Quantification des incertitudes et analyse de sensibilité pour codes de calcul à entrées fonctionnelles et dépendantes

Simon Nanty

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques appliquées**

Arrêté ministériel : 7 août 2006

Présentée par

Simon Nanty

Thèse dirigée par **Clémentine Prieur**

préparée au sein **du CEA Cadarache**
et de **l'Ecole Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique**

Quantification des incertitudes et analyse de sensibilité pour codes de calcul à entrées fonctionnelles et dépendantes

Thèse soutenue publiquement le **15 octobre 2015**,
devant le jury composé de :

Mme Béatrice Laurent
Professeur, INSA Toulouse, Rapporteur
M. Hervé Monod
Directeur de recherche, INRA, Rapporteur
M. Fabrice Gamboa
Professeur, Université Paul Sabatier, Examinateur
M. Bruno Sudret
Professeur, ETH Zurich, Président
Mme Clémentine Prieur
Professeur, Université Joseph Fourier, Directrice
Mme Céline Helbert
Maître de Conférence, Ecole Centrale de Lyon, Co-directrice
Mme Amandine Marrel
Ingénieur de recherche, CEA, Co-encadrante
Mme Nadia Pérot
Ingénieur de recherche, CEA, Co-encadrante

Thèse réalisée dans le cadre d'un Contrat de Formation par la Recherche au Commissariat à l'Energie Atomique et aux énergies alternatives de Cadarache.

Remerciements

Mes remerciements s'adressent tout d'abord à mes quatre encadrantes pour m'avoir accompagné et aidé durant ces trois ans ; leur suivi régulier et leurs conseils m'ont permis d'avancer. Je tiens à remercier Amandine Marrel pour sa grande implication, son soutien tout au long de cette thèse. Je remercie aussi Clémentine Prieur d'avoir accepté de diriger cette thèse ainsi que pour son expertise scientifique et sa grande disponibilité pour répondre à mes questions. Je voudrais aussi remercier Céline Helbert pour l'aide qu'elle m'a apportée, son suivi et sa disponibilité. J'adresse enfin mes remerciements à Nadia Pérot pour ses conseils et son suivi.

Je tiens ensuite à remercier Béatrice Laurent et Hervé Monod d'avoir accepté d'être les rapporteurs de cette thèse ainsi que pour leurs commentaires intéressants et très pertinents sur ce manuscrit. Je voudrais aussi remercier Fabrice Gamboa et Bruno Sudret d'avoir accepté de faire partie de mon jury de thèse.

Je voudrais aussi remercier Bertrand Iooss d'avoir suivi mes travaux de thèse ainsi que pour ses conseils. Je tiens aussi tout particulièrement à remercier Vincent Moutoussamy et Benoît Pauwels avec qui j'ai travaillé durant les 6 semaines du CEMRACS 2013 : une expérience très intéressante et enrichissante ! Je remercie aussi Sébastien Da Veiga pour le temps qu'il a consacré à nous aider, pour ses conseils très utiles et sa relecture finale, ainsi qu'Anthony Nouy pour l'aide et les idées qu'il nous a apportées pendant ce projet.

J'adresse aussi mes remerciements à tous mes collègues du CEA que j'ai eu le plaisir de côtoyer pendant ces trois ans.

Table des matières

Chapitre 1

Introduction

1.1 Contexte

Dans le cadre des études de sûreté et de maîtrise des risques pour les réacteurs nucléaires, les simulateurs numériques ou codes de calcul sont des outils essentiels pour modéliser, étudier et prédire les différents phénomènes physiques mis en jeu. Ces simulateurs calculent, à partir de plusieurs paramètres caractéristiques du phénomène physique étudié (caractéristiques du réacteur, paramètres thermo-hydrauliques ou neutroniques, etc.), des quantités d'intérêt (ou variables de sortie) décrivant l'évolution du phénomène. Dans le cadre des études de sûreté, les systèmes modélisés sont complexes, et de nombreux phénomènes physiques (thermo-hydrauliques, mécaniques, neutroniques) se déroulent et interagissent lors du scénario étudié. Par conséquent, ces codes peuvent s'avérer très coûteux en temps de calcul, ce qui limite le nombre de simulations possibles. La connaissance des paramètres d'entrée du code est souvent limitée à cause du manque de caractérisation (erreur de mesure ou manque de données), de la variabilité intrinsèque du phénomène, ou encore d'erreurs numériques. La prise en compte des incertitudes sur ces paramètres d'entrée et en particulier l'étude de leur effet sur la sortie du code est une étape essentielle dans les études de sûreté.

De nombreuses méthodes ont été développées pour étudier les incertitudes liées aux paramètres des codes de calcul et quantifier leur impact sur la sortie de ces codes. Dans la littérature consacrée à l'analyse et au traitement des incertitudes de codes de calcul (Helton et al., 2006; De Rocquigny et al., 2008; Kleijnen, 2007), la démarche est usuellement décomposée en quatre étapes, illustrées par la Figure 1.1 issue de Baudin et al. (2015). Dans l'étape A, le modèle ou l'ensemble de modèles étudiés est défini. Les grandeurs d'intérêt sont ensuite identifiées. L'étape B consiste à quantifier les sources d'incertitudes, c'est-à-dire à identifier les entrées incertaines et à modéliser leurs incertitudes. Dans l'étape C, les incertitudes des paramètres d'entrée quantifiées dans la deuxième étape sont propagées à travers le code de calcul. L'objectif de cette étape de propagation des incertitudes est de quantifier comment l'incertitude sur les entrées se répercute sur la sortie du simulateur. L'étape C', complémentaire de l'étape C et appelée analyse de sensibilité, vise à identifier comment la variation de chacun des paramètres d'entrée contribue, qualitativement ou quantitativement, à la variation de la sortie. En déterminant les paramètres ou les interactions entre paramètres les plus influents, ceux ayant une influence négligeable, ou encore en hiérarchisant leurs contributions à la variabilité de la sortie, l'analyse de sensibilité peut permettre de valider le modèle numérique étudié, d'orienter les efforts de caractérisation sur les paramètres les plus influents, de simplifier le modèle et d'améliorer la compréhension du phénomène modélisé en identifiant des relations entre les variables d'entrée et variables de sortie.

Dans cette thèse, on s'intéresse plus particulièrement au cas de paramètres d'entrée fonctionnels (des courbes temporelles, par exemple). La prise en compte des incertitudes liées à ce type de variables nécessite le développement d'outils statistiques spécifiques et performants. Cette thèse a pour objectif de proposer des méthodologies et des outils associés pour prendre en compte les incertitudes liées aux variables aléatoires fonctionnelles

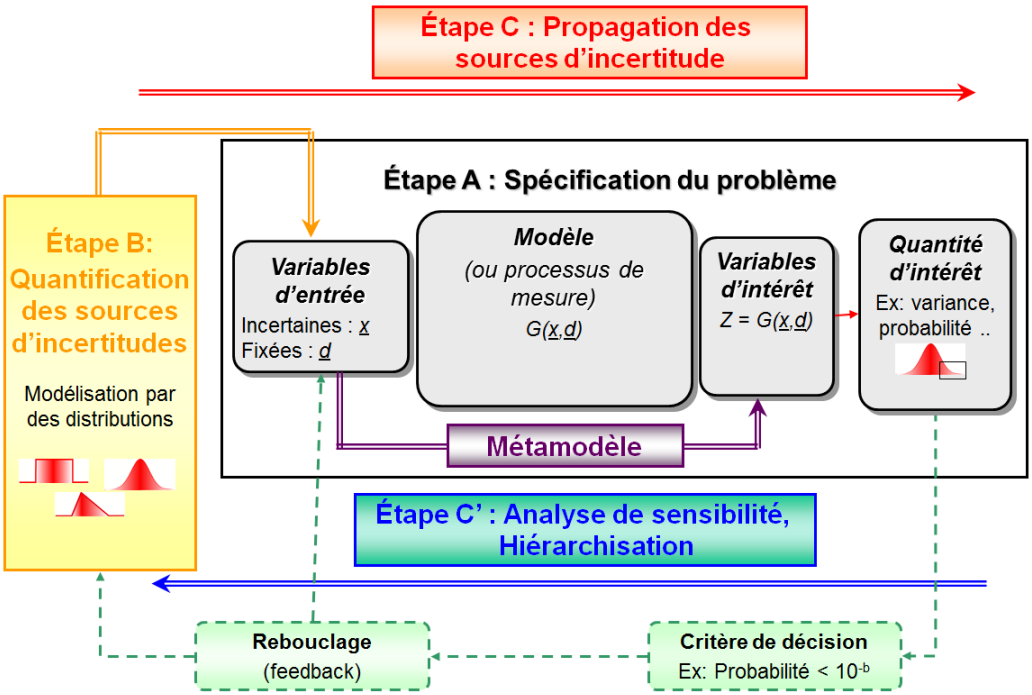

FIGURE 1.1 – Schéma de la méthodologie générale de traitement des incertitudes, issu de Baudin et al. (2015).

en support à l'analyse de sensibilité et à la propagation des incertitudes. Avant de présenter les différentes sous-problématiques abordées dans cette thèse, les deux cas d'étude qui ont motivé ces travaux sont détaillés.

1.2 Description des deux cas d'application

Deux exemples issus des études de sûreté nucléaire sont étudiés. Dans ces cas d'étude, des situations accidentelles sont modélisées :

- le choc thermique pressurisé (ou PTS pour *Pressurised Thermal Shock*) sur un réacteur à eau pressurisée dans le cadre des études en soutien aux réacteurs du parc actuel,
- la rupture de la Liaison Pompe-Sommier (LiPoSo) sur un réacteur rapide refroidi au sodium dans le cadre des études sur les réacteurs de quatrième génération.

Les phénomènes physiques en jeu dans les accidents étudiés, les codes de calcul les modélisant ainsi que les jeux de données disponibles sont décrits dans ce qui suit.

1.2.1 Choc thermique pressurisé (PTS)

1.2.1.1 Présentation du phénomène étudié

Dans le cadre de l'étude de la durée de vie des cuves de réacteurs à eau pressurisée, la tenue mécanique des cuves est étudiée notamment lors de situations accidentelles. L'un des types d'accidents étudiés, la perte de réfrigérant primaire dans le réacteur, peut entraîner un choc thermique pressurisé.

La perte de réfrigérant est initiée par une brèche dans le circuit primaire du réacteur. Ce circuit contient un fluide caloporteur, l'eau pressurisée qui permet le transfert de la chaleur de la cuve au circuit secondaire qui, par l'entraînement d'un alternateur, permet de générer de l'électricité. L'eau du circuit primaire a une pression d'environ 150 bar et une température moyenne de 285°C. Au cours de

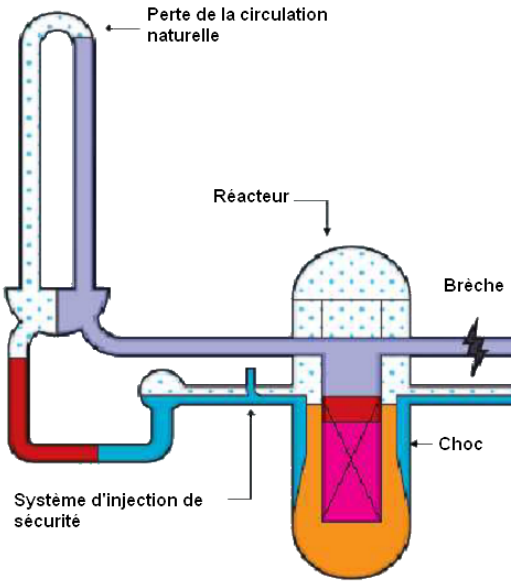

FIGURE 1.2 – Schéma d'une brèche sur le circuit primaire entraînant un choc thermique pressurisé.

cet accident hypothétique, la brèche entraîne une chute de la pression dans le circuit primaire et une perte de l'inventaire en eau de ce circuit. Plusieurs systèmes de sécurité sont alors mis en service, pour éviter l'échauffement des crayons de combustible. Parmi eux, le système d'injection de sécurité compense la perte d'eau due à la brèche en ajoutant au circuit primaire de l'eau issue d'un réservoir dédié. Cette eau à température ambiante entre en contact avec les parois plus chaudes de la cuve, ce qui entraîne une diminution rapide de la température dans le réacteur. Le choc thermique pressurisé correspond à cette baisse rapide de température et de pression. Si l'acier dont est composée la cuve comporte des défauts, ce choc peut entraîner la création de fissures et leur propagation sur toute l'épaisseur de la paroi. Le schéma de la Figure 1.2 illustre les principales étapes du processus du PTS.

Une fissure apparaît dans le matériau si l'intensité de contrainte appliquée au niveau de la fissure dépasse la ténacité de l'acier de la cuve. La ténacité d'un matériau, notée K_{IC}, est définie comme la capacité du matériau à résister à une fissure. Celle-ci dépend de la composition du matériau et de l'irradiation reçue. L'intensité de contrainte appliquée au niveau de la fissure lors du PTS dépend à la fois des caractéristiques du choc (la température, la pression...) et des caractéristiques du défaut où est appliquée la contrainte (taille, position sur la cuve...). Cette intensité de contrainte est notée K_I. L'évaluation de la possibilité de rupture se mesure à l'aide d'un Critère de Sécurité (noté CS). Ce critère est une grandeur adimensionnelle définie comme suit :

$$\mathrm{CS} = \min_t f\left(K_{IC}(t), K_I(t)\right), \tag{1.1}$$

où f est une fonction de deux variables, non précisée par souci de confidentialité. Si le CS est inférieur à une certaine valeur critique cs^*, il existe un risque de propagation de fissures et de rupture de la cuve. Cette valeur critique est un seuil à ne pas dépasser.

1.2.1.2 Workflow thermo-hydraulique – thermo-mécanique

Les différents phénomènes physiques intervenant dans le choc thermique pressurisé et décrits précédemment sont modélisés par le chaînage de plusieurs codes de calcul thermo-hydrauliques (T-H) et thermo-mécaniques (T-M), comme illustré par la Figure 1.3. Tout d'abord, les calculs T-H sont réalisés par l'ensemble des codes chaînés CATHARE2-SSTH-VESTALE que l'on appellera, par la suite, workflow

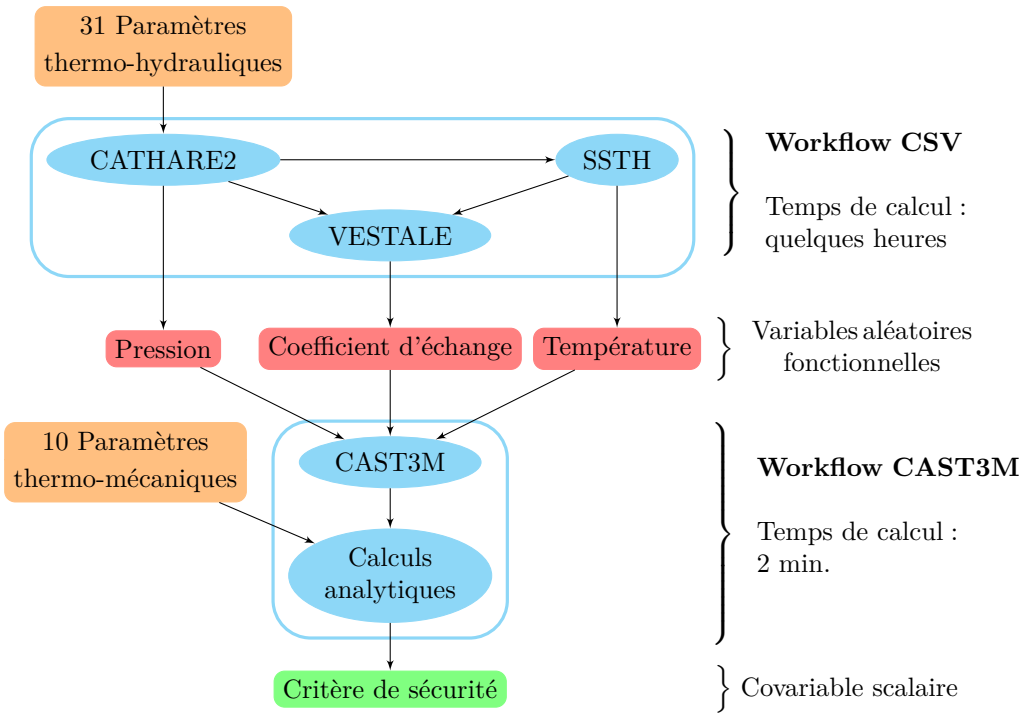

FIGURE 1.3 – Workflow pour le calcul du critère de sécurité dans le cas d'un choc thermique pressurisé.

CSV. Cet ensemble de codes permet le calcul des trois transitoires de température, pression et coefficient d'échange thermique entre le fluide caloporteur et la paroi de la cuve au niveau du défaut étudié. Ces transitoires sont des fonctions du temps qui décrivent ces trois grandeurs physiques durant 5000 secondes après le début de l'accident. Dans le scénario étudié, 31 paramètres T-H incertains interviennent en entrée du code CATHARE2. Ces variables T-H définissent l'état initial de la cuve, les caractéristiques du réacteur et celles du système d'injection de sécurité. Lors de précédentes études, leurs distributions de probabilité ont été attribuées par avis d'expert ou ont été estimées à partir de données disponibles. Une description de ces paramètres et de leurs lois de probabilité est donnée par la Table 1.1. A partir de ces paramètres d'entrée, le modèle CATHARE2 calcule les transitoires de plusieurs grandeurs physiques comme les températures, les pressions et les débits en différents endroits de la cuve, et notamment la pression au niveau du défaut. Ces données sont ensuite utilisées par le code SSTH pour calculer la température de l'eau pressurisée au niveau du défaut. Enfin, le code VESTALE calcule à partir des sorties de CATHARE2 et de SSTH le coefficient d'échange thermique, qui quantifie le flux d'énergie traversant le liquide par unité de surface. **Une simulation du workflow CSV peut ainsi nécessiter plusieurs heures.**

Les entrées du code CAST3M sont les trois transitoires de température, pression et coefficient d'échange au niveau de la fissure fournies par le workflow CSV, ainsi que 10 autres paramètres T-M scalaires et incertains. Ces entrées, présentées dans la Table 1.2, décrivent le défaut étudié, des caractéristiques physiques et chimiques du matériau ainsi que des paramètres neutroniques du réacteur. A partir de ces 13 entrées, le simulateur numérique CAST3M calcule les champs de contrainte et de température appliqués à la cuve du réacteur. Un modèle analytique, prenant en entrée les champs de contrainte et de température, calcule ensuite le CS défini dans l'équation (1.1). Par soucis de concision, l'ensemble composé de CAST3M et de ce modèle analytique est appelé workflow CAST3M par la suite. **Le temps d'exécution de ce workflow est d'environ deux minutes.**

PAR	Nom	Type Loi	Paramètre 1	Paramètre 2	Paramètre 3	Paramètre 4
1	Pression Pressuriseur (Pa)	Normale	0	107143	-321429	321429
2	Niveau Pressuriseur (m)	Normale	0	0,05102	-0,153	0,153
3	Vitesse Pompes (t/mn)	Normale	1	0,0102	0,969	1,031
4	Pression secondaire (Pa)	Normale	0	102041	-306122	306122
5	Puissance résiduelle	Uniforme	Courbe min f(t)	Courbe max f(t)		
6	Température d'alimentation en secours des générateurs de vapeur (°C)	Normale	33,5	13,52	7	60
7	Température d'injection de sécurité (°C)	Normale	20	4,7	10	35
8	Débit d'injection de sécurité	Uniforme	Courbe min f(P)	Courbe max f(P)		
9	Température accumulateurs (°C)	Normale	0	2,55	-5	10
10	Perte de charge ligne accumulateurs	Lognormale	0	0,354	0	2,23
11	Pression initiale accus (Pa)	Normale	0	102041	-306122	306122
12	Volume liquide accu (m3)	Normale	1	0,0255	0,923	1,077
13	Bypass dôme : coefficient de perte de charge singulière	Lognormale	0	0,354	0	2,23
14	Frottement interfacial : cœur	Lognormale	-0,490	0,718	0	2,743
15	Frottement interfacial : plenum supérieur	Lognormale	-5.10^{-4}	0,561	0	3,303
16	Frottement interfacial : stratifié	Normale	0,9225	0,152	0,467	1,378
17	Hauteur limite d'entrainement (brèche)	Lognormale	-0,420	0,661	0	2,633
18	Saturation (brèche) : Qle flashing	Lognormale	-0,715	0,0313	0,443	0,535
19	Saturation (brèche) : frottement Liquide-Paroi	Normale	14	1,0204	10,94	17,06
20	Saturation (brèche) : frottement interfacial	Lognormale	1,242	0,634	0	13,17
21	Sous-Saturation (brèche) : flashing	Lognormale	0,193	0,181	0,557	1,909
22	Sous-Saturation (brèche) : flashing delay	Normale	3,14	0,872	0,523	5,757
23	Sous-Saturation (brèche) : frottement Liquide-Paroi	Normale	4	2,91	0	12,72
24	Convection liquide tubes générateur de vapeur primaire	Normale	1	0,153	0,541	1,459
25	Echange de chaleur paroi-fluide : convection forcée liquide	Lognormale	0	0,354	0	2,230
26	Echange de chaleur paroi-fluide dans le downcomer : convection forcée liquide	Lognormale	0	0,354	0	2,230
27	Echange de chaleur liquide-interface due à l'impact du jet	Lognormale	-0,548	0,461	0	1,581
28	Echange de chaleur liquide-interface en écoulement stratifié	Lognormale	0,466	0,602	0	5,699
29	Echange de chaleur liquide-interface tous régimes	Lognormale	-5.10^{-4}	0,561	0	3,303
30	Taux de stratification	Triangle	1	1,3	2	
31	Puissance nominale	Normale	1	0,0102	0,969	1,031

Paramètre 1	Paramètre 2	Paramètre 3	Paramètre 4
• Moyenne pour loi normale • Moyenne du log pour loi lognormale • Borne min pour loi uniforme • Borne min pour loi triangle	• Ecart-type pour loi normale • Ecart-type du log pour loi lognormale • Borne max pour loi uniforme • Mode pour loi triangle	• Borne min (-3 écart-types) pour loi la loi normale tronquée • Borne min (-3 écart-types) pour loi lognormale tronquée • Borne max pour T	• Borne max (+3 écart-types) pour loi normale tronquée • Borne max (+3 écart-types) pour loi lognormale tronquée

TABLE 1.1 – Paramètres T-H incertains en entrée du code CATHARE2 et distributions de probabilité associées.

