualisation du graphe, nous avons choisi l'algorithme "Fast Multipole Multilevel Method" (FM^3)[2]. L'algorithme multi-niveaux FM3 a été introduit par Hachul et Jünger. Il est basé sur une combinaison d'une technique multi-niveaux efficace avec un algorithme d'approximation pour obtenir les forces répulsives entre toutes les paires de nœuds. Cet algorithme est rapide et efficace, et ses besoins en mémoire sont linéaires.

Notre graphe prétraité possède ainsi 1140 nœuds et 2373 arêtes. Afin d'améliorer la visualisation, une dernière étape de prétraitement a été effectuée : nous avons appliqué une algorithme de groupement d'arêtes (Edge Bundling) [3]. Cette méthode consiste à faire passer les arêtes sur une grille du graphe en utilisant l'algorithme de Dijkstra puis à les regrouper grâce une pondération de l'arête selon son chemin parcouru. Le lissage des arêtes ainsi obtenu s'effectue grâce aux courbes de Bézier qui sont des courbes polynomiales paramétriques.

FIGURE 2 – Graphe résultant du prétraitement



PARTITIONNEMENT DES GÈNES

Dans le but de pouvoir manipuler et analyser nos données pré-traitées nous avons construit un graphe complet ou autrement dit un graphe où chaque sommet est relié à tous les autres sommets par une arête. Les sommets du graphe complet sont les mêmes sommets que dans le graphe initial, cependant les arêtes sont pondérées par une distance entre les gènes calculée selon leur niveau d'expression.

Nous avons fait le choix de calculer cette distance grâce au coefficient de corrélation de Pearson défini par :

$$Cor(X,Y) = \frac{Cov(X,Y)}{\sigma_X \times \sigma_Y}$$

où Cov(X,Y) désigne la covariance des variables X et Y, σ_X et σ_Y leurs écarts-types. L'intérêt d'utiliser cette méthode est de pouvoir savoir s'il existe ou non une corrélation entre l'expression de deux gènes, ainsi si on l'applique à notre jeu de données on obtient des informations de co-expression génique pour

chaque arête reliant deux sommets.

Afin de calculer la corrélation de Pearson sur les nœuds de notre graphe, nous avons fais le choix d'utiliser la bibliothèque *scipy* de Python et son module *stats* utilisés de la manière suivante :

```
from scipy.stats.stats import pearsonr
scipy.stats.pearsonr(X,Y)
```

La fonction ci-dessus renvoie un tableau T dont la première valeur (T[0]) est le coefficient de corrélation de Pearson calculé et la deuxième valeur (T[1]) est la p-value du test statistique. La valeur de Pearson pouvant être négative, ainsi synonyme d'inhibition d'expression d'un gène envers l'autre, nous avons décidé d'utiliser la valeur absolue de ce résultat afin de catégoriser une corrélation négative avec les corrélations positives en tant que simple corrélation. Cela nous permet de distinguer une présence d'une absence de corrélation.
Graphiquement nous voulions que plus la corrélation entre deux gènes est élevée moins la distance qui sépare les deux sommets correspondants est grande, ainsi nous avons calculé l'inverse de la valeur obtenue afin d'obtenir cette distance.

Au vu du nombre important de nœuds et donc d'arêtes créées lors de la construction du graphe complet, nous avons dû filtrer les arêtes au moment de leur calcul afin de réduire leur nombre. Nous avons utilisé la p-value (T[1]) obtenue à partir du calcul du coefficient de Pearson pour sélectionner les arêtes que nous voulions ajouter. Ainsi toutes les arêtes dont la p-value était supérieure à 0.01 ont été supprimées du graphe complet.

Une fois le graphe complet généré, il s'agissait de regrouper les gènes entre eux selon leur expression génique. Pour ce faire nous avons utilisé l'algorithme "MCL" (Markov Cluster algorithm) [4], implémenté dans le logiciel Tulip, qui crée des clusters (groupes de sommets homogènes) à partir d'arêtes pondérées. Grâce aux valeurs obtenues via le calcul du coefficient de Pearson, on obtient donc une clusterisation des sommets sur l'ensemble de notre graphe.

