EURIA
Euro-Institut d'Actuariat

Rapport de Bureau d'étude

Mai 2018

Réplication de Bases de Données

Allianz

Taki Mohammed
Dehodang Matthieu
Master 1 EURIA

Tuteurs : M. Dominique Abgrall
M. Brice Franke

Année 2017 — 2018

Sommaire

Remerciements

Nous tenons à adresser nos remerciements à tous les enseignants et interlocuteurs ayant apporté leur aide à la réalisation de ce bureau d'étude. En particulier :

Monsieur Dominique ABGRALL, actuaire à Allianz et membre du collège de direction de l'Euro-Institut d'Actuariat, pour nous avoir accompagné pendant ce projet ;

Brice FRANKE, Docteur en Mathématiques à l'Université de Bretagne Occidentale, pour son accessibilité ;

Monsieur Anthony NAHELOU, actuaire à Sterenn Actuariat et membre du collège de direction de l'EURIA ;

Monsieur Marc QUINCAMPOIX, directeur de l'EURIA ;

Monsieur Franck VERMET, directeur des études de l'EURIA, pour son intérêt dans nos recherches.

1 Introduction

La question de pouvoir transmettre ou non les données est fréquente dans une compagnie d'assurance dans un but d'élaboration d'étude avec des partenaires professionnels.

"Une donnée est une description élémentaire d'une réalité. C'est par exemple une observation ou une mesure" Serge Abiteboul.

Cependant il n'est pas permis de diffuser librement les données personnelles des clients à un tiers.

Le présent bureau d'étude est une première approche de l'analyse de la réplication de bases de données. Des contraintes nous ont été imposées et des choix de simplification/généralisation ont été faits afin de déterminer les directions à suivre.

Dans un premier temps sera exposé le contexte dans lequel s'inscrit ce sujet, élément encadrant les réflexions faites au cours de l'année.

Puis nous aborderons les différents critères que nous aurons retenus afin de valider nos modèles. Ce pourront être des critères déjà existants dans la littérature ou bien des démarches mises en place de façon indépendante.

Enfin nous passerons aux méthodes de réplications à proprement parler que nous étudierons pour en extraire le plus d'informations possibles, aussi bien les aspects positifs que les limites apparentes.

Il sera également proposé pour terminer une introduction à une méthode sophistiquée non approfondie dans ce rapport.

2 Contexte assurantiel et données

L'objectif de ce Bureau d'étude est donc de répliquer des bases de données assurantielles.

"Une base de données permet de stocker et de retrouver l'intégralité de données brutes ou d'informations en rapport avec un thème ou une activité ; celles-ci peuvent être de natures différentes et plus ou moins reliées entre elles. Dans la très grande majorité des cas, ces informations sont très structurées, et la base est localisée dans un même lieu et sur un même support. Ce dernier est généralement informatisé" Colin Ritchie.

	Age (an)	Taille (m)	Poids (kg)
Individu 1	20	1.87	94
Individu 2	56	1.74	56
Individu 3	29	1.55	49
Individu 4	45	1.65	73

TABLE 1 – Exemple simple d'une base de données

Il s'agit dans cette problématique d'être capable, à partir d'un tableau de départ contenant p colonnes (variables) et n lignes (individus) de créer un nouveau tableau contenant des valeurs différentes mais avec des propriétés statistiques semblables (moyenne, écart-type, médiane, corrélations...).

Le but du travail effectué cette année n'est pas d'obtenir une réponse absolue à la problématique, mais de comprendre comment l'appréhender, quelles sont les questions à se poser, quelles seraient les méthodes qui pourraient correspondre à la problématique et quelles sont celles qui ne peuvent pas fonctionner et pourquoi.

Avant d'aller plus loin, évoquons le principe du bootstrap auquel on peut penser en premier lieu pour tenter de répondre au sujet posé.

Utilisé depuis la fin des années 1970 avec l'expansion des ordinateurs, le bootstrap est une méthode intéressante pour estimer les paramètres d'un échantillon, de façon paramétrique ou non. La description du principe est ici faite en dimension 1 mais il est tout à fait possible de raisonner en dimensions supérieures.

Sur la base d'un échantillon initial $X = (X[1], X[2], ..., X[n])$, on crée un grand nombre de nouvelles suites de même longueur $X^k, k \in \{1, ..., N\}$ avec N entier fixé, tirées de façon aléatoire et avec remises dans l'échantillon X. Une fois ces listes formées, nous obtenons une distribution des différents paramètres de l'échantillon X (exemple : moyenne, variance, médiane,...) et nous pouvons en déterminer des *estimateurs bootstrap*.

Cette méthode ne sera pas utilisée dans cette étude (sinon indirectement). En effet il est important de noter que la finalité est de générer une nouvelle base avec des individus (différents), et non de recréer un tableau avec un tirage parmi d'anciennes données.

2.1 Aspects juridiques

Du point de vue juridique le traitement des bases de données est suivi de près par le législateur, d'où la motivation de ce sujet de recherche.

En effet un assureur possédant un ensemble de bases de données sur ses assurés ne peut pas les revendre à des agences de marketing par exemple. Leurs utilisations sont soumises à des réglementations, dont l'ensemble est décrit dans le `Règlement Général sur la Protection des Données` (RGPD) de l'Union Européenne adopté le 14 avril 2016 par le Parlement européen. Ce règlement sera applicable à partir de mai 2018 et offrira un cadre harmonisé des règles concernant la protection des données. Nous pouvons citer par exemple l'application du règlement de façon *extra-territoriale* : sont concernées les entreprises établies en dehors de l'UE qui traitent d'informations relatives aux organisations de l'UE, mais également les sociétés non européennes possédant des données sur les citoyens européens ; Le *droit à l'oubli* est remplacé par le *droit à l'effacement* : un citoyen peut exiger l'effacement de ses données à la condition de fournir six motifs valables ; Ou encore l'obligation pour l'entreprise traitant les données de notifier l'autorité concernée de toute violation ou fuite de ces dernières. À noter qu'un manquement à ces obligations peut entraîner pour l'entreprise une sanction pouvant aller jusque 20 millions d'euros ou 4% du chiffre d'affaire mondial.

Remaque : Les aspects négatifs des bases de données sont souvent mis sur le devant de la scène mais la loi ne prend pas en compte que ces

faits, elle admet également le côté positif de ces informations notamment pour les gros investissements (besoin de connaissances avant d'entamer un projet de construction) et incite à s'informer avant d'agir.

En France, c'est principalement la `Commission Nationale Informatique et Libertés` (CNIL) qui détermine les conditions à respecter lorsqu'une personne ou un organisme détient des données à caractères personnels.

La CNIL, fondée en 1978 par la loi informatique et libertés, est un organisme administratif français quasi-indépendant dont l'objectif est de protéger les données informatiques existantes sur chaque citoyen.

Elle fut créée suite au projet gouvernemental SAFARI ayant pour but de rassembler toutes les informations sur une personne en un seul ensemble identifiable par un numéro unique. La crainte suscitée par un tel fichage conduisit l'État à créer une commission afin de réguler les transmissions d'informations d'aspect privé.

Composée d'un collège de 18 membres, employant pas loin de deux-cent agents en 2015 pour un budget de 17.1 millions d'euros, le pouvoir de la CNIL s'est renforcé au cours des ans lui donnant chaque fois une meilleure force de contrôle, de sanction, et de droit des individus.

La CNIL a plusieurs missions : elle informe les personnes sur leurs droits, elle autorise ou non les transmissions et utilisations de données sensibles, elle accompagne le citoyen lorsqu'il demande à accéder aux données le concernant, elle contrôle les entreprises, sanctionne tout non respect des règles relatives aux données, et se tient à jour des nouvelles applications et technologies afin de comprendre et parer les nouveaux risques liés aux traitements des données.

Cinq principes clés sont mis en avant par la commission :

— La finalité : l'organisme collecteur doit expliciter son but en collectant des données,

— La pertinence : ne sont demandées que les données nécessaires,

— La conservation : les données sont supprimées une fois leur fonction remplie,

— Les droits : obligation d'informer les personnes de la collecte de leurs données, de plus ces dernières ont des droits sur ces informations

(accès, rectification, opposition),

— La sécurité : les mesures indispensables sont prises afin de conserver et protéger la confidentialité des données.

Nous constatons le rôle primordial joué par la CNIL, d'une part du fait de la nécessité d'obtenir son aval dans certaines opérations impliquant le traitement de données, et d'autre part en tant qu'autorité compétente dans la protection des droits du citoyens.

Le graphique suivant présente les questions qu'un détenteur de données doit se poser lorsqu'il effectue des opérations dessus.

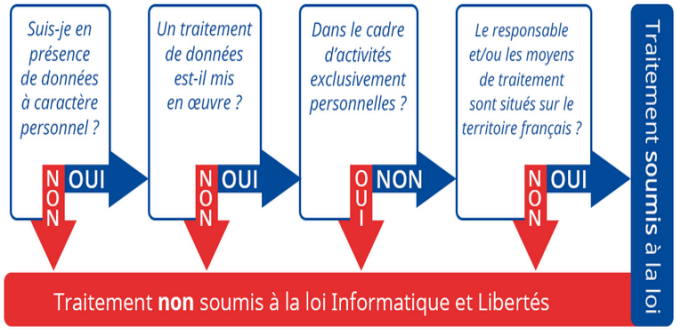

FIGURE 1 – Loi Informatique et Liberté : Suis-je concerné ?

La Norme de Pratique Actuarielle (NPA 5) :

Adoptée par l'Institut des Actuaires le 16 novembre 2017, la NPA 5 a été mise en place afin de guider l'actuaire face aux éventuels problèmes liés à la manipulation de données considérées comme sensibles (état de santé, génétique, appartenances diverses...). Cette norme rejoint les codes de déontologies déjà existants et les consignes édictées par la CNIL et le RGPD : l'actuaire doit considérer l'origines de ses données, la pertinance et la nécessité de leur utilisation, respecter les règles de confidentialité et de secret professionnel. Cette norme est applicable quelque soit le territoire sur lequel travaille l'actuaire mais ne supplante pas la loi déjà existante.

2.2 Données considérées

Le postulat de départ est de ne considérer que des bases de données à valeurs numériques. On ne considérera pas la possibilité de rencontrer

des chaînes de caractères. De plus il n'est pas nécessaire de faire des études sur les valeurs aberrantes ou autres aspects de la base à répliquer car l'on suppose que l'utilisateur connaît et maîtrise ses données.

2.2.1 Dimension 1

Le cas le plus simple de base de données est le tableau à une colonne (p=1). Plusieurs des méthodes abordées dans ce rapport s'appliquent sur chaque colonne de la base indépendamment les unes des autres. C'est pourquoi les raisonnements se feront parfois en dimension 1 dans un premier temps.

Cependant un travail supplémentaire devra se faire pour reconstruire les nouvelle bases (voir la sous section suivante).

2.2.2 Dimension 2 et supérieures

Lorsque la base considérée contient plus d'une colonne, il devient plus délicat de reproduire ses éléments. En effet il peut y avoir des dépendances significatives entre les différentes variables, par exemple nous savons bien que le poids et la taille des individus sont corrélés. En outre, plus il y aura de variables, plus le biais entre la table de base et la table obtenue sera grand. D'où la nécessité de prendre ce fait en considération, cependant les réflexions en dimension supérieure à 2 sont similaires, par conséquent nous n'étudierons que le cas de la dimension 2, les autres pouvant soit se faire par extension, soit en comparant les variables deux à deux.

Nous appliquerons régulièrement les méthodes proposées sur un tableau composé d'une loi normale centrée réduite en première variable et d'une loi exponentielle de paramètre 2 en seconde. Ce sont des lois usuelles dont les densités sont aisément identifiables. Le tableau sera souvent désigné sous le nom *Table_ init*.

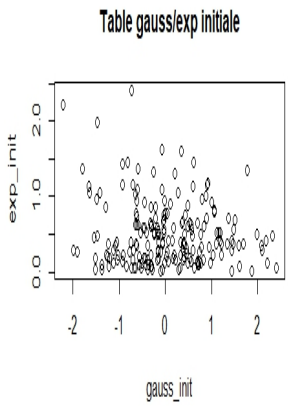

FIGURE 2 – Nuage de points du tableau Table_init

```
> summary(Table_init)
   gauss_init         exp_init
 Min.   :-2.2100   Min.   :0.0100
 1st Qu.:-0.6125   1st Qu.:0.1755
 Median :-0.0450   Median :0.3835
 Mean   : 0.0357   Mean   :0.5000
 3rd Qu.: 0.6125   3rd Qu.:0.6753
 Max.   : 2.4000   Max.   :2.4160
```

FIGURE 3 – Résumé de Table_init

Nous utiliserons également une couronne centrée en 0 et de rayons 8 et 10. Ce nuage de valeurs sera confondu avec un cercle (les données sont rarement disposées sur un cercle net). L'intérêt pour le cercle est d'avoir une structure très particulière et voir si elle peut être conservée après réplication. Le cercle sera souvent désigné sous le nom *cercle_ init*.

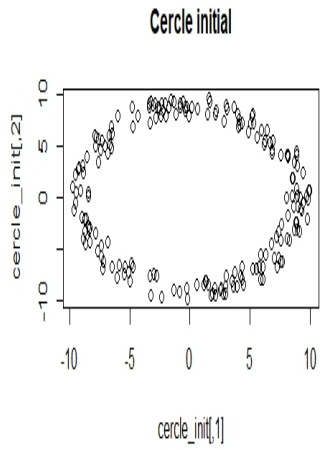

FIGURE 4 – Cercle cercle_init

```
> summary(cercle_init)
       V1                 V2
 Min.   :-9.720   Min.   :-9.8000
 1st Qu.:-5.165   1st Qu.:-6.2825
 Median : 1.480   Median : 0.2250
 Mean   : 0.638   Mean   : 0.3881
 3rd Qu.: 6.315   3rd Qu.: 6.9675
 Max.   : 9.930   Max.   : 9.7400
```

FIGURE 5 – Résumé de cercle_init

Première partie

Méthodes de validation

Nous parlerons dans cette section des divers outils utilisés pour comparer deux tableaux de données, et savoir à quel point ils sont similaires ou totalement différents. En d'autres termes, cette liste d'outils nous permettra de dire si oui ou non, la méthode de réplication est valable, voire même utilisable.

1 Nuage de points

En statistiques, un nuage de points est une représentation de données mettant en relation plusieurs variables. C'est un outil qui permet de mettre en évidence :

— Des tendances, des dépendances.
— Relations positives, négatives, directes, indirectes ou inverses.
— Des répartitions plus ou moins homogènes.
— Des données aberrantes s'écartant de l'écart type.

Comme un nuage de points montre si les données sont grandement réparties ou si elles sont concentrées dans un secteur, nous pouvons voir si les points du tableau répliqué sont répartis dans la même zone que ceux du tableau de départ.

2 Densité empirique de l'échantillon

En statistiques, la densité empirique f_n d'un échantillon $(x_1, ..., x_n)$ permet de représenter une estimation statistique de la distribution sous-jacente de l'échantillon.

Le principe de construction d'une densité empirique repose sur celui de la fonction de répartition empirique :

$$f_n(x) = \lim_{h \to 0} \frac{(F_n(x+h) - F_n(x-h))}{2h}$$

Donc pour un h petit, la densité empirique s'exprime ainsi :

$$f_n(x) = \frac{1}{2nh} \sum_{i=1}^{n} \mathbb{1}_{[-1,1]}(\frac{x - x_i}{h})$$

Toutefois, cet estimateur de la distribution de l'échantillon présente un gros défaut pour pouvoir estimer une densité : il n'est pas continu. Pour gagner la continuité, on utilise des estimateurs à noyau (on remplace l'indicatrice de l'expression précédente par une fonction réelle K), ce point sera approfondi dans la partie suivante.

Ainsi, dans tout notre bureau d'étude nous tracerons et comparerons les densités empiriques pour chaque variable des deux tableaux (initial et répliqué). Nous l'avons fait avec la fonction *density* sur R. Cette fonction permet de faire un tracé de l'estimateur à noyaux de la densité en prenant en argument l'échantillon $(x_1, ..., x_n)$, mais aussi la fonction K désirée et le paramètre h.