Variable	Symbole	Distribution	Param. 1	Param. 2
Hauteur du défaut (mm)	H	Weibull	1,8	3,08
Hauteur/Longueur	a/c	Log-normale	-1,52	0,5
Fluence maximum à l'interface bimétallique (cm^{-2})	F_0	Normale	6	0,18
Distribution azimutale du flux neutronique	K_α	Uniforme	0	1
Teneur en phosphore (%)	P	Normale	8.10^{-3}	10^{-3}
Teneur en cuivre (%)	Cu	Normale	9.10^{-2}	10^{-2}
Teneur en nickel (%)	Ni	Normale	0,72	0,1
Température de transition initiale (°C)	RT_{ndt0}	Normale	-20	10
Aléa sur la température de transition	RT_{ndt}	Normale	0	1
Aléa sur la ténacité	K_{IC}	Uniforme	0	1

TABLE 1.2 – Paramètres T-M incertains en entrée du code CAST3M et leurs distributions de probabilité associées. Pour la distribution normale, les paramètres sont respectivement la moyenne et la variance. Pour la loi uniforme, ce sont les bornes minimales et maximales de l'intervalle. Pour la loi log-normale, ce sont la moyenne et la variance du logarithme de la variable. Pour la loi de Weibull, ce sont les paramètres de forme et d'échelle.

1.2.1.3 Jeu de données

L'étude du cas du PTS doit répondre à deux objectifs :
- caractériser et quantifier les incertitudes sur les différents transitoires T-H en sortie du workflow CSV. Associé à cette caractérisation, un outil de visualisation est proposé afin d'identifier les caractéristiques des profils de ces transitoires et d'avoir une meilleure compréhension du phénomène.
- évaluer l'impact des transitoires T-H incertaines sur le CS et en particulier comparer cet effet à celui des variables T-M incertains en entrée du workflow CAST3M.

Pour répondre au premier objectif, un plan d'expériences de triplets de transitoires est obtenu en évaluant le workflow CSV sur un échantillon probabilisé de réalisations indépendantes de ses 31 paramètres d'entrée scalaires et incertains (cf. Table 1.1). Afin de construire un échantillon ayant de bonnes propriétés de répartition sur les marginales, un plan hypercube latin (McKay et al., 1979) est utilisé pour construire l'échantillon. De plus, ce plan est optimisé selon une mesure de l'uniformité de l'échantillonnage, appelée discrépance *wrap-around* (cf. Fang et al. 2006 et section 3.3.1.1), dans le but d'assurer un bon recouvrement de l'espace des entrées. En raison du temps de calcul du workflow CSV, il a été choisi de limiter la taille de l'échantillon à 1000. La Figure 1.4 présente une partie de l'échantillon des transitoires T-H. De la même manière, un échantillon de tirages indépendants des paramètres d'entrée T-M du workflow CAST3M est construit. Le workflow CAST3M est ensuite évalué sur le plan d'expériences combinant aléatoirement les réalisations des transitoires T-H et des variables T-M. 1000 valeurs du CS sont alors obtenues. Ces réalisations du CS pourront être utilisées lors de la caractérisation des incertitudes des transitoires T-H afin que celle-ci prenne bien en compte le lien avec le CS.

Pour répondre au deuxième objectif lié à l'analyse de sensibilité du CS, on pourra réaliser davantage de simulations du workflow CAST3M, celui-ci étant plus rapide à évaluer. Pour cela, on pourra utiliser la caractérisation probabiliste des transitoires T-H pour générer davantage de réalisations de ces transitoires, en entrée du workflow CAST3M.

1.2.2 Rupture de la liaison pompe-sommier (LiPoSo)

1.2.2.1 Présentation du phénomène étudié

Dans le cadre des études de préconception d'un prototype de réacteur à neutrons rapides refroidi au sodium liquide de génération IV, le CEA étudie différentes familles de scénarios d'accidents graves. Le cœur du réacteur est essentiellement composé de sous-assemblages contenus dans des tubes hexagonaux

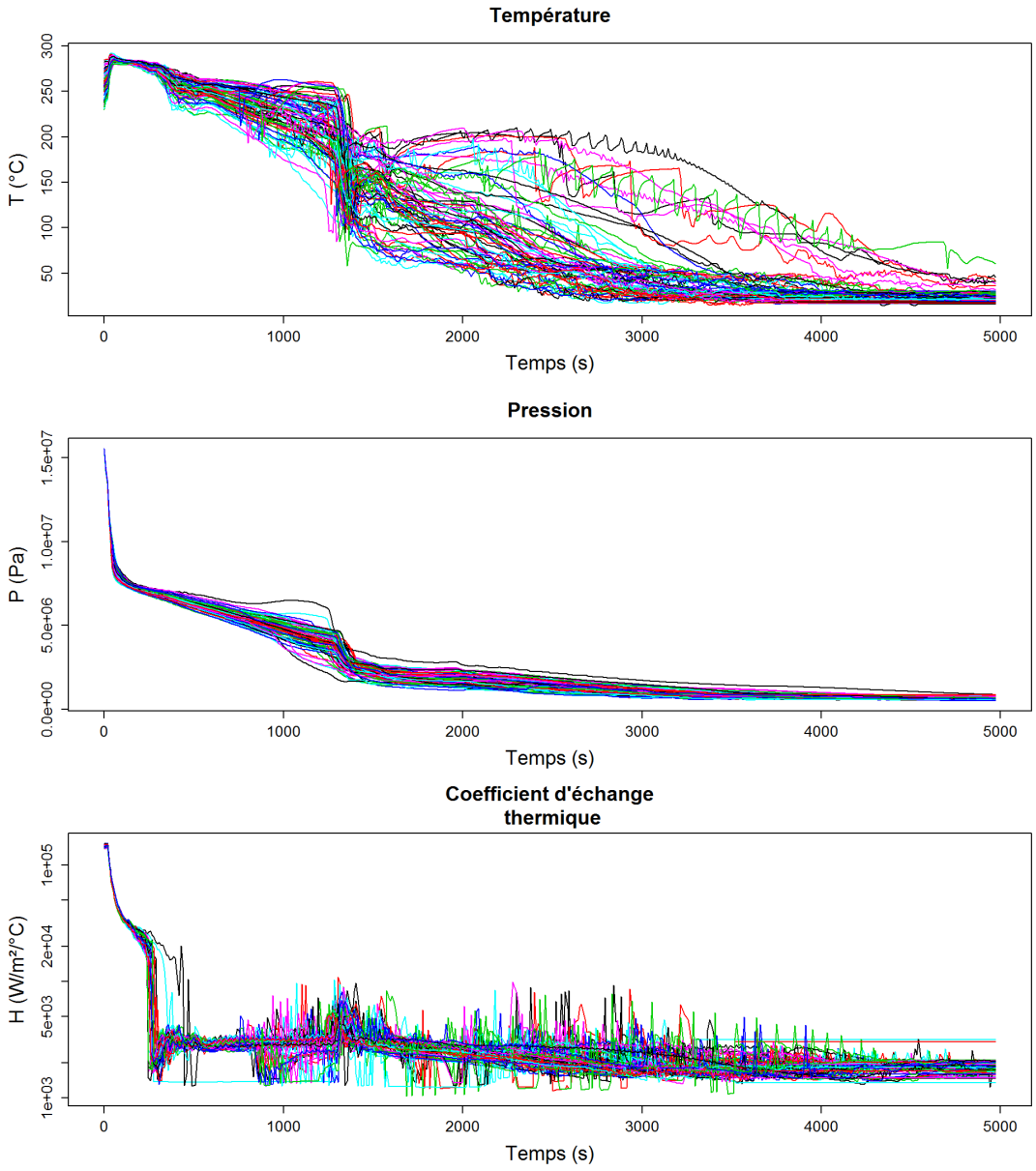

FIGURE 1.4 – Exemples de transitoires de température, pression et coefficient d'échange thermique en sortie du workflow CSV.

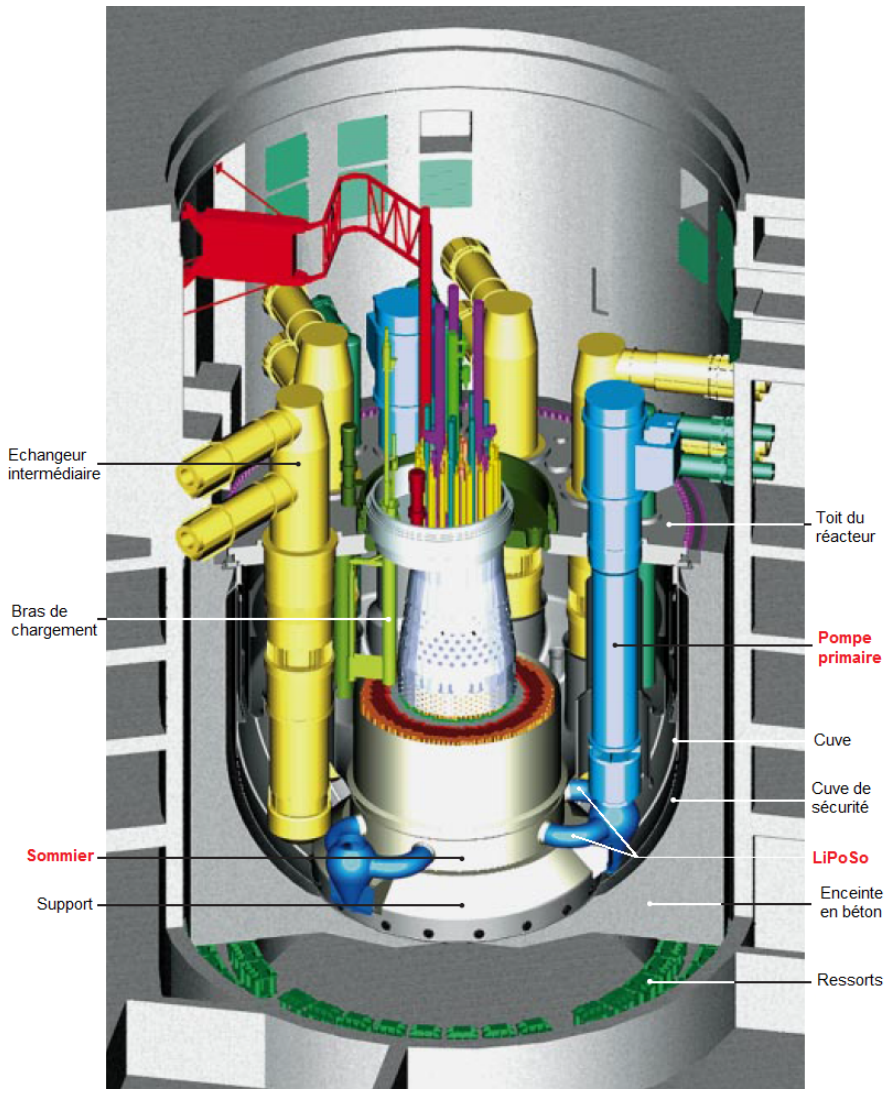

FIGURE 1.5 – Schéma d'un réacteur rapide au sodium (European Fast Reactor).

et comprenant les crayons de combustible. Ces crayons sont entourés d'un fluide caloporteur (du sodium liquide dans le type de réacteur considéré) qui transfère la chaleur du cœur vers l'échangeur thermique. Dans cette section, nous nous concentrons sur un accident initié par la rupture de la connexion entre la pompe du circuit primaire et le sommier du réacteur, sur lequel sont disposés les sous-assemblages du cœur, comme on peut le voir sur la Figure 1.5. La rupture de la conduite entre la pompe du système de refroidissement et le sommier, en bas du cœur, entraîne une baisse rapide de débit du fluide caloporteur dans le cœur du réacteur. Compte tenu du rapport des pertes de charge entre le circuit hydraulique normal (passant par le cœur et les échangeurs intermédiaires) et le nouveau circuit de bypass créé par la brèche, la température du sodium et des crayons de combustible augmente rapidement. Dans le scénario considéré, l'Arrêt Automatique Rapide (AAR) est déclenché si le rapport puissance sur débit calculé en utilisant la mesure de la différence de pression entre la pompe primaire et le collecteur froid dépasse un certain seuil, ou si les températures en sortie d'assemblage mesurées sur un second poste dépassent un autre seuil.

Les hautes températures subies par le cœur du réacteur peuvent l'endommager et, par exemple,

Type	Variable	Distribution	Param. 1	Param. 2	Param. 3
Caractéristiques cœur (scalaires)	Diamètre des aiguilles (m)	Normale	$9,7.10^{-3}$	$2,55.10^{-5}$	-
	Epaisseur des gaines (m)	Normale	5.10^{-4}	$1,53.10^{-5}$	-
	Diamètre des fils (m)	Uniforme	$0,98.10^{-3}$	$1,02.10^{-3}$	-
	Entreplat interne (m)	Normale	0,16155	$2,55.10^{-4}$	-
	Conductibilité acier ($W.m^{-1}.K^{-1}$)	Triangle	18	22	20
Coefficients multiplicatifs (scalaires)	Coefficient sur la puissance	Uniforme	0,92	1,08	-
	Coefficient sur le débit 1	Normale	1	$9,18.10^{-3}$	-
	Coefficient sur le débit 2	Uniforme	-6.10^{-4}	6.10^{-4}	-
Sorties CATHARE2 (fonctionnelles)	Transitoire de débit ($kg.s^{-1}$)	Non connue analytiquement			
	Transitoire de puissance (W)	Non connue analytiquement			
	Transitoire de température (°C)	Non connue analytiquement			

TABLE 1.3 – Variables incertaines en entrée de Trio_U MCT et leurs distributions de probabilité. Pour la distribution normale, les paramètres sont respectivement la moyenne et la variance. Pour la loi uniforme, ce sont les bornes minimales et maximales de l'intervalle. Pour la loi triangle, ce sont les deux bornes de l'intervalle de variation et le mode.

conduire à la rupture des gaines des crayons de combustible. Pour évaluer ce risque, il est nécessaire de prédire l'évolution des températures du sodium et des gaines suite à la rupture d'une LiPoSo. Pour cela, on utilise un workflow de codes thermo-hydrauliques chaînés.

1.2.2.2 Workflow CATHARE2-Trio_U MCT

Le code de calcul Trio_U MCT est utilisé pour modéliser la thermo-hydraulique du cœur du réacteur à l'échelle locale, appelée « échelle des sous-canaux » car le cœur y est divisé en sous-canaux dans lesquels s'écoule le fluide caloporteur. Le code Trio_U MCT est coûteux en temps de calcul puisqu'une évaluation dure deux heures. Dans cette étude, les paramètres d'entrée du code considérés comme incertains sont listés dans la Table 1.3. Les cinq premiers caractérisent le cœur du réacteur : quatre paramètres décrivent la géométrie du cœur et le cinquième est la conductibilité thermique de l'acier des gaines. Le simulateur Trio_U MCT prend aussi en entrée trois variables temporelles : les transitoires de débit, de puissance et de température, décrivant l'évolution temporelle de ces quantités au cours de l'accident. Ces variables sont dépendantes entre elles et sont issues du code de calcul thermo-hydraulique CATHARE2, utilisé pour modéliser à une échelle globale, appelée « échelle système », tous les composants du système primaire (cœur, plenum, pompes, échangeurs) et les boucles (système secondaire, système de refroidissement). La distribution de probabilité de ces trois variables aléatoires n'est donc pas directement connue. En effet, les incertitudes associées aux variables fonctionnelles sont une transformation de celles des paramètres d'entrée de CATHARE2. En entrée de ce code, quatre paramètres incertains sont pris en compte. Ces paramètres incertains ainsi que leurs distributions de probabilité sont décrits dans la Table 1.4. CATHARE2 est aussi considéré comme coûteux en temps de calcul puisqu'une évaluation de ce code prend environ une heure dans ce cas d'étude. Pour modéliser les incertitudes liées à des propriétés du système qui ne peuvent pas être ajustées dans les codes Trio_U MCT et CATHARE2, trois coefficients multiplicatifs appliqués au débit et à la puissance sont introduits. Leurs distributions de probabilité sont décrites dans la Table 1.3. **Le code Trio_U MCT a donc au total 11 paramètres incertains : 8 entrées scalaires (les paramètres géométriques, la conductibilité de l'acier et les coefficients multiplicatifs), notées $X_1, \dots X_8$, dont les distributions de probabilité sont connues, et 3 entrées fonctionnelles, notées Z_1, Z_2, Z_3, dont les distributions de probabilité ne sont pas directement connues.**

A partir de ces paramètres d'entrée, le simulateur Trio_U MCT calcule les températures des gaines et du sodium dans tout le cœur du réacteur. Le processus de calcul de ces températures est résumé par

Variable	Distribution	Param. 1	Param. 2
Délai sur le seuil de réactivité (s)	Uniforme	0	1
Durée d'ouverture de la brèche (s)	Uniforme	0,2	1,8
Perte de charge transverse dans le sommier (%)	Uniforme	-20	20
Inertie des pompes primaires (%)	Uniforme	-20	20

TABLE 1.4 – Variables incertaines en entrée de CATHARE2 et leurs distributions de probabilité. Les paramètres sont respectivement les bornes minimales et maximales du support de la loi uniforme.

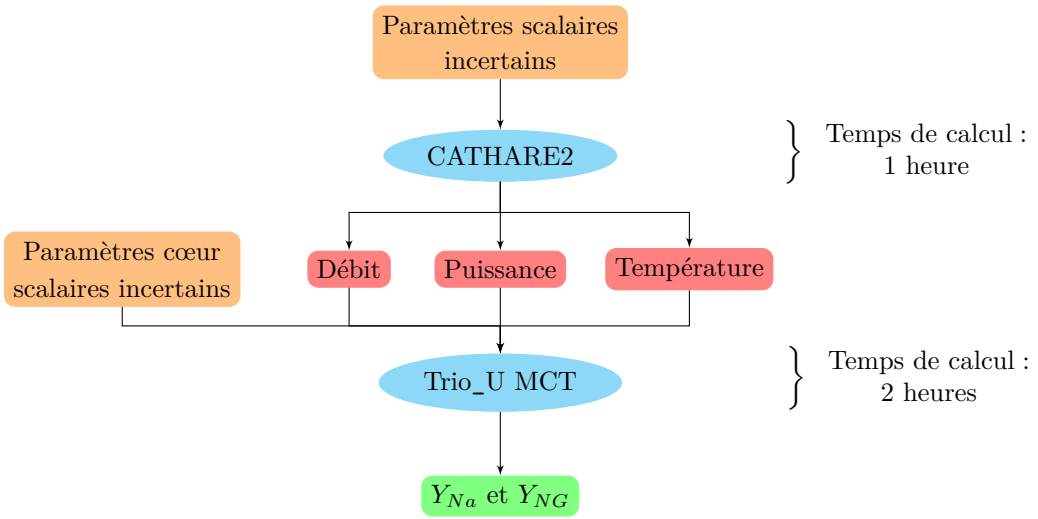

FIGURE 1.6 – Workflow pour le calcul des températures du sodium et de gaine dans le cas d'une rupture LiPoSo.

la Figure 1.6. On s'intéresse dans cette étude à quatre sorties du code Trio_U MCT :
- la température maximale du caloporteur sodium dans les assemblages de crayons de combustible, notée Y_{Na}, en fonction du temps,
- la température maximale des gaines des crayons situées en stockage interne, notée Y_{NG}, en fonction du temps,
- la valeur maximale de Y_{Na} sur l'intervalle de temps $I_s = [0; 20]$, notée Y_{Na}^{max},
- la valeur maximale de Y_{NG} sur l'intervalle de temps $I_s = [0; 20]$, notée Y_{NG}^{max}.

1.2.2.3 Jeu de données

Comme dans le cas du PTS, l'étude du cas de la rupture LiPoSo a deux objectifs :
- caractériser les incertitudes des transitoires T-H en sortie du code CATHARE2, afin d'obtenir une meilleure compréhension du phénomène,
- évaluer l'effet de ces transitoires incertains sur les températures de gaine et du sodium en sortie du code Trio_U MCT et le comparer à celui des autres variables scalaires incertaines en entrée de Trio_U MCT.

Pour répondre au premier objectif, on génère un échantillon probabilisé des transitoires en sortie de CATHARE2 en évaluant ce code sur un échantillon de réalisations indépendantes de ses quatre paramètres d'entrée scalaires incertains (cf. Table 1.4). Comme dans le cas du PTS, l'échantillon a été construit à partir d'un plan hypercube latin optimisé selon la discrépance *wrap-around* (cf. section 3.3.1.1 et Fang et al. 2006). Cet échantillon est limité à $n = 200$ réalisations en raison du temps de calcul élevé

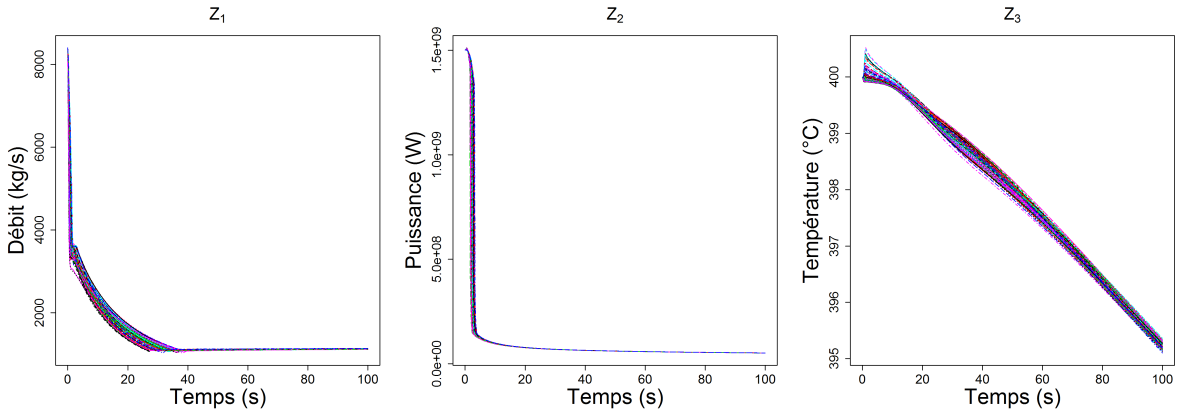

FIGURE 1.7 – Exemple de 70 réalisations des variables de débit, puissance et température, en entrée de Trio_U MCT.

du code CATHARE2. Quelques unes des réalisations des transitoires T-H sont représentées sur la Figure 1.7.

Pour répondre au deuxième objectif et réaliser l'analyse de sensibilité de ses différentes sorties, le code Trio_U MCT, très coûteux en temps de calcul, ne peut être utilisé directement. Une solution possible est de remplacer ce code par un modèle de substitution ou métamodèle, qui l'approche et a un temps d'évaluation très court. Pour construire ce métamodèle, un plan d'expériences spécifique doit être élaboré en utilisant la caractérisation des entrées fonctionnelles réalisée.

1.3 Problématiques et objectifs

Les deux cas d'application décrits dans la section précédente présentent des caractéristiques assez proches. Dans ces deux cas d'étude (PTS et LiPoSo), la modélisation des phénomènes mis en jeu repose sur le chaînage de deux simulateurs. Un premier code fournit des variables aléatoires fonctionnelles dépendantes dont on ne connaît pas la distribution de probabilité et qu'il est important de caractériser. Ces variables associées à d'autres variables scalaires incertaines constituent les variables d'entrée d'un second code. Cependant, ces deux cas-tests diffèrent par le temps de calcul nécessaire à chaque évaluation du simulateur : dans le cas du PTS, le second code a un temps d'exécution court (quelques minutes), alors que dans le cas de la rupture LiPoSo, il est coûteux en temps de calcul (quelques heures).

