resume2

2024-02-05

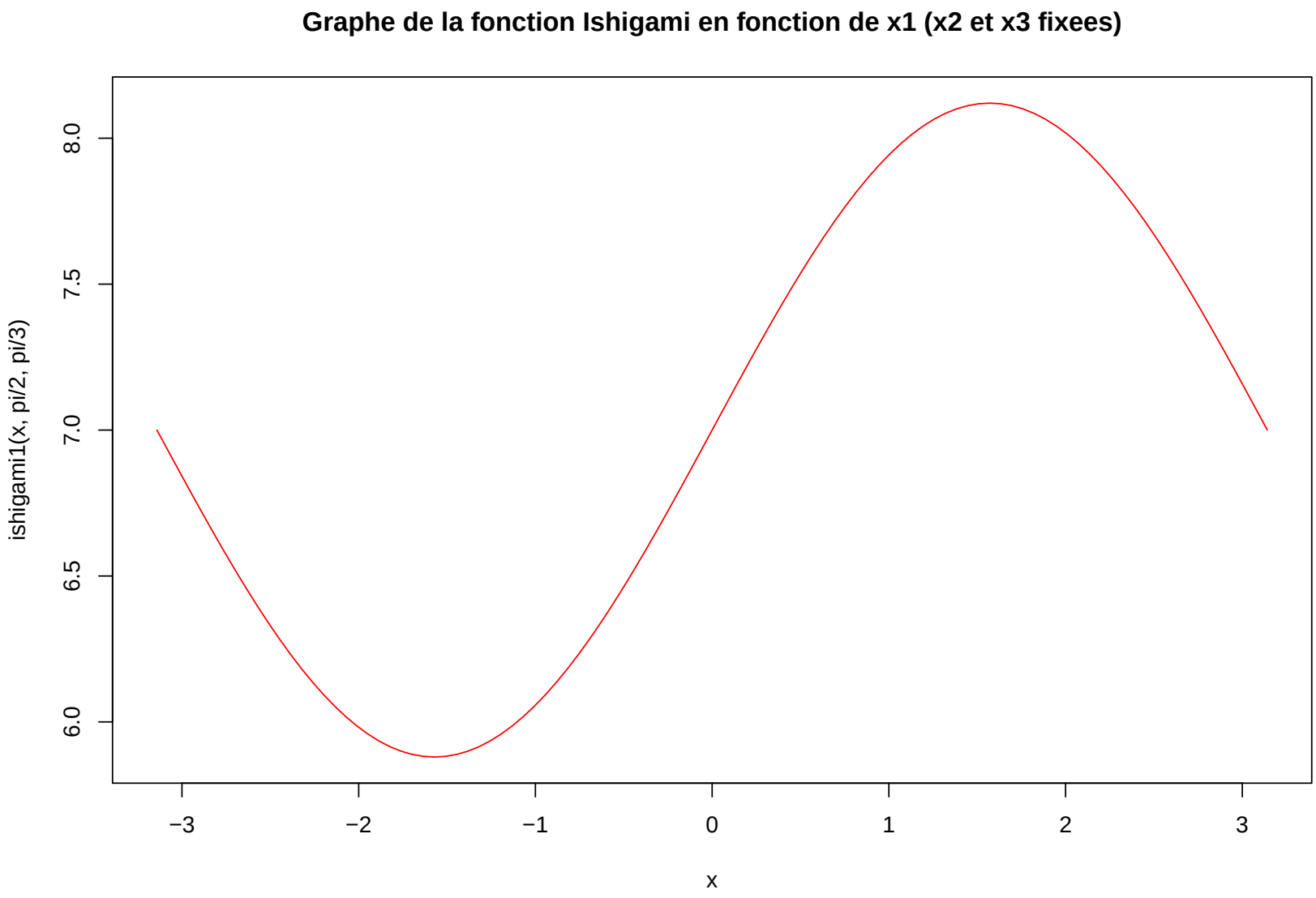
Graphes exploratoires - Fonction Ishigami - Code

```r
ishigami1 <- function(X1, X2, X3) {
  A <- 7
  B <- 0.1
  sin(X1) + A * sin(X2) ^ 2 + B * X3 ^ 4 * sin(X1)
}
curve(ishigami1(x,pi/2,pi/3), from = -pi, to = pi,
      col = "red", main = "Graphe de la fonction Ishigami e
```

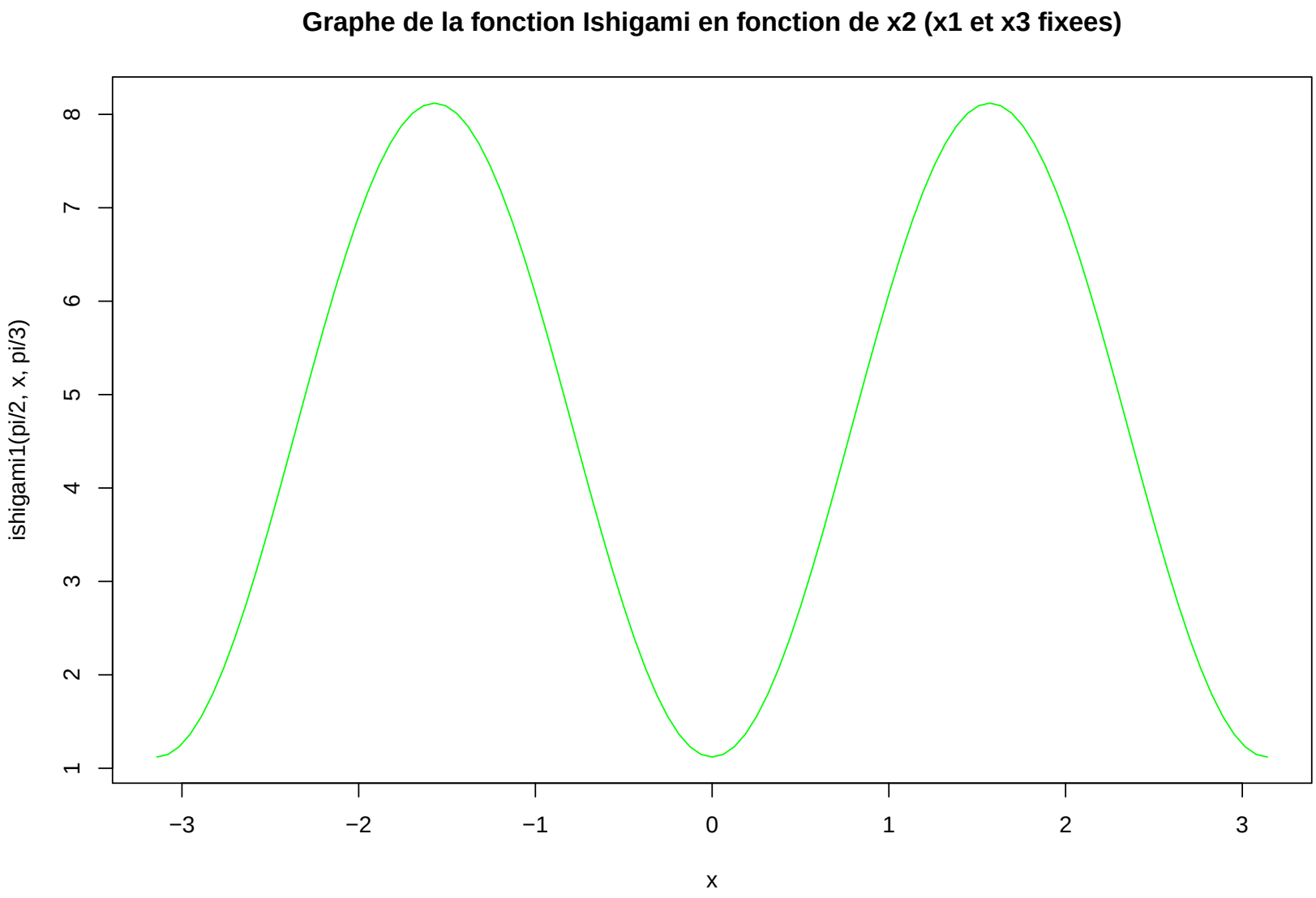
```r
curve(ishigami1(pi/2,x,pi/3), from = -pi, to = pi,
      col = "green", main = "Graphe de la fonction Ishigam
```

```r
curve(ishigami1(pi/2,pi/3,x), from = -pi, to = pi,
      col = "blue", main = "Graphe de la fonction Ishigami
```