3 Validation par étude des barycentres

Une autre idée est de générer une base de données B ainsi que 1000 bases $B_1, ..., B_{1000}$ suivant les mêmes lois que B. Ensuite nous appliquons une méthode de réplication sur B afin de la répliquer également en 1000 autres bases $B_1^{'}, ..., B_{1000}^{'}$. Le principe est de comparer ensuite le comportement de $(b_i)_{i\in\{1,...1000\}}$ la distribution des barycentres de $(B_i)_{i\in\{1,...1000\}}$, et celui de $(b_i^{'})_{i\in\{1,...1000\}}$ la distribution des barycentres de $(B_i^{'})_{i\in\{1,...1000\}}$. L'objectif est de savoir si les méthodes de réplications employées sont trop précises ou non.

Remarque : En pratique il sera difficile de "générer" des bases. Cependant on disposera de bases de données de tailles conséquentes, ainsi une base d'un million de lignes pourrait être scindée en mille bases de mille lignes pour correspondre à l'idée proposée dans ce paragraphe.

4 Validation par des réseaux de neurones

Avant d'aborder le dernier critère de comparaison retenu, nous faisons quelques rappels sur les réseaux de neurones. Ces éléments ne sont en

aucun cas un cours mais servent à mieux cerner les capacités des réseaux de neurones.

4.1 Rappels sur les réseaux de neurones

Il existe de nombreux types de réseaux de neurones, on peut les diviser en deux grandes catégories selon la nature de leur algorithme d'apprentissage. En effet, les premiers sont dits supervisés car lors de l'apprentissage, ils doivent disposer d'un guide capable de leur indiquer ce qui devrait être produit en sortie pour chacune des informations fournies en entrées. Les seconds sont dits non supervisés car ils arrivent à s'auto-organiser.

4.1.1 Les perceptrons multi-couches

C'est un réseau à couche qui permet de positionner en entrées des éléments devant être appris. Ceux-ci tracent un chemin à travers le réseau. Une fois l'apprentissage effectué, en repassant les mêmes éléments en entrée ils réutilisent le même chemin et activent les mêmes neurones de sorties. Plusieurs neurones de sorties sont activés pour chaque comparaison, les résultats ne sont pas identiques à chaque comparaison, des approximations sont effectuées.

On note Y la variable à expliquer, $(X_1, ..., X_p)$ les variables explicatives et (a, b) des paramètres à estimer lors de l'apprentissage.

Un perceptron multicouche réalise donc une transformation des variables d'entrée :

$$Y = f(X_1, ..., X_p, a)$$

Ainsi, en régression avec un perceptron à une couche cachée de q neurones et un neurone de sortie, on note a une matrice dans $R^p \times R^q$ contenant les paramètres $a_{i,j}$ représentant la i$^{\text{eme}}$ entrée du j$^{\text{eme}}$ neurone. On note aussi b le vecteur dans R^q contenant chacun des paramètres b_j de la j$^{\text{eme}}$ entrée, du neurone de sorti.

$$y = f(x, a, b) = b_0 + b^t z$$

avec $z_k = \phi(a_{k,0} + (a_k)^t x)$ où ϕ est la fonction d'activation.

— Apprentissage

Supposons que l'on dispose d'une base d'apprentissage de n observations $((x_1, y_1)..., (x_n, y_n))$ des variables explicatives et des variables à expliquer. Considérons le cas le plus simple de la régression avec un réseau constitué d'un neurone de sortie linéaire et d'une couche à q neurones dont les paramètres sont optimisés par moindres carrés. Ceci se généralise à toute fonction perte dérivable et donc à la discrimination à m classes.

L'apprentissage est l'estimation des paramètres $(a_{i,j})_{(i,j)\in[1,p]\times[1,q]}$ et $(b_i)_{i=1,...,q}$ par minimisation de la fonction perte quadratique (ou d'une fonction d'entropie en classification) :

$$Q(a, b) = \sum_{i=1}^{n} (y_i - f(x, a, b))^2$$

Différents algorithmes d'optimisation sont proposés, ils sont généralement basés sur une évaluation du gradient par rétro-propagation.

— Rétro-propagation de l'erreur

Il s'agit donc d'évaluer la dérivée de la fonction coût en une observation et par rapport aux différents paramètres.

Soient $z_{k,i} = \phi(a_{k,0} + a_k x_i)$ et $z_i = (z_{1,i}, ..., z_{q,i})$.

Les dérivées partielles de la fonction perte quadratique s'écrivent :

$$\frac{\partial Q_i}{\partial b_k} = -2(y_i - \phi(x_i))(b^t z_i) z_{k,i} = \zeta_i z_{k,i}$$

$$\frac{\partial Q_i}{\partial a_{k,j}} = -2(y_i - \phi(x_i))\ (b^t z_i)\ b_k\ (\ f((a_k)^t) x_i\ )^t x_{i,p} = s_{k,i} x_{i,p}$$

Les termes ζ_i et $s_{k,i}$ sont respectivement les termes d'erreur du modèle courant à la sortie et sur chaque neurone caché. Ces termes d'erreur vérifient les équations dites de rétro-propagation :

$$s_{k,i} = b_k\ (\ \phi((a_k)^t) x_i\ )^t \zeta_i$$

dont les termes sont évalués en deux phases. D'abord avec les valeurs courantes des poids : l'application des différentes entrées x_i au réseau permet de déterminer les valeurs ajustées $\hat{f}(x_i)$. On peut ensuite déterminer les (ζ_i) qui sont rétro-propagés afin de calculer les $s_{k,i}$ et ainsi obtenir les évaluations du gradient.

— Algorithme d'optimisation

Sachant évaluer les gradients, différents algorithmes plus ou moins sophistiqués sont implémentés. Le plus élémentaire est une utilisation itérative du gradient : en tout point de l'espace des paramètres, le vecteur gradient de Q pointe dans la direction de l'erreur croissante. Pour faire décroître Q il suffit donc de se déplacer en sens contraire. Il s'agit d'un algorithme itératif modifiant les poids de chaque neurone selon :

$$b_k^{(r+1)} = b_k^{(r)} - \tau(\sum_{i=1}^{n} \frac{\partial Q_i}{\partial b_k^{(r)}})$$

$$a_{k,p}^{(r+1)} = a_{k,p}^{(r)} - \tau(\sum_{i=1}^{n} \frac{\partial Q_i}{\partial a_{k,p}^{(r)}})$$

Le coefficient de proportionnalité τ est appelé coefficient d'apprentissage. Il peut être fixe, à déterminer par l'utilisateur, ou encore varier en cours d'exécution. Il paraît en effet intuitivement raisonnable que, grand au début pour aller plus vite, ce taux décroisse pour aboutir à un réglage plus fin au fur et à mesure que le système s'approche d'une solution. Bien d'autres méthodes d'optimisation ont été adaptées à l'apprentissage d'un réseau : méthodes du gradient avec second ordre utilisant une approximation itérative de la matrice Hessienne (algorithme BFGS).

— Régularisation

Dans les réseaux élémentaires, une option simple pour éviter le surapprentissage consiste à introduire une terme de pénalisation ou régularisation, dans le critère à optimiser. Celui-ci devient alors :

$$Q(\theta) + \gamma||\theta||_2^2$$

Plus la valeur du paramètre γ est importante et moins les poids

des entrées des neurones peuvent prendre des valeurs chaotiques contribuant ainsi à limiter les risques de sur-apprentissage.

4.1.2 Cartes de Kohonen

Ce réseau de neurones peut être considéré comme dynamique, des neurones peuvent être détruits et crées, le réseau n'a pas de taille fixe. Généralement ce réseau est appelé *carte de Kohonen*, en effet ce réseau est représenté à plat comme une grille rectangulaire à 1, 2, 3 ou 4 dimensions. Les applications sont multiples : sélection de données représentatives dans une grande base de cas, compression d'images, diagnostic de pannes, optimisation combinatoire (dont le fameux "voyageur de commerce", modélisation de la cartographie des aires visuelles.

4.2 Validation par quadrillage et scoring

Nous arrivons donc à la dernière possibilité retenue afin de tester la validité des méthodes. Le principe est de quadriller la zone du plan (dimension 2) occupée par les points du nuage, en la découpant en rectangles de même taille. L'idée est ensuite de dénombrer les éléments présents dans chacune des cases générées, et d'appliquer un réseau de neurones sur les nombres obtenus, dans le but de faire apprendre la répartition des points au réseau de neurones. Nous retiendrons les scores obtenus par les réseaux de neurones que nous utiliserons pour mettre en place un indicateur Ind défini plus tard.

Déroulement de la phase d'apprentissage en considérant une base initiale B :

i) Scinder la base de données initiale B en deux bases de même taille B_1 et B_2. Cela nous permettra de former deux valeurs finales à comparer.

ii) Quadriller la zone du plan $\mathcal{R}$=$[X_1^{min}, X_1^{max}] \times [X_2^{min}, X_2^{max}]$ occupée par le nuage de départ. Nous effectuons un quadrillage 10×10 donnant cent rectangles $\mathcal{R}_{i,j}$, $i, j \in \{1, ..., 10\}$ de même taille.

Pour B_1 :

iii) Déterminer $N_{i,j} = card(\mathcal{R}_{i,j})\ \forall i,j \in \{1,...,10\}$ et stocker dans un vecteur $Presence\_init$ la valeur 0 si $card(\mathcal{R}_{i,j}) = 0$, et 1 sinon. On choisira de stocker en choisissant d'abord i puis j.

iv) On considère pour compléter un vecteur $Absc\_init$ contenant l'abscisse du centre de chaque $\mathcal{R}_{i,j}$ et $Ordo\_init$ contenant son ordonnée. Ceci nous donne des coordonnées pour chaque case, chaque coordonnée se répétant dix fois du fait du quadrillage. Ces vecteurs sont ensuite normalisés.

v) Les trois vecteurs construits sont concaténés en un vecteur Z afin d'être utilisés pour l'apprentissage.

vi) Application d'un réseau de neurones sur Z.

vii) Prédiction de B_2, et stockage des scores obtenus dans une matrice $Scores$.

Ce procédé est réitéré sur $B_1^{'}$ et $B_2^{'}$, qui sont respectivement les tableaux répliqués de B_1 et B_2.

```
> quadrillage2(cercle_init,cases=10)
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0    1    2   14   11    5    6    2    0     0
 [2,]    2    3    1    1    0    0    3    8    1     0
 [3,]    2    7    0    0    0    0    0    0    7     5
 [4,]    5    1    0    0    0    0    0    0    0     5
 [5,]    5    0    0    0    0    0    0    0    0     9
 [6,]    3    0    0    0    0    0    0    0    0     7
 [7,]    8    0    0    0    0    0    0    0    3     6
 [8,]    2    4    0    0    0    0    0    1    5     2
 [9,]    0    5    5    3    0    0    2    6    4     0
[10,]    0    0    1    2    5    7   10    2    1     0
>
```

FIGURE 6 – Exemple : quadrillage de la zone correspondante à cercle_init

Sont définis ensuite les rapports Ind et $Ind^{'}$ choisis comme suit :

$$Ind = \frac{1}{\frac{n}{2}} \sum_{i,j=1}^{10} Scores_{i,j} \times N_{i,j}$$

et

$$Ind^{'} = \frac{1}{\frac{n}{2}} \sum_{i,j=1}^{10} Scores^{'}_{i,j} \times N^{'}_{i,j}$$

qui nous donnent un nouveau moyen de comparaison. Ces rapports peuvent être vus comme le nombre moyen de points dans chaque rectangles de la région $\mathcal{R}$.

Ces méthodes de validation nous ont principalement servi à nous conforter dans les différentes directions prises. Il est fortement conseillé pour l'avenir de construire de véritables test afin de mieux employer ces méthodes.

Par exemple pour la validation par étude des barycentres, il serait intéressant de générer un grand nombre de distributions $(b_i)_{i\in\{1,...1000\}}$ afin d'avoir un comportement moyen des barycentres provenant de la simulation, et de faire la comparaison avec la distribution $(b_i^{'})_{i\in\{1,...1000\}}$ en mettant en place un seuil d'acceptation.

Deuxième partie

Méthodes de réplication

1 Méthode Proportions

1.1 Formalisation mathématique en dimension p=1

Une première idée est d'observer la répartition des éléments $X[i]$, i=*1,...,n* de l'échantillon X que nous souhaitons répliquer. L'idée est de considérer un intervalle I dont la borne inférieure est la valeur minimum de l'échantillon ($I_{inf} = X_{min}$) et la borne supérieure la valeur maximum ($I_{sup} = X_{max}$). Puis nous divisons cet intervalle en une centaine de morceaux de même longueur $\frac{X_{max}-X_{min}}{100}$ (chaque borne ne correspond pas forcément à une valeur de l'échantillon). Enfin nous comptons le nombre d'éléments de l'échantillon présents dans chacun de ces intervalles. Cela nous donne des quantités p_1,p_2,...,p_{100} telles que $p_j = card(I_j)$, j=*1,...,100*, et $\sum_{j=1}^{100} p_j = n$.

Une fois ces données déterminées, nous simulons de nouvelles valeurs dans chacun des intervalles I_j en piochant $n \times p_j$ valeurs selon la loi uniforme $\mathcal{U}(I_j)$, j=*1,...,100*.

Cela nous donne un nouvel échantillon Y. Notons que cet échantillon tel que construit ne peut pas contenir de valeur dépassant les valeurs extrêmes de X. L'utilisateur pourra inclure une "marge" dans le programme originel selon l'objectif poursuivi.

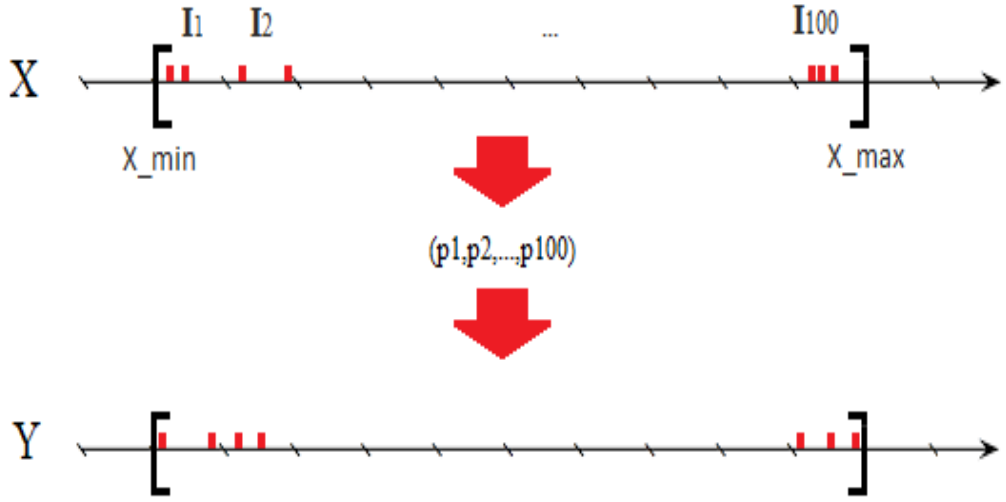

FIGURE 7 – Idée du procédé

Remarque : Nous pouvons former nos intervalles à partir des centiles de l'échantillon.

Soient $c_1, c_2, ..., c_{100}$ les centiles de X.

Pour k allant de 1 à 99, générer $Y[\frac{k}{100}n], ..., Y[\frac{k+1}{100}n]$ selon $\mathcal{U}[c_k, c_{k+1}]$.

Cette méthode se rapproche beaucoup de la première et ne sera donc pas explicitée d'avantage.

1.2 Formalisation mathématique en dimension p$\geqslant$ 2

En dimension supérieure, nous ne pouvons pas nous contenter de reproduire chaque colonnes indépendemment des autres. En effet le critère de corrélation intervient d'où la nécessité de conserver les mêmes liens entre les variables.