Dans Tulip nous avons associé l'algorithme de clusterisation "MCL" à l'algorithme "Equal Value" qui nous permet de visualiser graphiquement chaque cluster obtenu.

Construction d'une carte de chaleur

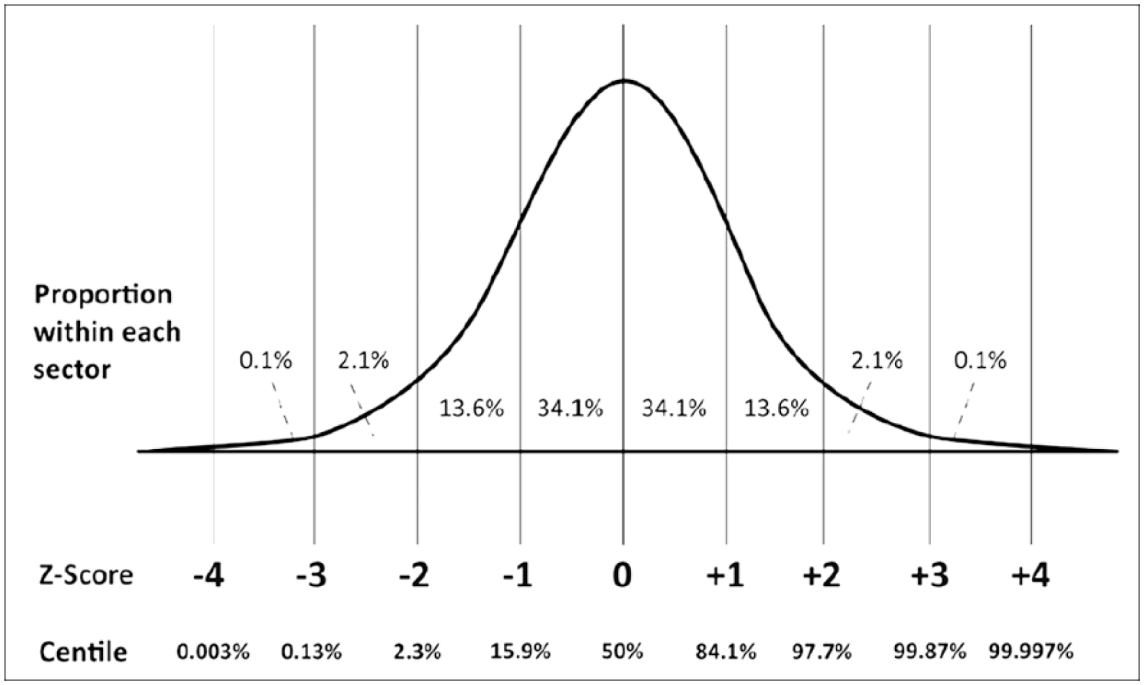
Une carte de chaleur (heatmap) est une représentation graphique de données où les valeurs individuelles contenues dans une matrice sont représentées comme des couleurs. Chaque sommet de cette carte correspond donc à un niveau d'expression d'un gène du réseau initial à un temps donné. Chaque sommet est disposé de manière contiguë à son voisin. Les sommets sont ainsi placés sur une grille de telle sorte que chaque ligne corresponde à un gène et chaque colonne à un pas de temps.

La visualisation de l'information est intégrée par un gradient de couleur du rouge au vert en passant par le noir. Le rouge correspondra aux valeurs faibles de l'expression d'un gène au cours du temps et le vert aux valeurs fortes. Afin d'établir ce gradient, toutes les valeurs d'expression ont été normalisées par la méthode de la variable centrée réduite afin d'obtenir des valeurs fluctuant généralement entre -2 et 2. Ainsi la valeur la plus forte de rouge sera -2 (et en dessous), ce gradient s'assombrira jusqu'à 0 (noir),

puis se teintera de vert avec la plus forte teinte a 2 (et au-dessus).