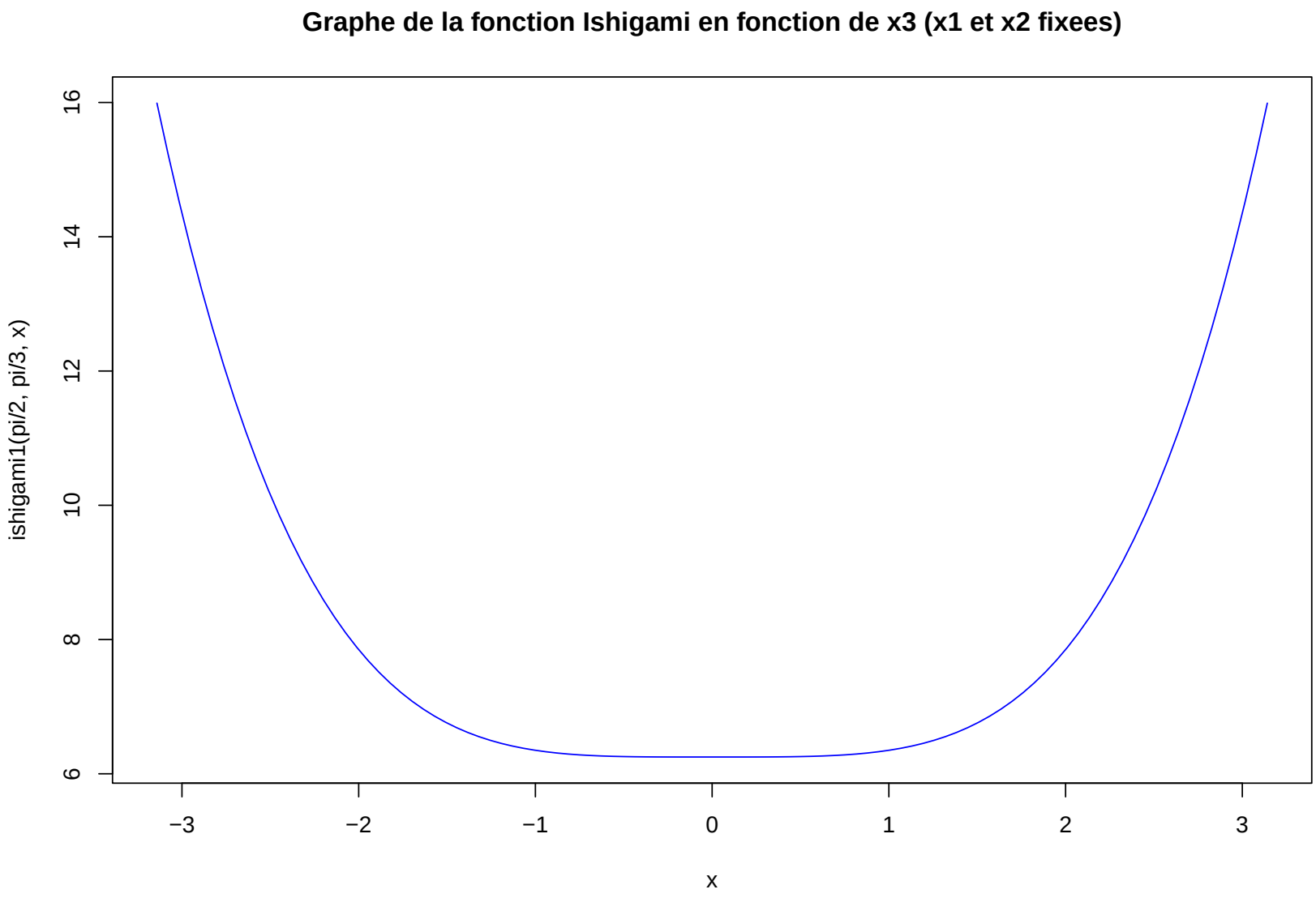
X1 varie


Graphe de la fonction Ishigami en fonction de x1 (x2 et x3 fixees)
ishigami1(x, pi/2, pi/3)
8.0
7.5
7.0
6.5
6.0
−3
−2
−1
0
1
2
3
x

X2 varie


Graphe de la fonction Ishigami en fonction de x2 (x1 et x3 fixees)
ishigami1(pi/2, x, pi/3)
1
2
3
4
5
6
7
8
−3
−2
−1
0
1
2
3
x