Pour exemple :

	Taille (m)	Poids (kg)
Individu 1	1.90	90
Individu 2	1.70	50

Ne doit pas devenir :

	Taille (m)	Poids (kg)
Individu 1	1.89	50.5
Individu 2	1.71	88

Pour se faire un premier raisonnement fut, pour p=2 variables répliquées Y_1 et Y_2 à partir de X_1 et X_2, de considérer un élément $Y_1[i]$ de Y_1, de déterminer l'élément $X_1[j]$ de X_1 le plus proche, puis considérer le $X_2[j]$ associé et déterminer l'élément $Y_2[k]$ le plus proche. On forme ensuite le couple ($Y_1[i]$,$Y_2[k]$). De façon plus formelle :

$\forall$ i $\in \{1, ..., n\}$:

i) Déterminer j^* réalisant $\min\limits_{j=1,...,n} |Y_1[i] - X_1[j]|$, puis

ii) déterminer k^* réalisant $\min\limits_{k=1,...,n} |X_2[j^*] - Y_2[k]|$, enfin

iii) constituer le couple ($Y_1[i]$,$Y_2[k^*]$).

Cette méthode a vite montré ses limites car lorsque le nuage de points a une structure particulière (un cercle par exemple) le nuage répliqué n'a plus rien à voir avec celui désiré.

D'où la nécessité de déterminer une méthode plus pertinente. Ce que l'on obtient en partant non pas des variables répliquées mais des variables de départ, en associant à chaque valeurs de la i$^{\text{eme}}$ ligne les valeurs les plus proches parmi celles obtenues. Ce procédé est non seulement plus performant mais également plus simple dans sa programmation. $\forall$ i $\in \{1, ..., n\}$:

i) Déterminer j^* réalisant $\min\limits_{j=1,...,n} |X_1[i] - Y_1[j]|$, et

ii) déterminer k^* réalisant $\min\limits_{k=1,...,n} |X_2[i] - Y_2[k]|$, puis

iii) constituer le couple ($Y_1[j^*]$,$Y_2[k^*]$).

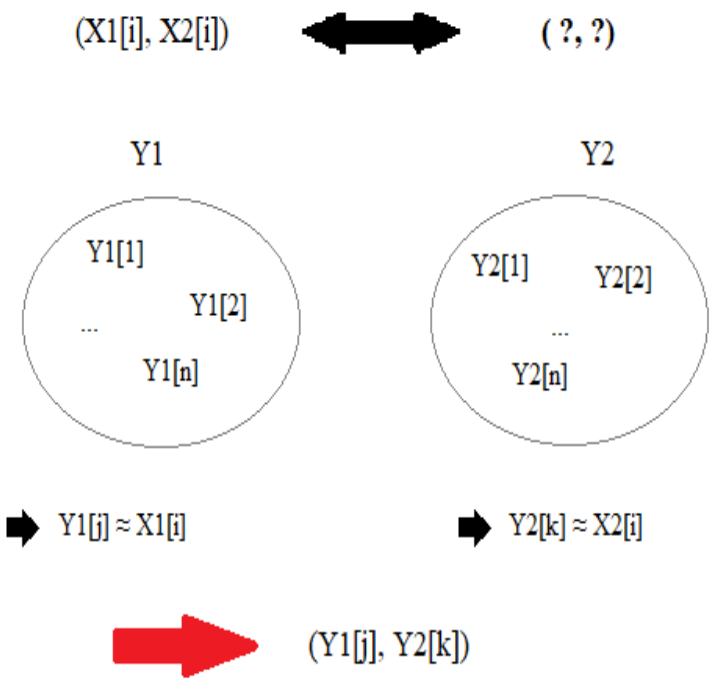

FIGURE 8 – Idée du procédé

Remarques :

- Afin d'améliorer la qualité de la réplication, nous simulons plus de valeurs que nécessaire et nous ne gardons ensuite que les plus intéressantes (proches de celles de départ).
- Le principe s'étend simplement par récurrence en dimensions supérieures.

1.3 Résultats obtenus et limites

Cette méthode a été appliquée sur les tableaux *Table_init* (loi gaussienne et loi exponentielle) et *cercle_init* (couronne de rayon 8 et 10). La réplication s'est faite en créant dix intervalles et deux fois plus de valeurs que nécessaire (puis un tri est fait afin de n'en conserver que n=200).

Comparaison des nuages de points

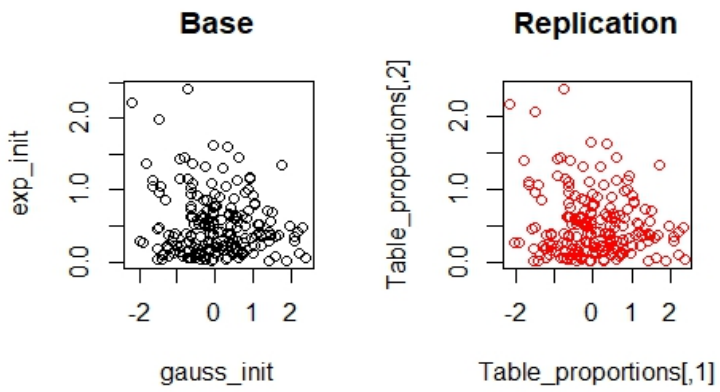

FIGURE 9 – Réplication de Table_init

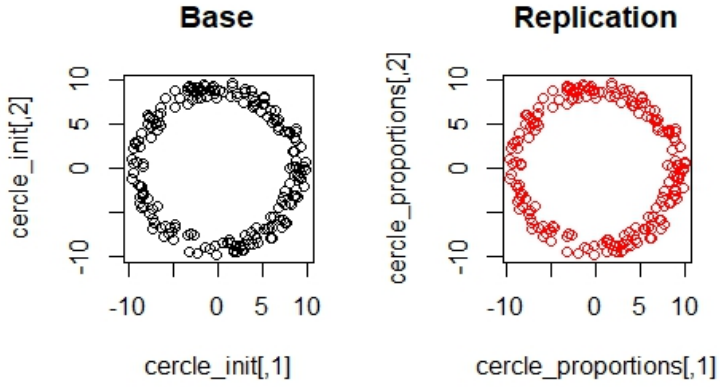

FIGURE 10 – Réplication de cercle_init

Les nuages de points sont très semblables à première vue. Notons que l'échelle et le le nombre de chiffres significatifs des valeurs expliquent la difficultés de discerner les écarts entre les nuages.

Comparaison des densités empiriques

Ces courbes sont obtenues par la fonction *density* sous R.

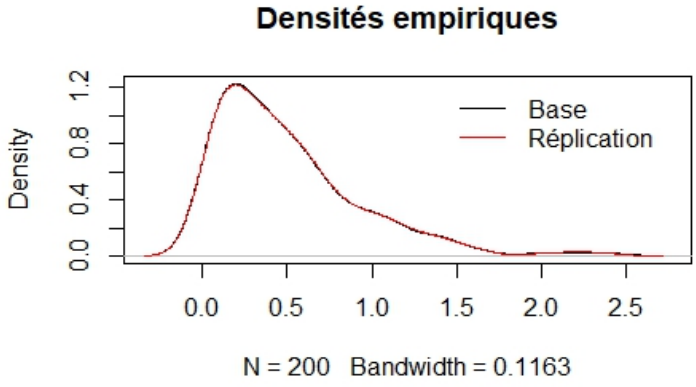

FIGURE 11 – Loi exponentielle de paramètre 2

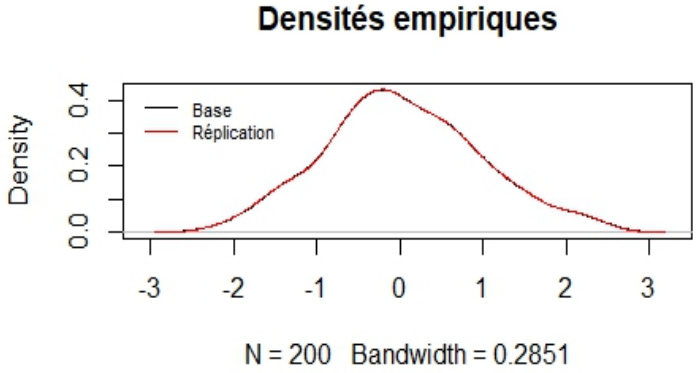

FIGURE 12 – Loi gaussienne centrée réduite

Nous constatons qu'il est difficile de distinguer la densité de la réplication (rouge) avec celle de base (noire).

Comparaison avec les réseaux de neurones

Notons Q le nombre de points moyen par cases de la zone quadrillée. Nous avons comparer les distributions de la quantité Q pour la méthode Proportion appliquée sur le tableau $Table\_init$.

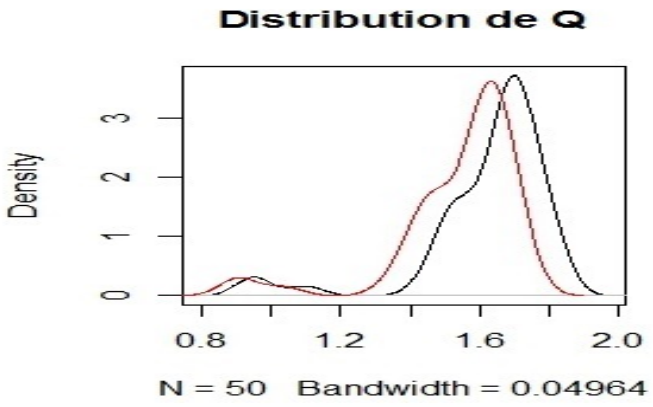

FIGURE 13 – Distributions de Q initiale (en noir) et répliquée (en rouge)

On remarque que les deux courbes se ressemblent bien que leur pics de densité respectifs soient légèrement translatés. Nous en comprenons que même si la structure du nuage de points est préservée, la zone de concentration des points est légèrement décalée. Nous en déduisons tous de même que notre méthode est bien validée.

Comparaison de la répartition des barycentres

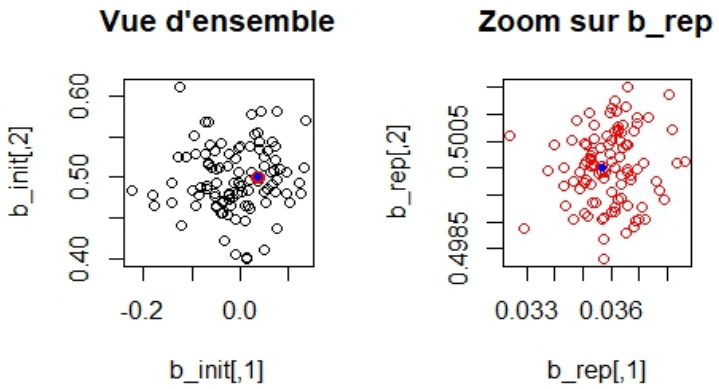

FIGURE 14 – Dispersion des barycentres pour Table_init

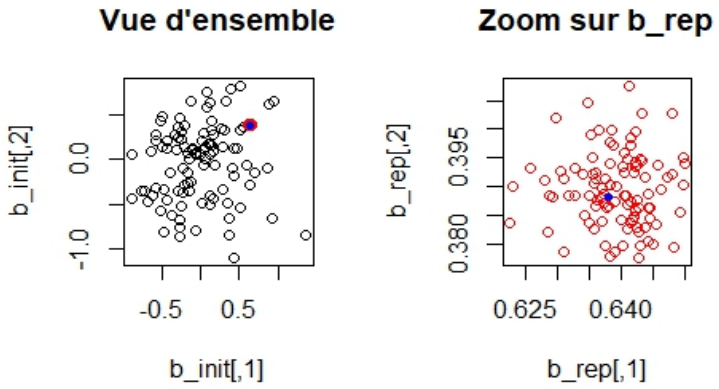

FIGURE 15 – Dispersion des barycentres pour cercle_init

La dispersion des barycentres pour les bases simulées comme la base *Table_init* (respectivement *cercle_init*) servant à la réplication est plus élevée que la dispersion des barycentres répliqués qui se trouvent concentrés autour du barycentre de *Table_init* (respectivement *cercle_init*). Ceci n'est pas surprenant puisque la réplication s'est faite à partir de cette base *Table_init* (respectivement *cercle_init*) mais nous conforte dans le bon fonctionnement de la méthode.

Il ne semble pas y avoir de véritable limite à cette méthode excepté sa trop grande précision qui joue contre elle. En effet il est difficile de distinguer les tables aussi bien sous forme de nuage qu'au niveau des valeurs répliquées. C'est pour cela que dans sa programmation le nombre d'intervalles est réglable, ainsi que le nombre de points simulés en plus (pour être sélectionnés ensuite) et l'arrondi de ses valeur. Il sera intéressant de s'en servir sur de réelles bases afin d'en connaître la portée.

2 Méthode Rot_theta

2.1 Principe de la méthode

Une autre méthode consiste à étudier directement le nuage de points du tableau de données, et de simuler nos nouveaux points dans une

région que nous pouvons nous-même définir. La région choisie ici est proche des points du nuage initial, afin de conserver la densité et la structure du nuage de points de départ.

Plus précisément, la méthode consiste à prendre un point du nuage et à déterminer la distance qui le sépare de son plus proche voisin.

On réitère ensuite le même processus pour tous les autres points du nuage ce qui nous mène à l'obtention d'une distribution de distances minimales que l'on note $(d_1,.....,d_n)$. Les nouveaux points sont construits sur des sphères centrées en les points initiaux et dont chaque rayon sera égal à la distance minimale d_i correspondante, suivant un angle aléatoire pris uniformément dans $[0, 2\pi]$. Plus formellement :

$\forall\ i^* \in \{1, ..., n\},\ X[i^*]=(X_1[i^*],X_2[i^*])$:

i) Calculer $d_{i^*} = \min_{i\in\{1,...,n\}\setminus\{i^*\}} \|X[i^*] - X[i]\|_2$,

ii) tirer un nombre aléatoire θ selon une loi $\mathcal{U}[0, 2\pi]$,

iii) poser le nouveau point $Y[i^*]=(X_1[i^*] + d_{i^*} \cos\theta\ ,\ X_2[i^*] + d_{i^*} \sin\theta)$.

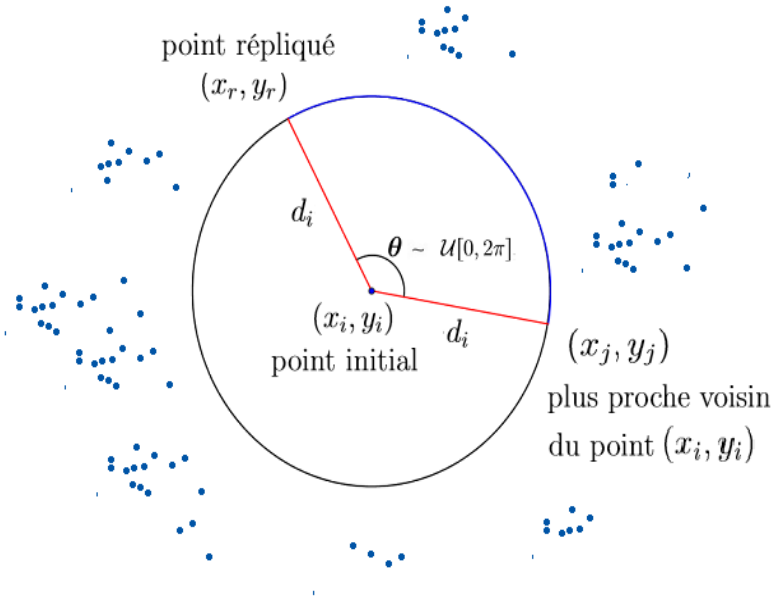

FIGURE 16 – Idée du procédé

Remarque : Cette méthode a été pensée en dimension 2, puis adaptée en dimension 3 grâce à l'expression des coordonnées sphériques pour la modélisation sur des sphères. Pour le cas des plus grandes dimensions il conviendrait d'utiliser la généralisation des coordonnées sphériques en dimension n pour établir la méthode ou plus modestement de l'appliquer aux colonnes deux à deux en dimension 2.

2.2 Résultats obtenus et limites

Cette méthode a été appliquée sur les tableaux *Table_ init* (loi gaussienne et loi exponentielle) et *cercle_ init* (couronne de rayon 8 et 10). La réplication s'est faite en créant dix intervalles et deux fois plus de valeurs que nécessaire (puis un tri est fait afin de n'en conserver que n=200).