Ainsi, nous nous intéressons, dans ces deux applications, aux objectifs suivants :

- **quantifier les incertitudes des variables aléatoires fonctionnelles potentiellement dépendantes entre elles**,
- **visualiser ces incertitudes**,
- **réaliser une analyse de sensibilité** afin de déterminer l'influence des variables aléatoires fonctionnelles sur la sortie du code,
- **construire un métamodèle** prenant en compte les variables fonctionnelles comme variables explicatives, ce métamodèle étant nécessaire pour réaliser l'analyse de sensibilité d'un simulateur coûteux (cas LiPoSo).

Remarque 1.1. *La distribution de probabilité des variables fonctionnelles en entrée des codes considérés n'est pas connue directement et seul un échantillon probabilisé de ces variables est disponible. De plus, on ne s'intéresse pas ici à la provenance de l'échantillon, et les outils proposés ont pour but d'être utilisables quelle que soit l'origine de l'échantillon. On suppose seulement que l'on dispose d'un échantillon de réalisations indépendantes et identiquement distribuées. Celui-ci peut être le résultat de mesures expérimentales ou d'un précédent code de calcul, comme c'est le cas dans les deux applications traitées*

(transitoires T-H en sortie du code CATHARE2).

Le traitement des différents objectifs précédemment listés soulève un certain nombre de problématiques décrites dans ce qui suit.

Quantification des incertitudes de variables aléatoires fonctionnelles potentiellement dépendantes entre elles. En plus de permettre une meilleure connaissance des variables d'entrée, la quantification des incertitudes est une étape nécessaire pour réaliser l'analyse de sensibilité ou la propagation d'incertitudes dans un code de calcul. Dans un contexte probabiliste, les paramètres incertains en entrée d'un code de calcul sont modélisés par des variables aléatoires auxquelles sont attribuées des distributions de probabilité. Ces distributions peuvent être choisies par avis d'expert ou, dans certains cas, estimées à partir de données. Dans notre cas, on dispose d'un échantillon de réalisations de variables aléatoires fonctionnelles dépendantes entre elles et notre objectif est d'estimer leur distribution de probabilité jointe.

Cette quantification des incertitudes doit servir, dans un second temps, à propager ces incertitudes à travers le code ou à réaliser une analyse de sensibilité de celui-ci. Il est donc important que, dans la modélisation de la densité de probabilité des paramètres d'entrée, leur lien avec la sortie du code, appelée ici covariable, soit aussi préservé. Conserver le lien entre les variables fonctionnelles et la covariable constitue ainsi le second objectif de cette étape de quantification des incertitudes. De nombreuses méthodes (Ghanem et Spanos, 1991; Anstett-Collin et al., 2015) existent pour modéliser des variables aléatoires fonctionnelles indépendantes. Le premier objectif des travaux de thèse présentés dans le chapitre 2 est donc d'améliorer ces méthodes et de les adapter au cas de variables aléatoires fonctionnelles dépendantes et liées à une covariable.

Visualisation de l'incertitude des variables aléatoires fonctionnelles. Visualiser les incertitudes de variables fonctionnelles peut être un moyen efficace pour explorer ces données et mettre en valeur certaines de leurs caractéristiques en résumant graphiquement ces informations complexes. Plusieurs techniques statistiques de visualisation de données fonctionnelles (Hyndman et Shang, 2010; Sun et Genton, 2011) ont été récemment développées avec plusieurs objectifs : la détection de tendances et de courbes extrêmes dans les échantillons de fonctions, ou encore la construction de domaines de confiance contenant la plupart des fonctions de l'échantillon. Dans cette thèse, l'objectif est aussi de proposer, à partir de la méthode de quantification des incertitudes développée, une extension des méthodes de visualisation existantes et adaptées aux problèmes considérés. Ces travaux sont aussi présentés dans le chapitre 2.

Analyse de sensibilité d'un code à entrées fonctionnelles. L'analyse de sensibilité a pour but d'identifier l'influence de paramètres ou de groupes de paramètres d'entrée sur la sortie du code de calcul. On peut distinguer deux types de méthodes d'analyse de sensibilité (Helton et al. 2006; Iooss 2011) : les méthodes locales, basées sur l'étude des variations locales du code, et les méthodes globales sur lesquelles on se concentre ici. L'analyse de sensibilité globale permet de quantifier l'influence de chaque paramètre d'entrée sur tout son domaine de variation, de les hiérarchiser par ordre d'influence et de détecter d'éventuelles interactions entre les entrées. Parmi les nombreuses techniques d'analyse de sensibilité globale, on s'intéresse plus particulièrement aux mesures de sensibilité basées sur la variance, appelées indices de Sobol' (Sobol', 1993), qui sont largement utilisées en pratique. L'un des objectifs de cette thèse est de proposer des méthodes pour estimer les indices de Sobol' dans les deux cas d'étude considérés : il s'agit plus particulièrement de comparer l'influence des différentes variables scalaires à celle du groupe de variables fonctionnelles dépendantes. Pour cela, des solutions sont proposées dans le cadre de ces travaux de thèse et présentées dans le chapitre 3.

Approximation par métamodèle d'un code de calcul à entrées fonctionnelles. Pour des simulateurs coûteux en temps de calcul, l'analyse de sensibilité globale n'est pas réalisable en pratique à cause du grand nombre d'évaluations du code nécessaire. Pour réduire ce nombre d'appels au code et,

par conséquent, le coût en temps de calcul de l'étude, il est possible d'approcher le code par un modèle de substitution (Sacks et al., 1989). Ce modèle de substitution, aussi appelé métamodèle ou émulateur, est une fonction mathématique qui doit approcher le plus précisément possible le code sur l'ensemble du domaine de variation de ses paramètres d'entrée et avoir un temps d'exécution très faible voire négligeable. Ce modèle simplifié est construit à partir d'évaluations du code sur un échantillon de paramètres d'entrée, appelé échantillon (ou base) d'apprentissage. Le métamodèle, très rapide à évaluer, est ensuite utilisé pour réaliser les études d'analyse de sensibilité. Ainsi, comme le code de calcul n'est évalué que pour la construction du métamodèle, le coût de calcul de l'analyse de sensibilité peut être considérablement réduit. En outre, les métamodèles sont des outils génériques qui peuvent être utilisés pour plusieurs études sur un même code comme la propagation d'incertitude, l'optimisation ou encore la calibration de codes. Dans cette thèse, l'un des objectifs est de construire un métamodèle pour des codes à entrées fonctionnelles et scalaires. Cela nécessite en particulier de construire un plan d'expériences spécifique sur l'espace des entrées afin de constituer la base d'apprentissage. Afin d'obtenir un métamodèle le plus prédictif possible, le plan d'expériences doit avoir des propriétés optimales en terme d'occupation de l'espace des entrées. Pour s'adapter au cas d'entrées fonctionnelles et scalaires, une méthode d'échantillonnage « mixte » a été développée et est proposée dans le chapitre 3. D'autres développements méthodologiques autour de la métamodélisation des codes à entrées fonctionnelles ont aussi été réalisés dans le cadre de cette thèse. Ces travaux, effectués en collaboration avec deux autres doctorants (Vincent Moutoussamy d'EDF R&D et Benoît Pauwels de l'IFPEN), sont détaillés dans le chapitre 4.

1.4 Organisation du document

Les problématiques introduites dans la section précédente sont étudiées dans les trois chapitres suivants. La quantification des incertitudes de variables fonctionnelles potentiellement dépendantes est étudiée dans le chapitre 2. Un état de l'art des méthodes existantes, ainsi que les méthodes que nous avons proposées au cours de cette thèse y sont détaillés. Les outils développés dans ce chapitre sont ensuite appliqués à un exemple analytique et aux applications PTS et LiPoSo. Dans ce même chapitre, la problématique de la visualisation de données fonctionnelles est aussi traitée. La méthode de quantification des incertitudes développée y est utilisée pour construire un nouvel outil de visualisation.

Le troisième chapitre est consacré à l'analyse de sensibilité. Dans un premier temps, l'analyse de sensibilité globale et les indices de sensibilité issus de la décomposition de la variance sont présentés. Une revue des méthodes d'estimation et des adaptations de ces indices dans le cas d'entrées fonctionnelles ou dépendantes est réalisée. Deux méthodes sont ensuite proposées pour conduire une analyse de sensibilité dans les configurations présentées dans la section 1.3. Ces méthodes sont appliquées sur le cas analytique étudié dans le chapitre 2 et aux études de sûreté PTS et LiPoSo.

Dans le chapitre 4, une approche pour approcher un code à entrées fonctionnelles par un métamodèle est explorée. Plusieurs méthodes de métamodélisation sont développées puis appliquées à deux exemples analytiques et au cas du PTS.

Enfin, nous concluons ce mémoire par un rappel sur les innovations méthodologiques proposées dans ce document pour le traitement des incertitudes dans les codes de calcul, ainsi que par une description des perspectives possibles d'amélioration de ces contributions.

Chapitre 2

Quantification des incertitudes de variables aléatoires fonctionnelles dépendantes

2.1 Problématiques et objectifs

Les simulateurs numériques utilisés pour modéliser les phénomènes physiques mis en jeu dans le cadre des études de sûreté prennent en entrée un certain nombre de paramètres caractéristiques des phénomènes physiques considérés. Par manque de connaissance ou du fait de leur nature intrinsèque, ces paramètres sont entachés d'une incertitude. Il est alors important d'évaluer comment ces incertitudes se répercutent sur les sorties du code de calcul, au travers d'études d'analyse de sensibilité et de propagation des incertitudes. Pour réaliser ces études, il est nécessaire d'identifier et de quantifier préalablement ces sources d'incertitudes en entrée du modèle ou du code. Cette quantification des incertitudes en entrée du simulateur est une étape essentielle dans le traitement des incertitudes (Helton et al., 2006; De Rocquigny et al., 2008). Dans le cadre d'une approche probabiliste, les paramètres d'entrée incertains sont représentés par des variables aléatoires et leurs incertitudes sont modélisées par des distributions de probabilité. Pour réaliser cette modélisation, il est possible de s'appuyer sur un avis d'expert et, dans certains cas, sur des données disponibles.

De nombreux travaux ont été réalisés sur la quantification des incertitudes dans le cas classique des entrées scalaires (Helton et al., 2006, 2010). On s'intéresse, dans ce chapitre, au cas plus complexe des paramètres d'entrée fonctionnels (des courbes temporelles, par exemple) et potentiellement corrélés. De plus, on suppose que l'on dispose d'un échantillon de réalisations indépendantes et identiquement distribuées de ces paramètres. Ces réalisations peuvent, par exemple, être le résultat de mesures expérimentales ou être les sorties d'un ou de plusieurs autres simulateurs numériques. L'objectif est alors de quantifier les incertitudes de ces paramètres fonctionnels, ce qui revient à estimer la densité de probabilité jointe d'un ensemble de variables aléatoires fonctionnelles, potentiellement dépendantes entre elles. La méthode proposée ne doit s'appuyer que sur les réalisations disponibles et ainsi pouvoir être appliquée quelle que soit l'origine de ces réalisations. Puisque la quantification des incertitudes est réalisée en vue de faire une analyse de sensibilité ou une propagation des incertitudes, il est important qu'elle prenne en compte la dépendance entre les variables d'entrées et la sortie du code. Cette dernière est aussi une variable aléatoire (fonction des variables aléatoires d'entrée) et est appelée, par la suite, covariable. Le deuxième objectif de la quantification des incertitudes étudiée ici est de modéliser la dépendance entre les variables fonctionnelles et la covariable : il s'agit en particulier de veiller à ce que les caractéristiques des variables aléatoires fonctionnelles qui jouent un rôle important dans l'explication de la covariable soient conservées et modélisées lors de l'étape de quantification. Sous l'hypothèse qu'une covariable est disponible, la quantification des incertitudes des variables fonctionnelles doit retenir à la fois les caractéristiques les

plus importantes des variables aléatoires fonctionnelles et celles qui sont le plus liées à la covariable.

Une littérature abondante a été consacrée à la quantification des incertitudes d'une variable fonctionnelle dans différents contextes. Par exemple, dans le cadre de l'étude des équations stochastiques aux dérivées partielles, la solution recherchée peut être une variable aléatoire fonctionnelle. Dans ce contexte, Ghanem et Spanos (1991) ont proposé une méthode de quantification des incertitudes en deux étapes. Tout d'abord, la variable fonctionnelle est décomposée sur une base fonctionnelle en utilisant la décomposition de Karhunen-Loève (Loève, 1955), aussi appelée Analyse en Composantes Principales (ACP), afin de réduire la dimension du problème. Dans un second temps, les coefficients de la variable sur la base fonctionnelle sont modélisés par une décomposition par chaos polynomial. Pour estimer les paramètres de la décomposition par chaos polynomial, deux grandes familles de méthodes existent : les méthodes intrusives et les approches non-intrusives dans lesquelles le code est considéré comme une boîte noire (privilégiées ici car seul un échantillon des données est disponible). L'approche de Ghanem et Spanos (1991) a été reprise et améliorée dans de nombreux travaux parmi lesquels on peut citer Xiu et Karniadakis (2002); Babuška et al. (2010); Ma et Zabaras (2011) ; etc. Dans le contexte de l'analyse de sensibilité, Anstett-Collin et al. (2015) considèrent que les variables fonctionnelles en entrée du code étudié sont des processus gaussiens et les approchent par une décomposition de Karhunen-Loève. Comme il supposent que les variables sont gaussiennes, leurs coefficients sur la base fonctionnelle sont indépendants et normalement distribués. Finalement, Hyndman et Shang (2010) proposent une méthode de quantification des incertitudes en support à la visualisation de données fonctionnelles. La variable fonctionnelle est décomposée sur les deux premières fonctions de la base obtenue par décomposition ACP. La densité de probabilité jointe des couples de coefficients est ensuite estimée par une méthode non-paramétrique (Scott, 1992). Hyndman et Shang (2010) utilisent la densité de probabilité ainsi estimée pour créer un outil de visualisation, le *High Density Region* boxplot. Toutes les méthodes citées précédemment ont en commun de procéder en deux étapes : une décomposition sur une base fonctionnelle et la modélisation de la densité de probabilité multivariée des coefficients de la décomposition.

Certains auteurs ont proposé des extensions de ces méthodes au cas de plusieurs variables fonctionnelles dépendantes. Pour la décomposition des variables sur une base fonctionnelle, Ramsay et Silverman (2005) puis Van Deun et al. (2009) proposent de décomposer simultanément les variables fonctionnelles sur une base construite par ACP. Quand les variables ont des variabilités sensiblement différentes ou ne sont pas du même ordre de grandeur, ils proposent de pondérer les variables fonctionnelles. Plus récemment, Perrin et al. (2013) ont développé une extension de la décomposition de Karhunen-Loève pour plusieurs variables fonctionnelles dans laquelle les pondérations associées aux variables fonctionnelles sont optimisées. En effet, Perrin et al. (2013) partent du constat qu'en appliquant la décomposition de Karhunen-Loève à plusieurs variables fonctionnelles, les erreurs sur les variables les plus élevées sont minimisées en priorité. Ils proposent alors une décomposition normalisée permettant soit de favoriser l'approximation de certaines variables fonctionnelles par rapport aux autres, soit de minimiser le maximum de l'erreur sur chaque variable.

Pour répondre aux objectifs de la quantification identifiés précédemment et en particulier tenir compte du lien entre les variables modélisées et la covariable, on propose dans ce chapitre une nouvelle méthode de décomposition simultanée de plusieurs variables fonctionnelles basée sur la décomposition *Partial Least Squares*. En effet, cette méthode présente l'avantage d'offrir un compromis entre l'approximation des variables fonctionnelles et l'approximation de leur lien avec la covariable. Quelle que soit la décomposition fonctionnelle utilisée, la densité de probabilité des coefficients de la décomposition peut ensuite être approchée à l'aide des méthodes citées précédemment (polynômes de chaos, estimation à noyau, etc.). On propose ici de modéliser leur densité de probabilité jointe par un mélange de gaussiennes. En effet, ce modèle semble adapté pour décrire les données disponibles dans les cas d'étude du PTS et de la rupture LiPoSo. Pour s'adapter au cas où la dimension (*i.e.* le nombre de coefficients) est élevée, une méthode d'estimation de matrices de covariance creuses est développée. Celle-ci a pour but de limiter le nombre de paramètres dans le modèle et d'en améliorer l'ajustement.

Dans ce chapitre, les deux étapes de la quantification des incertitudes pour des variables fonctionnelles sont étudiées en détail dans les sections 2.2 et 2.3 respectivement. Dans la section 2.2, plusieurs méthodes de décomposition d'une variable fonctionnelle seule ou de plusieurs variables fonctionnelles dépendantes

sont détaillées dans un premier temps. Plusieurs critères d'évaluation de la qualité des méthodes de décomposition sont ensuite proposés. Enfin, ces méthodes sont comparées sur un exemple analytique dont les caractéristiques sont connues, puis elles sont ensuite mises en œuvre sur le cas d'étude du choc thermique pressurisé (PTS) et celui de la rupture LiPoSo, présentés dans l'introduction. La section 2.3, qui aborde la seconde étape de la quantification, suit un plan similaire. Tout d'abord, les méthodes d'estimation de la densité de probabilité des coefficients de la décomposition (existantes et proposées) sont décrites. Dans un deuxième temps, des critères d'évaluation de ces méthodes ainsi que de la méthodologie globale de quantification des incertitudes sont proposés. Tous ces outils sont ensuite appliqués sur les trois cas d'étude (modèle analytique, PTS et LiPoSo). Dans la section 2.4, la problématique de la visualisation de données fonctionnelles incertaines est abordée. Une revue des méthodes existantes est d'abord réalisée. Puis, une extension de l'une de ces méthodes, basée sur la quantification des incertitudes développée, est proposée et appliquée aux deux cas d'étude PTS et LiPoSo. Enfin, une synthèse des travaux présentés dans ce chapitre est réalisée dans la section 2.5.

2.2 Décomposition fonctionnelle

La première étape de la méthodologie proposée pour la quantification des incertitudes des variables aléatoires fonctionnelles a pour objectif de réduire la dimension des données. Pour cela, on choisit de décomposer les variables aléatoires fonctionnelles étudiées sur une base de fonctions. Cette base fonctionnelle doit prendre en compte deux caractéristiques du problème étudié : les variables aléatoires fonctionnelles sont dépendantes entre elles et elles peuvent, selon le cas d'étude, être liées à une covariable. Cette section présente un état de l'art de bases fonctionnelles et des méthodes de construction associées. Ces bases peuvent être classées en deux types : les bases prédéfinies, c'est-à-dire dont les fonctions sont sélectionnées à l'aide des observations parmi un ensemble de fonctions fixé *a priori*, et les bases dont les fonctions sont estimées à partir des observations. Le principe des bases prédéfinies est de fixer l'ensemble de fonctions (l'ensemble des fonctions splines ou des ondelettes, par exemples) dans lequel peuvent être choisies les fonctions de la base de décomposition, puis des fonctions sont sélectionnées dans cet ensemble pour construire la base de décomposition par rapport aux variables fonctionnelles à décomposer. Les sections 2.2.1, 2.2.2 et 2.2.3 présentent trois types de bases prédéfinies : les splines, les ondelettes et les paquets d'ondelettes. Dans la deuxième famille de bases de décomposition, les fonctions de base sont construites à partir des données observées. Parmi ces bases fonctionnelles, l'Analyse en Composantes Principales (ACP) fonctionnelle est très largement employée dans la caractérisation de variables aléatoires fonctionnelles indépendantes (Ghanem et Spanos 1991; Perrin et al. 2013; Anstett-Collin et al. 2015; Hyndman et Shang 2010 ; etc.). La décomposition ACP fonctionnelle et son extension à la décomposition d'un groupe de variables aléatoires sont présentées dans la section 2.2.4. Une autre décomposition estimée à partir de données, la décomposition Partial Least Squares fonctionnelle, est présentée dans la section 2.2.5. Une adaptation de la décomposition Partial Least Squares au cas d'un groupe de variables fonctionnelles, développée au cours de cette thèse, est présentée dans cette même section. Des critères d'évaluation des méthodes de décomposition fonctionnelle sont définis dans la section 2.2.6. Enfin, dans la section 2.2.7, les méthodes de décomposition présentées sont testées sur un modèle analytique et sur les données des deux cas d'étude présentés dans l'introduction.

Soit un espace de probabilité $(\Omega, \mathcal{F}, P)$. On étudie les m variables aléatoires fonctionnelles

$$Z_1, \dots, Z_m : \Omega \times I \to \mathbb{R},$$

où I est un intervalle de $\mathbb{R}$. Ainsi, une réalisation $Z_i(\omega, .) : I \to \mathbb{R}$, pour $i \in \{1, \dots, m\}$ et $\omega \in \Omega$ fixé, est une fonction d'une variable réelle. Dans la suite, on considère que l'on dispose d'un échantillon de n réalisations de ces variables, notées

$$z_{1,j}, \dots, z_{m,j} : I \to \mathbb{R}, \ \forall j \in \{1, \dots, n\}.$$

Ces réalisations sont discrétisées sur les points $t_1, \dots, t_p$ de l'intervalle I. La version discrétisée de $z_{i,j}$ est notée $\mathbf{z}_{i,j} \in \mathbb{R}^p$, $i = 1, \dots, m$ et $j = 1, \dots, n$, telle que $(\mathbf{z}_{i,j})_k = z_{i,j}(t_k)$, $k = 1, \dots, p$. Enfin, on

introduit une covariable, notée Y, dépendant des m variables fonctionnelles. Les réalisations de cette variable correspondant aux réalisations $z_{1,j}, \ldots, z_{m,j}$ sont notées y_j pour $j \in \{1, \ldots, n\}$.

2.2.1 Splines

2.2.1.1 Définition des bases de splines

Une fonction spline d'ordre d aux nœuds $\{t_1, \ldots, t_p\}$ est une fonction polynômiale par morceaux dont les dérivées sont continues jusqu'à l'ordre $d-2$. Par exemple, les splines d'ordre 3 ou splines cubiques sont très largement utilisées (Hastie et al., 2001).

Définition 2.1 (Base de splines)**.** *La base naturelle des fonctions splines d'ordre d aux nœuds* $\{t_1, \ldots, t_p\}$ *peut s'écrire ainsi :*

$$\left\{(t \mapsto t^j)_{j=0,\ldots,d-1}, (t \mapsto (t-t_i)_+^{d-1})_{i=1\ldots p}\right\},$$

où $\forall a \in \mathbb{R}$, $a_+ = \begin{cases} a & si\ a > 0 \\ 0 & sinon \end{cases}$.

Soit $\Phi = (\varphi_1, \ldots, \varphi_p)$ la base de splines cubiques naturelles aux nœuds $\{t_1, \ldots, t_p\}$, les coefficients $\theta_1, \ldots, \theta_p$ de la fonction z sur la base Φ sont déterminés en résolvant le problème des moindres carrés pénalisés suivant :

$$\min_{\theta_1,\ldots,\theta_p \in \mathbb{R}} \sum_{i=1}^{p} \left(z(t_i) - f(t_i)\right)^2 + \lambda \int_I f''(t)^2 \mathrm{d}t,$$

où $f = \sum_{k=1}^{p} \theta_k \varphi_k$ est l'approximation de z sur la base de splines et λ est un paramètre contrôlant la régularité de l'approximation de z. L'ajout d'une pénalisation permet ainsi de contrôler le nombre de fonctions de base retenues dans la décomposition *via* le contrôle de la régularité. Cette minimisation peut être résolue analytiquement.