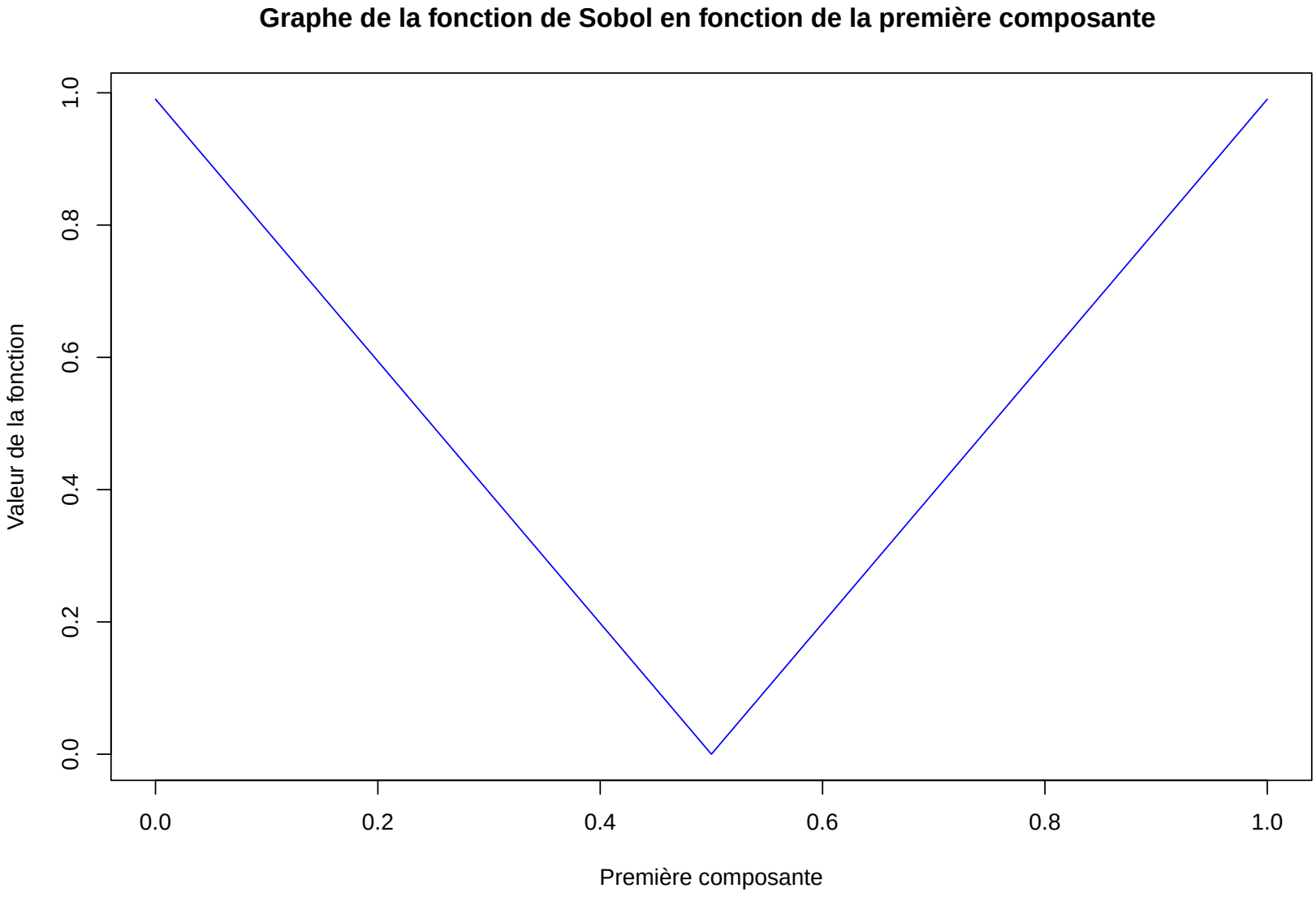
X3 varie


Graphe de la fonction Ishigami en fonction de x3 (x1 et x2 fixees)
ishigami1(pi/2, pi/3, x)
16
14
12
10
8
6
−3
−2
−1
0
1
2
3
x

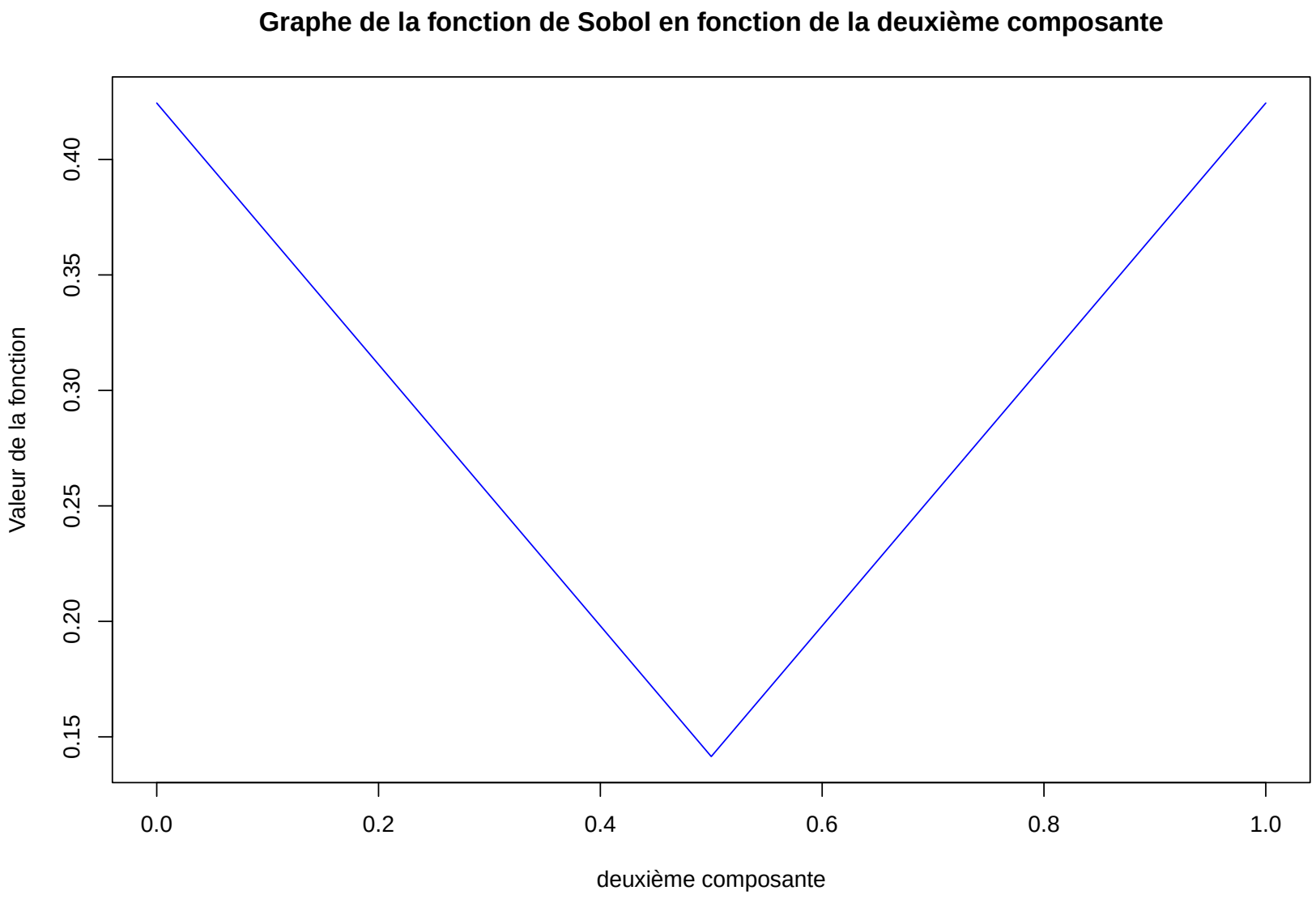
Graphes exploratoires - Fonction de Sobol - Code

```r
sobol_Fun1 <- function(X) {
  a <- c(0, 1, 4.5, 9, 99, 99, 99, 99)
  y <- 1

  for (j in 1:8) {
    y <- y * (abs(4 * X[j] - 2) + a[j])/(1 + a[j])
  }
  return(y)
}
#calcul dex valeurs de y en fonction de x1(les autres valeu
x_values <- seq(0, 1, length.out = 10000)
y_values <- sapply(x_values, function(x) sobol_Fun1(c(x,0.6

#tracé de la courbe
plot(x_values, y_values, type = "l", col = "blue",
     xlab = "Première composante", ylab = "Valeur de la fon
     main = "Graphe de la fonction de Sobol en fonction de
```