Comparaison des nuages de points

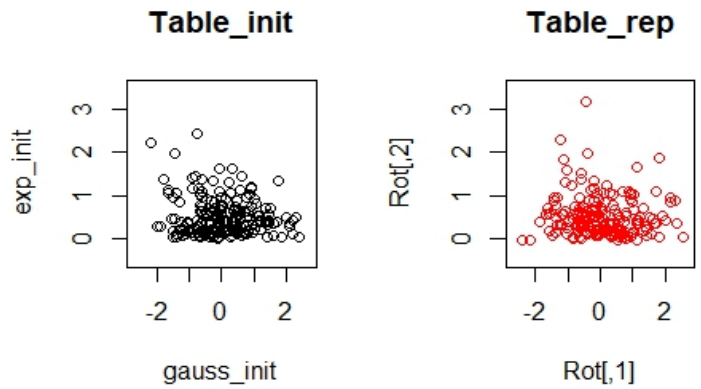

FIGURE 17 – Nuage initial (en noir) et répliqué (en rouge)

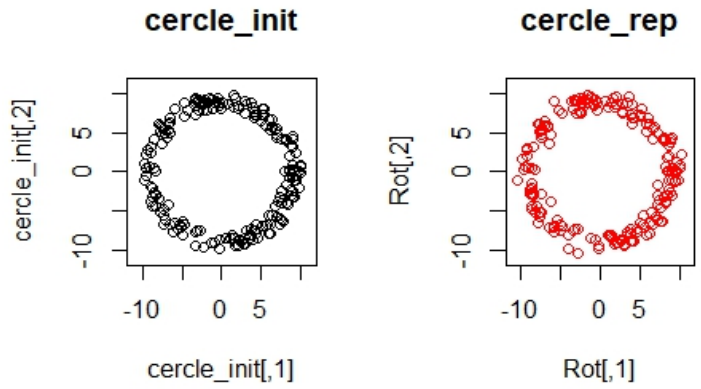

FIGURE 18 – Nuage initial (en noir) et répliqué (en rouge)

Cette fois les nuages sont plus faciles à distinguer, tout en restant similaires. Nous observons la même zone de concentration autour de (0 ;0) pour *Table_ init* et le cercle se trouve encore dans la zone de définition souhaitée.

Comparaison des densités empiriques

Ces courbes sont obtenues par la fonction *density* sous R.

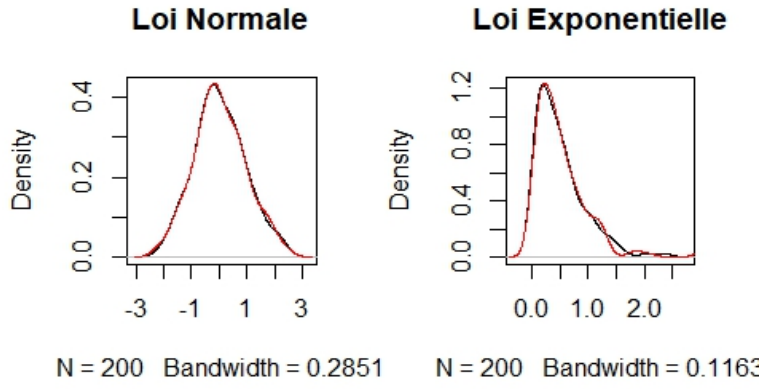

FIGURE 19 – Densités empiriques de base (noires) et après réplication (rouges)

Ici il est tout à fait possible de différentier les courbes malgré leur proximité.

Comparaison avec les réseaux de neurones

Notons Q le nombre de points moyen par case de la zone quadrillée. Nous avons illustré le graphique comparant les distribution de la quantité Q pour la méthode de réplication par simulation sur un cercle, appliquée sur la couronne de rayons 8 et 10.

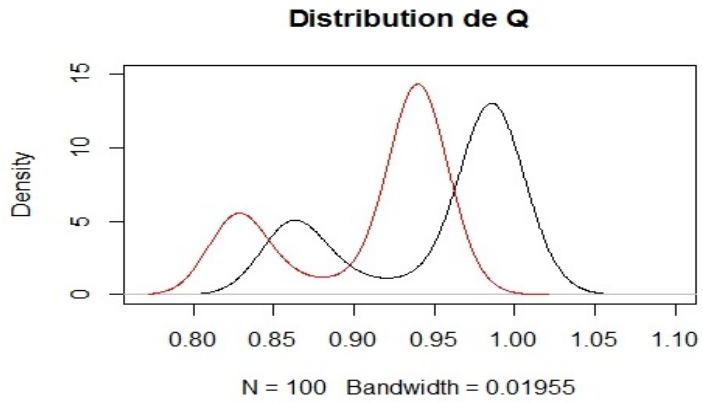

FIGURE 20 – Distributions de Q initiale (en noir) et répliquée (en rouge)

Nous remarquons ici que les deux courbes se ressemblent bien que leurs pics de densité respectifs sont légèrement translatés. Nous en déduisons que même si la structure circulaire du nuage de points est préservée, la zone de concentration des points ainsi légèrement est décalée. Ceci nous donne une observation plus fine qu'avec la seule comparaison des nuages de points.

Comparaison de la répartitions des barycentres

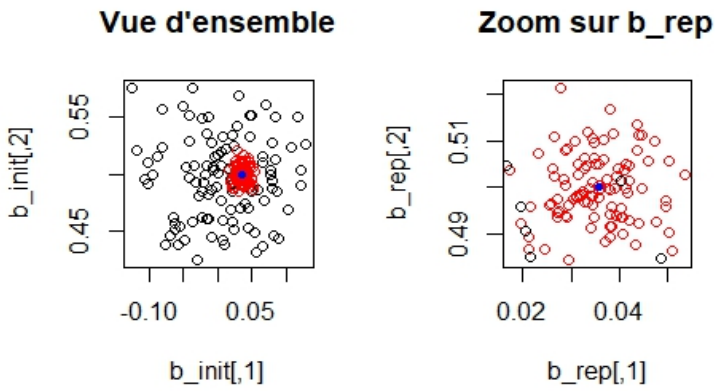

FIGURE 21 – Dispersion des barycentres pour Table_init

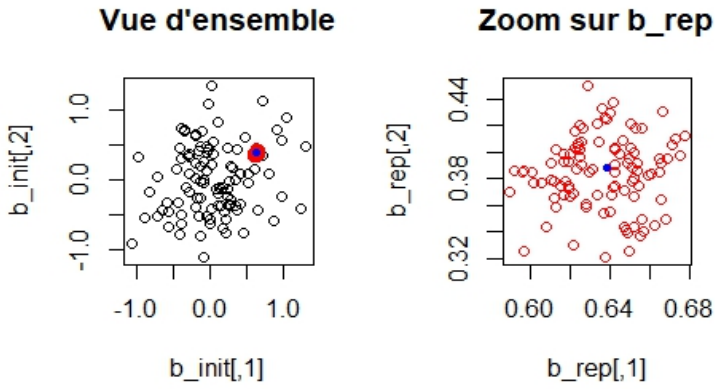

FIGURE 22 – Dispersion des barycentres pour cercle_init

En ce qui concerne les barycentres, nous pouvons remarquer encore une fois que cette méthode de réplication donne aussi des résultats trop précis, dans la mesure ou les barycentres répliqués sont trop proches du barycentre de base.

Si le programme décrivant cette méthode marche assez bien pour des échantillons de lois usuelles, et qu'il arrive aussi à préserver la structure circulaire du nuage de points de départ, cela ne veut pas pour autant dire que nous n'avons pas détecté de failles.

En effet, pour un échantillon dont l'étendue est très large et dont les valeurs dans cette étendue sont assez éparpillées, la méthode peut construire dans ce cas des nouveaux points sur des rayons qui seront trop grands (car les distances minimales seront devenues trop grandes du à l'éloignement mutuel des différents points du nuage).

C'est le cas par exemple pour un tableau de données dont une variable suit une loi uniforme sur un intervalle de longueur plus grande que le nombre d'individus (Figure ; 10 individus dans un tableau dont les deux lois sont uniformes sur $[0, 15000]$).

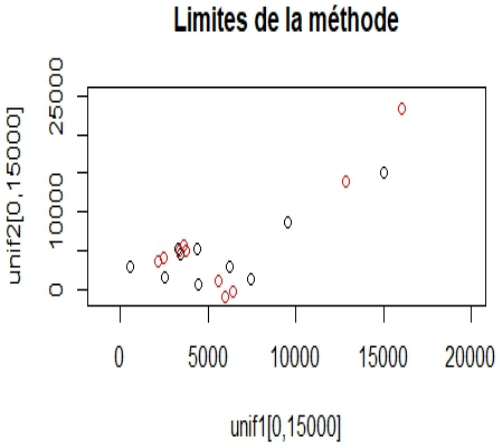

FIGURE 23 – En haut à droite : le point répliqué (16045,23270)

Ce problème peut avoir comme conséquence le débordement du support. En effet, la réplication d'un échantillon suivant une loi peut alors prendre des valeurs interdites, en dehors du support de la loi (Figure : la deuxième variable est une loi exponentielle).

```
> summary(Rot)
       V1                  V2
 Min.   :-2.41655   Min.   :-0.05331
 1st Qu.:-0.60565   1st Qu.: 0.16474
 Median :-0.04391   Median : 0.39598
 Mean   : 0.03703   Mean   : 0.49846
 3rd Qu.: 0.66438   3rd Qu.: 0.67229
 Max.   : 2.55751   Max.   : 3.18662
```

FIGURE 24 – La deuxième variable devrait être une loi exponentielle

Une solution peut être de répliquer des points tant que nous n'en avons pas le nombre souhaité dans le domaine de définition voulu et d'ignorer ceux mal positionnés.

3 Méthode SMOTE

3.1 Principe de la méthode

La méthode de réplication que nous allons décrire ci dessous emploie une technique d'échantillonnage désignée pour rééquilibrer des jeux de données dont les classes sont déséquilibrées, elle est connue sous le nom de méthode SMOTE (Synthetic Minority Oversampling Technique).

En effet : une approche simple à mettre en œuvre pour redresser la distribution asymétrique de données est de supprimer de manière aléatoire des individus de la classe majoritaire ou d'augmenter les individus de la classe minoritaire.

Dans le premier cas nous parlons de sous-échantillonnage et dans le deuxième de sur-échantillonnage. Les deux ont l'avantage d'être facile à déployer mais ils comportent aussi des inconvénients. Le sur-échantillonnage peut entraîner des problèmes d'overfitting. La méthode SMOTE est justement utilisée pour passer cet inconvénient.

L'idée de SMOTE est d'augmenter le rappel pour la classe minoritaire en générant des individus synthétiques. Ces individus artificiels sont créés à partir d'un choix aléatoire (selon le taux de sur-échantillonnage voulu) d'un certain nombre de voisins plus proches qui appartiennent à la même classe. Ensuite les individus synthétiques sont disséminés aléatoirement le long de la droite entre l'individu de la classe minoritaire et ses voisins sélectionnés. Ainsi, cette approche fait que la région de décision de la classe minoritaire est plus grande et général.

Dans notre contexte de réplication, nous allons modifier un peu le principe de la méthode SMOTE, dans la mesure où :

— chaque point, c'est-à-dire chaque individu, sera considéré comme une classe à part entière.

— un seul proche voisin sera généré pour chaque classe.

3.2 Formalisation mathématique

Soit $X = (X_1, ..., X_p)$ notre tableau initial que l'on veut répliquer, notons $V = (V_1, ..., V_p)$ le tableau dont la i^{eme} ligne représente les coordonnées du plus proche voisin du i^{eme} individu du tableau de départ X, on note aussi $(u_i)_{i\in\{1,...,n\}}$ des réalisations d'une variable aléatoire U suivant une loi uniforme sur $[0, 1]$ indépendante des colonnes de X et des colonnes de V.

Nous construisons alors le tableau répliqué Y dont les colonnes Y_j sont définies par :

$$Y_j[i] = X_j[i] + u_i(V_j[i] - X_j[i])$$

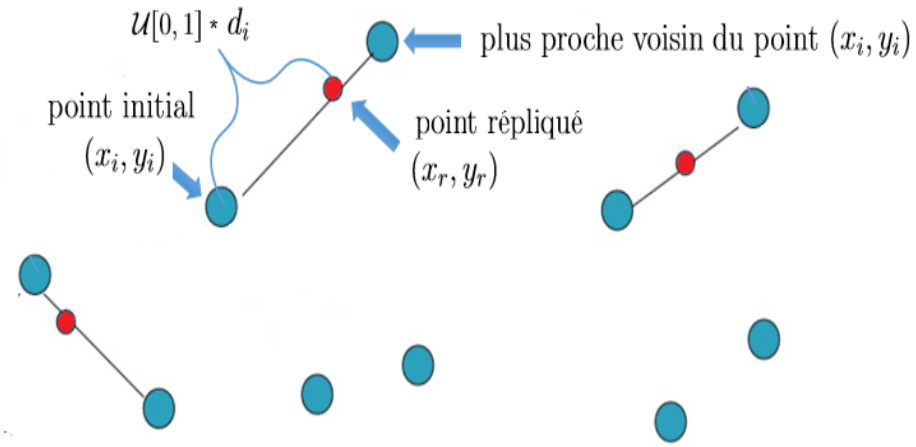

FIGURE 25 – Idée du procédé (dimension 2)

3.3 Résultats obtenus et limites

Cette méthode a été appliquée sur le tableau à deux colonnes et 200 lignes *cercle_ init* (couronne de rayon 8 et 10).

Comparaison des nuages de points

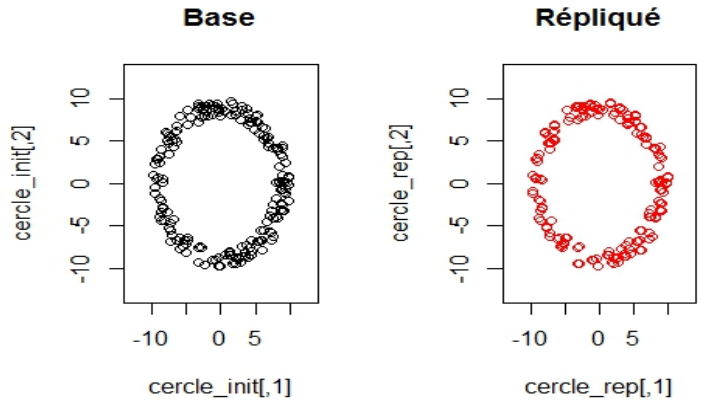

FIGURE 26 – Cercle initial (en noir) et répliqué (en rouge)

Nous voyons bien que la structure circulaire du nuage est bien répliquée avec cette méthode.

Comparaison des propriétés statistiques

```
> summary(cercle_init)
       V1                  V2
 Min.   :-9.720   Min.   :-9.8000
 1st Qu.:-5.165   1st Qu.:-6.2825
 Median : 1.480   Median : 0.2250
 Mean   : 0.638   Mean   : 0.3881
 3rd Qu.: 6.315   3rd Qu.: 6.9675
 Max.   : 9.930   Max.   : 9.7400
> summary(cercle_rep)
  cercle_rep1        cercle_rep2
 Min.   :-9.6621   Min.   :-9.7902
 1st Qu.:-5.2003   1st Qu.:-6.2550
 Median : 1.4569   Median : 0.3141
 Mean   : 0.6395   Mean   : 0.3914
 3rd Qu.: 6.2240   3rd Qu.: 6.9823
 Max.   : 9.9150   Max.   : 9.6223
> cor(cercle_init)
            [,1]       [,2]
[1,]  1.0000000 -0.0694076
[2,] -0.0694076  1.0000000
> cor(cercle_rep)
             cercle_rep1 cercle_rep2
cercle_rep1  1.00000000 -0.07177635
cercle_rep2 -0.07177635  1.00000000
```

FIGURE 27 – Comparaison des propriétés statistiques

Les minimums et maximums sont aussi bien répliqués, Nous ne constatons pas de débordements. De plus, les nouvelles corrélations sont similaires aux corrélations initiales.