Bien que très simple, la base des splines cubiques peut poser des problèmes numériques lors du calcul des coefficients de la décomposition, c'est pourquoi la base des B-splines, définie dans la section suivante, lui est souvent préférée.

2.2.1.2 B-splines

Les B-splines (Hastie et al., 2001) forment une base pour l'espace des fonctions splines. Pour construire cette base, on ajoute aux nœuds des fonction splines $\{t_1, \ldots, t_p\}$ deux points limites t_0 et t_{p+1}, qui définissent le domaine sur lequel calculer la spline. De plus, on définit la séquence de points $(\tau_i)_{i=1\ldots p+2M}$ telle que

- $\tau_1 \leqslant \cdots \leqslant \tau_M \leqslant t_0$
- $\forall j \in \{1, \ldots, p\}, \tau_{j+M} = t_j$
- $t_{p+1} \leqslant \tau_{p+M+1} \leqslant \cdots \leqslant \tau_{p+2M}$.

On peut prendre tous les points τ_i au-delà des points limites t_0 et t_{p+1} comme égaux respectivement à ces deux limites. La k-ème B-spline d'ordre j est notée B_k^j.

Définition 2.2 (B-splines)**.** *Les B-splines sont définies par récurrence par des différences divisées :*

$$\begin{aligned}
\forall k \in \{1, \ldots, p+2M-1\},\ B_k^1(t) &= \begin{cases} 1 & \text{si } \tau_k \leqslant t < \tau_{k+1} \\ 0 & \text{sinon} \end{cases} \\
\forall k \in \{1, \ldots, p+2M-j\},\ B_k^j(t) &= \begin{cases} 0 \text{ si } \tau_k = \tau_{k+j-1} \text{ ou } \tau_{k+j} = \tau_{k+1} \\ \frac{t-\tau_k}{\tau_{k+j-1}-\tau_k} B_k^{j-1}(t) + \frac{\tau_{k+j}-t}{\tau_{k+j}-\tau_{k+1}} B_{k+1}^{j-1}(t) \quad \text{sinon} \end{cases}.
\end{aligned}$$

Si $M = 4$, $(B_k^4)_{k=1\ldots p+4}$ est la base de B-splines cubiques pour les nœuds $t_1 \ldots t_p$.

2.2.2 Ondelettes

2.2.2.1 Construction de la transformée en ondelettes

Meyer et Salinger (1995) introduisent la notion d'ondelette via l'analyse multirésolution ou AMR.

Définition 2.3 (Analyse multirésolution)**.** *Une AMR de* $L^2(\mathbb{R})$ *se définit comme une suite* $(E_j)_{j\in\mathbb{Z}}$ *d'espaces fermés, croissante pour l'inclusion, telle que :*

1. $\cap_{j\in\mathbb{Z}} E_j = \{0\}$ *et* $\overline{\cup_{j\in\mathbb{Z}} E_j} = L^2(\mathbb{R})$
2. $\forall f \in L^2(\mathbb{R}), \forall j \in \mathbb{Z}, f \in E_j \Leftrightarrow [t \mapsto f(2^{-j}t)] \in E_{j+1}$
3. $\exists \varphi \in L^2(\mathbb{R}) : \{t \mapsto \varphi(t-k)\}_{k\in\mathbb{Z}}$ *est une base orthonormée de* E_0*.* φ *est nommée fonction d'échelle.*

On déduit des deux dernières propriétés d'une AMR que $\{\varphi_{j,k} : t \mapsto 2^{j/2}\varphi(2^j t - k)\}_{k\in\mathbb{Z}}$ est une base orthonormée de E_j. De plus, comme $E_0 \subset E_1$, la fonction d'échelle φ peut s'exprimer dans une base de E_1, selon la *relation à deux échelles* :

$$\forall k \in \mathbb{Z}, \exists h_k \in \mathbb{R} : \forall t \in \mathbb{R},\ \varphi(t) = \sum_{k\in\mathbb{Z}} h_k \varphi(2t-k). \tag{2.1}$$

On note W_j le complémentaire de E_j dans E_{j+1}, *i.e.* $E_{j+1} = E_j \oplus W_j$. D'après la première propriété de la définition 2.3 et par récurrence, on peut prouver que

$$L^2(\mathbb{R}) = \overline{E_{j_0} \oplus \bigoplus_{j=j_0}^{+\infty} W_j},\ \forall j_0 \in \mathbb{Z}. \tag{2.2}$$

De la même manière que précédemment, il existe une fonction ψ de W_0 telle que $\{\psi_{j,k} : t \mapsto 2^{j/2}\psi(2^j t - k)\}_{k\in\mathbb{Z}}$ est une base orthonormée de W_j et ψ est appelée ondelette. On définit de manière similaire une relation à deux échelles entre ψ et φ puisque $W_j \subset E_{j+1}, \forall j \in \mathbb{Z}$. Elle s'énonce ainsi :

$$\forall k \in \mathbb{Z}, \exists g_k \in \mathbb{R} : \forall t \in \mathbb{R},\ \psi(t) = \sum_{k\in\mathbb{Z}} g_k \varphi(2t-k). \tag{2.3}$$

Ainsi, l'équation (2.2) permet d'écrire la décomposition suivante pour une fonction $z \in L^2(\mathbb{R})$:

$$z = \sum_{k\in\mathbb{Z}} \alpha_{j_0,k}\varphi_{j_0,k} + \sum_{j=j_0}^{+\infty}\sum_{k\in\mathbb{Z}} \beta_{j,k}\psi_{j,k}, \tag{2.4}$$

où $\alpha_{j_0,k} = \int_{\mathbb{R}} z\varphi_{j_0,k}$ et $\beta_{j,k} = \int_{\mathbb{R}} z\psi_{j,k}$. Cette construction se transpose aisément aux fonctions de $L^2([0;1])$.

Définition 2.4 (Transformée en ondelettes)**.** *La transformée en ondelettes de* z *est définie comme la projection de* z *sur* E_J *et s'écrit comme la troncature de l'équation (2.4), à savoir :*

$$\hat{z} = \sum_{k=0}^{2^{j_0}-1} \alpha_{j_0,k}\varphi_{j_0,k} + \sum_{j=j_0}^{J-1}\sum_{k=0}^{2^j-1} \beta_{j,k}\psi_{j,k}.$$

Mallat (1989) a proposé des algorithmes de décomposition en ondelettes et de reconstruction à partir de la transformée. Les coefficients $\alpha_{j,k}$ et $\beta_{j,k}$ de la transformée en ondelettes sont calculés récursivement selon deux formules obtenues à partir des relations d'échelle des équations (2.1) et (2.3).

La décomposition en ondelettes construite dans cette section peut comporter un grand nombre de coefficients $\alpha_{j_0,k}$ et $\beta_{j,k}$. Afin de réduire le nombre de fonctions utilisées dans la base de décomposition, une sélection, appelée seuillage, est réalisée parmi les fonctions de base.

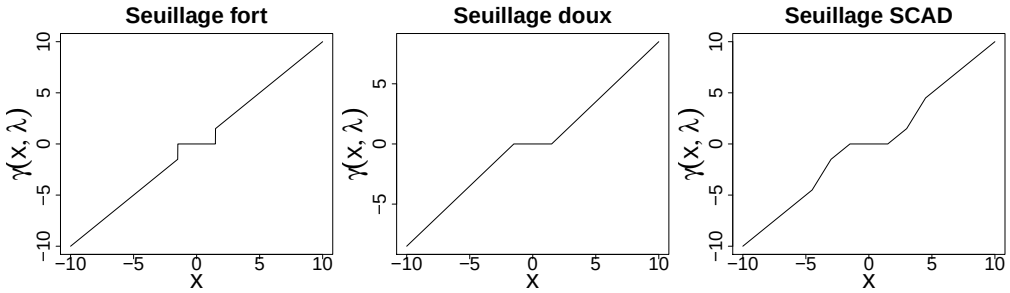

FIGURE 2.1 – Fonctions de seuillage doux, fort et SCAD pour la décomposition sur base d'ondelettes pour $\lambda = 1,5$ et $a = 3$.

2.2.2.2 Seuillage

Le seuillage de la base d'ondelettes a été introduit par Donoho et Johnstone (1994). Son objectif est de représenter la fonction z à approcher par un petit nombre d'ondelettes. Pour ce faire, les coefficients de la transformée en ondelettes les plus faibles sont supposés nuls. En effet, on considère que ceux-ci ne représentent que du bruit.

Pour cela, on introduit la fonction de seuil $\gamma(x, \lambda)$, où x est le coefficient à seuiller et λ le seuil. Parmi les fonctions de seuil, on peut citer, par exemple

– le seuillage fort :

$$\gamma(x, \lambda) = x 1_{|x|>\lambda} \ ;$$

– le seuillage doux :

$$\gamma(x, \lambda) = \text{sgn}(x)(|x| - \lambda)_+ \ ;$$

– le seuillage SCAD (Smoothly Clipped Absolute Deviation, Fan et Li 2001) :

$$\gamma(x, \lambda) = \begin{cases} \text{sgn}(x)(|x| - \lambda)_+ & \text{si } |x| \leqslant 2\lambda \\ \frac{1}{a-2}((a-1)x - a\lambda \text{sgn}(x)) & \text{si } 2\lambda < |x| \leqslant a\lambda \quad , a > 2, \\ x & \text{si} |x| > a\lambda \end{cases}$$

où sgn est la fonction signe. L'effet de ces trois seuillages est illustré par la Figure 2.1. Le seuillage fort possède deux discontinuités en $\pm\lambda$. Le seuillage doux a un biais pour $|x| > \lambda$, c'est-à-dire que, pour $|x| > \lambda$, le seuillage doux vaut $x - \lambda$ et non x. La méthode SCAD est un compromis entre les deux méthodes précédentes puisqu'elle ne présente pas de discontinuité et n'a pas de biais pour $|x| > a\lambda$.

Antoniadis et Fan (2001) ont montré que le seuillage est équivalent à une sélection de variables par moindres carrés pénalisés. Si $\mathbf{z} = (z_1, \ldots, z_p)^T$ est le vecteur de discrétisation de z à approcher et V désigne la matrice de projection correspondant à la transformée en ondelettes, l'expression à minimiser en fonction de la transformée en ondelettes, $\mathbf{w} = (w_1, \ldots, w_p)$, peut s'écrire ainsi :

$$\|V\mathbf{z} - \mathbf{w}\|^2 + l \sum_{i=1}^{n} \text{pen}(|w_i|),$$

où l est le paramètre de régularité et pen est une fonction de pénalité. Cette dernière est différente selon le type de seuillage choisi. Par exemple, la minimisation avec pénalisation pen telle que $\text{pen}(|\theta|) = |\theta|$ est équivalente au seuillage doux.

De nombreuses méthodes ont été proposées pour déterminer la valeur du paramètre de seuil λ. On présente ici le seuillage universel introduit par Donoho et Johnstone (1994). Ils proposent d'utiliser le seuil $\lambda = \sigma\sqrt{2\log(p)}$, où σ^2 est la variance du bruit. Donoho et Johnstone (1994) proposent d'estimer σ^2 par *maximum absolute deviation*, un estimateur robuste, appliqué aux coefficients de la transformée en

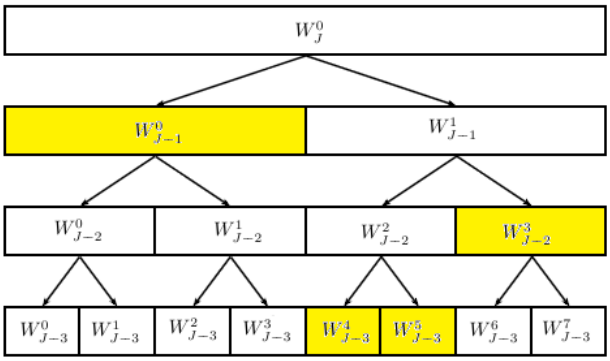

FIGURE 2.2 – Arbre binaire des décompositions successives de E_J pour les paquets d'ondelettes.

ondelettes de niveau de résolution maximal J. En définissant β comme le vecteur des 2^{J-1} coefficients de résolution maximale, il se définit ainsi :

$$\hat{\sigma} = \frac{mediane(|\beta - mediane(\beta)|)}{0,6745}.$$

Le seuil proposé par Donoho et Johnstone (1994) est asymptotiquement optimal, mais il est généralement trop grand en pratique et entraîne un lissage trop important (Gannaz, 2007), c'est pourquoi il est parfois remplacé par $\lambda = \sigma\sqrt{2\log(p) - \log(\log(p))}$.

2.2.3 Paquets d'ondelettes

Dans la section précédente, les espaces E_j d'une analyse multirésolution sont décomposés récursivement en un espace E_{j-1} et en son conjugué W_{j-1}. Coifman et Wickerhauser (1992) ont montré que cette décomposition récursive pouvait aussi être mise en œuvre sur les espaces W_j. Ce résultat permet de définir une nouvelle méthode de décomposition appelée décomposition en paquets d'ondelettes.

On reprend les mêmes notations que dans la section 2.2.2. L'objectif est ici de décomposer la fonction z sur l'espace E_J, $J \in \mathbb{N}$, noté W_J^0 par la suite. W_J^0 peut être décomposé en deux espaces complémentaires qui peuvent ensuite être eux-mêmes décomposés en deux espaces complémentaires, d'après le résultat de Coifman et Wickerhauser (1992). On peut alors construire récursivement une décomposition de l'espace W_j^p, pour $j \in \mathbb{Z}$ et $p \in \mathbb{N}$, sous cette forme :

$$W_j^p = W_{j-1}^{2p} \oplus W_{j-1}^{2p+1}.$$

Les décompositions successives de ces espaces peuvent être représentées sur un arbre binaire dont chaque nœud est un espace W_j^p, comme sur la Figure 2.2 (adaptée de celles de Mallat (2008) et de Auder et Fischer (2012)). La somme orthogonale des deux fils de chaque nœud est à chaque fois égale au nœud.

La transformée en paquets d'ondelettes est alors définie comme l'ensemble des représentations de z dans les bases de chacun des espaces de l'arbre binaire. Le nombre de coefficients ainsi calculé est supérieur au nombre de coefficients de la transformée en ondelettes. De plus, les coefficients de la transformée en paquets d'ondelettes sont redondants. On peut donc ne sélectionner qu'une partie de ces coefficients pour représenter la fonction z. Pour ce faire, on retient une décomposition particulière de W_J^0 en sous-espaces W_j^p, avec $j < J$ et $p \in \{0, \dots, 2^{J-j} - 1\}$. Par exemple, la décomposition $W_J^0 = W_{J-1}^0 \oplus W_{J-2}^3 \oplus W_{J-3}^4 \oplus W_{J-3}^5$ permet de représenter z dans la mesure où elle est composée d'une somme d'espaces orthogonaux. Cet exemple de décomposition est représenté sur l'arbre binaire de la Figure 2.2 par les cases colorées en jaune. Coifman et Wickerhauser (1992) ont proposé un algorithme glouton de sélection d'une décomposition particulière parmi le grand nombre de décompositions possibles de l'espace W_J^0. Pour ce faire, l'entropie de Shannon (Shannon 1948) est minimisée, par descente de l'arbre binaire. A

chaque étape, l'entropie du nœud parent et la somme de l'entropie de ses deux fils sont comparées. Si l'entropie du parent est plus petite que la somme des deux autres, alors le nœud parent est conservé, sinon les deux fils remplacent le parent dans la décomposition et la descente se poursuit dans chacun des sous-arbres dont les fils sont les racines.

2.2.4 Analyse en composantes principales

Dans cette section, on présente tout d'abord une méthode de réduction en dimension finie, l'analyse en composantes principales, sur laquelle est basée l'analyse en composantes principales fonctionnelle. Enfin, on décrit une extension de cette méthode pour la décomposition d'un groupe de variables fonctionnelles dépendantes. Dans la méthode présentée, la base de fonction n'est pas prédéfinie mais les fonctions de base sont construites à partir des observations, contrairement aux méthodes présentées dans les sections précédentes.

2.2.4.1 Définition

L'Analyse en Composantes Principales (ACP), aussi appelée décomposition de Karhunen-Loève (Pearson, 1901; Loève, 1955; Jolliffe, 2005), permet, pour des points de $\mathbb{R}^p$, de déterminer les directions orthogonales entre elles qui maximisent la variance projetée de ces points. Soit un échantillon de points $\mathcal{X} = \{\mathbf{x}_j : j = 1, \ldots, n\} \subset \mathbb{R}^p$ de moyenne nulle, la base orthonormée de projection de $\mathcal{X}$ est $(\boldsymbol{\varphi}_k)_{1 \leqslant k \leqslant p} \subset \mathbb{R}^p$. Si les points de $\mathcal{X}$ ne sont pas de moyenne nulle, ils sont préalablement centrés. On suppose, de plus, que $p \leqslant n$.

Définition 2.5 (Analyse en Composantes Principales)**.** *L'ACP revient à chercher les composantes principales orthonormales φ_k telles que*

$$\mathbf{x}_j = \sum_{k=1}^{p} \langle \mathbf{x}_j, \boldsymbol{\varphi}_k \rangle \, \boldsymbol{\varphi}_k,$$

et qui minimisent les variances projetées : $\sum_{j=1}^{n} \langle \mathbf{x}_j, \boldsymbol{\varphi}_k \rangle^2$, $k = 1, \ldots, p$.

Les $\langle \mathbf{x}_j, \boldsymbol{\varphi}_k \rangle$ sont les coefficients de $\mathbf{x}_j$ suivant la composante k. On peut sélectionner les $q \leqslant p$ premières composantes de l'ACP et approcher les points de $\mathcal{X}$ de cette manière pour $j = 1, \ldots, n$:

$$\hat{\mathbf{x}}_j = \sum_{k=1}^{q} \langle \mathbf{x}_j, \boldsymbol{\varphi}_k \rangle \, \boldsymbol{\varphi}_k.$$

Une des propriétés de l'ACP est qu'elle minimise, par définition, l'erreur quadratique d'approximation des points de $\mathcal{X}$. Pour tout $q \leqslant p$, pour toute famille $(\mathbf{e}_1, \ldots, \mathbf{e}_q)$ de $\mathbb{R}^q$, on a $\forall j = 1, \ldots, n$:

$$\left\| \hat{\mathbf{x}}_j - \sum_{k=1}^{q} \langle \hat{\mathbf{x}}_j, \boldsymbol{\varphi}_k \rangle \, \boldsymbol{\varphi}_k \right\|_2 \leqslant \left\| \hat{\mathbf{x}}_j - \sum_{k=1}^{q} \langle \hat{\mathbf{x}}_j, \mathbf{e}_k \rangle \, \mathbf{e}_k \right\|_2,$$

où $\|.\|_2$ est la norme L^2.

En pratique, l'ACP s'obtient par la décomposition en valeurs singulières (SVD) de X, la matrice dont les lignes sont les $\mathbf{x}_j$. En notant U et V des matrices orthogonales et D une matrice diagonale ordonnée décroissante, celle-ci s'écrit :

$$X = UDV^T.$$

Les composantes principales sont les colonnes de V.

2.2.4.2 ACP fonctionnelle

L'ACP présentée dans la section 2.2.4.1 est conçue pour décomposer des points de $\mathbb{R}^p$ sur une base vectorielle réduite. Une extension de cette méthode au cas fonctionnel, appelée ACP fonctionnelle (ACPF), a été proposée par Ramsay et Silverman (2005).

Définition 2.6. *Soit $(z_j)_{1\leqslant j\leqslant n}$ un échantillon de fonctions et $t \mapsto e(t)$ sa moyenne. L'ACPF consiste à chercher la base de fonctions orthonormales $\varphi_1, \ldots, \varphi_d$ et les coefficients $\alpha_{j,k}$, $j = 1, \ldots, n$, $k = 1, \ldots, d$ qui minimisent :*

$$\sum_{j=1}^{n} \int_I \left(z_j(t) - e(t) - \sum_{k=1}^{d} \alpha_{j,k}\varphi_k(t) \right)^2 dt.$$

En pratique, il existe plusieurs approches pour déterminer la base et les coefficients de l'ACPF. Ramsay et Silverman (2005) proposent d'exprimer les fonctions sur une base de splines (cf. section 2.2.1). Une ACP peut ensuite être appliquée aux coefficients des fonctions sur la base des splines. Une deuxième solution est d'appliquer l'ACP aux fonctions discrétisées. Rice et Silverman (1991) ont proposé une autre solution dans laquelle l'opérateur de covariance est lissé puis discrétisé.