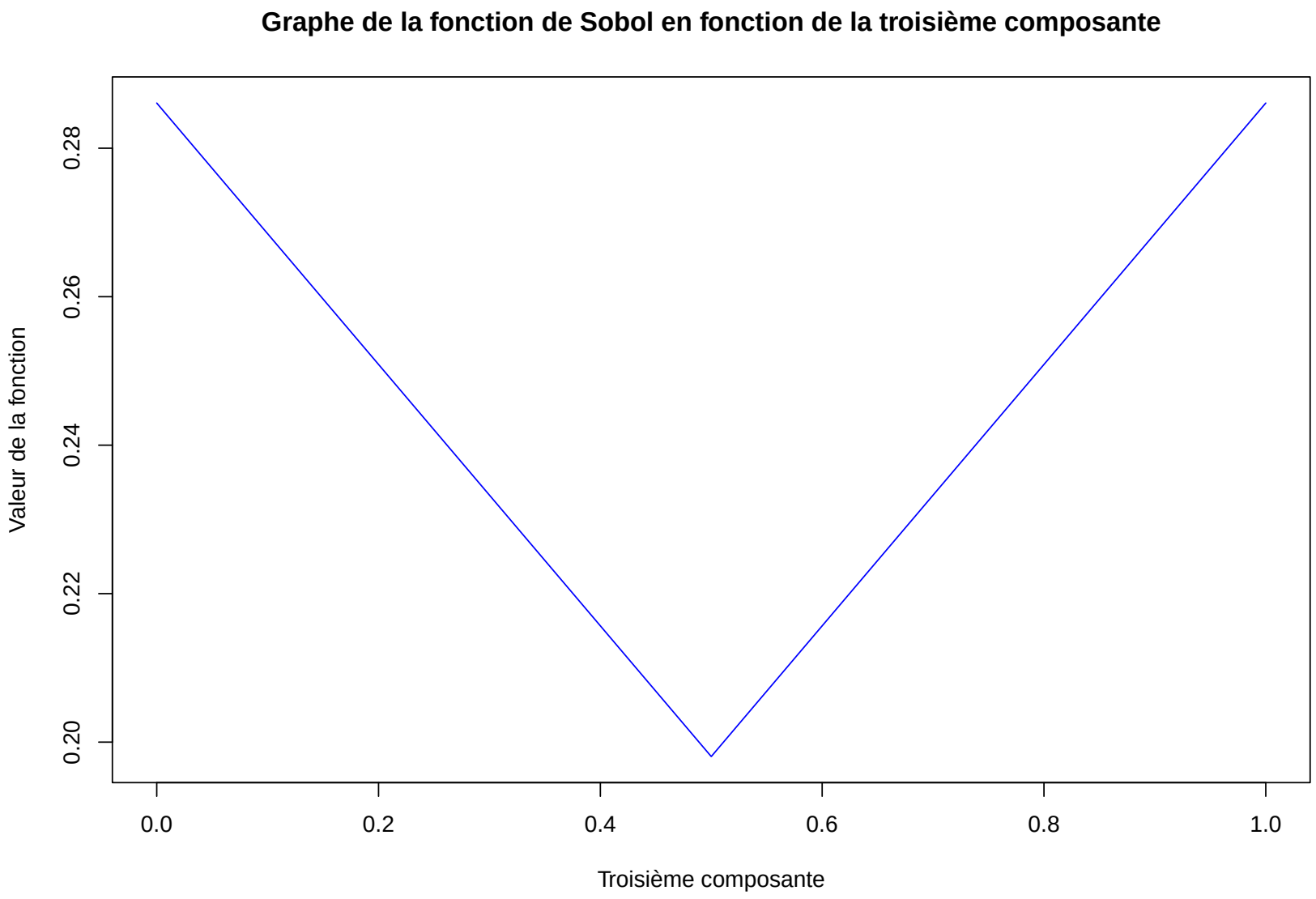
X1 varie


Graphe de la fonction de Sobol en fonction de la première composante
Valeur de la fonction
0.0
0.2
0.4
0.6
0.8
1.0
Première composante
0.0
0.2
0.4
0.6
0.8
1.0

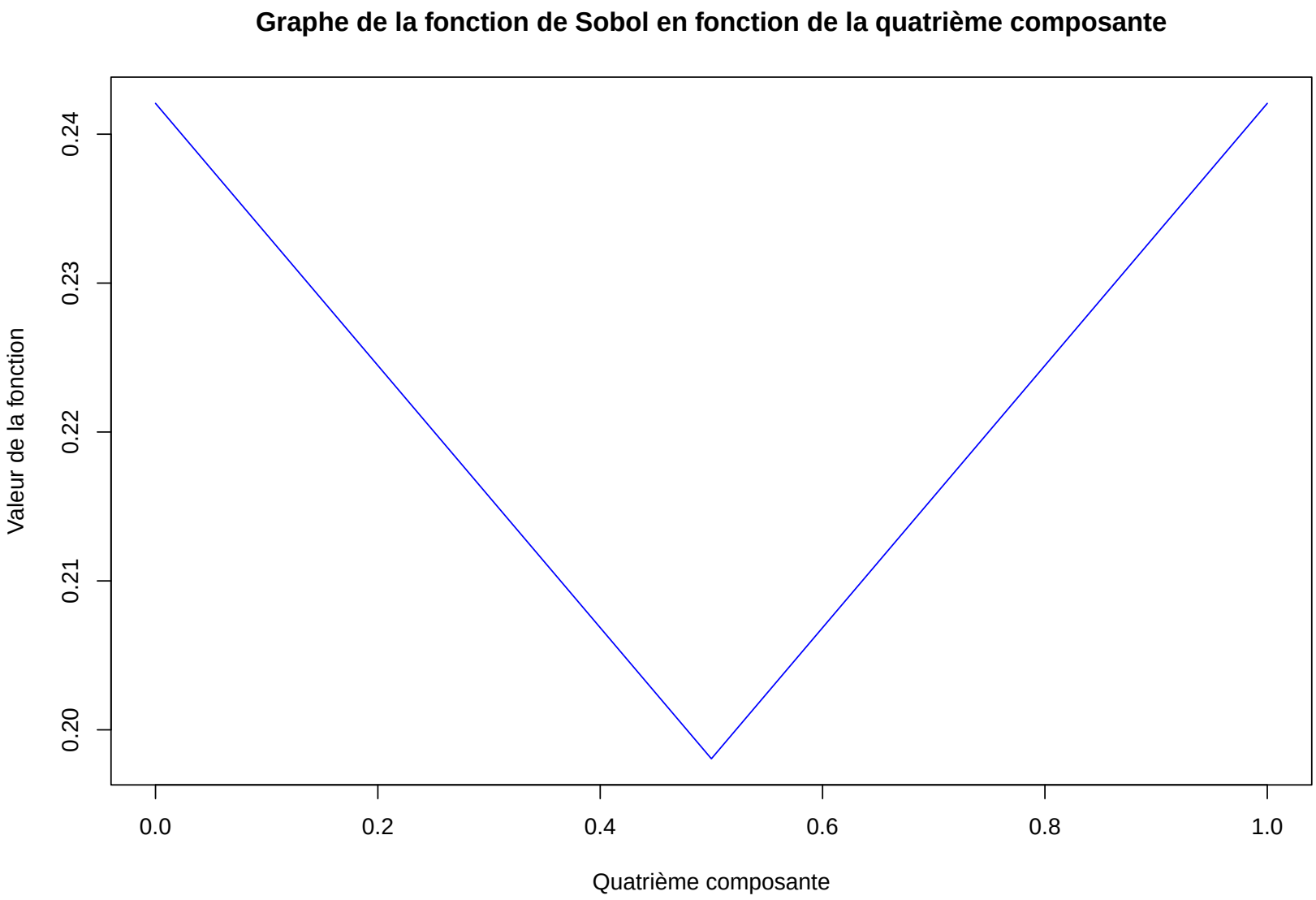
X2 varie


Graphe de la fonction de Sobol en fonction de la deuxième composante
Valeur de la fonction
0.15
0.20
0.25
0.30
0.35
0.40
0.0
0.2
0.4
0.6
0.8
1.0
deuxième composante