Comparaison avec les réseaux de neurones

Notons Q le nombre de points moyen par cases de la zone quadrillée (qui a été expliqué précédemment). Nous avons comparé les distributions de la quantité Q pour la méthode de réplication SMOTE appliquée sur la couronne de rayons 8 et 10.

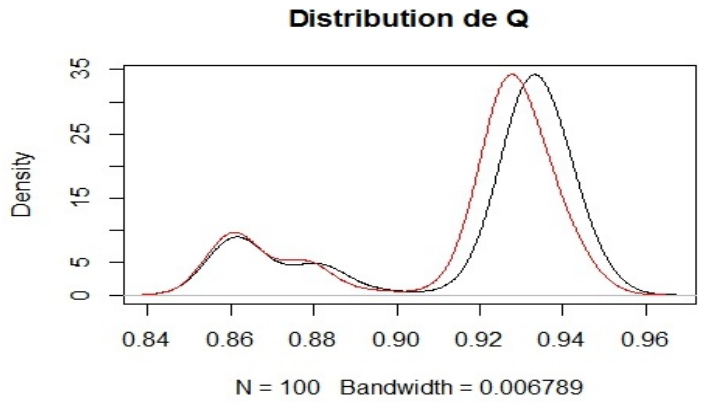

FIGURE 28 – Distributions de Q initiale (en noir) et répliquée (en rouge)

On remarque que les deux courbes se ressemblent bien que leur pics de densité respectifs sont légèrement translatés. On en déduit que même si la structure circulaire du nuage de points est préservée, la zone de concentration des points est légèrement décalée.

Comparaison de la répartition des barycentres

D'après le procédé décrit dans la partie "Méthodes de validation", nous avons généré les barycentres des bases simulées selon la base d'origine, puis les barycentres répliqués à partir des bases répliquées de la base d'origine (avec la méthode de réplication SMOTE).

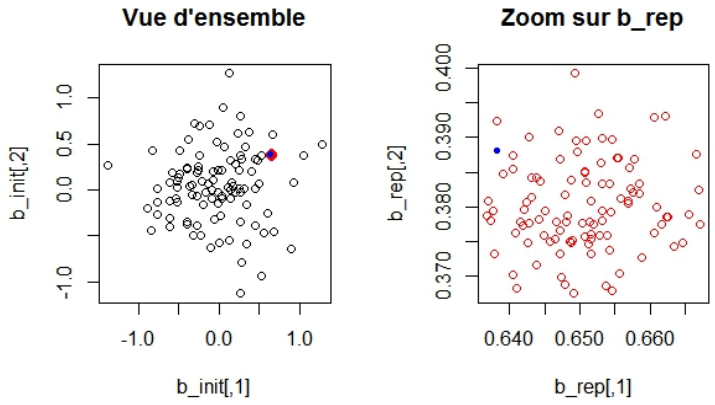

FIGURE 29 – Comparaison des nuages de points des barycentres

Nous voyons nettement au niveau de l'échelle des deux graphes que les barycentres répliqués sont très concentrés autour du barycentre du tableau initial, et même trop concentrés par rapport aux barycentres simulés.

Nous en déduisons que la méthode de réplication SMOTE crée des points globalement trop prés de ceux de la base initiale, et donc qu'elle peut être considérée comme trop précise.

Sur nos exemples traités nous n'avons pas réussi à expliciter d'éventuelles faiblesses de la méthode, hormis le fait qu'elle soit trop précise. En effet, elle ressemble à la méthode de réplication par simulation sur un cercle, sauf qu'on simule chaque point répliqué entre deux points initiaux, sans prendre en compte une rotation selon un angle aléatoire, réduisant ainsi considérablement le risque de débordement du support.

Pour des échantillons dont la forme du nuage est assez simple (par exemple : blocs de points regroupés) ou bien pour ceux avec des nuages en couronne, la méthode marche assez bien que ce soit en termes de marginales restituées ou de corrélations restituées.

4 Méthode paramétrique

Ce paragraphe traite de la réplication d'un tableau de données dans le cas particulier où on peut déterminer explicitement la loi de chaque marginale de ce tableau.

4.1 Principe de la méthode

Nous disposons d'un tableau X dont les colonnes sont notées X_j, $j \in \{1, ..., p\}$, chacune est caractérisée par une loi notée F_j a priori inconnue. On dispose également de la matrice de corrélation de ce tableau qui sera notée $\sum_X$.

Afin de répliquer X ainsi que ses caractéristiques, nous allons utiliser une méthode se basant sur des hypothèses et suppositions paramétriques.

La première étape consiste à supposer que nous avons à disposition une famille de lois $P_{i,\theta}$ à ajuster sur chaque variable X_j. Avec les différents tests d'ajustement, on peut facilement obtenir une p-value ou un graphique QQ-plot, nous affirmant avec un risque α si oui ou non, la loi peut s'ajuster sur l'échantillon décrivant la variable X_j.

Cette démarche serait ensuite à répéter pour toutes les colonnes X_j de X, ce qui nous donne à la fin plusieurs lois ajustées. Et la réplication va simplement consister à resimuler chaque marginales du tableau de données X selon les lois ajustées correspondantes.

Cependant la procédure ne s'arrête pas là, car même si les marginales ont été ajustées puis simulées une à une, il reste néanmoins un autre critère, qui n'est pas vérifié avec cette première étape qu'on vient de décrire ici, à savoir les corrélations.

Ainsi, pour la deuxième étape de notre méthode, nous sommes confrontés au problème suivant : "Comment simuler un tableau $X = (X_1, X_2, ..., X_p)$ de lois marginales respectives $(F_1, F_2, ..., F_p)$ et de matrice de corrélation $\sum_X$?"

Il existe dans la littérature une méthode qui répond au problème : la méthode NORTA (Normal to Anything). Cette méthode, comme nous

le verrons dans la suite, nécessite de faire une hypothèse supplémentaire sur la structure de dépendance de notre tableau.

4.2 Rappels sur les tests statistiques d'adéquation

Dans cette sous-section, nous faisons des rappels sur les divers tests d'adéquation statistiques pour déterminer des lois théoriques qui conviennent le plus aux marginales du tableau de données.

4.2.1 Test d'adéquation de Kolgomorov-Smirnov

Le test de Kolmogorov-Smirnov est un test qui compare la distribution observée d'un échantillon statistique à une distribution théorique. On l'utilise de préférence au test d'adéquation du khi-deux lorsque le caractère observé peut prendre des valeurs continues. Il est basé sur la comparaison des fonctions de répartition.

Données : on dispose de n observations $(x_1, ..., x_n)$ d'une variable aléatoire X.

Hypothèse testée : La fonction de répartition de X, notée F, est égale à F_0, avec risque d'erreur a.

Déroulement du test :

1) On ordonne les valeurs observées,
2) On pose $F(x_1) = \frac{1}{n}, F(x_2) = \frac{2}{n}, ..., F(x_n) = 1$ ce qui définit la fonction de répartition de F en escalier,
3) On calcule $K = \sup_{x \in R} |F(x) - F_0(x)|$,
4) On lit la valeur critique D_n dans la table de la loi de Kolmogorov-Smirnov. Si $K < D_n$, on accepte l'hypothèse, sinon, on la rejette.

4.2.2 Test d'adéquation du khi-deux

Le but de ce test est de comparer une distribution théorique d'un caractère à une distribution observée. Pour cela, le caractère doit prendre un nombre fini de valeurs, ou bien ces valeurs doivent être rangées en un nombre fini de classes.

Données : On dispose d'un caractère A dont les valeurs possibles sont réparties en k classes $A_1, ..., A_k$. La probabilité théorique dans chacune des classes est notée $p_1, ..., p_k$. On dispose également de n observations, qui donnent un effectif n_1 pour la classe A_1,..., n_k pour la classe A_k, avec $n_1 + ... + n_k = n$.

Hypothèse testée : La distribution observée est conforme à la distribution théorique avec un risque d'erreur a.

Déroulement du test :

1) On calcule les effectifs théoriques np_j (conditions d'application : il faut que $np_j > 5$ pour tout j=1,...,k),
2) On calcule la valeur observée K de la variable de test,
3) On cherche la valeur critique D_n dans la table de la loi du khi-deux à $k - 1$ degrés de liberté,
4) On lit la valeur critique D_n dans la table de la loi du khi-deux. Si $K < D_n$, on accepte l'hypothèse, sinon, on la rejette.

4.2.3 Test graphique : QQ-plot

Le QQ-plot est un outil graphique permettant de valider l'ajustement d'une distribution donnée à une loi théorique. Son nom provient du fait que l'on compare la position de certains quantiles dans la population observée avec la position des quantiles de la population théorique.

Si les points obtenus sont alignés sur la première bissectrice, cela veut dire que la distribution observée suit probablement la loi de distribution théorique.

4.3 Rappel sur les copules

4.3.1 Définition et caractérisation

Une copule C est une fonction de répartition, définie sur $[0,1]^d, d \in \mathbf{N}$, dont les marginales sont des lois uniformes sur $[0,1]$.

Une fonction C définie sur $[0,1]^d$ à valeur dans $[0,1]$, est une copule si et seulement si :

i) $C(u_1, ..., u_d) = 0$, pour tout i tel que $u_i = 0$,

ii) $C(1, ..., 1, u_i, 1, ..., 1) = u_i, i \in \{1, ..., d\}$,

iii) $u_i \mapsto C(u_1, .., u_i, .., u_d)$ est croissante pour tout $i \in \{1, ..., d\}$.

4.3.2 Théorèmes de Sklar

Théorème 1 : *Soit F une fonction de répartition multivariée avec $(F_1, F_2, ..., F_p)$ les fonctions de répartition continues des lois marginales, alors il existe une unique fonction C définie sur $[0, 1]^n$ à valeur dans $[0, 1]$ de sorte que : $F(x_1, x_2, ..., x_p) = C(F_1(x_1), ..., F_p(x_p))$.*

Théorème 2 : *Soit C une copule et $(F_1, F_2, ..., F_p)$ des fonctions de répartitions univariées, alors : $F(x_1, x_2, ..., x_p) = C(F_1(x_1), ..., F_n(x_p))$ est une fonction de répartition multivariée dont les marginales sont $(F_1, F_2, ..., F_p)$.*

Interprétation : La connaissance d'une copule et des lois marginales des composantes du vecteur X, est équivalente à la connaissance de la loi jointe F du vecteur X.

4.3.3 Simulation de vecteurs aléatoires de copule C

On souhaite simuler les réalisations d'un vecteur aléatoire $X = (X_1, ..., X_p)$ de copule C, dont les fonctions de répartitions des marginales sont respectivement $(F_1, F_2, ..., F_p)$.

Étape 1 : Simuler un vecteur uniforme sur $[0, 1]^n$ dont la fonction de répartition est C.

Étape 2 : Appliquer les transformations croissantes inverses $F^{-1}_{X_j}$ à chacune des marginales X_j.

Les copules sont bien adaptées à la construction de distributions conjointes possédant une structure de dépendance donnée.

En effet, dans certaines applications, on ne cherche pas à simuler un vecteur selon sa loi jointe (car elle est soit inconnue, soit trop difficile à simuler), mais plutôt à simuler seulement ses composantes selon leurs lois marginales respectives, tout en respectant un niveau donné de liaison entre ces composantes représentée par une copule C. En pratique,

cette copule C inconnue sera remplacée par l'un des modèles de copules présentés dans la littérature statistique, qui est en relation avec l'objet modélisé et qui capture le mieux la structure de dépendance entre les composantes de X.

Voici deux modèles de copules parmi les plus utilisés :

Copule gaussienne : $C(x_1, ..., x_p) = F_N(\phi(x_1), ..., \phi(x_p))$

où F_N est la fonction de répartition de la loi normale standard multivariée et ϕ la fonction de répartition de la loi normale standard unidimensionnelle.

Copule archimédienne : $C(x_1, ..., x_p) = \phi^{-1}(\phi(x_1) + ... + \phi(x_p))$

où ϕ est une fonction définie sur $[0, 1]$ à valeurs dans $\mathbf{R}$ strictement décroissante et convexe.

4.4 Méthode NORTA (Normal to Anything)

Nous voulons ici répliquer un tableau sous forme de vecteur $X = (X_1, ..., X_p)$ avec l'approche paramétrique décrite plus haut.

Comme nous connaissons les lois marginales et la matrice de corrélation $\sum_X$ de ce vecteur X, nous allons utiliser la méthode NORTA pour simuler une nouvelle version de X, en respectant les corrélations fixées $\sum_X$.

4.4.1 Formalisation mathématique

On suppose que les X_j sont centrées et réduites.

L'idée de cette méthode est de considérer que la structure de dépendance du vecteur X est décrite par une copule normale, c'est à dire que l'on considère que la loi conjointe de X est décrite par :

$$X = (F_{X_1}^{-1}(\phi(Y_1)), ..., F_{X_p}^{-1}(\phi(Y_p))$$

où $Y = (Y_1, ..., Y_p)$ est un vecteur gaussien centré de matrice de covariance $\sum_Y$ et ϕ est la fonction de répartition de la loi normale centrée

réduite. Chacune des variables Y_i est également réduite, de sorte que $\sum_Y$ est en fait une matrice de corrélation avec des uns sur la diagonale.

On doit alors choisir $\sum_Y$ de sorte qu'en simulant X à partir de Y, on retrouve les corrélations $\sum_X$.

En notant $p_{i,j} = \frac{E(X_iX_j)-E(X_i)E(X_j)}{\sigma_{X_i}.\sigma_{X_j}} = E(X_iX_j)$ le terme générique de la matrice $\sum_X$, et $r_{i,j}$ le terme générique de la matrice $\sum_Y$, on obtient l'égalité suivante :

$$p_{i,j} = E(X_iX_j) = E(\ F_{X_i}^{-1}(\phi(Y_i))\ F_{X_j}^{-1}(\phi(Y_j))\ )$$

$$= \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} F_{X_i}^{-1}(\phi(y_i))\ F_{X_j}^{-1}(\phi(y_j)) f(y_i, y_j, r_{i,j})\, dy_i\, dy_j \qquad (1)$$

où $f(x,y,r) = \frac{1}{2\pi(1-r)} \exp(\frac{x^2+y^2-2rxy}{2(1-r)})$ est la densité gaussienne bivariée.

Un résultat théorique démontré par Kruskal (1958) énonce que :

Si $p_{i,j}$ est remplacé par le terme générique de la matrice de corrélation du rang $E(F_{X_i}(X_i)F_{X_j}(X_j))$ du vecteur X, alors l'équation (1) se simplifie pour donner :

$$r_{i,j} = 2\sin(\frac{\pi p_{i,j}}{6})$$

Ainsi, avec cette expression analytique, nous obtenons les termes de la matrice de corrélation $\sum_Y$ avec laquelle nous simulons le vecteur gaussien Y avant d'appliquer les transformations inverses sur chacune des marginales (comme mentionné plus haut). Le vecteur résultant sera le vecteur répliqué.

4.5 Résultats obtenus et limites

Comparaison des nuages de points

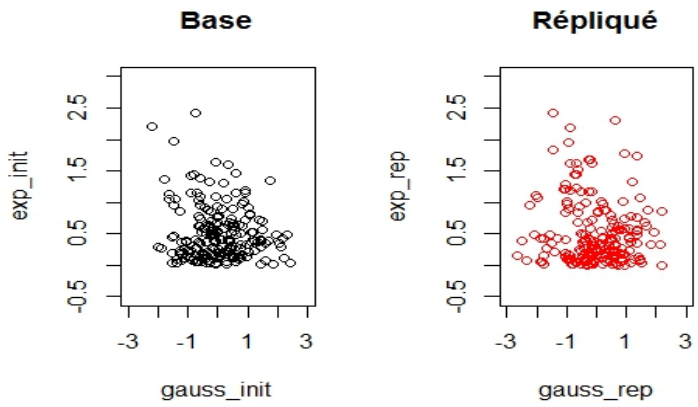

FIGURE 30 – Nuage de points initial (en noir) et répliqué (en rouge)

Les nuages de points sont assez similaires. Ici, les points répliqués ne franchissent pas la limite du support semi-borné de la loi exponentielle.