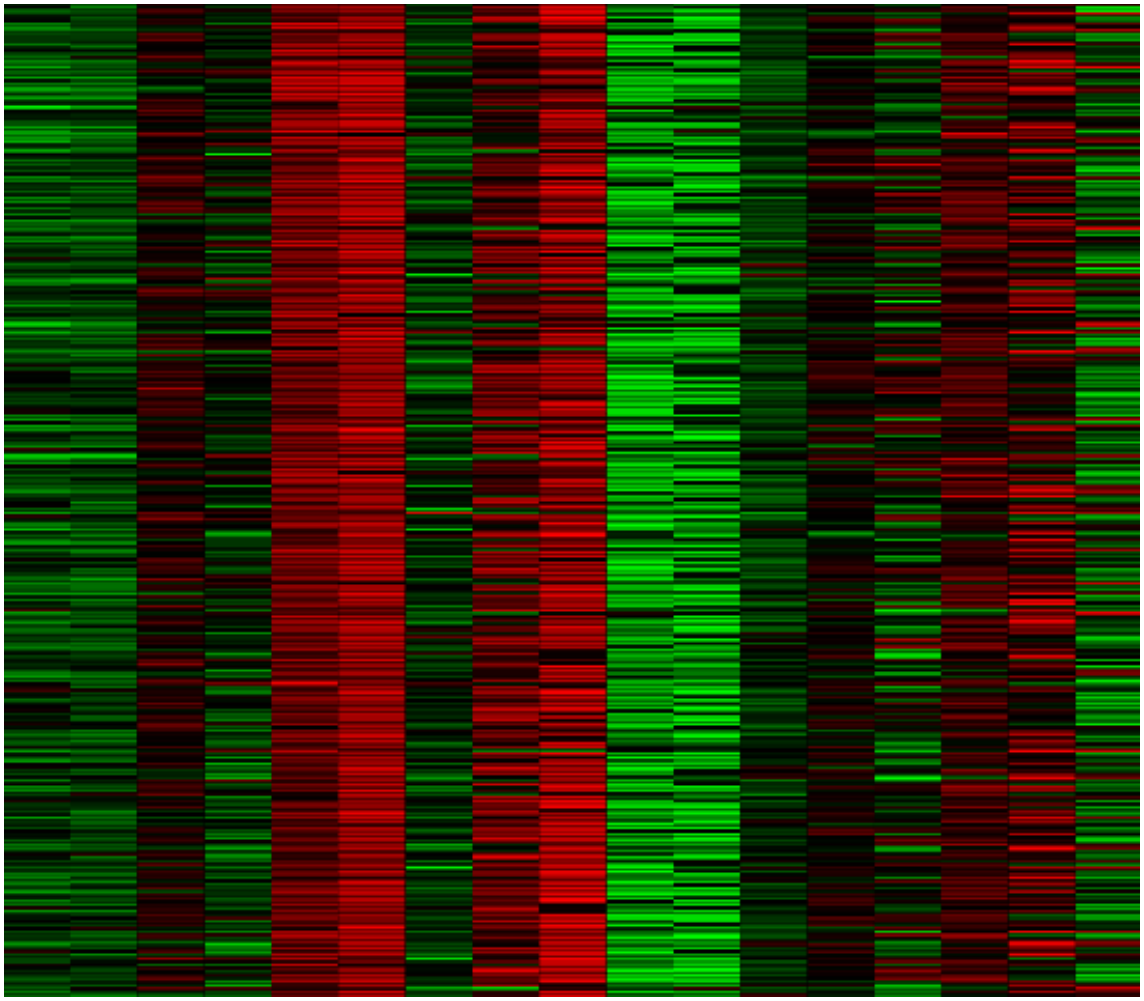
Pour la normalisation, deux choix s'offrait à nous : l'utilisation du log2 après avoir divisé par la moyenne afin de bien identifier quand l'expression double, ou l'utilisation de la variable centrée réduite qui passe par l'écart-type. Nous avons opté pour le second choix, afin de réutiliser l'écart-type pour trier les donner, car il représente la différence d'expression d'un gène au cours du temps.