Remarque 2.1. *Dans la définition 2.6, la base construite est une base finie de taille d. En pratique, le paramètre d peut être choisi en fonction, par exemple, de la précision d'approximation souhaitée.*

2.2.4.3 ACPF simultanée

Dans le cas étudié ici, on cherche à décomposer plusieurs variables fonctionnelles sur une base et non une seule variable comme dans l'ACPF. Ainsi, il est possible de tenir compte de la dépendance entre ces variables fonctionnelles. Dans ce but, Van Deun et al. (2009) et Ramsay et Silverman (2005) étendent l'analyse en composantes principales fonctionnelle au cas de m variables fonctionnelles. On définit les matrices $X_1, \ldots, X_m$, de taille $n \times p$, représentant pour chacun des n points de l'échantillon, les m fonctions discrétisées sur p points ou les p coefficients des fonctions sur une base de spline, selon la méthode de construction de la base ACPF choisie. On peut écrire la décomposition ACP de ces matrices X_i ainsi :

$$X_i = T_i P_i^T + E_i, \forall i \in \{1, \ldots, m\},$$

où E_i est la moyenne de X_i, T_i la matrice des composantes $(n \times d)$, P_i la matrice des fonctions de base $(p \times d)$ et d le nombre de composantes retenues. On définit la matrice $X = [X_1, \ldots, X_m]$ de taille $n \times mp$, ainsi que le vecteur $[E_1, \ldots, E_m]$ de taille $1 \times mp$. L'analyse en composantes principales simultanée (ACPS) des $X_1, \ldots, X_m$ consiste à appliquer une ACP à la matrice X. Cette décomposition s'écrit

$$X = TP^T + [E_1 \ldots E_m],$$

où la matrice T de taille $n \times d$ est la matrice des coefficients et la matrice P de taille $mp \times d$ est la matrice contenant, en colonne, la discrétisation des fonctions de base. L'ACPS revient donc à appliquer une ACP aux m fonctions discrétisées et concaténées (ou aux coefficients des fonctions sur une base de splines concaténés) :

$$[\mathbf{z}_{1,j}, \ldots, \mathbf{z}_{m,j}], \ \forall j \in \{1, \ldots, n\}.$$

L'avantage de l'ACPS est que les d composantes de la base résument les m fonctions à la fois. Cette idée se révèle d'autant plus intéressante que les fonctions à décomposer sont dépendantes. Pour que les m fonctions aient un rôle équivalent dans la décomposition et éviter que les valeurs les plus élevées soient mieux approchées, on propose d'appliquer une normalisation à chaque fonction. Chaque fonction discrétisée, notée $\mathbf{z}_{i,j}$, est alors divisée par la normalisation N_i. On propose plusieurs normalisations :

– Normalisation 1 par le maximum de la variable fonctionnelle :

$$N_i = \max_{k\in\{1,\ldots,p\}, j\in\{1,\ldots,n\}} (\mathbf{z}_{i,j})_k \ ;$$

– Normalisation 2 par la somme des écarts-types de chaque pas de temps k :

$$N_i = \sum_{k=1}^{p} \sqrt{\frac{1}{n}\sum_{j=1}^{n}((\mathbf{z}_{i,j})_k - \sum_{j=1}^{n}(\mathbf{z}_{i,j})_k)^2} = \sum_{k=1}^{p}\sqrt{\mathrm{var}((\mathbf{z}_{i,.})_k)}\ ;$$

– Normalisation 3 par la racine carrée de la somme des variances de chaque pas de temps k :

$$N_i = \sqrt{\sum_{k=1}^{p}\frac{1}{n}\sum_{j=1}^{n}((\mathbf{z}_{i,j})_k - \sum_{j=1}^{n}(\mathbf{z}_{i,j})_k)^2} = \sqrt{\sum_{k=1}^{p}\mathrm{var}((\mathbf{z}_{i,.})_k)}.$$

Plus récemment, deux approches ont été développées par Perrin et al. (2013) pour sélectionner de manière optimale les pondérations des variables fonctionnelles dans l'ACP simultanée. Ils proposent dans ces approches d'optimiser deux critères : l'erreur quadratique pondérée et l'erreur quadratique maximale de l'ACPS sur les variables fonctionnelles, définies respectivement par :

$$\varepsilon_\beta^2 = \sum_{i=1}^{m}\beta_i^2\varepsilon_i^2 \tag{2.5}$$

$$\varepsilon_\infty^2 = \max_{1\leq i\leq m}\varepsilon_i^2, \tag{2.6}$$

où $\beta = (\beta_1, \ldots, \beta_m) \in \mathbb{R}^m$ est le vecteur des pondérations, et ε_i^2 est l'erreur relative quadratique sur la variable Z_i, pour $i \in \{1, \ldots, m\}$, définie comme suit :

$$\varepsilon_i^2 = \frac{\left\|Z_i - \hat{Z}_i\right\|_2^2}{\|Z_i\|_2^2}. \tag{2.7}$$

De plus, on définit $\|.\|_2$ pour une variable aléatoire fonctionnelle Z :

$$\|Z\|_2 = \left(\mathbb{E}\left[\int_I Z(t)^2 \mathrm{d}t\right]\right)^{1/2}$$

La première approche consiste donc à choisir des normalisations $N = (N_1, \ldots, N_m)$ afin de minimiser l'erreur pondérée ε_β. Perrin et al. (2013) montrent que les pondérations, pour lesquelles l'ACPS minimise ε_β sont définies de la manière suivante pour tout $i \in \{1, \ldots, m\}$:

$$N_i^\beta = \frac{\|Z_i\|_2}{\beta_i}.$$

Le choix des poids β dans l'erreur pondérée dépend de l'importance donnée à chaque variable fonctionnelle. Dans le cas où les variables fonctionnelles sont liées à une quantité d'intérêt, Perrin et al. (2013) proposent d'utiliser les résultats d'une analyse de sensibilité entre les variables fonctionnelles et cette quantité d'intérêt pour définir les poids. Si les variables fonctionnelles ne sont pas liées à une quantité d'intérêt ou s'il n'est pas possible de faire d'analyse de sensibilité, le choix reste subjectif puisqu'il faut choisir quelles variables avantager ou désavantager.

Dans leur deuxième approche, Perrin et al. (2013) proposent un algorithme itératif pour déterminer la normalisation maximisant l'erreur quadratique maximale définie dans l'équation (2.6). La méthode proposée est donnée dans l'algorithme 2.1. A chaque itération de l'algorithme, l'ACPS est réalisée avec la valeur actuelle des normalisations et les erreurs quadratiques sont calculées. La normalisation est ensuite mise à jour en fonction de ces erreurs. L'algorithme s'arrête après un certain nombre d'itérations t_{max} ou quand le critère

$$\mathcal{UB}(N) = \frac{\sum_{i=1}^{m}\|Z_i\|_2^2\left(\varepsilon_\infty^2 - \varepsilon_i^2\right)/N_i}{\sum_{i=1}^{m}\|Z_i\|_2^2/N_i}$$

passe sous un seuil τ. En effet, Perrin et al. (2013) démontrent que la différence entre l'erreur quadratique maximale, notée $\varepsilon^2_\infty(N)$, obtenue avec la normalisation N et l'erreur quadratique maximale optimale, notée ε^{2*}_∞, est bornée de la manière suivante :

$$0 \leq \varepsilon^2_\infty(N) - \varepsilon^{2*}_\infty \leq \mathcal{UB}(N)$$

Ainsi, quand le critère passe en dessous de τ, l'écart entre la normalisation optimale et la normalisation obtenue est très faible, pour tout τ suffisamment faible.

Algorithme 2.1 Estimation de la normalisation optimale selon ε^2_∞ (Perrin et al., 2013)

1. Initialiser $N_i = \|Z_i\|_2$ puis normaliser le vecteur $1/N$
2. Pour t allant de 1 à t_{max}
   (a) Calculer l'ACPS normalisée par $N = (N_1, \ldots, N_m)$
   (b) Si $\mathcal{UB}(N) > \tau$,
   - $N_i = \frac{N_i}{\left(\varepsilon_i^2\right)^\gamma}$
   - Normaliser $1/N$

   (c) Sinon, fin de la boucle.
3. $N^\infty = N$

Dans l'algorithme 2.1, le paramètre γ contrôle la vitesse de convergence. Les paramètres γ et τ sont pris égaux à $1/2$ et 10^{-3} respectivement dans Perrin et al. (2013). Il est préférable de relancer l'algorithme avec plusieurs initialisations différentes de N puisque l'algorithme n'est pas assuré de converger vers le minimum global.

2.2.5 Décomposition Partial Least Squares

On présente dans cette section une méthode de décomposition fonctionnelle basée sur la régression Partial Least Squares. Cette décomposition permet de tenir compte dans l'approximation d'une variable fonctionnelle de son lien avec une covariable scalaire ou vectorielle. Une extension de la décomposition Partial Least Squares au cas d'un groupe de variables est ensuite proposée.

2.2.5.1 Régression Partial Least Squares

L'objectif de la régression aux moindres carrés partiels (PLS), proposée par Wold (1966), est d'expliquer q variables de sortie $(Y_l)_{l=1,\ldots,q}$ par les p variables $(X_{l'})_{l'=1,\ldots,p}$. L'ensemble des variables est supposé centré et réduit. On définit les matrices X de taille $n \times p$ et Y de taille $n \times q$, où n est la taille de l'échantillon. Elles contiennent en ligne les réalisations des variables $(X_{l'})_{l'=1,\ldots,p}$ et $(Y_l)_{l=1,\ldots,q}$ respectivement. Le principe de la PLS est de faire une régression non pas entre les $(X_{l'})_{l'=1,\ldots,p}$ et $(Y_l)_{l=1,\ldots,q}$ mais entre les variables latentes ξ_k et ω_k $(k = 1, \ldots, d)$ définies comme des combinaisons linéaires des variables $(X_{l'})_{l'=1,\ldots,p}$ et $(Y_l)_{l=1,\ldots,q}$ respectivement. Les poids de ces combinaisons linéaires sont respectivement les vecteurs u_k et v_k. Ces poids sont choisis de manière à maximiser la covariance entre les variables explicatives et les variables de sortie. Les variables latentes ξ_k et ω_k sont déterminées itérativement par l'algorithme 2.2. A la première étape de la méthode, on cherche donc les vecteurs u_1 et v_1 tels que

$$\xi_1 = Xu_1 \text{ et } \omega_1 = Yv_1,$$

et solutions de

$$\max_{\|u_1\|=\|v_1\|=1} \mathrm{Cov}(Xu_1, Yv_1).$$

On réalise ensuite l'opération de « déflation » sur les matrices X et Y. Il existe deux variantes de l'algorithme PLS selon le mode de déflation de Y :

$$\begin{aligned} X_1 &= X - \xi_1 c_1^T \\ Y_1 &= Y - \xi_1 d_1^T \text{ (Regression mode)} \\ Y_1 &= Y - \omega_1 e_1^T \text{ (Canonical mode)} \end{aligned}$$

avec $c_1 = \frac{X^T \xi_1}{\xi_1^T \xi_1}$, $d_1 = \frac{Y^T \xi_1}{\xi_1^T \xi_1}$ et $e_1 = \frac{Y^T \omega_1}{\omega_1^T \omega_1}$. Avec la déflation canonique, les variables d'entrée et de sortie jouent un rôle symétrique puisque ce qui est retiré de Y dépend de Y et de ω_1. A l'inverse, dans la déflation « Regression mode », ce qui est retiré de Y dépend à la fois de Y et de ξ_1. Les variables de Y sont donc expliquées par les variables de X. La déflation « Regression mode » peut s'appliquer quand on sait que les variables de X sont bien les variables explicatives de Y, c'est-à-dire qu'on fait une régression des unes par rapport aux autres. Quand on n'a aucune connaissance *a priori* sur le fait qu'un des deux groupes de variables implique l'autre, il est préférable d'appliquer la déflation canonique. Il s'agit alors d'une approche exploratoire plutôt que d'une régression.

A la deuxième étape de l'algorithme, on applique la même méthode que précédemment sur X_1 et Y_1 au lieu de X et Y, pour estimer les vecteurs ξ_2, ω_2, u_2 et v_2. On continue ces opérations jusqu'à obtenir le nombre voulu de composantes. L'algorithme avec lequel la PLS est calculée en pratique est NIPALS (Non linear PArtial Least Squares), qui a l'avantage de ne pas nécessiter d'inversion de matrice.

Algorithme 2.2 Partial Least Squares

1. $X_0 = X$, $Y_0 = Y$
2. Initialiser ω_1 avec la première colonne de Y
3. Pour k dans $1, \ldots, d$
   (a) Jusqu'à convergence faire
      i. $u_k = X_{h-1}^T \omega_k / \omega_k^T \omega_k$
      ii. $u_k = u_k / u_k^T u_k$
      iii. $\xi_k = X_{k-1} u_k$
      iv. $v_k = Y_{h-1}^T \xi_k / \xi_k^T \xi_k$
      v. $v_k = v_k / v_k^T v_k$
      vi. $\omega_k = Y_{k-1}^T v_k$
   (b) $c_k = X_{h-1}^T \xi_k / \xi_k^T \xi_k$
   $d_k = Y_{h-1}^T \xi_k / \xi_k^T \xi_k$
   $e_k = Y_{h-1}^T \omega_k / \omega_k^T \omega_k$
   (c) $X_k = X_{h-1} - \xi_k c_k^T$
   (d) $\begin{cases} Y_k = Y_{k-1} - \xi_k d_k^T & \text{(Regression mode)} \\ Y_k = Y_{k-1} - \omega_k e_k^T & \text{(Canonical mode)} \end{cases}$

Par récurrence sur les déflations successives, on peut écrire les décompositions de X et Y de cette

manière :

$$\begin{aligned} X &= \sum_{k=1}^{d} \xi_k c_k^T + X_{d+1} \\ Y &= \sum_{k=1}^{d} \xi_k d_k^T + Y_{d+1} \quad \text{(Regression mode)} \\ Y &= \sum_{k=1}^{d} \omega_k e_k^T + Y_{d+1} \quad \text{(Canonical mode)} \end{aligned}$$

Ainsi, les variables d'entrée et de sortie se décomposent pour une déflation de type régression :

$$X = \Xi C^T + \varepsilon_1 \quad Y = \Xi D^T + \varepsilon_2, \tag{2.8}$$

et pour une déflation canonique :

$$X = \Xi C^T + \varepsilon_1 \quad Y = \Omega E^T + \varepsilon_2, \tag{2.9}$$

où les vecteurs colonnes de C, D, E, Ξ et Ω sont respectivement c_k, d_k, e_k, ξ_k et ω_k, pour $k = 1, \ldots, d$, et où $\varepsilon_1 = X_{d+1}$ et $\varepsilon_2 = Y_{d+1}$ sont les matrices des résidus.

2.2.5.2 Proposition d'une PLS fonctionnelle simultanée

A partir de la méthode de régression PLS définie dans la section 2.2.5.1, une méthode de décomposition fonctionnelle peut être construite. Soit un échantillon de fonctions $z_1, \ldots, z_n$ discrétisées aux points $t_1, \ldots, t_p$. On définit la matrice $X \in \mathcal{M}_{n,p}(\mathbb{R})$ dont les lignes sont les versions discrétisées des fonctions z_j, $j = 1, \ldots, n$. La régression PLS appliquée sur X fournit une décomposition de ces fonctions sur une base grâce à l'équation (2.8) ou (2.9) selon la déflation utilisée. Les d fonctions composant la base sont les colonnes de la matrice C. La matrice Ξ représente donc les coefficients des fonctions Z_i sur la base $(c_1, \ldots, c_d)$. On appelle cette décomposition une décomposition PLS.

On peut montrer (Höskuldsson, 1988) que les composantes ξ_k sont orthogonales entre elles : $\xi_k^T \xi_h = 0, \forall h, k \in \{1, \ldots, d\}, h \neq k$. De plus, les fonctions de base c_k sont orthogonales selon le produit scalaire associé à la matrice $X^T X$: $c_k X^T X c_h = 0, \forall h, k \in \{1, \ldots, d\}, h \neq k$.

On propose ici de définir une décomposition PLS simultanée (SPLS), de la même manière que pour l'ACP simultanée afin de décomposer, sur une même base, un groupe de variables fonctionnelles. La décomposition PLS décrite précédemment est alors appliquée aux concaténations des fonctions discrétisées comme présenté dans la section 2.2.4.3. Plus précisément, on considère les fonctions $z_{i,j}$ définies au début du chapitre et leurs discrétisations $\mathbf{z}_{i,j}$. On note X la matrice dont les lignes $X_{j,.} \in \mathbb{R}^{mp}$ pour $j = \{1, \ldots, n\}$ sont définies comme la concaténation des vecteurs $(\mathbf{z}_{1,j}, \ldots, \mathbf{z}_{m,j})$:

$$X_{j,.} = [\mathbf{z}_{1,j}, \ldots, \mathbf{z}_{m,j}],$$

La PLS simultanée, notée SPLS, consiste à réaliser la décomposition PLS de la matrice X ainsi définie. Contrairement à l'ACPS, aucune normalisation n'est appliquée aux discrétisations car ces variables sont centrées et réduites avant la décomposition PLS.

2.2.6 Critères d'évaluation développés pour évaluer la décomposition fonctionnelle

Les méthodes de décomposition fonctionnelle présentées dans cette section ont pour but d'approcher les variables d'entrée fonctionnelles d'un code de calcul. Les deux objectifs qui ont été fixés sont d'approcher au mieux les variables fonctionnelles ainsi que leur lien avec la variable de sortie du code de calcul, appelée covariable. La décomposition fonctionnelle doit donc retenir les caractéristiques des

variables fonctionnelles les plus importantes pour l'approximation et les plus corrélées avec la covariable. Pour estimer la qualité de la décomposition fonctionnelle selon ces objectifs, on propose trois critères notés C_1^d, C_2^d et C_3^d. Le premier critère correspond à l'erreur quadratique moyenne (RelMSE pour *Relative Mean Squared Error*). Pour la i-ème variable fonctionnelle, la RelMSE entre les fonctions observées $\mathbf{z}_{i,1}, \ldots, \mathbf{z}_{i,n}$ et leurs approximations $\hat{\mathbf{z}}_{i,1}, \ldots, \hat{\mathbf{z}}_{i,n}$ s'écrit ainsi :

$$\text{RelMSE}_i = \frac{1}{n} \sum_{j=1}^{n} \frac{(\mathbf{z}_{i,j} - \hat{\mathbf{z}}_{i,j})^T (\mathbf{z}_{i,j} - \hat{\mathbf{z}}_{i,j})}{\mathbf{z}_{i,j}^T \mathbf{z}_{i,j}}, \ \forall i \in \{1 \ldots m\}. \tag{2.10}$$

Le critère C_1^d est défini comme le vecteur des RelMSE des différentes variables fonctionnelles $C_1^d = (\text{RelMSE}_1, \ldots, \text{RelMSE}_m)^T$.

Le deuxième critère proposé permet d'étudier la qualité de l'approximation des fonctions par les fonctions projetées sur la base. Pour cela, on considère la part de variance expliquée par la décomposition. Ainsi, pour des fonctions discrétisées $\{\mathbf{z}_{1,j}, \ldots, \mathbf{z}_{m,j}\}_{j=1,\ldots,n}$ et leurs projections respectives $\{\hat{\mathbf{z}}_{1,j}, \ldots, \hat{\mathbf{z}}_{m,j}\}_{j=1,\ldots,n}$, **ce second critère, noté C_2^d, est donné par :**

$$C_2^d = 1 - \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} (\mathbf{z}_{i,j} - \hat{\mathbf{z}}_{i,j})^T (\mathbf{z}_{i,j} - \hat{\mathbf{z}}_{i,j})}{\sum_{i=1}^{m} \sum_{j=1}^{n} (\mathbf{z}_{i,j} - \bar{\mathbf{z}}_i)^T (\mathbf{z}_{i,j} - \bar{\mathbf{z}}_i)}, \tag{2.11}$$

où $\bar{\mathbf{z}}_i = \sum_{j=1}^{n} \mathbf{z}_{i,j}$ est la moyenne de l'échantillon pour la i-ème variable aléatoire.

Pour quantifier le lien entre les variables projetées et la covariable, un métamodèle (Sacks et al., 1989), est utilisé pour prédire la covariable en fonction des coefficients de la décomposition. Un métamodèle est une fonction mathématique qui doit approcher le plus précisément possible le code sur l'ensemble du domaine de variation de ses paramètres d'entrée et avoir un temps d'exécution très faible voire négligeable. Ce modèle simplifié est construit à partir d'évaluations du code sur un échantillon de paramètres d'entrée. Si le métamodèle prédit efficacement la covariable, cela peut confirmer que les caractéristiques des variables fonctionnelles qui expliquent le mieux la covariable sont préservées dans la décomposition. Notons que dans le cas contraire, soit la covariable est mal expliquée par les coefficients, soit le métamodèle n'est pas adapté. Il n'est cependant pas possible de trancher entre ces deux alternatives. Parmi tous les types de métamodèles, on utilise ici le métamodèle processus gaussien (Oakley et O'Hagan, 2002; Rasmussen et Williams, 2006). Ce métamodèle est décrit dans la section 3.3.2. De nombreux auteurs (par exemple Welch et al. 1992; Marrel 2008) ont montré que ces métamodèles peuvent être d'efficaces approximations de codes de calcul, même en grande dimension. Pour évaluer la qualité du métamodèle, on considère le coefficient de prédictivité Q^2. Pour un échantillon de validation $Y_1, \ldots, Y_{n_t}$ avec $n_t \in \mathbb{N}$, indépendant de l'échantillon d'apprentissage, le Q^2 est défini comme suit :

$$Q^2 = 1 - \frac{\sum_{j=1}^{n_t} \left(Y_j - \hat{Y}_j\right)^2}{\sum_{j=1}^{n_t} \left(Y_j - \bar{Y}\right)^2}, \tag{2.12}$$

où $\bar{Y} = \frac{1}{n_t} \sum_{j=1}^{n_t} Y_j$ est la moyenne de l'échantillon de validation, et pour $j = 1, \ldots, n_t$, $\hat{Y}_j$ est la prédiction de Y_j par le métamodèle. En pratique, on ne dispose pas toujours d'un échantillon de validation et le Q^2 peut alors être calculé par validation croisée (Hastie et Tibshirani, 1990). Pour les tests numériques, celui-ci sera donc calculé sur une base de test et pour les applications industrielles traitées ici par validation croisée. **Par la suite, le Q^2 constitue notre critère C_3^d.**

2.2.7 Application des méthodes de décomposition fonctionnelle

Les méthodes de décomposition fonctionnelle présentées dans ce chapitre sont appliquées à un modèle analytique et aux deux cas d'étude présentés dans l'introduction : le choc thermique pressurisé et la rupture LiPoSo. Dans ces trois applications, la décomposition fonctionnelle de l'échantillon de données est choisie conditionnellement à trois objectifs : elle doit permettre de fournir une bonne représentation des variables aléatoires fonctionnelles $Z_1, ..., Z_m$ dans un espace de dimension finie, ses composantes doivent être liées à la covariable, et elle doit tenir compte de la dépendance entre les variables.

2.2.7.1 Application au modèle analytique

Dans le premier cas d'application, $m = 2$ variables fonctionnelles définies sur $I = [0\,;1]$ sont étudiées. On définit, tout d'abord, sur $\mathbb{R}$ un processus gaussien Z_1' de moyenne 0, de variance 1, dont le noyau de covariance est un noyau de Matèrn 5/2 (Rasmussen et Williams, 2006) défini comme suit :

$$k(t_1, t_2) = \left(1 + \frac{\sqrt{5}|t_1 - t_2|}{\theta} + \frac{5|t_1 - t_2|^2}{3\theta^2}\right) \exp\left(-\frac{\sqrt{5}|t_1 - t_2|}{\theta}\right),$$

où θ est un paramètre du noyau, appelé usuellement portée, qui vaut 0,2 pour Z_1'. Z_1' est conditionné en trois points :

$$Z_1'(-0.01) = 0,\ Z_1'(1.01) = 0 \text{ et } Z_1'(0.49) = 1.$$

De la même manière, on définit sur $\mathbb{R}$ un autre processus gaussien Z' de mêmes moyenne et covariance que Z_1'. Z' est conditionné en deux points comme suit :

$$Z'(-0.01) = 0 \text{ et } Z'(1.02) = 1.$$

La première variable aléatoire fonctionnelle étudiée est Z_1 définie comme la restriction à I de Z_1'. La seconde variable est $Z_2 = Z + Z_1$, où Z est la restriction de Z' à I. Des réalisations de Z_1 et Z_2 sont présentées sur la Figure 2.3. Par définition, les deux variables fonctionnelles sont dépendantes et leur corrélation temps par temps est représentée sur la Figure 2.4.

On définit un code $\mathcal{M}$ comme suit :

$$\mathcal{M} : (x_1, x_2, x_3, z_1, z_2) \mapsto 2x_1^2 x_2 + 2\int_0^1 (z_1(t) + z_2(t))\,\mathrm{d}t + x_3 \max_{t\in[0,1]} z_2(t) + x_2. \tag{2.13}$$

On définit la covariable Y par $Y = \mathcal{M}(X_1, X_2, X_3, Z_1, Z_2)$, où Z_1, Z_2 sont les variables fonctionnelles précédemment définies et X_1, X_2, X_3 sont 3 variables aléatoires scalaires indépendantes et uniformes sur $[0; 1]$.