Comparaison des densités empiriques

Ensuite, nous allons comparer la densité estimée de l'échantillon de départ à celle de l'échantillon répliqué. Ces courbes sont obtenues par la fonction *density* sous R.

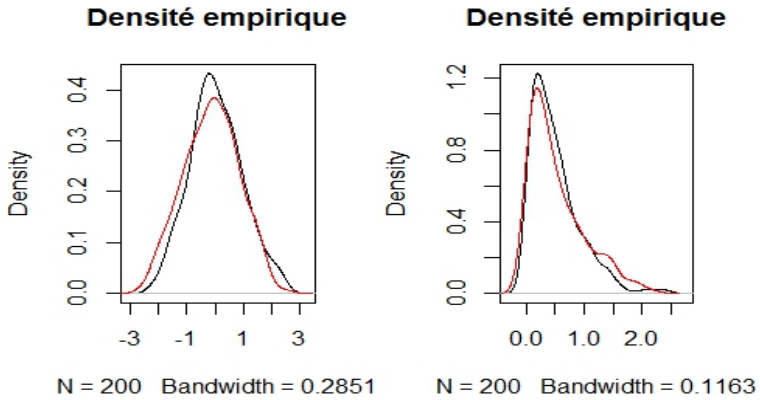

FIGURE 31 – Densités empiriques initiale (en noir) et répliquée (en rouge)

On remarque que nos densités empiriques sont très proches, mais pas identiques. Cela parait évident, car les marginales sont bien de même lois.

Comparaison avec les réseaux de neurones

Notons Q le nombre de points moyen par cases de la zone quadrillée (qui a été expliqué précédemment). Nous avons comparé les distributions de la quantité Q pour la méthode de réplication paramétrique appliquée sur le tableau $Table\_init$.

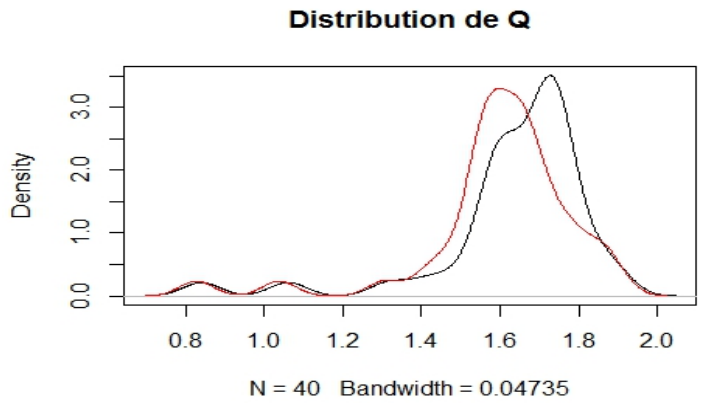

FIGURE 32 – Distributions de Q initiale (en noir) et répliquée (en rouge)

On remarque que les deux courbes se ressemblent bien que leur pics de densité respectifs sont légèrement translatés. On en déduit que même si la structure du nuage de points est préservée, la zone de concentration des points est décalée.

Comparaison de la répartition des barycentres

D'après le procédé décrit dans la partie "Méthodes de validation", nous avons généré les barycentres simulés à partir des bases simulées selon la base d'origine, puis les barycentres répliqués à partir des bases répliquées de la base d'origine (avec la méthode de réplication paramétrique).

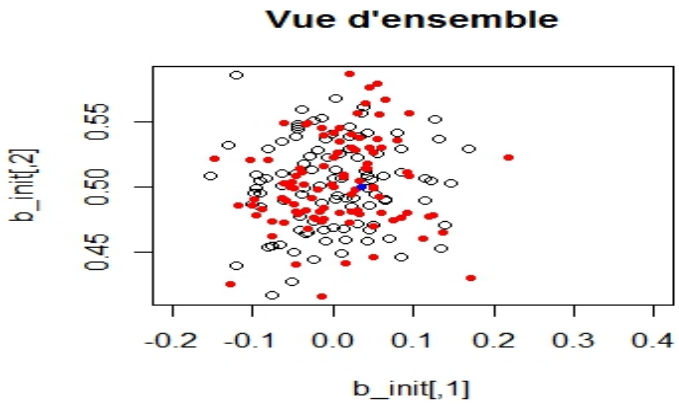

FIGURE 33 – Comparaison des distributions des barycentres

Nous voyons bien que les barycentres répliqués sont dispersés autour du barycentre de base et nous constatons la même chose pour les barycentres simulés. Nous en déduisons qu'il est difficile de distinguer les deux sortes de barycentres, et donc que notre méthode arrive à créer des points assez proches des points initiaux sans pour autant être identifiables.

Par définition, cette méthode est une restriction à des cas particuliers où des hypothèses bien précises sont faites.

Or ces hypothèses ne sont pas toujours vérifiées, et même quand elles le sont, elles peuvent comporter des inconvénients liés à la modélisation.

Tout d'abord, le contexte paramétrique dans lequel nous nous plaçons a ses limites, nous pouvons disposer a priori de n'importe quel jeu de données et nous ne savons pas si les tests d'ajustement aboutiront forcément à des lois connues.

De plus, un grand nombre de variables supposera un nombre croissant de tests à effectuer sur chacune d'entre elles, nous pouvons redouter un temps de calcul assez long.

Ensuite, modéliser la structure de dépendance d'un jeu de données par une copule gaussienne a ses avantages (les expressions analytiques vues plus haut, sont assez simples), mais aussi ses inconvénients. Par

exemple, la copule gaussienne n'est pas adaptée pour la modélisation de la dépendance entre des événements extrêmes. Elle est aussi moins bien adaptée en assurance car elle s'applique à des distributions symétriques.

Un moyen d'amélioration pourrait être de considérer des autres modèles de copules mieux adaptés (copule archimédienne, copule de Gumbel,....). Les copules archimédiennes ont le grand avantage de décrire des structures de dépendance très diverses dont notamment les dépendances dites asymétriques, où les coefficients de queue inférieure et de queue supérieure diffèrent. Nous n'avons pas eu le temps de travailler sur cette idée.

Enfin, dès que le tableau de données excède un nombre de variables supérieur à 2, il se peut que l'équation (1) fournisse une matrice de corrélation $\sum_Y$ non semi-définie-positive. Par conséquent, les corrélations obtenues peuvent être différentes et l'efficacité de la méthode n'est plus garantie.

```
> summary(Table_init)
   gauss_init          exp_init
 Min.   :-2.2100   Min.   :0.0100
 1st Qu.:-0.6125   1st Qu.:0.1755
 Median :-0.0450   Median :0.3835
 Mean   : 0.0357   Mean   :0.5000
 3rd Qu.: 0.6125   3rd Qu.:0.6753
 Max.   : 2.4000   Max.   :2.4160
> summary(Table_rep)
   gauss_rep           exp_rep
 Min.   :-3.08690   Min.   :0.00434
 1st Qu.:-0.81475   1st Qu.:0.14483
 Median : 0.19298   Median :0.31570
 Mean   : 0.07505   Mean   :0.45991
 3rd Qu.: 0.86491   3rd Qu.:0.60969
 Max.   : 3.25831   Max.   :2.72730
> cor(Table_init)
            gauss_init   exp_init
gauss_init  1.0000000 -0.1349164
exp_init   -0.1349164  1.0000000
> cor(Table_rep)
           gauss_rep   exp_rep
gauss_rep 1.0000000 0.1057787
exp_rep   0.1057787 1.0000000
>
```

FIGURE 34 – Comparaison des propriétés statistiques

En effet, pour notre exemple précédent, la forme du nuage est conservée dans son ensemble, mais les corrélations entre nos deux variables différent et sont même de signe opposés $r_{init} = -0.1349164$ alors que $r_{rep} = 0.1057787$.

Un moyen d'amélioration pourrait être de chercher la matrice semi-définie-positive la plus proche possible de $\sum_Y$ (au sens d'une norme).

Nous concluons que cette méthode arrive à bien répliquer les marginales du tableau, mais pas les corrélations, et qu'elle ne peut répliquer des vecteurs uniquement selon les lois théoriques ajustées. Ainsi pour notre nuage de points circulaire, la méthode ne peut être appliquée, en raison de cette structure circulaire complexe qui ne peut être appréhendé en approche paramétrique. Sous l'attente d'améliorations, cette méthode n'est donc pas utilisable pour l'instant.

5 Méthode par estimation des noyaux gaussiens

Nous avons vu précédemment que se placer dans un contexte paramétrique nous permettait de générer de nouveaux échantillons en resimulant selon les lois ajustées aux marginales du tableau de données, notamment avec des méthodes de simulation comme l'inverse de la fonction de répartition, ou la méthode de rejet-acceptation qui se base sur la connaissance explicite d'une fonction de densité f dépendant de la loi et des paramétres ajustés.

Comme nous ne connaissons pas f pour chaque marginale, cette approche semble difficilement concevable en non-paramétrique. C'est pourquoi, il nous a été conseillé une méthode utilisant un estimateur non paramétrique de la densité, en particulier un estimateur à noyaux gaussiens. Nous allons présenter le principe en dimension 1, puis en dimensions supérieures.

5.1 Rappels sur les estimateurs empiriques de la densité

5.1.1 Fonctions noyaux

Un noyau est une fonction utilisée dans les techniques d'estimation non-paramétrique. Un noyau $\mathtt{K}$ est une fonction intégrable à valeurs réelles, qui vérifie les deux propriétés suivantes :

- $\int_{-\infty}^{+\infty} \mathtt{K}(x)dx = 1$
- $\mathtt{K}(u) \geqslant 0$

Plusieurs types de noyaux sont couramment utilisés : uniforme, triangle, epanechnikov, quadratique, cubique, gaussien, et circulaire. Nous utiliserons pour la suite le noyau gaussien.

5.1.2 Estimateur à noyaux de la densité

Nous allons estimer la densité inconnue de notre échantillon par son estimateur à noyaux, qui a pour expression :

$\hat{f}_n(x) = \frac{1}{nh} \sum_{k=1}^{n} \mathtt{K}(\frac{x-xi}{h})$ avec $\mathtt{K}(u) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-\frac{u^2}{2}}$

Soient $(X_1, ..., X_n)$ des variables aléatoires $i.i.d$, et h un réel positif.

alors $\hat{f}_n(x) = \frac{1}{nh} \sum_{k=1}^{n} \mathtt{K}(\frac{x-X_i}{h})$ est une variable aléatoire vérifiant :

Quand h tend vers 0 et nh tend vers $+\infty$:

$\hat{f}_n(x)$ converge vers $f_{X_1}(x)$

$\sqrt{nh}(\hat{f}_n(x) - f(x))$ converge vers $\mathcal{N}(0, f_{X_1}(x)(\int_{-\infty}^{+\infty} \mathtt{K}^2(x)dx))$ p.s.

5.1.3 Choix du paramètre de lissage h

Le paramètre de lissage a un impact très significatif sur les résultats.

Une valeur trop faible pour h produit des résultats sans intérêt, la densité étant non nulle seulement à proximité immédiate des observations, alors qu'une valeur trop élevée produit un lissage excessif avec perte des détails.

Le choix du paramètre h est donc critique pour obtenir des résultats satisfaisants. Parmi les méthodes de choix de la valeur optimale de h, nous ne présentons que la méthode : Par minimisation de la MSE ou de la MISE.

— Par minimisation de la MSE ou de la MISE

Définissons la MSE (Mean Square Error) et la MISE (Mean Integrated Square Error).

$\mathrm{MSE}(x, n, h) = Var(\hat{f}_n(x)) + Biais^2(\hat{f}_n(x)) = \mathbb{E}((\hat{f}_n(x) - f(x))^2)$

$\text{MISE}(n,h) = \int_{-\infty}^{+\infty} \text{MSE}(x,n,h)dx$

On choisira alors, la valeur $h = h_{opt}$ tel que :

$h_{opt} = argmin_h(MSE(x,n,h))$ ou $h_{opt} = argmin_h(MISE(x,n,h))$

5.1.4 Choix du noyau

Pour comparer deux noyaux symétriques, on peut calculer le rapport de leur MISE.

$\text{eff}(\mathtt{K}_1, \mathtt{K}_2) = \frac{MISE(\mathtt{K}_1,n,h)}{MISE(\mathtt{K}_2,n,h)}$

Le noyau $\mathtt{K}_1$ est choisi au détriment du noyau $\mathtt{K}_2$ si la quantité $\text{eff}(\mathtt{K}_1, \mathtt{K}_2)$ est $\leqslant 1$.

Le choix de la fonction noyau a cependant peu d'impact sur la qualité d'estimation (à condition que ce noyau soit lisse ou symétrique).

5.2 Réplication et Formalisation mathématique en dimension 1

5.2.1 Rappel sur la méthode de rejet

Théorème :

Soit f une fonction de densité de probabilité. On suppose qu'il existe une densité de probabilité g telle que : $\ni M \geqslant 0, \forall x \in \mathbf{R}\, f(x) \leqslant M.g(x)$

Soit Z suivant la loi de densité g, et $Y \sim \mathcal{U}\,[0, M.g(Z)]$. Alors, la variable aléatoire X telle que $\mathcal{L}(X) = \mathcal{L}(Z|Y \leqslant f(Z))$ suit une loi de densité f.

— Interprétation,

Soient f et g deux densités de probabilités. D'après la théorie, si la densité f est majorée par Mg sur un intervalle $[a,b]$ contenant son support, nous pouvons dans ce cas génèrer des variables uniformément distribuées dans l'aire $[a,b] \times [0, Mg]$. Ensuite nous observons les abscisses des points situés sous le graphe de f ; Ces points vont constituer un f-échantillon (échantillon dont la loi est de densité f)

Ainsi, l'algorithme de rejet consiste donc à itérer les étapes :

— tirer Y de densité g,
— tirer U de loi uniforme sur $[0, 1]$, indépendante de Y,

Tant que U$\geqslant f(Y)/Mg(Y)$
— Accepter Y comme un tirage aléatoire de densité de probabilité

Dans un algorithme de rejet, on sait que le nombre de tirages nécessaires à chaque étape suit une loi géométrique de paramètre $\frac{1}{M}$. Le nombre moyen de tirages est alors de l'ordre de M. Ainsi, il convient de choisir M le plus petit possible.

5.2.2 Formalisation mathématique

Essayons maintenant de simuler un f_n-échantillon en dimension 1 avec la méthode de rejet.

La méthode de rejet-acceptation s'applique généralement pour des densités à support compact. Il existe cependant une généralisation pour quelques lois à support dans $\mathbf{R}$ ou $\mathbf{R}_+$. Comme nous avons choisi un noyau gaussien, cela revient à simuler une réalisation de lois gaussiennes avec la méthode de rejet. La densité de probabilité g majorant le noyau $\mathtt{K}(x)$ s'obtient de la manière suivante :

$$\mathtt{K}(\frac{x-xi}{h}) = \frac{1}{\sqrt{2\pi}}\,\mathrm{e}^{-\frac{(\frac{x-xi}{h})^2}{2}} \leqslant \frac{1}{\sqrt{2\pi}}\,\mathrm{e}^{(\frac{1}{2}-|\frac{x-xi}{h}|)} = \frac{2h}{\sqrt{2\pi}}\,\mathrm{e}^{(\frac{1}{2})}\,\frac{\mathrm{e}^{-|\frac{x-xi}{h}|}}{2h} = M\frac{\mathrm{e}^{-|\frac{x-xi}{h}|}}{2h}$$

Nous notons ici M la constante de majoration. Nous reconnaissons ici la densité d'une loi de Laplace de paramètres (x_i, h).

Ainsi, nous obtenons les éléments nécessaires à l'application de la méthode de rejet pour simuler des réalisations provenant de cette densité empirique à noyaux gaussiens, qui forment alors notre échantillon répliqué.

5.3 Réplication et formalisation mathématique en dimensions supérieures

Soit un ensemble de n observations vectorielles $(x_1, ..., x_n)$ dans $\mathbf{R}^p$.

Nous rappelons que l'objectif général de l'estimation de la densité est d'approcher la fonction de densité qui a généré l'échantillon aléatoire D_n.