FIGURE 3 – Diagramme des Z-scores



Les lignes de la carte de chaleur ont été organisées selon le résultat du clustering des valeurs de corrélation entre les gènes.

FIGURE 4 – Carte de chaleur partielle de variation d'expression génique normalisée



Certaines caractéristiques se démarquent sur la carte de chaleur. Les valeurs d'expression sont globalement élevés (vert) (t1-t4) au début des expériences, puis s'affaiblissent (rouge) (t5-t9) avant de remonter (t10-t14) et faiblir de nouveau (t15-17).
On peut ainsi supposer que l'organisme fonctionne correctement lors de la phase de consommation du glucose jusqu'aux alentours du temps 6 où la diauxie se met en place, l'organisme manque alors de glucose et la majorité des gènes ne sont plus autant exprimés. Lorsque l'organisme se met à consommer du lactose, l'expression des gènes revient. Enfin, lorsque le lactose est épuisé, la valeur de l'expression de la plupart des gènes diminue.

Analyse de données

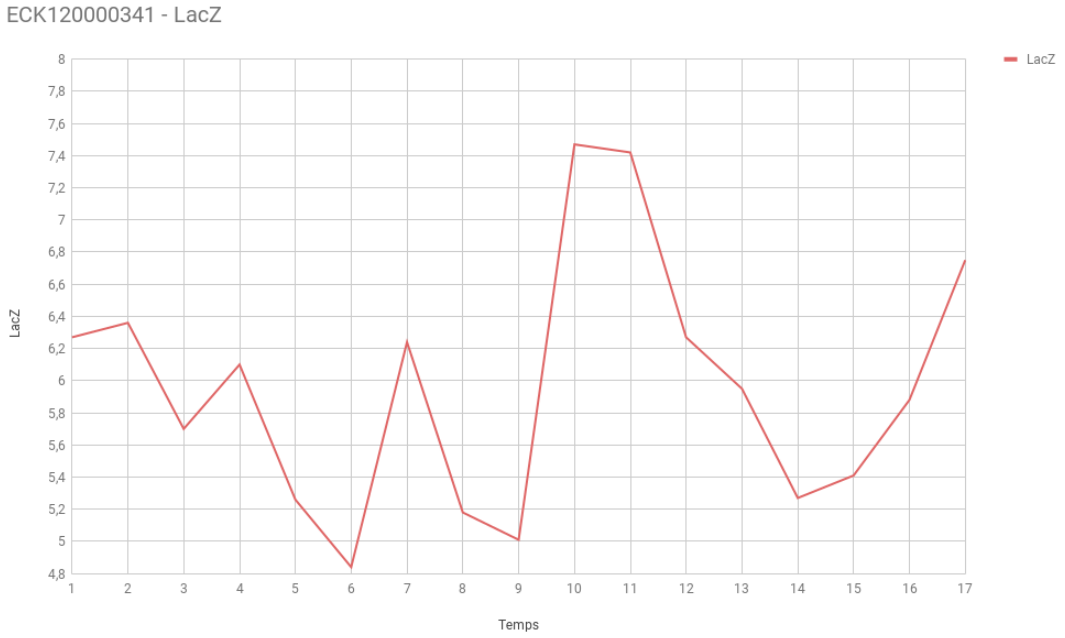
Afin d'obtenir une analyse approfondie des données, nous avons voulu utiliser la base de donnée RegulonDB[5]. Nous avons ordonné la carte de chaleur selon l'écart-type de chaque gène, afin d'avoir les gènes avec la plus grande différence d'expression au cours de l'expérience en ordre décroissant. Nous espérions que ces gènes aient le plus d'impact lors de la diauxie ou qu'ils soient responsables d'un mécanisme spécifique lors du passage de la consommation du glucose à celle du lactose. Nous souhaitions ainsi interroger la base RegulonDB afin d'obtenir de meilleurs clusters et/ou des labels pour certains nœuds

et ainsi avoir une meilleure visualisation en général. Mais parmi les locus testés à la main, aucun n'était pertinent. En effet, tout gène largement affecté par un manque de glucose ou de lactose peut avoir un écart-type de sa valeur d'expression au cours du temps élevé.