Un échantillon de $n = 200$ réalisations $\{z_{i,j}\}_{j=1,...,n}$, pour $i \in \{1, 2\}$, est connu. n réalisations indépendantes et identiquement distribuées (i.i.d.) des variables scalaires $\{x_{1,j}, x_{2,j}, x_{3,j}\}_{j=1,...,n}$ sont générées et les sorties du code $\mathcal{M}$ correspondantes $\{y_j = \mathcal{M}(x_{1,j}, x_{2,j}, x_{3,j}, z_{1,j}, z_{2,j})\}_{j=1,...,n}$ sont alors calculées. L'échantillon utilisé dans les tests ci-dessous est représenté sur la Figure 2.3. Les fonctions de l'échantillon sont discrétisées sur $p = 512$ points équirépartis et notés $t_1, \ldots, t_p \in I$. La version discrétisée de $z_{i,j}$ est notée $\mathbf{z}_{i,j} = (z_{i,j}(t_1), \ldots, z_{i,j}(t_p))$, $i = 1, 2$ et $j = 1, \ldots, n$.

Les variables fonctionnelles Z_1 et Z_2 sont d'abord décomposées séparément sur les bases présentées dans les sections précédentes. La Figure 2.5 représente les critères C_1^d (erreurs quadratiques moyennes relatives) pour les fonctions Z_1 et Z_2 sur des bases de taille 4 à 20, obtenues avec l'ACP (ronds noirs), la PLS (triangles rouges), la transformée en ondelettes (croix vertes), les paquets d'ondelettes (« x » bleus) et les splines cubiques (losanges orange). L'ordonnée des deux graphiques est en échelle logarithmique. Par définition, l'ACP donne la RelMSE la plus faible. La décomposition PLS présente aussi de très bons résultats avec des erreurs d'approximation très faibles. Pour la variable Z_1, la décomposition sur une base de splines cubiques donne d'assez bons résultats, en particulier pour des tailles de base supérieures à 15. Les erreurs des deux autres décompositions basées sur les ondelettes sont beaucoup plus élevées. Pour

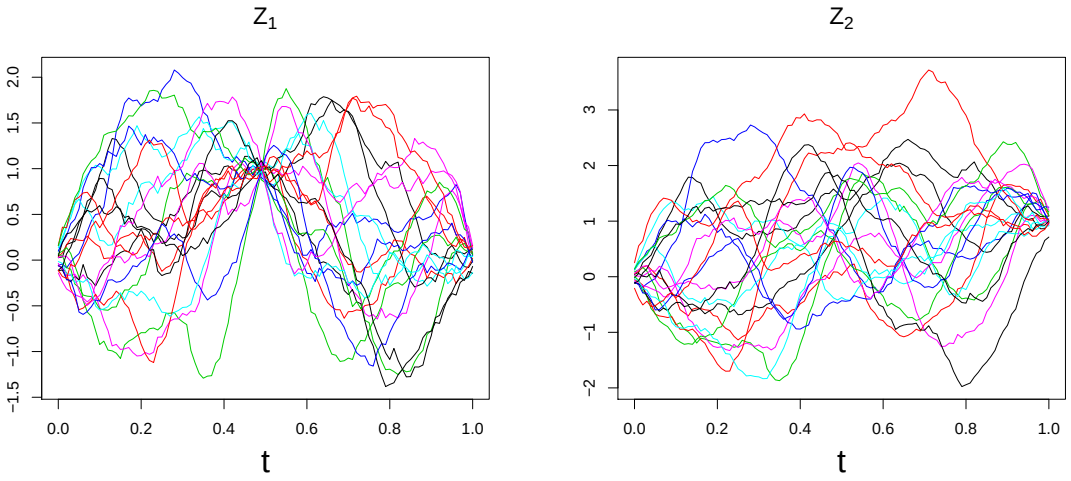

FIGURE 2.3 – Cas analytique : échantillon de 100 réalisations des variables aléatoires Z_1 (à gauche) et Z_2 (à droite) sur $[0; 1]$.

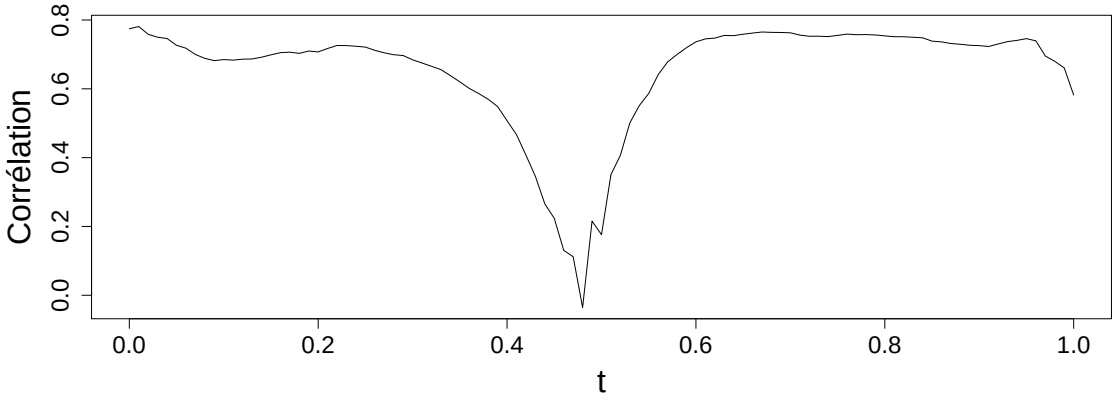

FIGURE 2.4 – Cas analytique : corrélation terme à terme entre les variables Z_1 et Z_2.

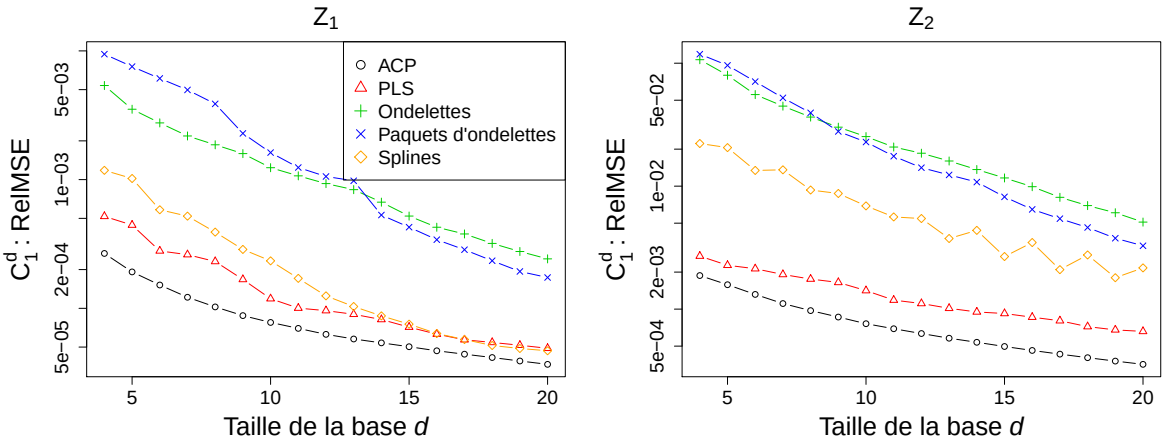

FIGURE 2.5 – Cas analytique : critère C_1^d (erreur quadratique moyenne relative d'approximation) pour Z_1 (à gauche) et Z_2 (à droite), avec les différentes méthodes de décomposition (échelle logarithmique).

la variable Z_2, les trois décompositions sur des bases non estimées sur les données donnent des résultats bien moins bons que l'ACP et la PLS, mieux adaptées. **L'un des objectifs de la décomposition étant de bien reproduire les fonctions observées, seules les décompositions ACP et PLS seront utilisées et étudiées dans le reste de l'étude.**

On s'intéresse maintenant aux décompositions ACP et PLS simultanées. Dans la section 2.2.4, l'ACP simultanée est définie comme la méthode qui consiste à appliquer la décomposition ACP aux vecteurs $[\mathbf{z}_{1,j}/N_1, \ldots, \mathbf{z}_{m,j}/N_m]$, où $N_1, \ldots, N_m$ sont des facteurs de normalisation. La PLS simultanée est définie de la même manière à la différence qu'aucune normalisation n'est appliquée aux données puisque celles-ci sont standardisées au cours de l'algorithme de la PLS. Pour l'ACPS, les trois facteurs de normalisation présentés dans la section 2.2.4.3 ainsi que la normalisation obtenue en minimisant l'erreur quadratique maximale selon la méthode de Perrin et al. (2013) sont comparés. La Figure 2.6 présente la RelMSE des ACPS réalisées sans normalisation et avec chacune des normalisations étudiées. Quelle que soit la normalisation, les APCS donnent des résultats assez similaires. Cependant, sans normalisation (en bleu), la deuxième variable fonctionnelle est mieux approchée par l'ACPS que la première. Les ACPS avec les trois normalisations proposées (par le maximum, la variance et l'écart-type) sont très proches les unes des autres, celle basée sur le maximum donnant des résultats très légèrement meilleurs. L'ACPS avec la normalisation sélectionnée par la méthode de Perrin et al. (2013) donne aussi de bons résultats. Cela est dû au fait que les deux variables fonctionnelles ainsi que leurs variations sont du même ordre de grandeur. On s'intéresse à présent à l'erreur quadratique maximale des variables fonctionnelles, notée ε_∞ et définie dans l'équation (2.6). La Figure 2.7 montre que les différentes normalisations ont des erreurs ε_∞ très proches. On peut aussi remarquer que l'algorithme de Perrin et al. (2013) qui cherche la normalisation minimisant ε_∞ n'a pourtant pas donné la normalisation optimale. Cela peut s'expliquer par le fait que l'optimum est difficile à déterminer dans cet exemple. **Au vu des différents résultats, la normalisation par maximum proposée dans la section 2.2.4.3 est sélectionnée pour la suite du traitement du cas analytique.**

Les RelMSE de ces décompositions simultanées sont ensuite comparées aux RelMSE des ACP et PLS simples, *i.e* faites indépendamment sur chaque variable aléatoire fonctionnelle Z_i pour $1 \leq i \leq 2$. La Figure 2.8 représente en fonction du nombre de composantes d de l'ACPS (respectivement SPLS) le nombre de composantes d_1 et d_2 dans les ACP (respectivement PLS) de Z_1 et Z_2 pour obtenir une erreur d'approximation équivalente. Plus précisément, les triangles rouges et les ronds noirs représentent le nombre maximal de composantes (d_1 et d_2) des décompositions de Z_1 et Z_2 tel que les RelMSE de ces décompositions soient supérieures ou égales à celle de la décomposition simultanée. Les croix bleues représentent la somme des deux autres courbes, *i.e.* la somme $d_1 + d_2$. Ces croix bleues sont strictement

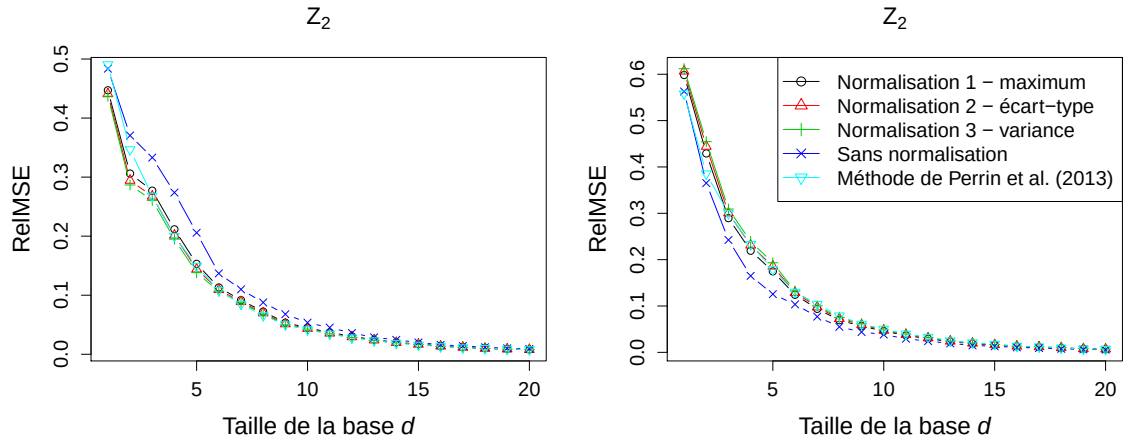

FIGURE 2.6 – Cas analytique : RelMSE d'approximation de Z_1 (à gauche), et de Z_2 (à droite) obtenues avec l'ACP simultanée sans normalisation, avec les trois normalisations proposées dans la section 2.2.4.3 et avec la normalisation sélectionnée par la méthode de Perrin et al. (2013).

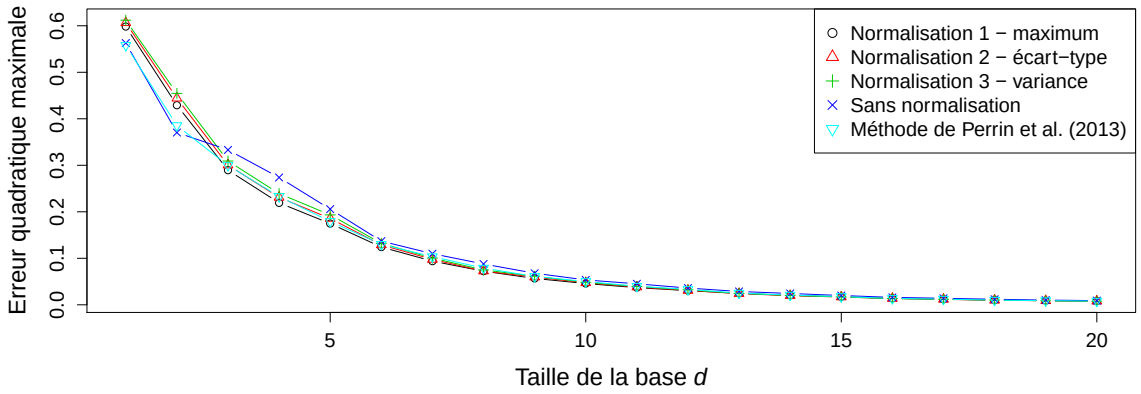

FIGURE 2.7 – Cas analytique : erreur quadratique maximale de l'ACP simultanée avec les trois normalisations proposées dans la section 2.2.4.3, sans normalisation et avec la normalisation sélectionnée par la méthode de Perrin et al. (2013).

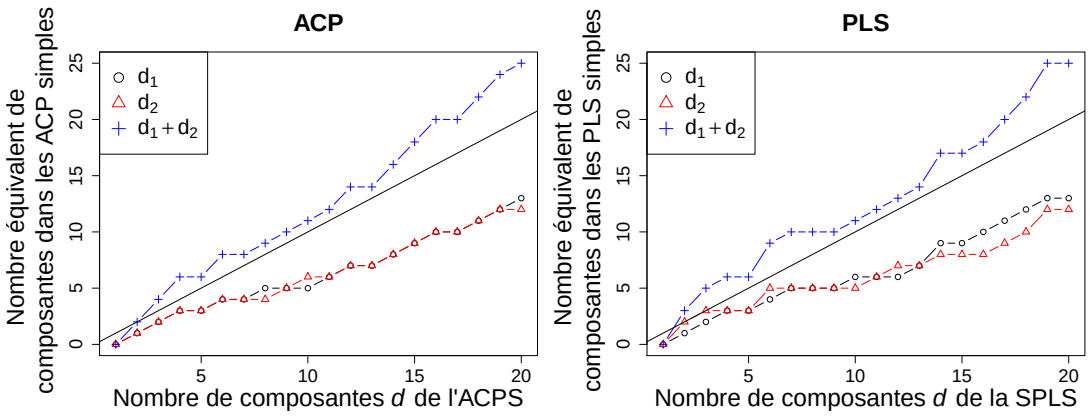

FIGURE 2.8 – Cas analytique : nombre maximal de composantes des décompositions sur chaque transitoire tel que les RelMSE des décompositions soient supérieures ou égales à celles de la décomposition simultanée pour l'ACP (à gauche) et pour la PLS (à droite).

supérieures à la courbe "$y = x$" en noir à partir de 3 composantes pour l'ACPS et 2 pour la SPLS. Cela signifie qu'avec 3 composantes dans l'ACPS on obtient une meilleure RelMSE pour Z_1 et Z_2 qu'en décomposant chaque variable sur des ACP simples dont la somme des nombres de composantes est 3. A partir de 3 composantes (respectivement 2), l'ACPS (resp. SPLS) simultanée approche donc mieux Z_1 et Z_2 à nombre de composantes égal que l'ACP (resp. PLS). Ce résultat n'est pas étonnant car les variables aléatoires fonctionnelles Z_1 et Z_2 sont très corrélées (cf. Figure 2.4). **Dans la suite du cas test analytique, on conserve donc les versions simultanées des décompositions fonctionnelles ACP et PLS.**

Ici, la décomposition fonctionnelle a deux objectifs. Elle doit permettre d'approcher les variables fonctionnelles tout en conservant les caractéristiques de ces variables qui expliquent le mieux la covariable. Pour quantifier la qualité des décompositions étudiées vis-à-vis de ces deux objectifs, les critères C_2^d et C_3^d définis dans les équations (2.11) et (2.12) sont respectivement utilisés. Le premier est la variance expliquée par la décomposition, c'est-à-dire le pourcentage de variabilité qui a été conservée dans l'approximation de l'échantillon initial. Le deuxième critère est le Q^2 du métamodèle processus gaussien qui approche le lien entre les coefficients de la décomposition, les variables scalaires X_1, X_2, X_3 et la covariable Y. Pour choisir la décomposition la plus adaptée aux objectifs et aux données, les valeurs de ces deux critères appliqués aux deux décompositions sont comparées. La Figure 2.9 présente respectivement à gauche et à droite les deux critères, en noir pour la SPLS et en rouge pour l'ACPS. La variance expliquée par la décomposition ACPS est supérieure à celle expliquée par la SPLS par définition. Cependant, à partir d'une base de 6 ou 7 composantes, la différence entre les deux variances expliquées devient assez faible. De plus, elles sont toutes les deux assez hautes : celle de la SPLS vaut environ 0,89 pour une base de 6 fonctions. Le coefficient Q^2 de la SPLS reste supérieur à celui de l'ACPS pour des bases de moins de 15 composantes, mais la différence entre les deux Q^2 devient faible à partir de 12 composantes. La décomposition SPLS semble dans ce cas d'étude réaliser un bon compromis entre l'approximation des variables fonctionnelles et l'approximation du lien entre ces variables fonctionnelles et la covariable. Le Q^2 reste assez élevé quelle que soit la taille de la base. Cela peut s'expliquer par le fait qu'il ne tient pas seulement compte du lien entre les approximations des variables fonctionnelles et la covariable, mais aussi du lien entre la covariable et les variables scalaires X_1, X_2, X_3 qui ne sont pas approchées par les décompositions. **La décomposition SPLS, qui réalise un bon compromis entre variance expliquée et explication de la covariable, sera donc retenue dans la suite du traitement du cas-test analytique dans la section 2.3.6.1.**

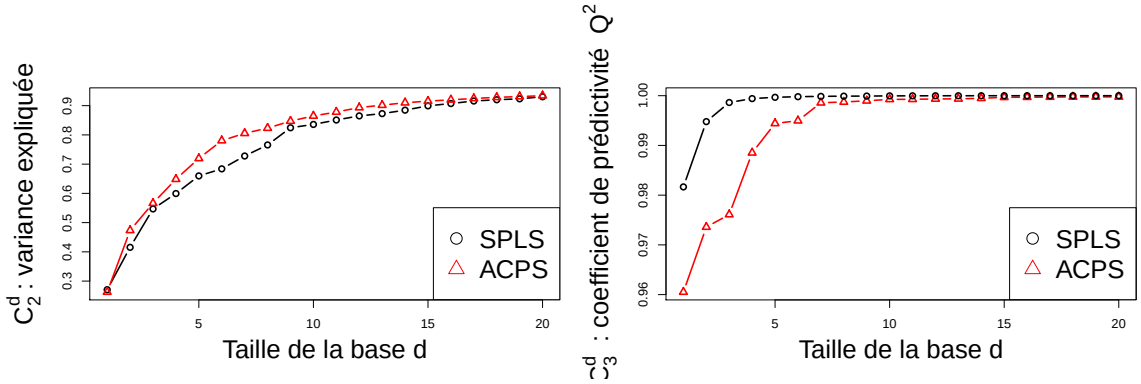

FIGURE 2.9 – Cas analytique : part de variance expliquée par la SPLS et l'ACPS en fonction de la taille de la base (à gauche) et coefficient de prédictivité Q^2 des métamodèles processus gaussien liant les composantes de l'ACPS et de la SPLS à Y en fonction de la taille de la base (à droite).

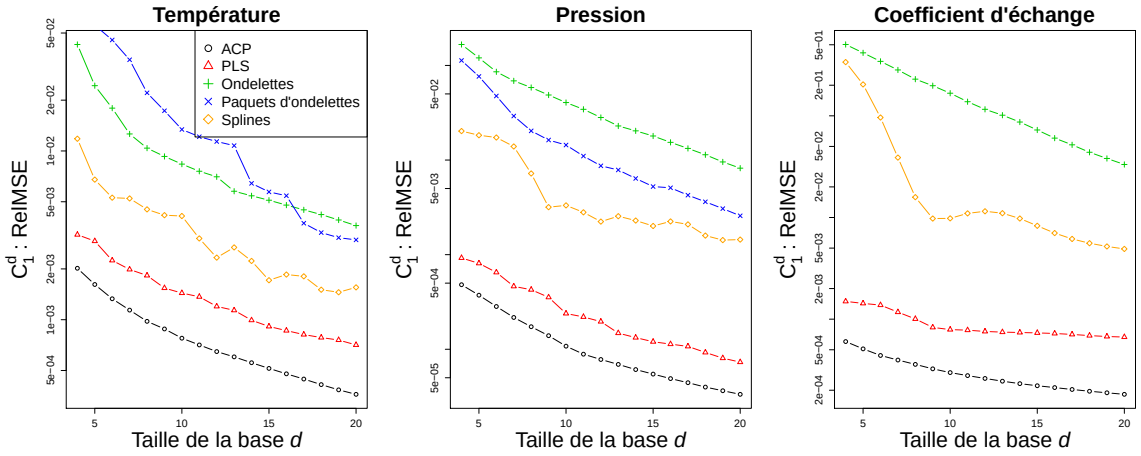

FIGURE 2.10 – Cas PTS : critère C_1^d (RelMSE) pour la température (à gauche), la pression (au milieu) et le coefficient d'échange (à droite) pour les différentes décompositions (échelle logarithmique).