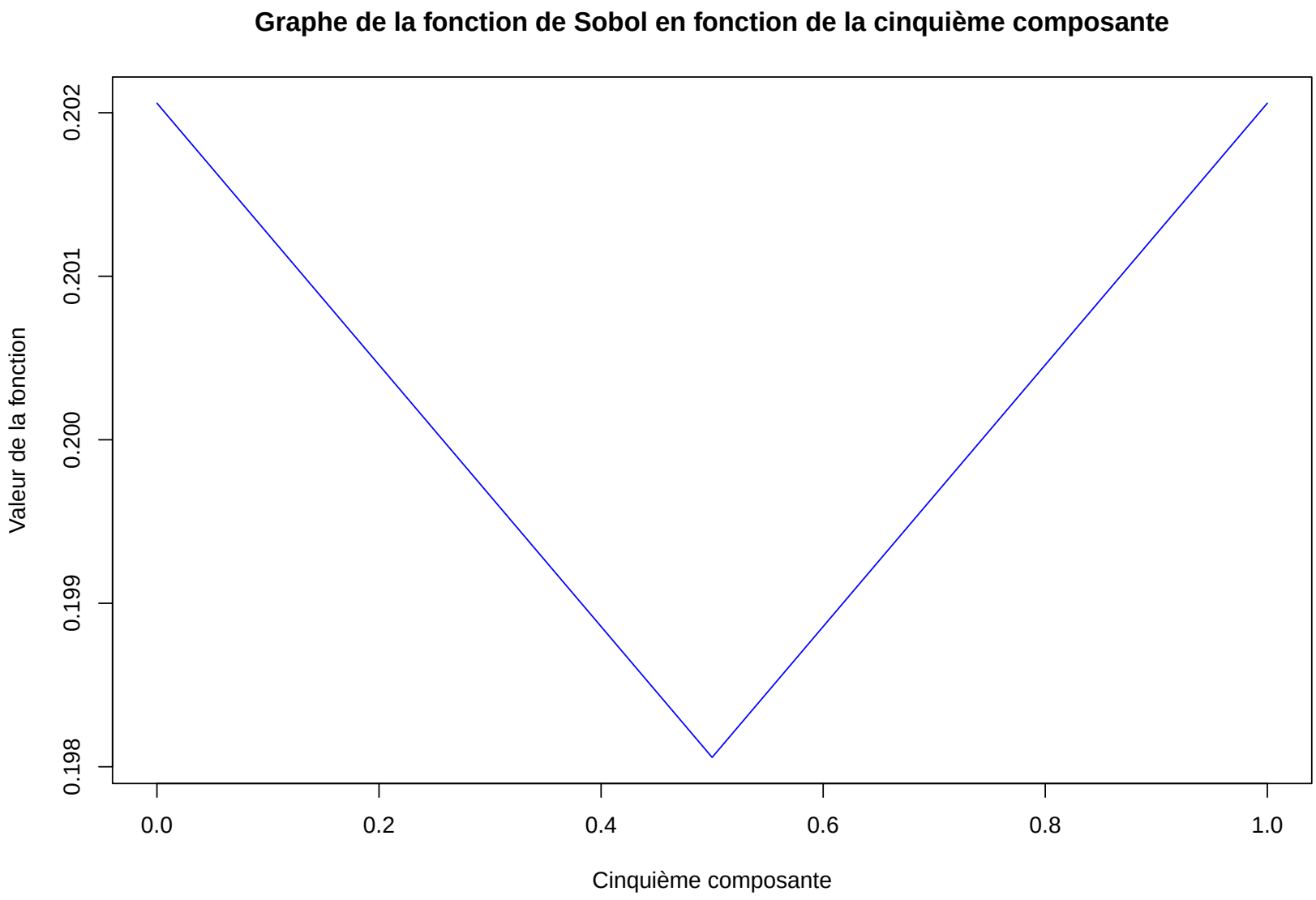
X3 varie


Graphe de la fonction de Sobol en fonction de la troisième composante
Valeur de la fonction
0.20
0.22
0.24
0.26
0.28
0.0
0.2
0.4
0.6
0.8
1.0
Troisième composante

X4 varie


Graphe de la fonction de Sobol en fonction de la quatrième composante
Valeur de la fonction
0.20
0.21
0.22
0.23
0.24
0.0
0.2
0.4
0.6
0.8
1.0
Quatrième composante

X5 varie


Graphe de la fonction de Sobol en fonction de la cinquième composante
Valeur de la fonction
0.198
0.199
0.200
0.201
0.202
0.0
0.2
0.4
0.6
0.8
1.0
Cinquième composante

Analyse de sensibilité avec entrées dépendantes - Introduction

Lorsque les paramètres d'entrée incertains sont indépendants, le vecteur $X = (X_1, .., X_d)$ est distribué selon une loi de probabilité construite comme le produit des lois de probabilité de chaque paramètre d'entrée, c'est-à-dire $P_X = \prod_{j=1}^{d} P_{X_j}$. Cela n'est plus le cas si au moins deux variables sont dépendantes. Nous allons alors présenter brièvement plusieurs méthodes permettant de calculer des mesures de sensibilité avec des entrées dépendantes, puis nous nous intéressons plus en détails à une méthode utilisant les effets de Shapley. Ensuite nous montrerons comment l'algorithme Random Forest peut être utile pour l'analyse des variables d'importance. Pour finir nous verrons les indices Hsic.

Méthode Grouping 1

-valuer les indices de Sobol de sous-ensembles de X, ces sous-ensembles sont constitués de variables dépendantes entre elles et ils sont indépendants entre eux

-interprétation compliquée, d'où l'introduction des indices de Sobol groupés par Jacques et al.

-L'indice de Sobol groupé d'ordre 1 est défini de la même manière que l'indice de Sobol fermé associé à un sous-ensemble dans le cas des entrées indépendantes

Méthode Grouping 2

-L'indice de Sobol groupé du second ordre donné par $S_{k,k'} = S^{clos}_{A_k \cup A_{k'}} - S^{clos}_{A_k} - S^{clos}_{A_{k'}}$ mesure l'impact de l'interaction entre X_k et $X_{k'}$ sur la sortie $G(X)$

-L'indice de Sobol groupé fermé du second ordre est donné par $S^{clos}_{k,k'} = S^{clos}_{A_k \cup A_{k'}}$ mesure l'impact de X_k et $X_{k'}$ sur la sortie $G(X)$, individuellement et en interaction les uns avec les autres.

-Cette méthode permet donc de quantifier l'influence d'un groupe de variables d'entrée sur la sortie mais pas d'une seule variable précise si celle-ci est corrélée à d'autres variables

Méthodes basées sur la covariance

- consistent à décomposer la variance partielle d'une entrée en une partie corrélée et une partie décorrélée.
- permet de mettre en évidence les entrées qui ont un un impact conséquent sur la sortie uniquement grâce à leur forte corrélation avec d'autres entrées
- utilisation la décomposition de Hoeffding ou la décomposition de Hoeffding généralisée qui nécessite une hypothèse et qui présente l'avantage d'être unique

Présentation effets de Shapley

-permettent d'attribuer la valeur créée par une équipe à ses membres

-résultent d'une allocation directe d'une part de la variance de la sortie à chaque entrée

-somme des effets associés au variables d'entrée vaut la variance de la sortie Y (dans le cas où ils ne sont pas normalisés)

Effets de Shapley - cadre mathématique

-fonction charactéristique $val : \mathcal{P}_d \to \mathbb{R}^+$ telle que $val(\emptyset) = 0$

-basée sur cette fonction charactéristique une valeur Φ_j est attribuée à chaque variable d'entrée, la méthode d'attribution doit vérifier 4 propriétés :

-efficacité : $\sum_{j=1}^{d} \phi_j = \text{val}([1:d])$

-symétrie : Si $\text{val}(A \cup \{i\}) = \text{val}(A \cup \{j\})$ pour tout $A \subseteq -\{i,j\}$, alors $\phi_i = \phi_j$.