Après cet échec nous avons voulu observer les gènes qui n'adoptaient pas le comportement général de l'organisme. Ceux qui seraient en vert sur la carte de chaleur (donc fortement exprimés) lorsque tous les autres seraient rouges au moment de la diauxie. Ou bien le cas où un gène serait rouge pendant la phase de consommation du glucose (peu exprimé) alors que les autres seraient verts. Malheureusement cet essai n'a pas été concluant, en effet selon la base de données aucun gène n'était lié aux sucres ou bien à la diauxie. Ces gènes étaient pour la plupart responsables d'autres processus biologiques tels que le transport du carbone.

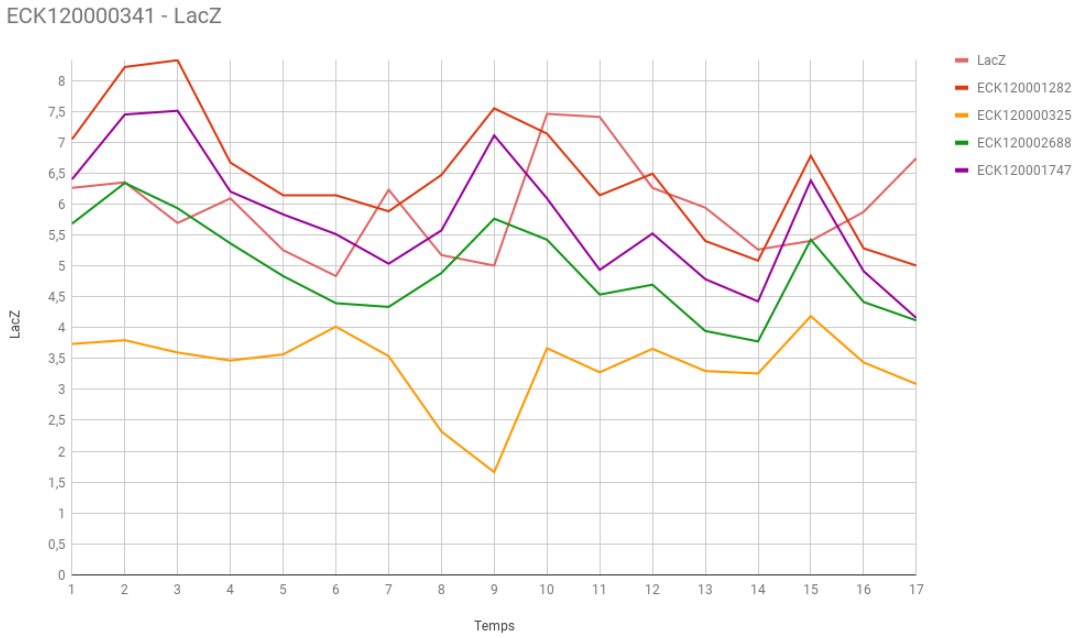
Nous nous sommes ensuite intéressé de plus près au métabolisme de la bactérie. En effet cette souche étant cultivée sur un milieu contenant du lactose, l'opéron lactose qui est nécessaire au transport et au métabolisme du lactose chez *E.Coli* doit être représenté dans le génome de l'organisme. L'opéron lactose est composé de trois gènes structurels : lacA (ECK0339), lacY (ECK0340) et lacZ (ECK0341), nous avons ainsi émis l'hypothèse que ces gènes devaient faire partie de notre jeu de données. Une fois ces gènes identifiés au sein de notre jeu de données l'idée était de comparer leur niveau d'expression au cours du temps, d'observer leur répartition dans la clusterisation réalisée au préalable (les gènes de l'opéron appartiennent-ils tous au même cluster ? Un cluster est-il spécifique au lactose ?) et ainsi conclure sur leur rôle dans la croissance post-diauxie de la bactérie. Cependant le gène lacZ est le seul gène caractéristique de l'opéron lactose présent dans notre jeu de données, nous avons décidé malgré tout de le repérer dans notre graphe et d'étudier l'évolution de son niveau d'expression (Figure 5)