La densité est alors estimée par :

$$\hat{f}_n(x) = \frac{1}{nh} \sum_{i=1}^{n} K(\frac{||x-x_i||}{h}) \,, \quad \text{avec } \mathtt{K}(u) = \frac{1}{\sqrt{2\pi}} \mathrm{e}^{-\frac{u^2}{2}} \ \forall x \in \mathbf{R}^p$$

Le choix du paramètre de lissage h se fait de la même manière que dans le paragraphe précèdent. Pour appliquer la méthode en dimension $p \geqslant 1$, nous avons utilisé le package ks de R.

5.4 Résultats obtenus et limites

Avec cette méthode, nous avons essayer de répliquer un tableau à deux colonnes et 200 individus, dont la forme du nuage de points est une couronne de rayon 8 et 10.

Ensuite, nous avons essayer de répliquer un tableau à deux colonnes et 200 individus. Les colonnes sont respectivement des réalisations de la loi normale centrée réduite, et de la loi exponentielle de paramètre 2

Comparaison des nuages de points

On remarque selon la figure ci-dessous, que les points répliqués forment une couronne, mais débordent du support de la couronne initiale.

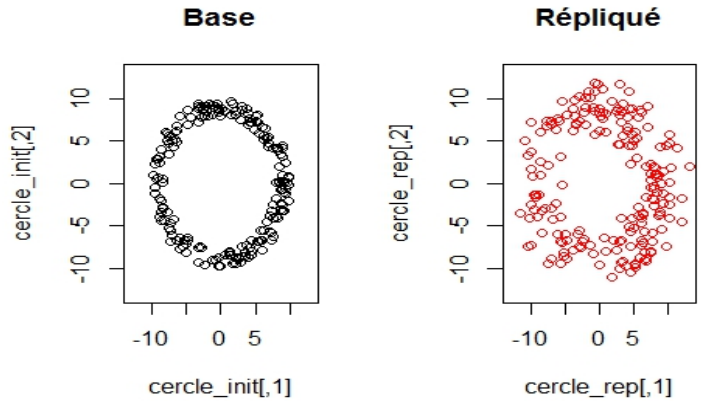

FIGURE 35 – Nuage de points initial (en noir) et répliqué (en rouge)

```
> summary(cercle_init)
       V1                  V2
 Min.   :-9.720   Min.   :-9.8000
 1st Qu.:-5.165   1st Qu.:-6.2825
 Median : 1.480   Median : 0.2250
 Mean   : 0.638   Mean   : 0.3881
 3rd Qu.: 6.315   3rd Qu.: 6.9675
 Max.   : 9.930   Max.   : 9.7400
> summary(cercle_rep)
  cercle_rep1         cercle_rep2
 Min.   :-11.17925   Min.   :-11.946
 1st Qu.: -5.66649   1st Qu.: -4.900
 Median : -0.16887   Median :  2.810
 Mean   :  0.05584   Mean   :  1.615
 3rd Qu.:  5.91213   3rd Qu.:  7.263
 Max.   : 12.55046   Max.   : 12.315
> cor(cercle_init)
           [,1]       [,2]
[1,]  1.0000000 -0.0694076
[2,] -0.0694076  1.0000000
> cor(cercle_rep)
             cercle_rep1 cercle_rep2
cercle_rep1  1.00000000 -0.08778877
cercle_rep2 -0.08778877  1.00000000
>
```

FIGURE 36 – Comparaison des propriétés statistiques

Ce débordement du support se constate aussi par des minimums et maximums répliqués qui différent des minimums et maximums initiaux.

Pour remédier à ce problème, nous avons répliqué plus de points qu'il n'y en avait dans le nuage initial (ici 600 au lieu de 200 points) et sur ces 600 points répliqués, nous avons conservés 200 points qui étaient strictement dans la couronne. Les résultats obtenus sont nettement meilleurs.

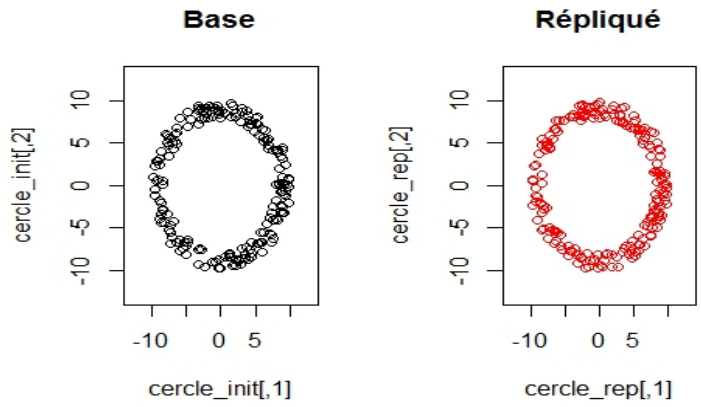

FIGURE 37 – Nuage de points initial (en noir) et répliqué (en rouge)

```
> summary(cercle_init)
       V1                  V2
 Min.   :-9.720   Min.   :-9.8000
 1st Qu.:-5.165   1st Qu.:-6.2825
 Median : 1.480   Median : 0.2250
 Mean   : 0.638   Mean   : 0.3881
 3rd Qu.: 6.315   3rd Qu.: 6.9675
 Max.   : 9.930   Max.   : 9.7400
> summary(cercle_rep)
  cercle_rep1        cercle_rep2
 Min.   :-9.7327   Min.   :-9.7885
 1st Qu.:-5.4392   1st Qu.:-6.8281
 Median : 2.2054   Median :-1.3904
 Mean   : 0.7377   Mean   :-0.1715
 3rd Qu.: 5.7988   3rd Qu.: 6.6635
 Max.   : 9.8799   Max.   : 9.9173
> cor(cercle_init)
            [,1]       [,2]
[1,]  1.0000000 -0.0694076
[2,] -0.0694076  1.0000000
> cor(cercle_rep)
             cercle_rep1 cercle_rep2
cercle_rep1  1.00000000 -0.05392098
cercle_rep2 -0.05392098  1.00000000
```

FIGURE 38 – Comparaison des propriétés statistiques

Comparaison avec les réseaux de neurones

Notons Q le nombre de points moyen par case de la zone quadrillée (qui a été expliqué précédemment). Nous avons illustrer le graphique comparant les distribution de la quantité Q pour la méthode de réplication par estimateur de noyaux gaussiens appliquée sur la couronne de rayons 8 et 10.

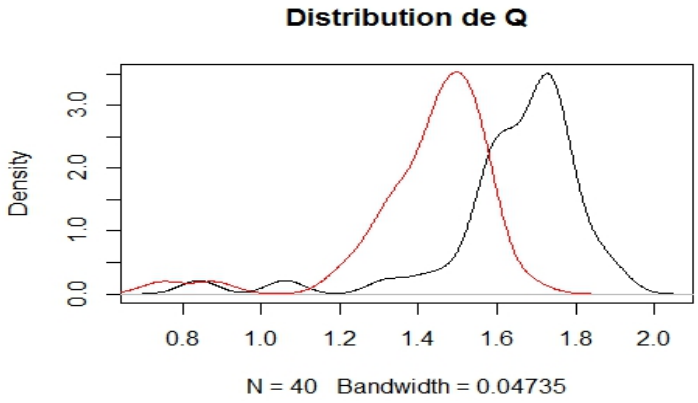

FIGURE 39 – Distributions de Q initiale (en noir) et répliquée (en rouge)

On remarque que les deux courbes se ressemblent bien que leur pics de densité respectifs sont légèrement translatés. On en déduit que même si la structure circulaire du nuage de points est préservée, la zone de concentration des points est décalée.

Comparaison de la répartition des barycentres

D'après le procédé décrit dans la partie "Méthodes de validation", nous avons généré les barycentres simulés à partir des bases simulées selon la base d'origine, puis les barycentres répliqués à partir des bases répliquées de la base d'origine (avec la méthode de réplication des noyaux gaussiens).

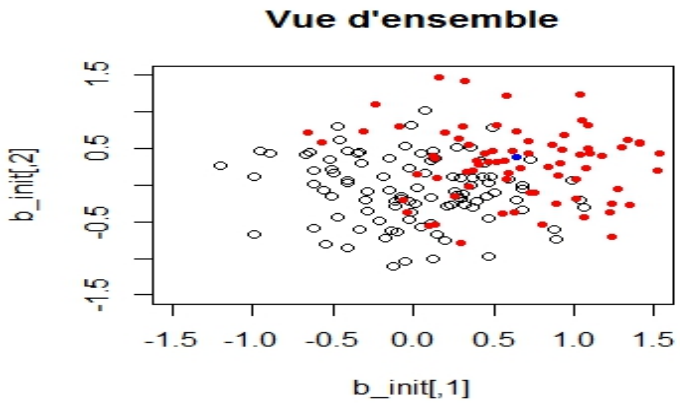

FIGURE 40 – Comparaison des distributions des barycentres

Nous voyons bien que les barycentres répliqués sont dispersés autour du barycentre de base et nous constatons la même chose pour les barycentres simulés. Nous en déduisons qu'il est difficile de distinguer les deux sortes de barycentres, et donc que notre méthode arrive à créer des points assez proches des points initiaux sans pour autant être identifiables.

En général, on utilise le noyau gaussien en raison de sa simplicité et de ses propriétés asymptotiques. Cependant, l'inconvénient principal de ce noyau est qu'il peut attribuer des poids positifs à des coordonnées qui

peuvent être à l'extérieur du support de notre échantillon. Or cela peut causer des problèmes lorsque l'on cherche l'estimateur de la densité de probabilité d'un échantillon provenant de la loi d'une variable aléatoire bornée ou semi-bornée.

Comme nous l'avons vu ci-dessus, c'est le cas par exemple pour notre tableau de données dont le nuage est une couronne de rayons 8 et 10, ou même pour un tableau de données dont une variable suit une loi à semi-borné comme la loi exponentielle, la version répliquée de cette variable peut alors prendre des valeurs négatives.

Comme nous l'avons fait ci-dessous, on pourrait alors penser à générer plus de points que de points dans le tableau de base, et à ne conserver que ceux qui ne dépassent pas du support.

Mais cela implique de connaître et de savoir définir le support en question, chose qui peut être extrêmement difficile pour des tableaux de données à nuages de points à forme complexe comme une étoile ou un croissant de lune.

Une solution envisagée, mais qui n'a pas eut l'occasion d'être étudiée, serait de rechercher d'autres noyaux et ne pas se limiter au noyau gaussien. Il en existe quelques-uns, comme le noyau beta (pour les densités à support compact) et le noyau gamma (pour les densités à support positif).

Troisième partie

Extensions

1 Possibilité de réplication par réseaux de neurones : GAN

Au cours de l'année, nous avons été renseignés par un de nos enseignants sur de possibles pistes à explorer, notamment sur les modèles génératifs. Malheureusement, nous n'avons pas trouvé le temps nécessaire d'établir entièrement la méthode, donc nous ne la présenterons que brièvement en tant que méthode à découvrir pour l'avenir.

La modélisation générative a pour ambition de trouver une structure pour la distribution jointe d'un tableau $(x_1, ..., x_n)$ de données observées. Elle est donc parfaitement en concordance avec notre contexte de réplication.

1.1 Principe

La description qui suit s'appuie sur l'article de Ian Goodfellow. Soit x notre tableau de données observées.

Le principe général des modèles génératifs est d'approcher la vraie distribution inconnue $p(x)$ de notre tableau de données par une distribution générée $\hat{p}_\theta(x)$.

Plus précisément, $\hat{p}_\theta(x)$ est généralement définie par un bruit gaussien que l'on fait passer à travers le modèle génératif (qui peut être un réseau de neurones de paramètres θ).

On remarque alors que modifier les paramètres θ reviendrait à modifier la distribution de sortie $\hat{p}_\theta(x)$.

Notre but est de faire apprendre à notre réseau de neurones la distribution jointe de départ, et donc de trouver les paramètres θ tel que $\hat{p}_\theta(x)$ approche au mieux la distribution jointe de départ $p(x)$.

1.2 Generatives Adversial Networks (GAN)

Il existe plusieurs types de modèles génératifs. Nous ne présenterons que le plus connu d'entre eux : les Generatives Adversial Networks.

Cette fois, le modèle génératif n'est plus un seul réseau de neurones, mais deux réseaux de neurones enchaînes. L'apprentissage se fera grâce à ses deux réseaux de neurones en concurrence :

Le premier réseau de neurones appelé "Générateur" ne connaît pas du tout le vrai tableau de données x et va essayer de le reproduire : Il va tout d'abord créer à l'aveugle un faux échantillon à partir d'un bruit aléatoire quelconque ("gaussien" dans la littérature).

L'autre réseau de neurones appelé "Discriminateur" qui lui est en possession du véritable tableau de données x va ensuite prendre le relais :

Il va comparer les deux tableaux (le faux au vrai) pour prélever les erreurs et défauts du faux échantillon généré par le premier. Il va effectuer la rétro-propagation des erreurs dans le premier réseau de neurones afin que celui-ci s'améliore à chaque boucle de l'apprentissage, et rendre ainsi le faux échantillon généré de plus en plus proche de l'échantillon du tableau de départ jusqu'à pouvoir tromper le Discriminateur.

Synthèse

Cette étude, outre le fait de nous donner l'occasion d'effectuer un véritable travail de recherche, nous a permis d'analyser un aspect primordial dans le métier d'actuaire : les données. Part indispensable à toute analyse, nous avons du imaginer, penser et mettre en place des moyens non triviaux afin de les reproduire en respectant leur diverses caractéristiques. Nous ne copions pas les données, mais notre espoir est d'en conserver un nombre raisonnable de propriétés.

Cinq méthodes se profilent dans ce support. Les méthodes précédemment nommées *Proportions*, *Rot_theta* et *SMOTE* sont puissantes dans la conservation des caractéristiques des bases employées, ce qui semble également représenter leur faiblesse : des valeurs trop similaires à celles de la base initiale. Dans une autre catégorie nous trouvons la méthode *Paramétrique*. Utilisable dans les cas où les lois des variables sont connues ou peuvent être ajustées, inutile dans tous les autres. De plus cette méthode coûte très cher en temps de calcul du fait de devoir faire des ajustements de lois. Enfin la méthode des *Noyaux gaussiens* est applicable en dimension "petite" (inférieure à 10), mais devient inutilisable au delà. La difficulté est de déterminer l'équilibre entre similitudes et différences.

La conclusion est alors que ces méthodes sont acceptables dans certaines limites, et qu'il serait très intéressant de les appliquer dans un cas réel. Pour aller plus loin, il faudra se tourner vers le machine learning et les modèles génératifs (GAN).

