Un algorithme de normalisation des données à l'aide de graphes pour le traitement non-linéaire des données : Application à l'optimisation des cartes de Kohonen

Catherine Aaron[1]

SAMOS-MATISSE- Université Paris I
90 rue de Tolbiac,
75013 PARIS, France
(e-mail: catherine_aaron@hotmail.com)

Abstract. Dans le cas où la structure des données à étudier est fortement non-linéaire les méthodes de normalisation "classiques" sont inefficaces pour rendre compte de l'organisation des données. Pour pallier ce problème on propose un algorithme de normalisation des données reposant sur le choix d'un graphe et visant à rendre les voisinages des points sphériques. La version "exhaustive" d'un tel algorithme étant coûteuse en temps de calcul, on en présentera, aussi, sa version stochastique.

En illustration de cette méthode de normalisation, nous proposerons un indicateur permettant de choisir le nombre de lignes et de colonnes à demander en entrée d'une carte de Kohonen.

Keywords: Normalisation, Graphes, Distance Curviligne, Cartes de Kohonen.

1 Normalisation et Analyse des données non linéaire

1.1 La distance curviligne

Dans le cas de l'analyse des données linéaire, l'hypothèse topologique sous-jacente est la convexité des données qui permet de lier les points par des segments en restant dans l'ensemble considéré. En revanche, dans le cas de l'analyse des données non-linéaire, la seule hypothèse topologique est la connexité, qui ne garantit que l'existence d'un chemin continu liant les points deux à deux.

Ainsi dans le cas où la structure des données serait non linéaire, la mesure de distance entre les points représentant le mieux l'organisation des données est la distance curviligne (ou géodésique) qui, en résumé, représente la longueur minimum d'un chemin continu, liant les points, au sein de l'ensemble considéré.

Cette distance (curviligne) est utilisée dans de nouvelles méthodes d'analyse des données non linéaires telles qu' "ISOMAP" ou "curvilinear distance analysis" et, on verra, dans la dernière partie, en quoi son étude peut aider au paramétrage des cartes de Kohonen.

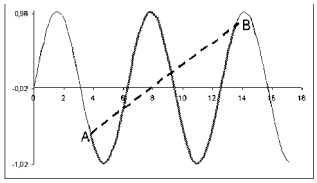

Fig. 1. distance curviligne vs distance euclidienne dans le cas d'un ensemble non convexe.

1.2 Impact de la normalisation sur la distance curviligne

Dans la pratique on approche la distance curviligne en deux étapes : dans un premier temps on détermine un graphe sur les points ($k-$ plus proches voisins, $\varepsilon-$ voisins, MST...) qui lie les points si on peut considérer qu'ils sont "suffisamment proches". Puis l'algorithme de Disjkstra permet de rechercher le plus court chemin liant les points et d'en donner sa longueur et d'obtenir ainsi une approximation de la distance curviligne).

Le problème est que les le graphe des liaisons est très sensible aux changements d'échelles. En illustration le graphique si dessous montre le Minimum spanning tree d'un même tirage sinusoïdal pour différentes échelles sur l'axe horizontal.

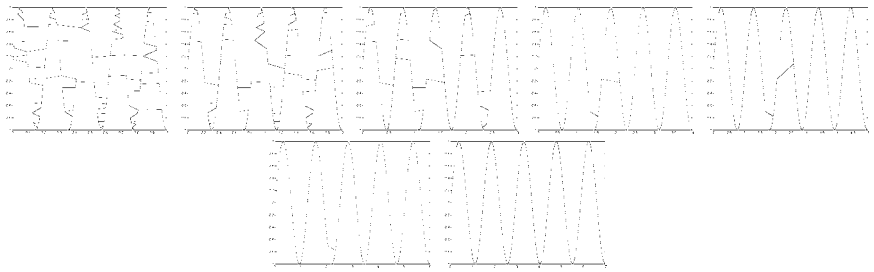

Fig. 2. MST pour différentes échelles horizontales (de 1 à 7) d'un même tirage sinusoïdal.

Le chapitre suivant présente un algorithme qui recherche les transformations à effectuer sur les données pour construire un graphe "correct" c'est a dire résumant l'information topologique des données.