-dummy : Si $\text{val}(A \cup \{i\}) = \text{val}(A)$ pour tout $A \in \mathcal{P}_d$, then $\phi_i = 0$.

-additivité : Si val et val0 ont des valeurs de Shapley respectives Φ et $\Phi + \Phi'$, alors le jeu avec une valeur de val + val0 a une valeur de Shapley de Φ'pour $j \in [1:d]$.

Effets de Shapley - définition

- L'unique valeur ϕ qui satisfait les 4 propriétés attribue une valeur aux variables X_j selon la formule suivante :

$$\phi_j = \frac{1}{d} \sum_{A \subset -j} \binom{d-1}{|A|}^{d-1} [val(A \cup \{j\}) - val(A)]$$

- Pour tout $j = 1, \ldots, d$, on définit l'effet Shapley pour X_j comme

$$\mathrm{Sh}_j = \frac{1}{d} \sum_{A \subset -j} \binom{d-1}{|A|}^{d-1} \left(S^{clos}_{A \cup \{j\}} - S^{clos}_{A} \right)$$

Cette définition correspond à la valeur ϕ_j obtenue en définissant la fonction de valeur $val : \mathcal{P}_d \to \mathbb{R}^+$ comme

$$\mathrm{val}(A) = S^{clos}_A = \frac{\mathrm{Var} E[G(X)|X_A]}{V} \quad (1)$$

Effets de Shapley - propriétés

- compris entre 0 et 1
- En supposant (1) on a $1 = \sum_{j=1}^{d} \mathrm{Sh}_j$
- On peut choisir de définir la fonction charactéristique $\mathrm{val}_0 : \mathcal{P}_d \to \mathbb{R}^+$ par $\mathrm{val}_0(A) = \frac{E\,\mathrm{Var}[G(X)|X_{\overline{A}}]}{V}$ (2) pour obtenir les mêmes résultats.

Effets de Sapley - procédures d'estimation

- complexité d'obtention des indices car tous les sous-ensembles de variables d'entrée doivent être considérés, pour cela il faut considérer toutes les permutations possibles de variables d'entrée
- on note Π une permutaion de tous les d joueurs et on définit l'ensemble $P_j(\Pi)$ comme les joueurs qui précèdent le joueurs j dans Π Ainsi le coût supplémentaire d'inclure le joueur j dans $P_j(\Pi)$ est val($P_j(\Pi) \cup \{j\}$)-val($P_j(\Pi)$). En prenant toutes les permutations Π_d des d joueurs en considération la valeur de Shapley Φ_j peut se réécrire Shj = $\sum_{\pi \in \Pi^d} \frac{1}{d!} (\text{val}(P_j(\pi) \cup \{j\}) - \text{val}(P_j(\pi)))$ (3). Pour calculer les indices de Shapley avec (3) on doit évaluer val(A) pour tout $A \in P_d$ (c'est à dire $2^d - 1$ évaluations) et d permuTations.

Effets de Shapley - méthode de Castro et al. améliorée par Song et al.

-algorithme d'approximation

-basé sur la fonction (2) val_0

-pour une grande dimension d ils prennent indépendemment et uniformément m permutations $\Pi_1, \Pi_2, ..., \Pi_m$ dans Π_d et ils estimment SH_j avec un schéma de Monte-Carlo

Effets de Shapley - autres méthodes d'estimation 1

- algorithme de Broto et al. ("subset agregation procedure) : consiste à estimer $E(Var(Y|X_A^c)$ avec une double procédure de Monte-Carlo ou d'estimer $Var(E(Y|X_A))$ avec un schéma pick freeze pour tout $A \in P_d$ dès le départ et de les stocker. Quand le nombre de variables devient grand Broto et al. suggère qu'on n'est pas obligé de tous les estimer avec la même précision et qu'une grande partie pourra être approximée par 0.
- algorithme de Plischke et al. : basé sur les inverse de MÖbius, à partir de la relation $\Phi_j(val) = \sum_{A \in P_d, j \in A} \frac{mob(A)}{|A|}$ où mob(A) $= P \cdot \sum_{B \subseteq A} (-1)^{|A|+|B|} \cdot \text{val}(B)$ et de la fonction val_0 il est possible de calculer les indices de Shapley en calculant $2^d - 1$ valeurs de val(A), $A \in P_d$ et en résolvant un système d'équations défini par une matrice peu remplie de taille $(2^d - 1)(2^d - 1)$.

Effets de Shapley - autres méthodes d'estimation 2

- algorithme one-sample : algorithme basé sur le plus proche voisin, peut présenter des biais important réduits grâce à l'augmentation de la taille de l'échantillon. Algorithme du voyageur de commerce (TSP) pour trouver les plus proches voisins.

Effets de Shapley - fonction R

Fonctions calculant les indices de Shapley avec respectivement un algorithme de permutations exacte et un algorithme de permutations aléatoires :

- shapleyPermEx : Estimation of Shapley effects by examining all permutations of inputs (Agorithm of Song et al, 2016), in cases of independent or dependent
- shapleyPermRand : Estimation of Shapley effects by random permutations of inputs (Agorithm of Song et al, 2016), in cases of independent or dependent

Fonctions pour calculer les indices de Shapley en utilisant l'algorithme du plus proche voisin :

- shapleysobol_knn : Data given Shapley effects estimation via nearest-neighbors procedure
- shapleySubsetMc : Estimation of Shapley effects from data using nearest neighbors method

Random Forest - Idée générale

- faire des simulations avec ISIS-Fish puis donner ces données à un algorithme Random forest qui va construire un une “boîte noire” qui pourra simuler de nouvelles prédictions en ayant appris avec l'échantillon qu'on lui a fourni. On cherchera ensuite à mesurer l'importance des différentes variables d'entrée de cette boîte noire.

Random Forest - Arbre de décision

-à partir d'un ensemble de données on regroupe les données "proches", à partir de maintenant chaque donnée appartient donc à un groupe de données

-on construit l'arbre de façon à ce que chaque donnée de l'échantillon, quand elle suit un chemin de l'arbre selon sa valeur, tombe dans la cellule finale de l'arbre correspondant à la moyenne de son groupe

Random Forest

- on prend un échantillon de données
- on forme d'autres échantillon à partir de celui-ci grâce à la méthode de Bootstrappe
- on choisit aléatoirement des sous-ensembles de charactéristiques des variables d'entrée pour chaque échantillon
- on crée un arbre de décision basé sur les charactéristiques associées à l'échantillon pour chaque échantillon
- ensuite, pour obtenir une prédiction, on fait passer une variable d'entrée par chacun des arbres et on moyenne les valeurs de sorties obtenues par chaque arbre pour trouver la valeur de prédiction de sortie

Sobol-MDA pour Random Forest - Introduction 1

-MDA : Mean Decrease Accuracy, montre de combien la précision de notre modèle diminue si nous enlevons une des covariables ou si les valeurs d'une covariables sont permutées

-MDI : somme les diminutions pondérées de l'impureté sur tous les noeuds qui se divisent selon une covariable donnée, moyennées sur l'ensemble des arbres de la forêt

-preuve théorique que le MDA ne cible pas la bonne quantité pour détecter les covariables influentes dans un contexte dépendant

-Sobol-MDA corrige les lacunes du MDA originel et estime de manière cohérente la diminution de précision de la forêt

Sobol-MDA pour Random Forest - Introduction 2

-l'algorithme Sobol-MDA mime l'algorithme de force brute de réentraînement de forêt avec une covariables en moins pour mesurer la diminution de précision mais avec un coût plus faible qu'un véritable réentraînement de forêt

-Sobol MDA requiert seulement d'obtenir des prédictions du Random-Forest, qui se calcule plus vite qu'une construction de forêt

-l'indice de Sobol-MDA converge vers l'indice de Sobol total si l'on modifie légèrement la construction du random forest, donc il vise bien la quantité appropriée

Analyse de sensibilité - objectifs

1) Maximiser la précision de la sortie avec un petit nombre de variables
2) Détecter et classer les variables les plus influents pour l'interprétation du modèle

Si deux variables influentes sont fortement corrélées alors si on vise l'obj. 1) on en garde une seule des deux mais si on vise l'obj. 2) on les garde toutes les deux.