A Codes

Voici une partie des principaux codes utilisés dans l'élaboration des méthodes décrites tout au long du rapport.

```
# Les différentes bibliothèques utilisées

library(MASS) # pour certaines fonctionnalités statistiques
library(nnet) # pour l'utilisation des réseaux des neurones
library(ks) # pour générer des densités multidimensionnelles

########################

# Simulation d'un cercle (couronne de rayons a et b)

cercle=function(n,a,b){
  x1=NULL
  y1=NULL
  k=0
  while(k<n){ # On crée autant de points que nécessaires
    x=runif(1,min = -sqrt(b),max=sqrt(b))
    y=runif(1,min = -sqrt(b),max=sqrt(b))
    if((x^2+y^2)<b & (x^2+y^2)>a ){
      x1=c(x1,x)
      y1=c(y1,y)
      k=k+1
    }
  }
  return(matrix(c(x1,y1),n,byrow=FALSE))
}

########################

# Définitions des principales tables utilisées lors de cette étude

n=200
```

```
set.seed(1) # variables identiques à chaque compilation
gauss_init=round(rnorm(n,0,1),3) # la loi normale centrée réduite,
                                 # avec arrondi à 3 décimales
plot(density(gauss_init))

set.seed(1)
exp_init=round(rexp(n,2),3) # la loi exponentielle de paramètre 2,
                            # avec arrondi à 3 décimales
plot(density(exp_init))

Table_init=cbind(gauss_init,exp_init)
plot(Table_init)

set.seed(1)
cercle_init=round(cercle(200,64,100),3)
# couronne de rayons 8 et 10 avec arrondi à 3 décimales
plot(cercle_init)



########################
# Méthode Proportions
########################

# Méthode Proportions sur une entrée x
# simulant par défaut coef=2 fois plus de valeurs
# sur I=100 intervalles de même longueur
# arrondies à arr=2 décimales

Proportions=function(x,coef=2,I=100,arr=2){

  m=(max(x)-min(x)) # longueur de l'étendue des valeurs de x
  t=seq(min(x),max(x),by=m/I) # vecteur contenant les I+1 bornes
                              # des I intervalles
  t[I+1]=max(x) # On force la dernière valeur de t
                # car pas toujours bonne (problème d'arrondi avec R)
  p=NULL # vecteur des effectifs
```

```
  for(i in 1:I){
    p[i]=length(which(x<=t[i+1]))-length(which(x<t[i]))
    # détermination des effectifs de chaque intervalle
  }
  y=NULL # nouveau vecteur
  for(i in 1:I){
    y=c(y,runif(coef*p[i],t[i],t[i+1]))
    # pioche uniforme de nouvelles valeurs
  }
  y=round(y,arr) # arrondi des nouvelles valeurs
  return(y)
}

########################
# Méthode Proportions pour les quantiles
########################

# Méthode Proportions selon les quantiles de l'entrée x
# simulant par défaut coef=2 fois plus de valeurs
# sur des I=1/quant intervalles de longueurs variables
# arrondies à arr=2 décimales

Proportions_quant=function(x,coef=2,quant=.01,arr=2){
  n=length(x)
  I=1/quant # nombre d'intervalles selon quant
  t=quantile(x,seq(0,1,quant))
  t[I+1]=max(x)
  y=NULL
  for(i in 1:I){
    y=c(y,runif(coef*n*quant,t[i],t[i+1]))
  }
  y=round(y,arr)
  return(y)
}

########################
# Méthode Rot_theta
```

```
#########################

rtheta=function(x){
  n=length(x[,1])
  X=NULL
  Y=NULL
  theta=runif(n,0,2*pi) # tirage de n angles sur [0,2pi]
  d=distance2(x[,1],x[,2]) # distribution des distances
                           # au plus proche voisin
  for(i in 1:n){
  # coordonnées des nouveaux points
    X[i]=x[i,1]+cos(theta[i])*d[i]
    Y[i]=x[i,2]+sin(theta[i])*d[i]
  }
  return(matrix(c(X,Y),n,byrow=F))
}

#########################
# Méthode Rot_theta en dimension 3
#########################

# z matrice réelle à 3 colonnes

rthetaphi=function(z){
  n=length(z[,1])
  X1=NULL
  X2=NULL
  X3=NULL
  theta=runif(n,-pi/2,pi/2)
  phi=runif(n,-pi,pi)
  r=distance3(z[,1],z[,2],z[,3])
  for(i in 1:n){
    X1[i]=z[i,1]+r[i]*cos(theta[i])*cos(phi[i])
    X2[i]=z[i,2]+r[i]*cos(theta[i])*sin(phi[i])
    X3[i]=z[i,3]+r[i]*sin(theta[i])
  }
  return(matrix(c(X1,X2,X3),n,byrow=F))
```

```
}

########################
# Méthode SMOTE
########################

# en dimension 2
# fonction retournant les coordonnées du plus proche
# voisin de chaque points ((x1,y1),....,(xn,yn))

plus_proches_voisins=function(x,y){
  n=length(x)
  x1=NULL
  y1=NULL
  d=matrix(rep(0,n*n),nrow=n)
  for(i in 2:n){
    for(j in 1:(i-1)){
      d[i,j]=sqrt((y[j]-y[i])^2+(x[j]-x[i])^2)
    }
  }
  d=d+diag(max(d),n)+t(d)
  for(k in 1:n){
    x1[k]=x[which.min(d[k,])]
    y1[k]=y[which.min(d[k,])]
  }
  f=cbind(x1,y1)
  return(f)
}

# méthode smote dans le cas particulier où on génère
# un seul nouveau point (x'i,y'i) entre le point (xi,yi)
# et son plus proche voisin

smote=function(x){
  n=length(x[,1])
  X=NULL
  Y=NULL
```

```
  theta=runif(n,0,1)
  d=plus_proches_voisins(x[,1],x[,2])
  for(i in 1:n){
    X[i]=x[i,1]+theta[i]*(d[i,1]-x[i])
    Y[i]=x[i,2]+theta[i]*(d[i,2]-y[i])
  }
  return(matrix(c(X,Y),n,byrow=F))
}

########################
# Méthode NORTA
########################

  # Avant d'écrire la fonction, on ajuste des lois théoriques
  # sur les marginales du tableau x

  # QQplot 1  (loi gamma)
  fit1=fitdistr(x[,1],"gamma")
  fit1

  ks.test(x[,1], "pgamma", fit1$estimate[1],fit1$estimate[2])
  u1=rgamma(length(x[,1]),shape=fit1$estimate[1],
                                          rate=fit1$estimate[2])
  qqplot(x[,1],u1)
  abline(0,1)

  # QQplot 2 (loi normale)
  fit2=fitdistr(x[,2],"normal")
  fit2

  ks.test(x[,2], "pnorm", fit2$estimate[1],fit2$estimate[2])

  u2=rnorm(length(x[,3]),mean=fit2$estimate[1],sd=fit2$estimate[2])
  qqplot(x[,3],u2)
  abline(0,1)

  # QQplot 3 (loi normale)
```

```
  fit3=fitdistr(x[,3],"normal")
  fit3

  ks.test(x[,3], "pnorm", fit3$estimate[1],fit3$estimate[2])

  u3=rnorm(length(x[,3]),mean=fit3$estimate[1],sd=fit3$estimate[2])
  qqplot(x[,3],u3)
  abline(0,1)

replication_param=function(x){
  # on prend les corrélation du rang "spearman"
  # car théoriquement les expressions de cor_y se
  # simplifient et donc peuvent se coder facilement

  cor_init=cor(x,method = "spearman")

  # création de la matrice de corrélation cor_y
  # à partir de la matrice de corrélation
  # de Spearmann cor_init du tableau de départ tab_init

  n=length(x[,1])
  p=length(x[1,])
  cor_y=matrix(0,nrow=p,ncol=p)
  for(i in 1:p){
    for(j in 1:p){
      cor_y[i,j]=2*sin(pi*cor_init[i,j]/6)
    }
  }
  for(i in 1:p){cor_y[i,i]=1}

  # il faut simuler un vecteur normal y de matrice de corrélation
  # cor_y afin d'avoir en retour
  # après les transformations inverses,
  # les bonnes marginales avec les bonnes corrélations cor_init

  tab_rep=mvrnorm(n,mu=c(0,0,0),Sigma = cor_y)
```

```
  tab_rep[,1]=qgamma(pnorm(tab_rep[,1]),
                shape=fit1$estimate[1],rate=fit1$estimate[2])
  tab_rep[,2]=qnorm(pnorm(tab_rep[,2]),
                mean=fit2$estimate[1],sd=fit2$estimate[2])
  tab_rep[,3]=qnorm(pnorm(tab_rep[,3]),
                mean=fit3$estimate[1],sd=fit3$estimate[2])

  return(tab_rep)
}

########################
# Méthode des noyaux gaussiens
########################

# x est le tableau de départ

rep_noyaux_gaussien=fonction(x){
  estimateur_de_f=kde(x)
  y=rkde(length(x[,1]),estimateur_de_f)
  return(y)
}

########################
# Quadrillage du plan
########################

quadrillage2=function(X,cases=10){
  n=length(X[,1])

  # délimitation de la zone quadrillée
  t1=seq(min(X[,1]), max(X[,1]), by=(max(X[,1])- min(X[,1])) /cases)
  t2=seq(min(X[,2]), max(X[,2]), by=(max(X[,2])- min(X[,2])) /cases)
  # un faible dépassement des extrêmes est permis
  t1[1]=min(X[,1])-10^-3 ; t1[length(t1)]=max(X[,1])+10^-3
  t2[1]=min(X[,2])-10^-3 ; t2[length(t2)]=max(X[,2])+10^-3

  l1=length(t1)
```

```
  l2=length(t2)

  # les lignes suivantes permettent d'une part
  # de compter les effectifs de chaque case, et d'autre part
  # de pouvoir comparer visuellement le nuage de point
  # avec le quadrillage
  # (sinon le quadrillage n'est pas dans le bon sens)

  p=matrix(rep(0,(l1-1)*(l2-1)),(l1-1),byrow=T)
  for(i in 1:(l1-1)){
    for(j in 1:(l2-1)){
      p[i,j]=sum( (X[,1]<t1[i+1]) * (X[,1]>=t1[i])
                  *(X[,2]<t2[j+1]) * (X[,2]>=t2[j]) )
    }
  }
  p=t(p)

  q=matrix(rep(0,(l1-1)*(l2-1)),(l1-1),byrow=T)
  for(i in 1:(l1-1)){
    for(j in 1:(l2-1)){
      q[i,j]=sum( (Y[,1]<t1[i+1]) * (Y[,1]>=t1[i])
                  *(Y[,2]<t2[j+1]) * (Y[,2]>=t2[j]) )
    }
  }
  q=t(q)

  Base=matrix(rep(0,(l1-1)*(l2-1)),(l1-1),byrow=T)
  Replic=matrix(rep(0,(l1-1)*(l2-1)),(l1-1),byrow=T)
  for(i in 1:(l1-1)){
    Base[i,]=p[(l1-1)-i+1,]
    Replic[i,]=q[(l1-1)-i+1,]
  }
  c=rep(-1,(l1-1)) # permet de séparer visuellement Base et Replic
  Comp=cbind(Base,c,Replic)
  return(Comp)
}
```

```
########################
# Quadrillage de l'espace
########################

# X matrice réelle à 3 colonnes
# par défaut le programme crée 3*3*3 cases

# même principe qu'en dimension 2 mais seul un tableau est considéré
# pour un affichage plus pratique
# L'espace est vu comme des tranches
# qui subissent le traitement de quadrillage2

quadrillage3=function(X,cases=3){
  n=length(X[,1])

  t1=seq(min(X[,1]), max(X[,1]), by=(max(X[,1])- min(X[,1])) /cases)
  t2=seq(min(X[,2]), max(X[,2]), by=(max(X[,2])- min(X[,2])) /cases)
  t3=seq(min(X[,3]), max(X[,3]), by=(max(X[,3])- min(X[,3])) /cases)
  t1[1]=min(X[,1])-10^-3 ; t1[length(t1)]=max(X[,1])+10^-3
  t2[1]=min(X[,2])-10^-3 ; t2[length(t2)]=max(X[,2])+10^-3
  t3[1]=min(X[,3])-10^-3 ; t3[length(t3)]=max(X[,3])+10^-3

  l1=length(t1)
  l2=length(t2)
  l3=length(t3)

  Total=rep(-1,l1-1)
  c=rep(-1,l1-1)

  for(k in 1:(l3-1)){
    p=matrix(rep(0,(l1-1)*(l2-1)),(l1-1),byrow=T)
    Base=matrix(rep(0,(l1-1)*(l2-1)),(l1-1),byrow=T)
    for(j in 1:(l2-1)){
      for(i in 1:(l1-1)){
        p[i,j]=sum( (X[,1]<t1[i+1]) * (X[,1]>=t1[i])
                    *(X[,2]<t2[j+1]) * (X[,2]>=t2[j])
                      *(X[,3]<t3[k+1]) * (X[,3]>=t3[k]) )
```

```
      }
    }
    p=t(p)
    for(i in 1:(l1-1)){
      Base[i,]=p[(l1-1)-i+1,]
    }
    Total=cbind(Total,Base,c)
    # l'affichage de chaque couche de l'espace est séparé par c
  }
  return(Total)
}

########################

# Validation Barycentres pour Table_init et méthode Proportions
# pour des raisons de puissance et de visibilité,
# nous ne simulons que 100 tables et non 1000

n=200

b_init=NULL
for(i in 1:100){ # génération des barycentres simulés
  x1=round(rnorm(n,0,1),2)
  x2=round(rexp(n,2),3)
  b_init=rbind(b_init,c(mean(x1),mean(x2)))
}

b_rep=NULL
for(i in 1:100){ # génération des barycentres répliqués

  y1=Proportions(Table_init[,1], I=10, arr=2)
  y2=Proportions(Table_init[,2], I=10, arr=3)

  Table_rep=matrix(rep(1,2*n),n,byrow=T)

  for(i in 1:n){
```

```
  # formation de gauss_rep
    a1=Table_init[i,1]
    b1=which( abs(a1-y1) == min( abs(a1-y1) ) )[1]
    Table_rep[i,1]=y1[b1]
    y1=y1[-b1]

  # formation de exp_rep
    a2=Table_init[i,2]
    b2=which( abs(a2-y2) == min( abs(a2-y2) ) )[1]
    Table_rep[i,2]=y2[b2]
    y2=y2[-b2]
  }
  b_rep=rbind(b_rep,c(mean(Table_rep[,1]),mean(Table_rep[,2])))
  # matrice contenant les barycentres issus de la réplication
}

par(mfrow=c(1,2))
plot(b_init,col='black',main="Vue d'ensemble")
points(b_rep,col='red')
points(b[1],b[2],col='blue',pch=20)

plot(b_rep,col='red',main="Zoom sur b_rep")
points(b[1],b[2],col='blue',pch=20)

########################

# Utilisation de Proportions sur le cercle

n=200

x1=cercle_init[,1]
x2=cercle_init[,2]

y1=Proportions(x1,coef=2,I=100)
y2=Proportions(x2,coef=2,I=100)

Y=matrix(rep(1,2*n),n,byrow=T)
```

```
for(i in 1:n){
  a1=x1[i]
  b1=which( abs(a1-y1) == min( abs(a1-y1) ) )[1]
  Y[i,1]=y1[b1]
  y1=y1[-b1]

  a2=x2[i]
  b2=which( abs(a2-y2) == min( abs(a2-y2) ) )[1]
  Y[i,2]=y2[b2]
  y2=y2[-b2]
}

cercle_proportions=Y

par(mfrow=c(1,2))
plot(cercle_init,col='black',main="Base")
points(cercle_proportions,col='red',main="Replication")

########################

# Utilisation de Rot_theta sur Table_init

Rot=rtheta(gauss_init,exp_init)

plot(Table_init,col='black',main='Rot_theta')
points(Rot,col='red')
```

B Bibliographie

Références

[1] Sciences des données : de la logique du premier ordre à la Toile, Serge Abitboul, 8 mars 2012,
http://books.openedition.org/cdf/529

[2] Informatique et libertés, suis-je concerné ?, CNIL,
https://www.cnil.fr/fr/informatique-et-libertes-suis-je-concerne

[3] Package ks version 1.11.1, Cran, avril 2018,
https://cran.r-project.org/web/packages/ks/ks.pdf

[4] Norme de Pratique relative à l'utilisation et la protection des données massives, des données personnelles et des données de santé à caractère personnel, Institut des Actuaires, 16 novembre 2017,
https://www.institutdesactuaires.com/global/gene/link.php?doc_id=11773

[5] SMOTE : Synthetic Minority Over-sampling Technique, Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall and W. Philip Kegelmeyer, 2002,
https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume16/chawla02a-html/chawla2002.html

[6] Generative Adversarial Nets, Ian Goodfellow, 2014,
https://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[7] Behaviour of the NORTA Method for Correlated Random Vector Generation as the Dimension Increases, Soumyadip GHOSH and Shane G. HENDERSON,
https://pdfs.semanticscholar.org/3c6b/4dbe5e320b09c5425b479611af0e094f5290.pdf

[8] Preuve de Kruskal 1958 pour NORTA, Kruskal W. 1958. Ordinal measures of association. Journal of the American Statistical Association 53 :814–861

[9] Database Principles and Design, Colin Ritchie, Cengage Learning EMEA, 2008, ISBN 9781844805402

[10] — Cours sur les copules de Monsieur Brice FRANKE, version 2017-2018

— Cours d'apprentissage statistique de Monsieur Franck VERMET, version 2017-2018

— Cours de Data Mining de Monsieur Philippe LENCA, version 2017-2018