2 Algorithme de normalisation proposé

2.1 Principe

Travailler sur des données fortement non linéaire implique de s'intéresser localement aux données. Les méthodes "classiques" reposant sur la dispersion

générale autour d'un indicateur central seront ici inefficaces. Pour illustrer ce propos, dans tous les exemples proposés le point de départ des algorithmes suivant sera le résultat des données centrées réduites de manière classique (division par l'écart type).

La méthode de normalisation proposée a pour principe de rendre, en moyenne, les voisinages de forme sphérique et de rayon moyen égal à 1. Autrement dit, on veut rendre les voisinages isotropes en moyenne.

2.2 Version exhaustive

On se fixe un type de graphe ($k-$plus proches voisins, Minimum spanning tree...) qui sert à construire les voisinages que l'on veut rendre isotropes. Puis on itère l'algorithme suivant qui effectue à chaque étape :

- (1) Calcul du graphe G
- (2) Stockage de Y matrice de toutes les vecteurs liaisons
- (3) Effectue une ACP sur Y (les résultat sont une isométrie P et $Y := YP$)
- (4) Application de l'isométrie à X : $X := XP$ (comme P est une isométrie le graphe ne change pas $G(X) = G(XP)$)
- (5) Effectue $Z = |Y|$ vecteur constitué des valeurs absolu des liaisons dans toutes les (nouvelles) directions.
- (6) $pds = mean(Y)$ longueur moyenne des liaisons dans toutes les directions.
- (7) Pour tout j tel que $pds(j) \neq 0$ on effectue $X(:, j) = X(:, j)/pds(j)$

Les points (2) à (4) visent essentiellement à faire tourner les axes de manière à rendre toutes les directions de liaisons possibles. Les points (5) à (7) visent à rendre les liaisons de tailles équivalentes sur tous les axes significatifs.

Remarque : S'il existe un système de $d' < d$ axes linéaires permettant résumant complètement l'information celui ci est obtenu par l'algorithme. On obtient dans ce cas là les même résultats qu'ISOMAP mais avec un temps de calcul largement plus court car seul les graphes sont calculés et non toutes les distances curvilignes.

2.3 Version Stochastique

La version stochastique a, uniquement, comme but d'accélérer le temps de calcul du au calcul du graphe (en $O(N^2)$ pour les $k-$plus proches voisins et en $O(N^2 log(N))$ pour le MST) pour cela, à chaque étape on tire (sans remise) $N' < N$ points sur lesquels on calcul le graphe, la rotation et la pondération des axes qu'on applique sur toutes les données.

2.4 Quelques résultats

Les exemples suivants présentent tous les résultats de l'algorithme exposé ci-dessus pour des données sinusoïdales en dimension 2. Le graphe de référence le MST. Les graphiques résultats se lisent verticalement :

- (1) Graphe et données dans le cas de la normalisation standard.
- (2) Pourcentage d'inertie expliquée (cumulée) par axe.
- (3) Angle de la plus grande rotation de l'isométrie.
- (4) Poids de chaque axe.
- (5) Graphe pour les données après normalisation.

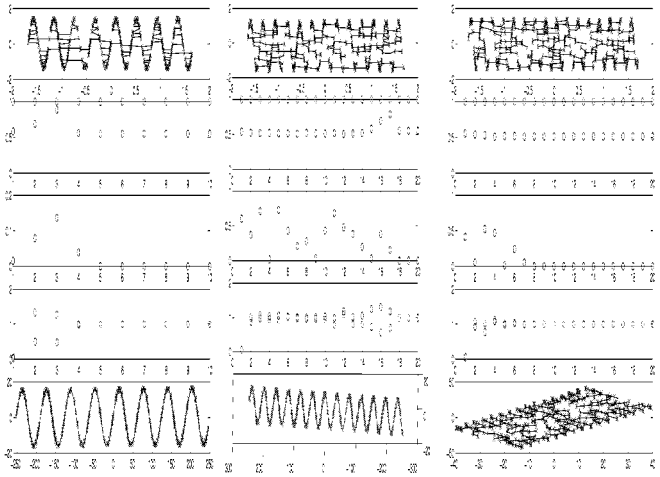

Fig. 3. 500 points avec $X(:,1) = unifrnd(0,1)$ et $X(:,2) = sin(\omega X(:,1))$ avec $\omega \in \{50, 80, 100\}$

On voit que l'algorithme donne de relativement bons résultats jusqu'à ce qu'il y ait un effet "saturation" pour des fréquences trop élevées.