2.2.7.2 Application au cas-test du choc thermique pressurisé

Le cas d'application présenté ici est celui du choc thermique pressurisé (PTS) décrit dans l'introduction. Trois variables aléatoires fonctionnelles sont étudiées : les transitoires de température, pression et coefficient d'échange. On considère un échantillon i.i.d. de $n = 1000$ triplets de transitoires. 1000 réalisations i.i.d. des 10 variables scalaires sont générées. Les 1000 covariables correspondantes, à savoir les critères de sécurité (CS), sont ensuite obtenues en évaluant le code CAST3M avec les 1000 jeux de paramètres (transitoires et variables scalaires). Chacune des trois variables fonctionnelles est décomposée séparément sur les bases présentées dans la section 2.2. La Figure 2.10 présente les RelMSE (critère C_d^1) des transitoires de température (à gauche), pression (au centre) et coefficient d'échange (à droite) en fonction du nombre de composantes dans la décomposition. Pour ce critère, les décompositions ACP et PLS donnent de bien meilleurs résultats que les autres décompositions sur l'ensemble des 3 variables. L'erreur d'approximation de l'ACP ou de la PLS avec 5 composantes est, par exemple, inférieure aux erreurs des ondelettes et paquets d'ondelettes avec 20 composantes. Comme attendu, l'ACP est meilleure que la PLS pour le critère C_d^1. **Dans la suite du traitement du cas PTS, on s'intéressera donc à l'ACP et à la PLS.**

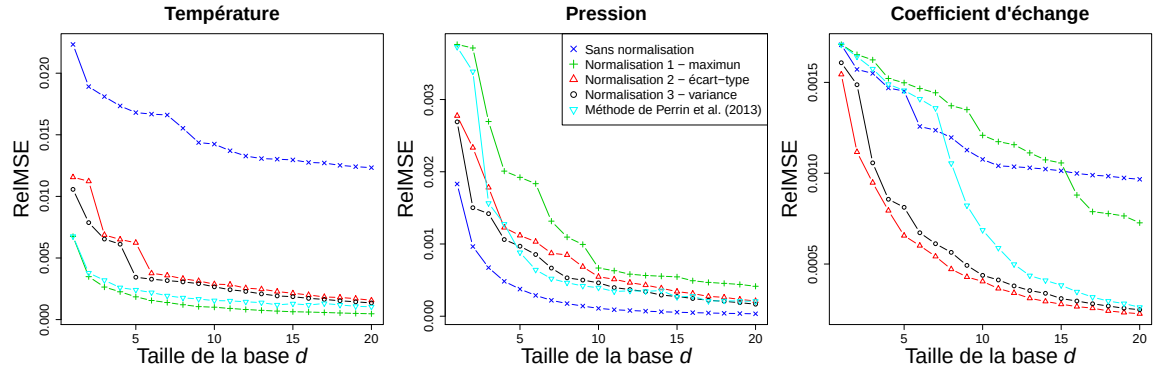

FIGURE 2.11 – Cas PTS : RelMSE des trois transitoires pour l'ACP simultanée sans normalisation, avec les trois normalisations proposées dans la section 2.2.4.3 et celle obtenue avec la méthode de Perrin et al. (2013).

On s'intéresse plus particulièrement aux décompositions ACP et PLS simultanées puisque l'objectif est de décomposer simultanément les trois variables fonctionnelles dépendantes. De plus, comme dans le cas analytique, on montre dans l'annexe C.1 que les décompositions simultanées (ACPS ou SPLS) permettent de mieux approcher les variables fonctionnelles que les décompositions simples (ACP ou PLS) sur chaque variable à nombre total de composantes égal.

On compare ici les trois mêmes facteurs de normalisation que ceux considérés dans le cas analytique (normalisation par le maximum, la variance et l'écart-type) ainsi que la normalisation obtenue en minimisant l'erreur quadratique maximale (Perrin et al., 2013). La Figure 2.11 présente la RelMSE des ACPS réalisées avec chaque normalisation et sans normalisation (en bleu). Sans normalisation, le transitoire de pression est beaucoup mieux approché que les deux autres. En effet, l'ordre de grandeur de la pression est 10^7 alors que l'ordre de grandeur de la température est 10^2 et celui du coefficient d'échange est 10^4. L'ACP simultanée avec la normalisation 1 (*i.e.* la normalisation par le maximum des transitoires) favorise la température. En effet, la courbe de la RelMSE de cette méthode est bien inférieure à celle des autres normalisations pour la température, alors qu'elle est supérieure pour les autres transitoires. Cela s'explique par le fait que les variations entre courbes sont plus importantes sur ce transitoire que sur les deux autres. Les deux autres normalisations par la somme des écarts-types et la racine carrée de la somme des variances diminuent l'importance du transitoire de température. En effet, ces normalisations sont plus grandes pour des variables ayant des variations plus importantes. La normalisation estimée par la méthode de Perrin et al. (2013) avantage plus particulièrement la température et le coefficient d'échange. Pour comparer ces normalisations, on utilise l'erreur quadratique maximale, ε_∞, définie dans l'équation (2.6). Elle est représentée sur la Figure 2.12. L'APCS avec la normalisation 1 minimise cette erreur parmi les cinq courbes pour toutes les tailles de base sauf pour $d = 2$ où la normalisation obtenue avec la méthode de Perrin et al. (2013) minimise l'erreur maximale. La normalisation sélectionnée par l'algorithme de Perrin et al. (2013) dont l'objectif est de minimiser ε_∞ est sous-optimale, mais son erreur est très proche de celle obtenue avec la normalisation 1. Les normalisations 2 et 3 donnent des résultats moins bons mais assez proches des deux normalisations pour des bases de plus de 5 fonctions. Sans normalisation, l'erreur ε_∞ est beaucoup plus élevée. **Au vu de ces différents résultats, on choisit de sélectionner la normalisation 1 par le maximum de chaque variable pour le cas-test du choc thermique pressurisé.**

Les décompositions SPLS et ACPS sont maintenant comparées selon leur qualité d'approximation des transitoires et d'explication de la covariable, à savoir le CS. La partie gauche de la Figure 2.13 présente le critère C_2^d, la variance expliquée par les deux décompositions fonctionnelles, en fonction de leur nombre de composantes. La variance expliquée par la SPLS est plus faible que celle de l'ACPS (par définition de l'ACPS) mais, à partir de 5 composantes, la variance expliquée par la SPLS devient assez proche de la

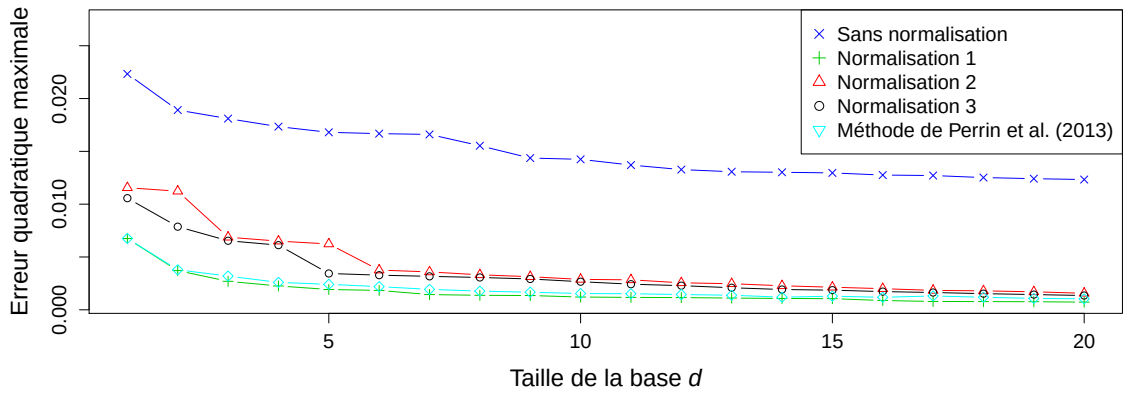

FIGURE 2.12 – Cas PTS : erreur quadratique maximale d'approximation des trois transitoires pour l'ACP simultanée sans normalisation, avec les trois normalisations proposées dans la section 2.2.4.3 et celle obtenue avec la méthode de Perrin et al. (2013).

variance expliquée par l'ACPS. Pour quantifier le lien entre les coefficients de la décomposition et le critère de sécurité, un métamodèle processus gaussien est estimé entre les coefficients, les variables scalaires en entrée de CAST3M et le CS. La partie droite de la Figure 2.13 représente le Q^2 du métamodèle pour les deux décompositions. Comme attendu, la SPLS donne des Q^2 plus élevés que l'ACPS quelle que soit la taille de la base ; la SPLS fournit des composantes qui permettent de mieux expliquer le CS. Les valeurs des Q^2 obtenues sont élevées ($> 0{,}8$) quelle que soit la taille de la base ; cela peut s'expliquer par le fait que le Q^2 ne tient pas seulement compte du lien entre les approximations des variables fonctionnelles et la covariable, mais aussi du lien entre les variables scalaires, qui ne sont pas approchées, et la covariable. La décomposition SPLS semble donc être un bon compromis puisqu'elle permet à la fois de bien approcher les transitoires et de conserver le lien entre les approximations des transitoires et le critère de sécurité. **Ainsi, la décomposition SPLS est retenue, pour la suite du traitement du cas-test PTS.**

Enfin, on compare les transitoires de la base d'apprentissage et leurs approximations obtenues avec une décomposition SPLS de dix composantes, représentés sur la Figure 2.14. La tendance générale des transitoires de température est assez bien approchée ainsi que les 1000 premières secondes. La deuxième partie de l'intervalle est moins bien approchée. On remarque aussi un pic vers 1500 secondes sur les courbes approchées qui n'est pas présent dans l'échantillon d'apprentissage. Le transitoire de pression est globalement bien approché. Cependant, les approximations semblent moins variables autour de 1500 secondes que les courbes réelles. Enfin, le coefficient d'échange est bien approché avant 500 secondes. Au-delà, la variabilité de l'échantillon d'apprentissage n'est pas retrouvée dans l'échantillon approché.

2.2.7.3 Application au cas-test de la rupture LiPoSo

Dans le cas-test de la rupture LiPoSo, trois variables aléatoires fonctionnelles dépendantes sont étudiées : le débit, la puissance et la température, notées respectivement Z_1, Z_2, Z_3. Les réalisations de ces variables fonctionnelles sont des fonctions du temps $t \in [0; T] \subset \mathbb{R}$, avec $T = 100s$. Un échantillon i.i.d. de $n = 200$ triplets de réalisations de ces variables est disponible. Dans ce cas-test, on ne considère pas de covariable.

Les réalisations de l'échantillon sont décomposées à l'aide des méthodes présentées dans la section 2.2. La PLS n'est pas appliquée ici car on ne considère pas de covariable. En effet, le code de calcul Trio_U MCT étant très coûteux en temps de calcul, il n'a pas été évalué sur les réalisations des variables fonctionnelles utilisées pour la quantification des incertitudes. La Figure 2.15 présente les RelMSE des transitoires de température (à gauche), pression (au centre) et coefficient d'échange (à droite) en fonction

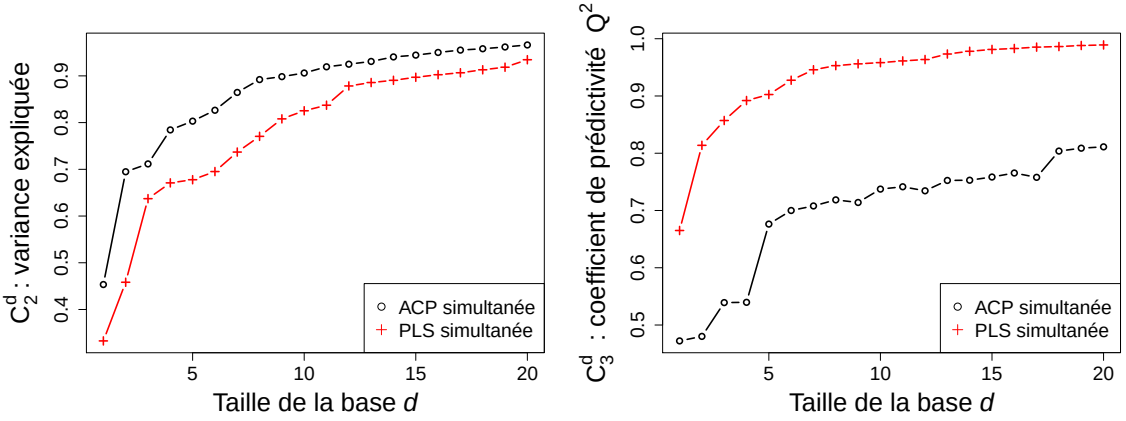

FIGURE 2.13 – Cas PTS : part de variance expliquée par la SPLS ou l'ACPS en fonction de la taille de la base (à gauche) et coefficient de prédictivité Q^2 des métamodèles processus gaussien liant les composantes de l'ACPS ou de la SPLS à la covariable en fonction de la taille de la base (à droite).

de la taille de la base. Les erreurs sont faibles pour les trois variables fonctionnelles. En particulier, la RelMSE de la température est très faible. La décomposition ACP donne les meilleurs résultats sur les trois variables fonctionnelles, par définition. Les trois autres décompositions donnent des erreurs bien plus élevées que l'ACP. L'erreur obtenue avec les ondelettes et paquets d'ondelettes sont notamment très importantes sur la température. **On n'étudie dans la suite que la décomposition ACP.**

Cependant, les réalisations de la variable de puissance sont mal approchées par l'ACP (ainsi que par les autres méthodes de décomposition utilisées). Sur la Figure 2.16, sont représentées les approximations de la puissance obtenues avec une décomposition ACP de 5 composantes sur l'intervalle $[0;5]$. Les réalisations sont mal approchées entre 2 et 3 secondes. Pour pallier ce problème, on propose d'appliquer l'ACP à une transformation de la puissance et non directement à cette variable. La transformation consiste à appliquer une translation aux réalisations de la variable Z_2 pour que le saut (*i.e.* l'instant où la décroissance de température devient forte) soit le même pour toutes les courbes translatées. Plus précisément, on note t_i^0 l'emplacement du saut pour la réalisation $z_{2,i}$, et on note T^0 la variable aléatoire associée. A partir des transitoires $\{z_{2,i} : I = [0,T] \to \mathbb{R}\}_{i=1,\dots,n}$, on définit les transitoires translatés $\{\tilde{z}_{2,i} : I = [0,T] \to \mathbb{R}\}_{i=1,\dots,n}$:

$$\tilde{z}_{2,i}(t) = z_{2,i}(t - t_i^0 + t_{min}^0), \forall t \in \left[t_i^0 - t_{min}^0; T - (t_{max}^0 - t_i^0)\right],$$

où $t_{max}^0 = \max_{1\leq i\leq n}(t_i^0)$ et $t_{min}^0 = \min_{1\leq i\leq n}(t_i^0)$. Pour une réalisation $z_{2,i}$, la variable transformée, $\tilde{z}_{2,i}$ est restreinte à un intervalle plus petit que $[0;T]$ afin d'appliquer la décomposition à des fonctions pour lesquelles le saut se produit au même temps. L'ACP est alors appliquée aux variables Z_2 et T^0. Pour transformer l'approximation de la variable $\tilde{Z}_2$ par la décomposition ACP en l'approximation de Z_2, la translation inverse est appliquée. On note $\hat{Z}_2$ la variable ainsi transformée. Cependant, comme la variable $\tilde{Z}_2$ est définie sur $\left[t_i^0 - t_{min}^0; T - (t_{max}^0 - t_i^0)\right]$, $\hat{z}_{2,i}$ doit être extrapolée sur les intervalles $I_1 = \left[0; t_i^0\right[$ et $I_2 = \left]T - (t_{max}^0 - t_i^0); T\right]$. Des extrapolations linéaires et constantes sont réalisées respectivement sur I_1 et I_2. L'effet de la transformation est illustrée sur la Figure 2.17.

On s'intéresse uniquement à l'ACP simultanée puisque l'objectif est de décomposer simultanément les trois variables fonctionnelles dépendantes. De plus, de la même manière que dans le cas analytique, on montre dans l'annexe C.2 que, dans le cas de la rupture LiPoSo, la décomposition ACPS permet de mieux approcher les variables fonctionnelles que la décomposition ACP seule, à nombre total de composantes égal.

Le choix de la normalisation appliquée à chacune des variables aléatoires fonctionnelles est prépondé-

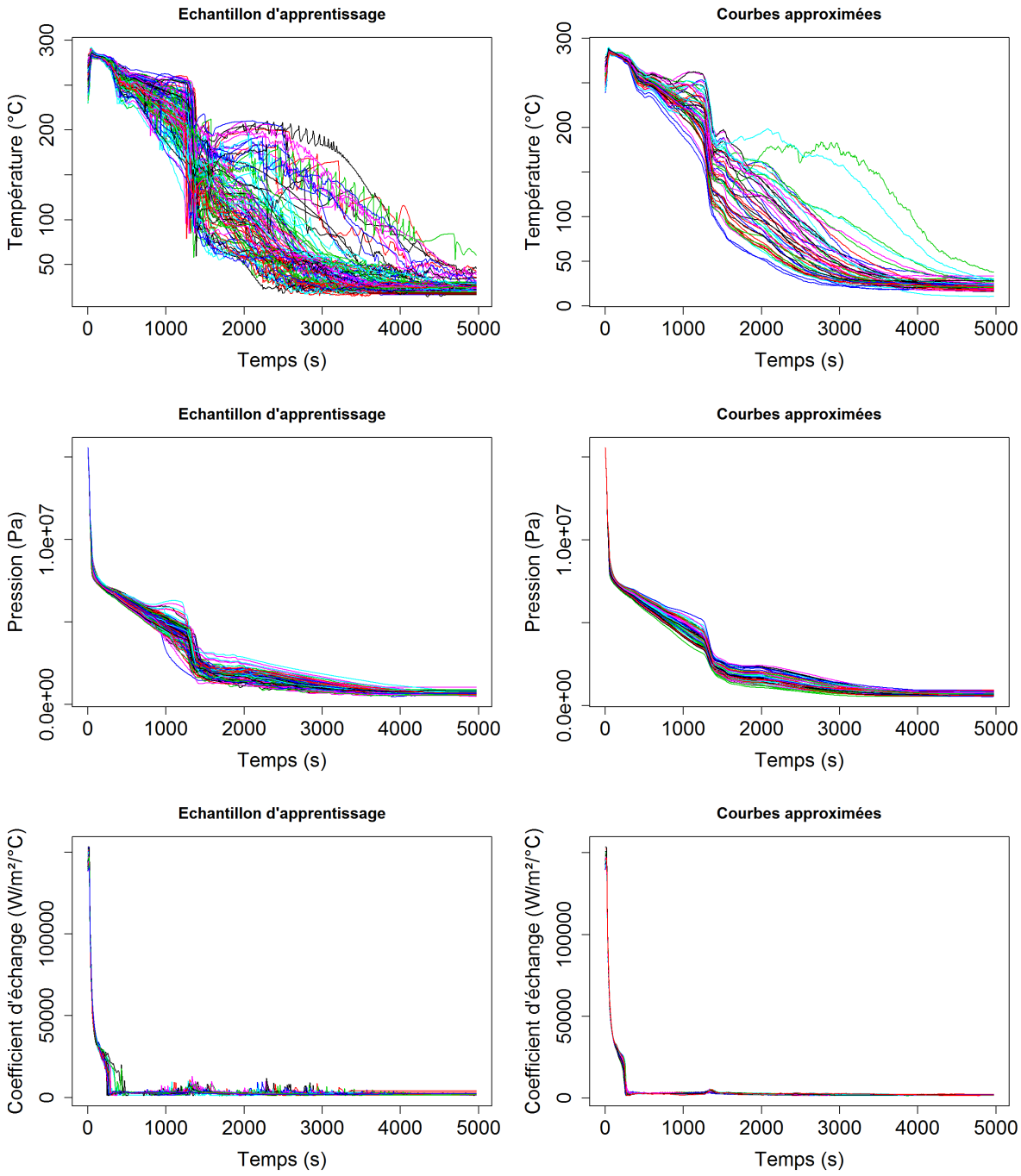

FIGURE 2.14 – Cas PTS : 100 courbes de la base d'apprentissage (à gauche) et leurs approximations sur base SPLS de taille dix (à droite) pour les variables de température (en haut), pression (au milieu) et coefficient d'échange (en bas).

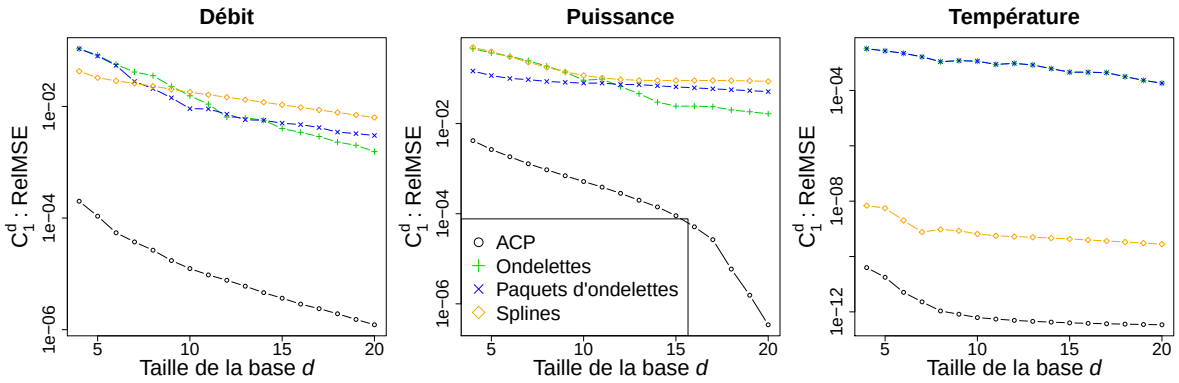

FIGURE 2.15 – Cas LiPoSo : critère C_1^d (RelMSE) pour le débit (à gauche), la puissance (au milieu) et la température (à droite) pour les décompositions ACP, splines, décomposition en ondelettes et en paquet d'ondelettes (échelle logarithmique).

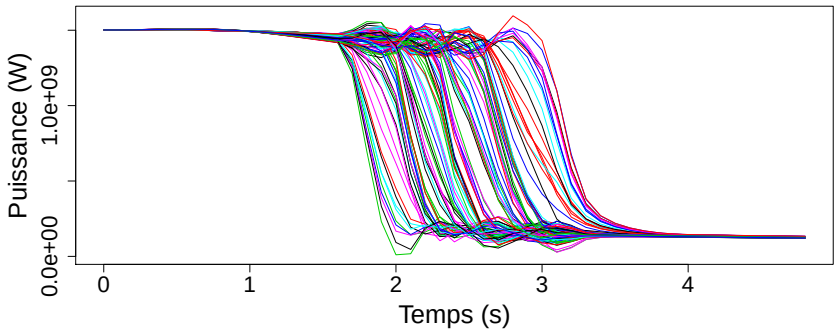

FIGURE 2.16 – Cas LiPoSo : approximations de réalisations de la variable fonctionnelle de puissance obtenues avec une décomposition ACP de 5 composantes.