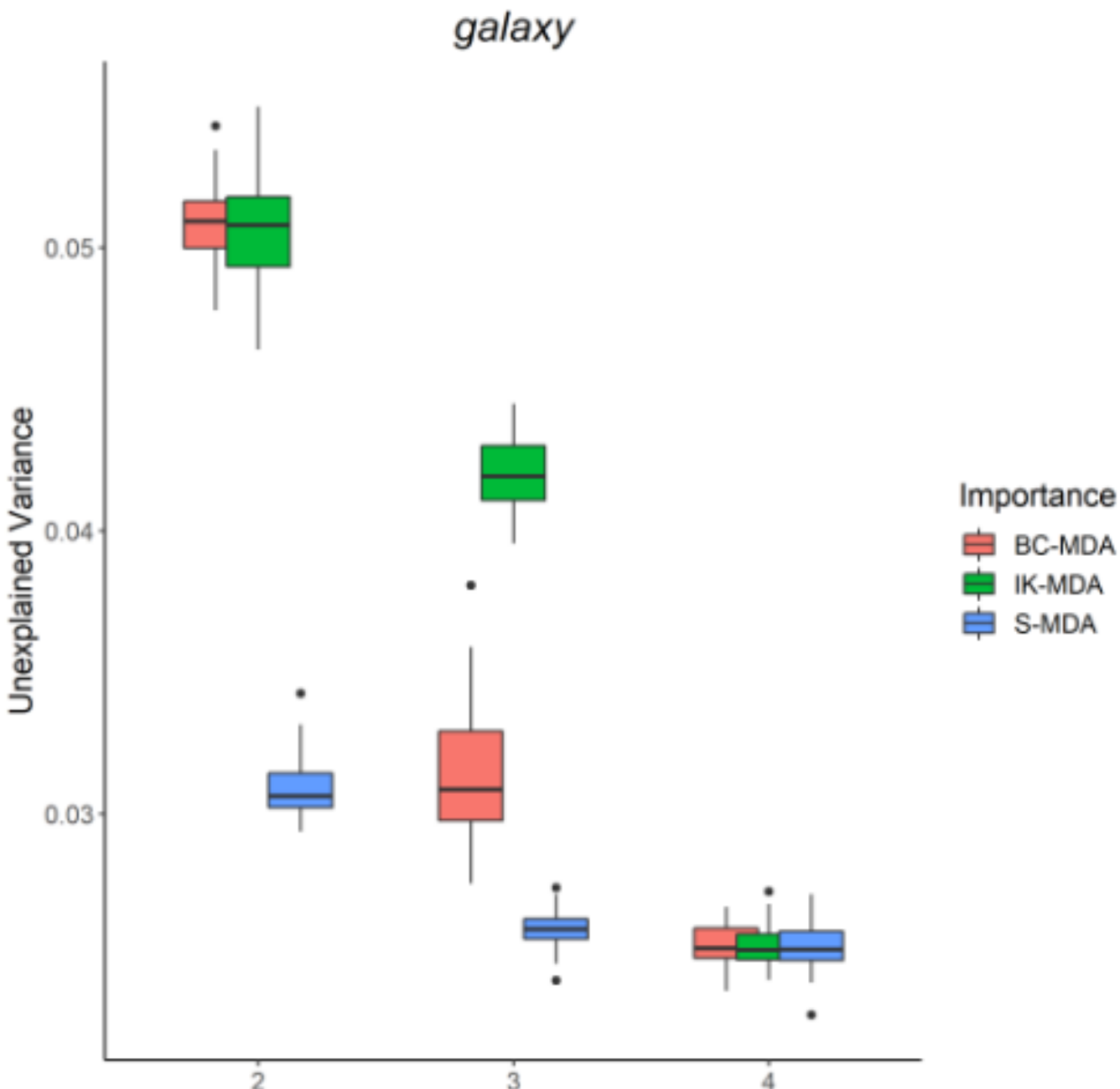
Sobol-MDA - algorithme

Algorithm 3 Projected-CART

Input: A Θ-random CART built with $\mathscr{D}_n$, and a variable index $j \in \{1, \ldots, p\}$. (Note that if a terminal leave occurs before the final tree level, it is copied at each level down the tree.)
Initialize both in-bag and OOB samples at the root node of the tree;
for all tree levels:
    for all level nodes:
        if the splitting variable is not j:
            send each data point to the right or left children node according to the node split;
        if the splitting variable is j:
            send the node sample to both the right and left children node ignoring the split;
    for all data points:
        retrieve the collection of nodes where the data point falls at the current tree level;
    for all OOB data points:
        retrieve the set of in-bag points which fall in the same node collection;
        if all nodes in the considered node collection are terminal:
            compute the output average of the in-bag points;
            set this average as the prediction for the considered OOB observation;
        if no in-bag points fall in the same node collection:
            retrieve the corresponding in-bag data points at the previous tree level;
            set the output average of these in-bag points as the prediction for the considered OOB observation;
return predictions;

Figure 1: Sobol-MDA algorithme

Sobol-MDA - exemple



Sobol-MDA - exemple

	BC-MDA*	$\widehat{\text{BC-MDA}}$	IK-MDA*	$\widehat{\text{IK-MDA}}$	ST*	$\widehat{\text{S-MDA}}$	$\widehat{\psi_{n,j}}$	$\widehat{\text{S-MDA}}_{Ldg}$
$X^{(3)}$	0.47	0.37 (0.03)	0.47	0.43 (0.02)	*0.47*	0.45 (0.03)	0.42 (0.06)	0.43 (0.03)
$X^{(4)}$	0.21	0.10 (0.02)	0.37	0.14 (0.01)	*0.10*	0.08 (0.01)	0.06 (0.04)	0.13 (0.01)
$X^{(5)}$	0.21	0.09 (0.01)	0.37	0.13 (0.01)	*0.10*	0.08 (0.01)	0.06 (0.04)	0.13 (0.01)
$X^{(1)}$	0.64	0.24 (0.02)	1.0	0.29 (0.02)	*0.07*	0.05 (0.01)	0.03 (0.04)	0.22 (0.02)
$X^{(2)}$	0.64	0.24 (0.02)	1.0	0.28 (0.02)	*0.07*	0.05 (0.01)	0.03 (0.04)	0.23 (0.01)

Table 2: BC-MDA (normalized by $2\mathbb{V}[Y]$), IK-MDA (normalized by $\mathbb{V}[Y]$), Williamson et al. (2021) ($\widehat{\psi_{n,j}}$), and Sobol-MDA estimates for Example 1 (standard deviations over 10 repetitions in brackets). Theoretical counterparts are defined in Proposition 2.