Les données suivantes ont été tiré de la manière suivante : $X(:,1)$ suit une loi uniforme sur $[0,1]$ et $X(:,2) = sin(\omega X(i,1)) + \sigma\varepsilon$ avec ε suivant une loi normale centrée réduite. Puis on a appliqué une rotation d'angle $pi/4$ aux données. On observe dans ce cas que la saturation est plus rapide.

2.5 Conclusion et perspective

Les résultats sont encourageants mais nous ne sommes pas encore parvenus à quantifier la performance de l'algorithme. On sait aujourd'hui qu'il n'existe

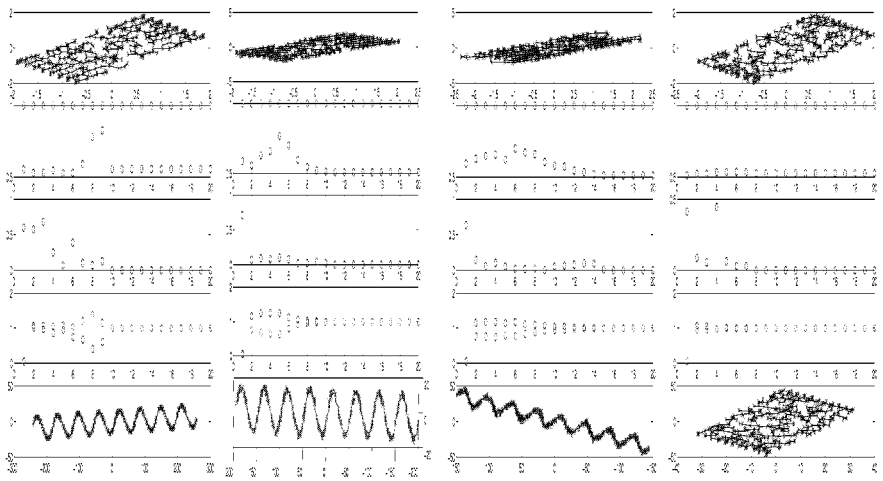

Fig. 4. exemples avec rotation : (1) $\omega = 50, \sigma = 0$,(2) $\omega = 50, \sigma = 0.1$, (3) $\omega = 50, \sigma = 0.2$,(4) $\omega = 70, \sigma = 0$

pas forcément une unique solution au problème de l'existence d'une transformation rendant les voisinages (exactement) isotropes en moyenne (mais la probabilité d'existence tend vers 1 lorsque le nombre d'individus augmente). On aimerait surtout quantifier le fait que la distance curviligne estimée à l'issue de la normalisation corresponde "au mieux" à la "vraie" distance curviligne par une vraie démonstration et, non, uniquement par des simulations.

3 Paramétrage d'une carte de Kohonen

3.1 Les cartes de Kohonen

L'algorithme des cartes de Kohonen vise à projeter les données sur une "carte" i.e. une structure de voisinage fixé a l'avance. Plusieurs types d'utilisations en sont faite. Essentiellement en représentation des données en plus petite dimension (pendant non linéaire à l'ACP) et en classification. Nous nous intéresserons ici plus particulièrement à l'aspect "projection" et représentation des données et non a l'aspect "classification".

Brièvement, pour projeter des données sur une carte de Kohonen, on se fixe une topologie c'est a dire un nombre de cellules et leurs voisinages associés.