FIGURE 5 – Evolution du niveau d'expression du gène lacZ au cours du temps



On remarque, en observant cette courbe, une augmentation significative du niveau d'expression de ce gène aux temps 9 et 10 de l'expérience. En se référant à la figure 1, ces temps correspondent à la phase exponentielle de croissance de la bactérie lors de la consommation du lactose post-diauxie. Ces résultats étant cohérents selon les critères biologiques liés à la croissance bactérienne, nous pouvons émettre l'hypothèse qu'en présence du gène lacA et lacY dans notre jeu de données, ils auraient présenté une évolution similaire au niveau de leur expression génique. Le gène lacZ est groupé dans un cluster (viewMCLMetric = 86) en présence de 4 autres gènes, nous avons ainsi comparé le niveau d'expression de ces 5 gènes (figure 6).

FIGURE 6 – Evolution du niveau d'expression du gène lacZ et des gènes appartenant au même cluster au cours du temps



Conclusion

Le but de ce projet était de réaliser une analyse visuelle de données d'expression génique à partir d'un jeu de données issue de la croissance d'une souche d'*E.Coli*. Nous avons pu, grâce au logiciel Tulip, construire un graphe complet dont les sommets sont les gènes provenant du génome de la bactérie et dont la longueur de chaque arête est corrélée à la co-expression des gènes concernés. Un algorithme implémenté dans le logiciel Tulip nous a permis de rassembler nos gènes en cluster (groupe de gènes similaires) selon leur corrélation (coefficient de Pearson). Nous avons également pu dessiner une carte de chaleur permettant de mettre en avant l'évolution du niveau d'expression de chaque gène au cours de la croissance de la bactérie.
Nous avons conclu notre travail sur l'étude des gènes caractéristiques de l'opéron lactose qui est responsable du transport et du métabolisme du lactose chez *E.Coli*.

Afin de mieux observer le passage de la consommation préférentielle de la bactérie du glucose au lactose, nous aurions pu nous concentrer sur la corrélation entre les gènes aux étapes autour de la diauxie. Par exemple entre le temps 4 et 8. Ce choix nous aurait permis d'obtenir des clusters différents et peut-être de meilleurs résultats. En effet, on peut voir sur la figure 6 que la corrélation entre les gènes est très importante sur la fin des expériences, aux étapes sans consommation de sucres. La corrélation est moins évidente autour de la diauxie.

Bibliographie

[1] Tulip, http ://tulip.labri.fr/TulipDrupal/

[2] S. Hachul, M.Jünger, Journal of Graph Algorithms and Applications, http ://jgaa.info/ vol. 11, no. 2, pp. 345–369 (2007)

[3] Antoine Lambert, Romain Bourqui, David Auber. Winding Roads : Routing edges into bundles. Computer Graphics Forum, Wiley, 2010, 29 (3), pp.853-862. <hal-00495279>

[4] Markov Cluster algorithm, Graph Clustering by Flow Simulation, Stijn van Dongen PhD Thesis, University of Utrecht (2000).

[5] RegulonDB, http ://regulondb.ccg.unam.mx/

Annexes

Quelques bugs sont susceptibles d'apparaître lors du lancement du programme. Ils ne sont généralement pas présent au premier lancement du logiciel Tulip mais peuvent parasiter le code après plusieurs essais.

— Bibliothèque scipy non reconnue (arrive à chaque premier lancement du fichier), il suffit de recharger le fichier python dans l'IDE pour que l'erreur n'apparaisse plus.
— Apparition d'une arête étrange (n° 6267). (Il faut relancer tout le programme).
— Temps d'exécution supérieur à 10 minutes.