Figure 3: Comparaison

Analyse de sensibilité globale avec des mesures de dépendance

-Dans certains cas la variance ne représente pas très fidèlement la variabilité de la distribution c'est pour cela qu'à été introduite la mesure d'importance du moment indépendant.

-définir la dépendance entre la sortie Y et chaque paramètre d'entrée X_k d'un point de vue probabiliste

-idée générale : trouver une fonction qui mesure la similarité entre la distribution de Y et celle de $Y|X_k$, l'impact de X_k sur Y est donné par $S_{X_k} = E_{X_k}(d(Y, Y|X_k))$

Hsic

- Le critère HSIC est défini comme la norme de Hilbert-Schmidt de l'opérateur de la covariance croisée :

$$\mathrm{HSIC}(X, Y)_{F,G} = \|\mathcal{C}_{XY}\|^2_{HS} = \mathrm{E}_{X,X',Y,Y'} k_{\mathcal{X}}(X, X') k_{\mathcal{Y}}(Y, Y')$$
$$+\mathrm{E}_{X,X'} k_{\mathcal{X}}(X, X') \mathrm{E}_{Y,Y'} k_{\mathcal{Y}}(Y, Y')$$
$$-2\mathrm{E}_{X,Y} \left[\mathrm{E}_{X'} k_{\mathcal{X}}(X, X') \mathrm{E}_{Y'} k_{\mathcal{Y}}(Y, Y')\right]$$

- $\mathrm{HSIC}(X, Y)_{F,G}$ vaut 0 si X et Y sont indépendantes
- indice de sensibilité basé sur le critère Hsic :

$$S_{X^k}^{HSIC_{F,G}} = R(X^k, Y)_{F,G}$$

où la corrélation de distance basée sur le noyau de kernel est donnée par

$$R^2(X, Y)_{F,G} = \frac{\mathrm{HSIC}(X, Y)_{F,G}}{\sqrt{\mathrm{HSIC}(X, X)_{F,F} \mathrm{HSIC}(Y, Y)_{G,G}}}$$

kernel-embedding Shapley effects

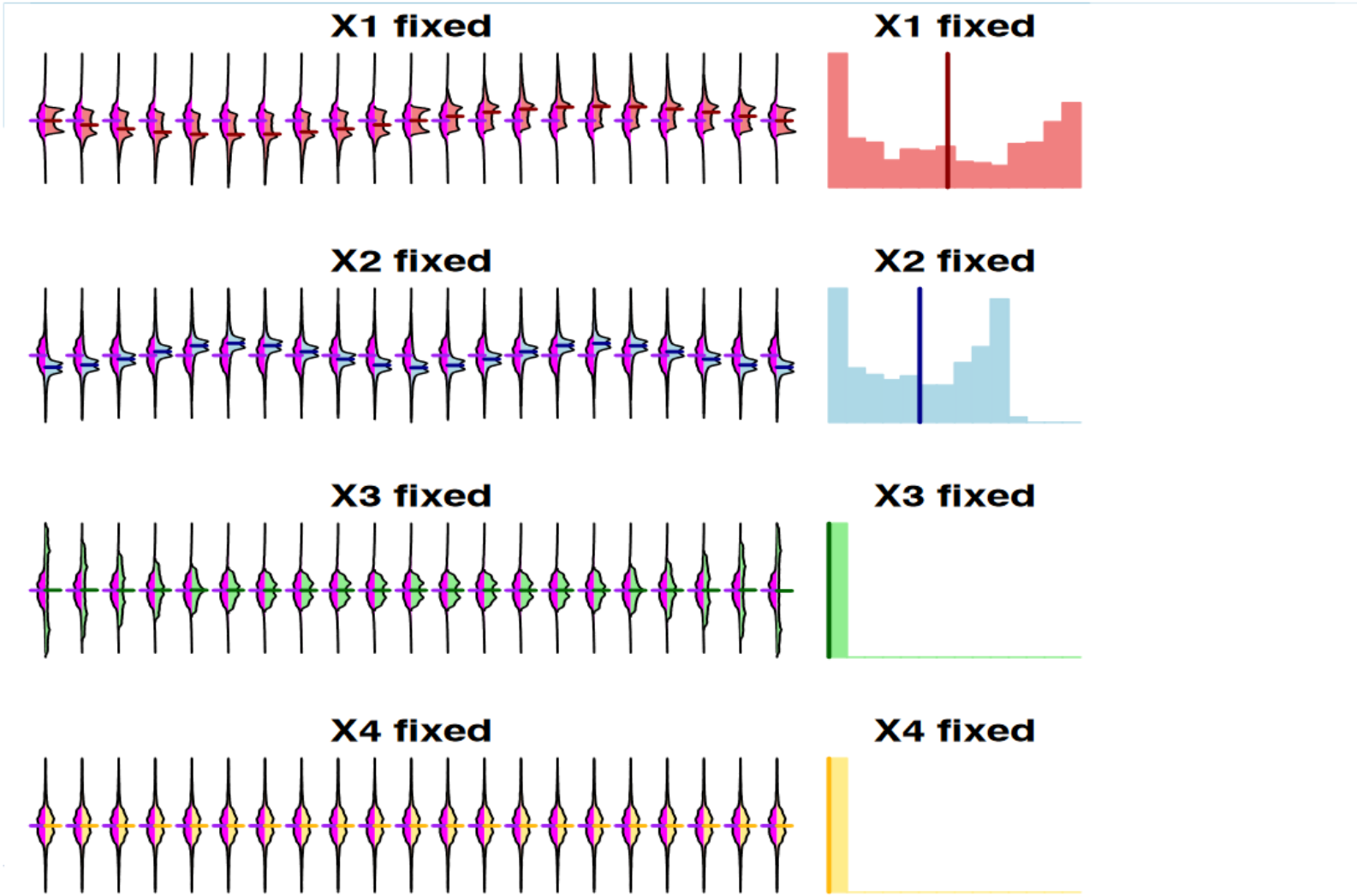
- L'effet de Shapley-HSIC est défini comme suit :

$$Sh_j^{HSIC} = \frac{1}{HSIC(X,Y)} \frac{1}{p} \sum_{A \subset -j} \binom{p-1}{|A|}^{-1} (\text{HSIC}(X_{A \cup \{j\}}, Y) - \text{HSIC}(X_A, Y$$

- Propiété :

$$\sum_{j=1}^{d} Sh_j^{HSIC} = 1$$

Répartition



Sobol-MDA - explications - notations mathématiques

On note $A_n(X, \Theta)$ la cellule de la partition de l'arbre d'origine où X

Sobol-MDA - explications 2 - explication de l'algorithme

Les données d'entraînement et les échantillons OOB sont mis dans l'arbre et envoyés à droite et gauche du noeud si celui fait une division sur la covariable j. A la fin chaque observation peut donc se retrouver dans plusieurs feuille terminales. Pour chaque donnée OOB, la prédiction associée est donc la moyenne des sorties des données d'entraînement qui sont tombées sur les mêmes feuilles terminales. En d'autres termes, on calcule l'intersection de ces feuilles terminales pour sélectionner les observations d'entraînement appartenant à chaque cellule de cette collection afin d'estimer la prédiction. Cette intersection donne la cellule projetée. Ce mécanisme est équivalent à projeter la partition de l'arbre sur le sous-espace engendré par $X^{(-j)}$