A chaque case (i, j) dans la carte correspond donc un vecteur code $\boldsymbol{C_{i,j}}$ dans l'espace des données. L'algorithme de Kohonen repose sur une conservation de la topologie c'est a dire sur le fait que la topologie induite par une

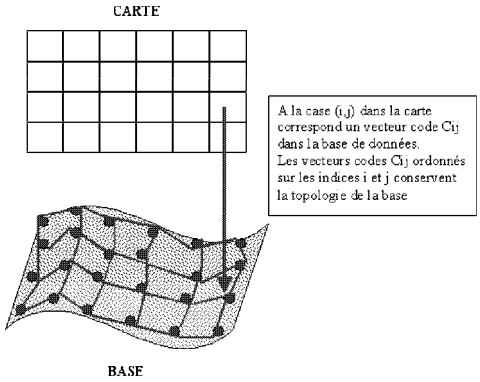

Fig. 5. cartes de Kohonen

distance sur Z^2 (distance entre les cases) respecte la topologie induite par une distance sur $\Re^d$ (distance entre les vecteurs codes).

3.2 Un indicateur de respect de la topologie

Le propos précédent nous permet de définir un indicateur de respect de la topologie. On choisit comme distance sur Z^2 la distance euclidienne : $d_1((i,j),(i',j')) = \sqrt{(i-i')^2+(j-j')^2}$. Le choix et la construction d'une distance sur les vecteurs codes est légèrement plus délicat. Etant donné que les données peuvent être dans un espace non linéaire on va choisir la distance curviligne mais, comme le nombre de vecteurs codes est relativement faible la détermination de la distance curviligne se fait en s'autorisant des chemins qui passent par des points de la base de donnée.

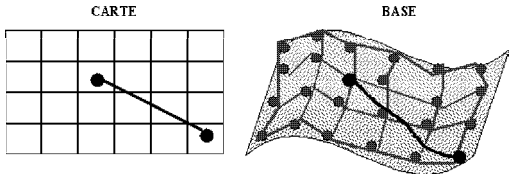

Fig. 6. Distance sur la grille et distance entre les vecteurs codes

Les données sont, en préliminaire, normées par l'algorithme décrit dans la section précédente.

Les résultats seront alors présentés de la manière suivante : Pour une base de donnée, et pour une carte de Kohonen (ici un nombre de ligne et un nombre de colonne), on va tracer les nuages de points liant la distance entre les cases et distances entre les vecteurs codes, et indiquer leur corrélation.

3.3 Résultats

Un premier exemple est constitué d'un tirage uniforme de 200 points sur $[0,1]^2$. On note alors que, comme escompté ce sont les cartes "carrées" (i.e. comportant autant de lignes que de colonnes) qui reconstituent au mieux la topologie de l'espace.

Le premier graphique donne les nuages de points entre les deux distances : première lignes pour des cartes $(1,3)$ jusqu'à $(1,10)$ deuxième lignes pour des cartes $(2,2)$ jusqu'à $(2,10)$... Le second graphique présente les coefficients de corrélation entre les distances pour toutes ces des cartes

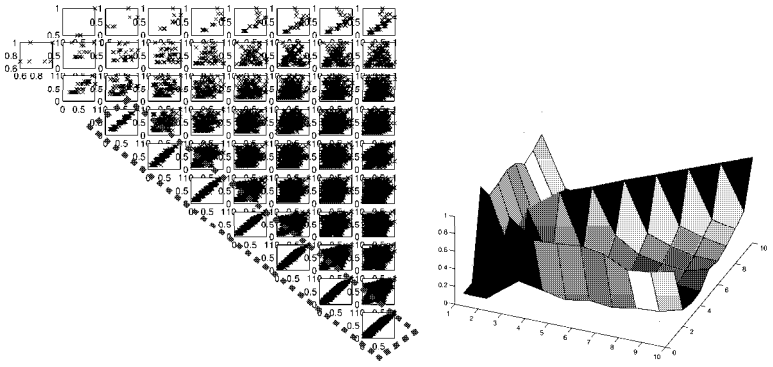

Fig. 7. Distance sur la grille et distance entre les vecteurs codes et correlation entre ces dernières dans le cas d'un tirage uniforme

Un deuxième exemple est constitué d'un tirage de 200 points avec $X(:,1)$ suivant une loi uniforme $[0,1]$ et $X(:,2) = sin(15X(:,1))$. Là aussi, comme prévu, on observe les meilleurs résultats pour une carte de Kohonen de largeur 1, c'est a dire pour une "ficelle", ce qui correspond au fait que la dimension intrinsèque des données est 1 (voir figure 8).

Pour finir, on s'est placé en dimension 3 avec $[X(:,1)X(:,2)]$ uniformément tirés dans $[0,1]^2$ et $X(:,3) = sin(15X(:,1))$. Le résultat obtenu qui semblait étonnant a première vue (préconisation d'une ficelle) est confirmé par la représentation du nuage de point et des vecteurs codes associés a une ficelle (15 cellules) voir figure 9.

References

1.De Bodt,E. , Cottrel, M., Verleysen, M.: : Statistical tools to asses the reliability of self-organizing maps In: Neural Networks 15 (2002) 967-978

2.E.W. Dijkstra, A note on two problemes in connection with graphs, *Mathematics*, (1):269-271, 1951.

3.J.B. Tenenbaum, V. de Silva, A global geometric framework for non-linear dimensionality reduction, *Science*, (290):2319-2323, December 2000.

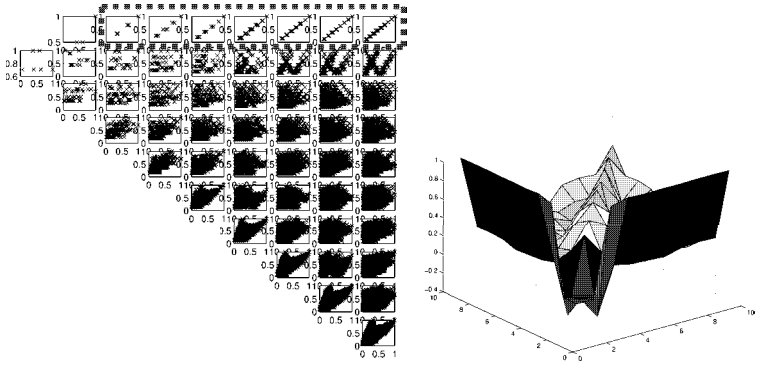

Fig. 8. Distance sur la grille et distance entre les vecteurs codes et correlation entre ces dernières dans le cas d'un tirage sinusoïdal

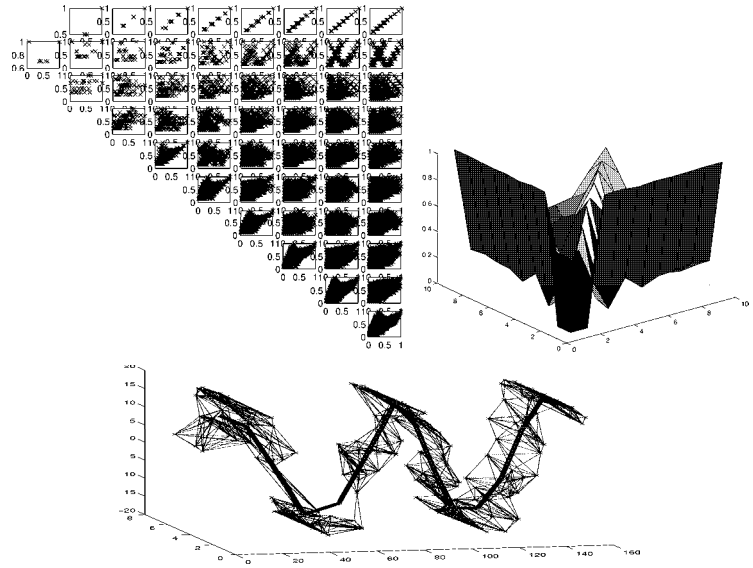

Fig. 9. Distance sur la grille et distance entre les vecteurs codes et correlation entre ces dernières dans le cas d'un tirage sinusoïdal en dimension 3

4.J.A. Lee, A. Lendasse and M. Verleysen, Curvilinear Distance analysis versus isomap In M. Verleysen, editor, *proceedings of the* 8^{th} *European Symposium on Artificial Neural Networks* (ESANN 2000), d-side pub., pages 13-20, April, Bruges (Belgium), 2000.