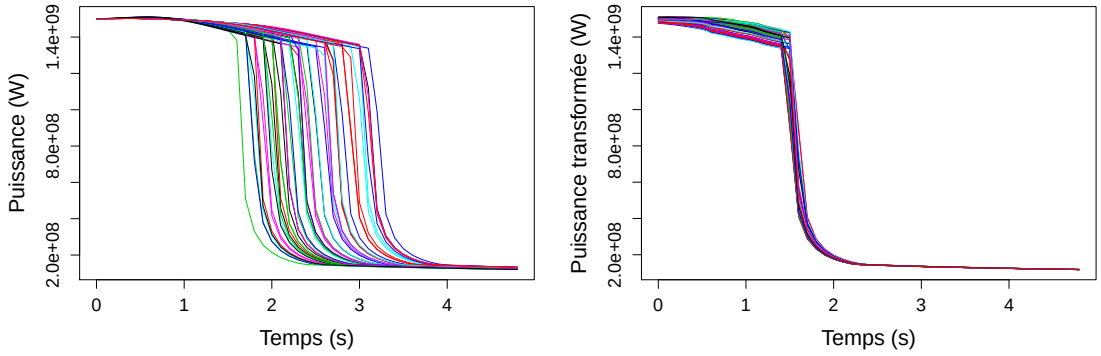

FIGURE 2.17 – Cas LiPoSo : Echantillon de 50 courbes de puissance (à gauche) et de leurs transformations (à droite).

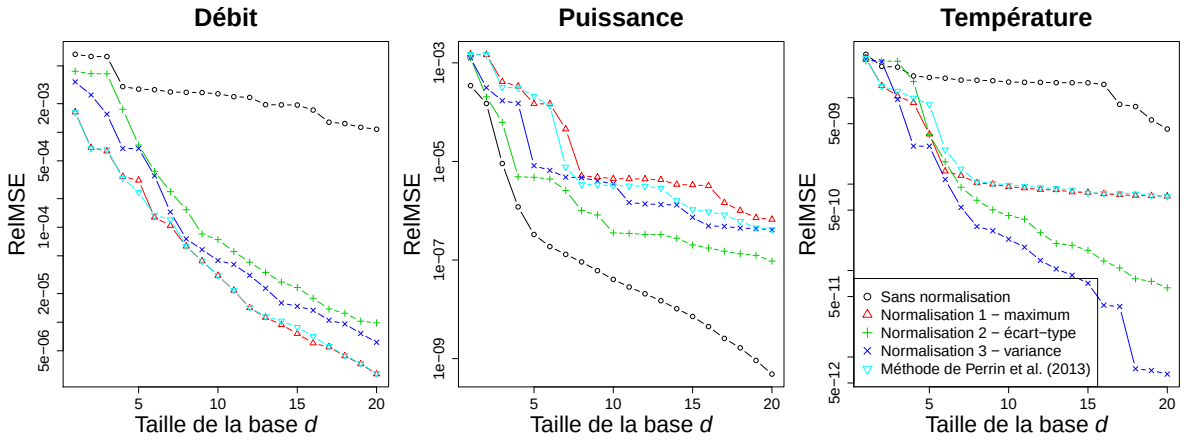

FIGURE 2.18 – Cas LiPoSo : RelMSE d'approximation des trois transitoires pour l'ACP simultanée sans normalisation, avec les trois normalisations proposées dans la section 2.2.4.3 et celle obtenue avec la méthode de Perrin et al. (2013).

rant dans cette décomposition. On compare ici les ACPS sans normalisation, avec les trois normalisations proposées dans la section 2.2.4.3 et avec la normalisation obtenue par la méthode proposée par Perrin et al. (2013). Tout d'abord, on compare les RelMSE obtenues sans normalisation et celles obtenues avec les normalisations (Figure 2.18). Sans normalisation, la puissance, qui a l'ordre de grandeur le plus élevé, est la plus avantagée. La normalisation 1 et celle obtenue avec la méthode de Perrin et al. (2013) donnent des résultats très proches en particulier pour le débit et la puissance. Pour comparer ces normalisations, on utilise l'erreur quadratique maximale, ε_∞, définie dans l'équation (2.6). Elle est représentée sur la Figure 2.19 en fonction de la taille de la base de décomposition. L'ACPS sans normalisation donne une erreur quadratique maximale bien plus élevée que les autres décompositions. En effet, dans celle-ci, l'effort d'approximation est concentré sur la variable dont la valeur moyenne est la plus élevée, à savoir la puissance. Ainsi, son erreur relative devient très faible alors que celles des deux autres variables restent plus élevées. La première normalisation proposée (par le maximum de chaque variable fonctionnelle) donne des erreurs maximales très proches de celles obtenues avec la normalisation calculée par la méthode de Perrin et al. (2013). Les deux autres normalisations donnent une erreur plus élevée pour des bases de moins de 7 ou 8 composantes mais donnent ensuite des résultats équivalents à ceux de la première normalisation et de la normalisation obtenue avec la méthode de Perrin et al. (2013). **Dans la suite, on utilise la normalisation optimisée par la méthode de Perrin et al. (2013).**

On trace le critère C_2^d pour la décomposition ACPS avec la normalisation obtenue avec la méthode de Perrin et al. (2013) dans la Figure 2.20. La variance expliquée dépasse 90% à partir de 4 composantes retenues dans la décomposition et 95% à partir de 6 composantes. Ce critère confirme donc que la décomposition utilisée approche bien les trois variables fonctionnelles. Le critère C_3^d n'est pas calculé dans cet exemple, car aucune covariable n'est prise en compte.

Finalement, on compare les fonctions de l'échantillon d'apprentissage et leurs approximation sur base ACPS de taille $d = 10$, dans la Figure 2.21. Les fonctions sont très bien modélisées, comme on pouvait s'y attendre puisque la variance expliquée par une base de taille dix est très élevée (supérieure à 99,5%). En particulier, la puissance est bien mieux approchée avec la transformation proposée que sans transformation (cf. Figure 2.16).

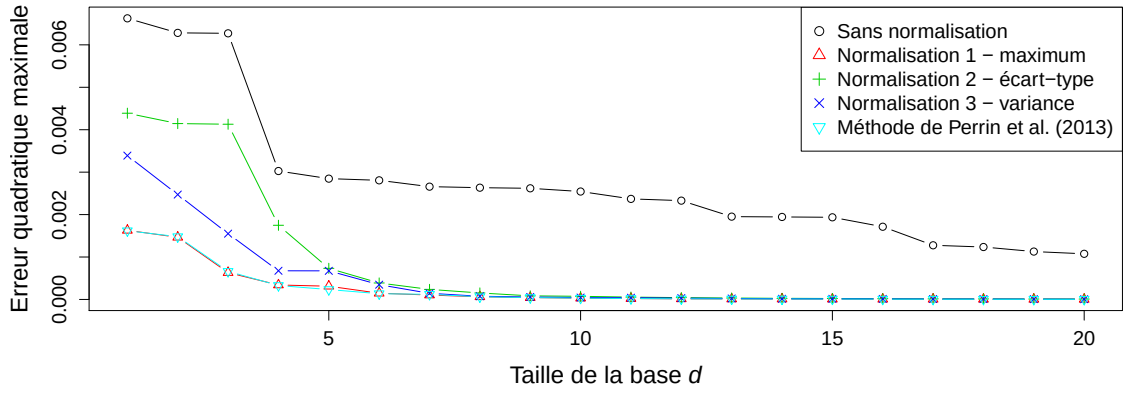

FIGURE 2.19 – Cas LiPoSo : erreur quadratique maximale d'approximation des trois transitoires pour l'ACP simultanée sans normalisation, avec les trois normalisations proposées dans la section 2.2.4.3 et avec celle obtenue avec la méthode de Perrin et al. (2013).

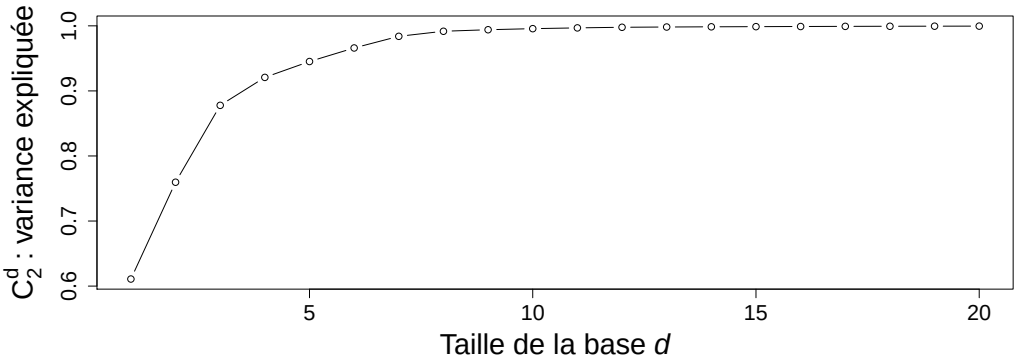

FIGURE 2.20 – Cas LiPoSo : part de variance expliquée (critère C_2^d) par l'ACP simultanée en fonction de la taille de la base.

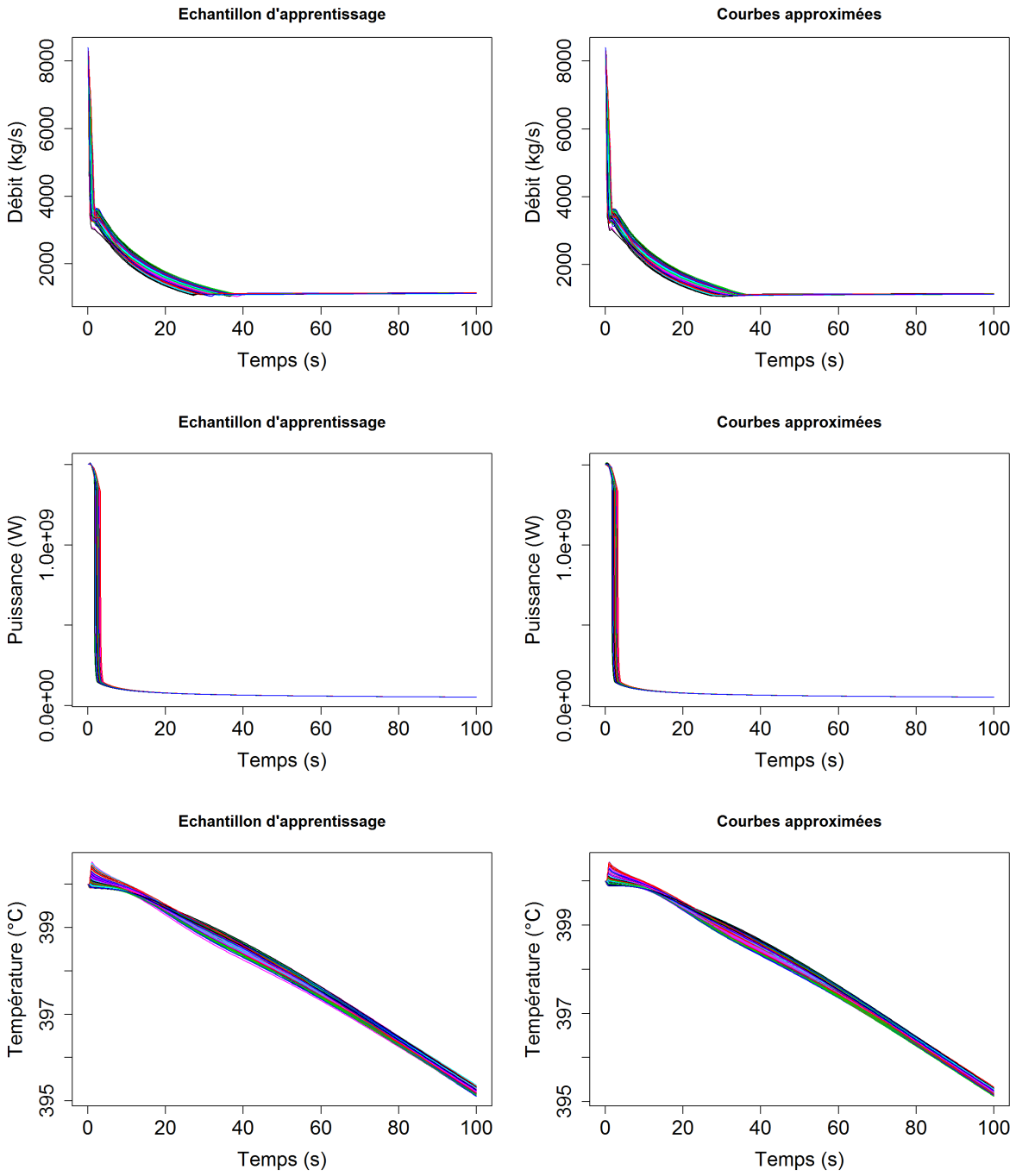

FIGURE 2.21 – Cas LiPoSo : 100 courbes de la base d'apprentissage (à gauche) et leurs approximations sur une base ACPS de dix composantes (à droite) pour les variables de débit (en haut), de puissance (au milieu) et de température (en bas).

Synthèse des tests sur les trois cas d'application

Sur les trois cas-tests étudiés dans cette section, on a appliqué la première étape de la méthodologie de quantification des incertitudes de variables fonctionnelles proposée dans ce chapitre, consistant à décomposer ces variables sur une base fonctionnelle. Les différentes méthodes proposées dans la section ont été comparées. Dans l'exemple analytique et le cas du PTS, la décomposition SPLS a été retenue car elle permet de décomposer simultanément les différentes variables fonctionnelles et qu'elle présente un bon compromis entre l'approximation des variables et de leur lien avec la covariable. Dans le cas de la rutpure LiPoSo où on ne considère pas de covariable, la décomposition ACPS est préférée. De plus, plusieurs remarques peuvent être faites sur les résultats obtenus dans ces trois cas d'application :

- les décompositions simultanées permettent, à nombre total de composantes de la décomposition égal, d'obtenir une meilleure approximation des variables fonctionnelles que les décompositions ACP ou PLS simples sur chaque variable (cf. Figure 2.8),
- la normalisation de chaque variable fonctionnelle avant la décomposition ACPS permet de ne pas avantager l'approximation d'une variable par rapport à une autre pendant la décomposition,
- la première normalisation proposée (par le maximum de chaque variable) et la normalisation obtenue avec la méthode proposée par Perrin et al. (2013) donnent les meilleurs résultats.

Dans la section suivante, on traite la deuxième partie de la méthodologie proposée, à savoir l'estimation de la densité de probabilité des coefficients issus de la décomposition.

2.3 Estimation de densité de probabilité

Dans la section 2.2, les variables aléatoires fonctionnelles étudiées ont été décomposées simultanément sur une base fonctionnelle réduite. Ces variables sont alors représentées par les coefficients de la décomposition correspondant aux composantes sélectionnées. Ces coefficients sont des variables aléatoires caractérisées par une densité de probabilité multivariée. L'objectif de cette section est de proposer une méthode d'estimation de cette densité.

Trois catégories de méthodes d'estimation pour des variables vectorielles sont présentées respectivement dans les sections 2.3.1, 2.3.2 et 2.3.3 : l'estimation de densité à noyau, la décomposition par chaos polynomial et la modélisation par mélange de gaussiennes. De plus, une extension de cette dernière méthode est proposée dans la section 2.3.4 afin d'estimer des mélanges de gaussiennes avec des matrices de covariance creuses. Plusieurs critères sont introduits dans la section 2.3.5 afin d'évaluer et de comparer la qualité des différentes méthodes d'estimation. Enfin, dans la section 2.3.6, les différentes méthodes proposées sont appliquées aux trois cas d'application étudiés dans la section précédente.

On reprend les notations de la section 2.2 et on considère que d composantes de la décomposition ont été retenues et que l'on dispose d'un échantillon de n réalisations des coefficients associés. Cet échantillon est noté $\gamma^s = (\boldsymbol{\gamma}_i)_{1 \leqslant i \leqslant n}, n \in \mathbb{N}$, avec $\boldsymbol{\gamma}_i = (\gamma_{i,1}, \ldots, \gamma_{i,d}) \in \mathcal{G} \subset \mathbb{R}^d$. A chaque vecteur de coefficients $\boldsymbol{\gamma}_i$, $i \in \{1, \ldots, n\}$, correspond une réalisation des m variables fonctionnelles : $(z_{1,i}, \ldots, z_{m,i})$. On note $\Gamma = (\Gamma_1, \ldots, \Gamma_d)$ la variable aléatoire vectorielle dont les coefficients sont des réalisations. La densité de probabilité de Γ est notée $g : \mathcal{G} \to \mathbb{R}$.

2.3.1 Estimation de densité de probabilité à noyau

La méthode d'estimation à noyau présentée dans cette section est un modèle non-paramétrique d'estimation de la densité de probabilité d'une variable aléatoire à partir d'un échantillon de points.

Définition 2.7 (Estimateur à noyau)**.** *Soit H une matrice de taille $d \times d$ dont les éléments sont positifs, et appelée largeur de bande. Soit $K_H : \mathbb{R}^d \to \mathbb{R}$ un noyau intégrable tel que $\int K_H(\mathbf{x}) \mathrm{d}\mathbf{x} = 1$. L'estimateur à noyau de la fonction de densité g, proposé par Rosenblatt (1956) et Parzen (1962), est défini pour tout*

$\boldsymbol{\gamma} \in \mathcal{G}$ *comme suit :*

$$\hat{g}(\boldsymbol{\gamma}) = \frac{1}{n} \sum_{i=1}^{n} K_H \left(\boldsymbol{\gamma}_i - \boldsymbol{\gamma} \right).$$

Cette méthode d'estimation requiert donc le choix de deux paramètres : la forme du noyau K_H et la largeur de bande H. Le noyau d'Epanechnikov (Epanechnikov, 1969) minimise l'erreur quadratique moyennée intégrée asymptotique (AMISE pour *Asymptotic Mean Integrated Squared Error*). Cependant, Wand et Jones (1995) montre que l'AMISE d'autres noyaux comme les noyaux triangulaire, gaussien ou uniforme est en général très proche de celle du noyau optimal. Ainsi, le choix du noyau a peu d'influence sur l'estimation (Wand et Jones, 1995). Le choix de la largeur de bande est quant à lui plus important. De nombreuses méthodes ont été proposées pour l'estimer ; on peut citer entre autres Sheather et Jones (1991); Scott (1992); Wand et Jones (1995) ou encore Duong et Hazelton (2003).

De nombreuses variantes de l'estimation à noyau ont été développées. Parmi elles, on peut citer les méthodes adaptatives (Muller et Stadtmuller, 1987; Terrell et Scott, 1992), dans lesquelles la largeur de bande peut varier sur l'espace des entrées. L'utilisation de largeurs de bande variables permet de tenir compte de la variabilité de la densité de probabilité et améliore donc l'estimation dans de nombreuses situations.

Un des problèmes majeurs de cette méthode est dû au fléau de la dimension. La qualité de l'estimation de la largeur de bande se détériore rapidement quand la dimension d du problème augmente. Ainsi, l'estimation de densité à noyau n'est pas adaptée en dimension supérieure à 6 ou 7.

2.3.2 Décomposition par chaos polynomial de variables aléatoires

La décomposition par chaos polynomial, introduite par Wiener (1938), repose sur le fait que toute variable aléatoire de carré intégrable peut être écrite sous la forme d'un développement en polynômes orthogonaux de plusieurs variables aléatoires.

Définition 2.8 (Décomposition par chaos polynomial)**.** *Soit* $\{\eta_i\}_{i=1,\dots,\infty}$ *des variables aléatoires i.i.d. Soit* $\Psi = \{\psi_j^{i_1,\dots,i_r}, j, i_1, \dots, i_r \in \mathbb{N}, i_1 + \dots + i_r = j\}$ *une base infinie de polynômes orthonormaux selon ces variables aléatoires. La variable* Γ *de carré intégrable admet la décomposition par chaos polynomial suivante :*

$$\Gamma = \sum_{j \geq 0} \sum_{i_1 + \dots i_r = j} \sum_{\rho_1, \dots, \rho_r} \beta_{\rho_1,\dots,\rho_r}^{i_1,\dots,i_r} \psi_j^{i_1,\dots,i_r}(\eta_{\rho_1}, \dots, \eta_{\rho_r}),$$

où $\beta_{\rho_1,\dots,\rho_r}^{i_1,\dots,i_r} \in \mathbb{R}$ *est un coefficient, et* $\psi_j^{i_1,\dots,i_r}$ *est le polynôme d'ordre* j *de* r *variables aléatoires distinctes parmi l'ensemble* $\{\eta_i\}_{i=1,\dots,\infty}$ *et dans lequel la variable* η_{ρ_k} *a une multiplicité* i_k*.*

En pratique, le chaos polynomial de dimension infinie introduit dans la définition 2.8 et comportant une infinité de terme n'est pas utilisé directement. La décomposition est tronquée pour obtenir une décomposition de N variables aléatoires de polynômes d'ordre P.

Remarque 2.2. *L'approximation est alors complètement caractérisée par les coefficients de la décomposition. Une approximation de la densité de probabilité de* Γ *est alors disponible, puisque les densités de probabilité des variables* η_i *sont connues.*

Cameron et Martin (1947) montrent que la décomposition par chaos polynomial converge au sens L^2 vers la variable Γ quand P et N tendent vers l'infini. Dans les travaux de Wiener (1938), la décomposition par chaos polynomial a été introduite avec des variables aléatoires gaussiennes et une base polynomiale de Hermite. Cette décomposition a ensuite été généralisée au cas de variables distribuées selon la loi de Poisson par Ogura (1972) puis à d'autres distributions de probabilité par Xiu et Karniadakis (2002). Dans cette extension, appelée chaos polynomial généralisé, Xiu et Karniadakis (2002) ont utilisé la classification d'Askey des polynômes hypergéométriques (Askey et Wilson, 1985) pour associer à chaque type de variable aléatoire une base polynomiale orthonormale selon ces variables. Par exemple, les polynômes de Legendre sont orthonormaux pour des variables η_i uniformes, et sont donc utilisés dans la décomposition par chaos polynomial en association avec des variables η_i uniformes.

Pour estimer les coefficients de la décomposition, de nombreuses méthodes ont été proposées. On peut les classer en deux grandes catégories. La première repose sur la méthode de Galerkin et a été introduite par Ghanem et Spanos (1991) pour résoudre des équations aux dérivées partielles stochastiques. Ces méthodes sont dites intrusives car elles nécessitent une connaissance du code de calcul ou de l'équation dont la solution est Γ. Un des principaux défauts de ce type de méthodes est leur coût de calcul élevé dû à la résolution d'un système d'équations linéaires de grande dimension. La deuxième catégorie de méthodes, dites non intrusives, n'utilisent que des réalisations de la variable à modéliser. Parmi ces méthodes, on peut citer notamment la collocation stochastique (Xiu et Hesthaven, 2005; Babuška et al., 2010), l'approche par minimisation des moindres carrés développée par Berveiller (2005), les développements par chaos polynomiaux creux et adaptatifs (Blatman et Sudret, 2010), et ceux utilisant des approximations de faible rang (Chevreuil et al., 2013). Un état de l'art de ces méthodes peut être trouvé dans Xiu (2009).

2.3.3 Modélisation par un mélange de gaussiennes

2.3.3.1 Définition d'un mélange de gaussiennes

La troisième possibilité envisagée pour estimer la densité de probabilité des coefficients est de la modéliser par un modèle paramétrique. La densité $g(.|\theta)$ est alors caractérisée par des paramètres $\theta \in \Theta$. On choisit ici d'approcher la distribution de probabilité des coefficients par un mélange de lois normales.

Définition 2.9 (Mélange de gaussiennes)**.** *En dimension d, la densité g d'un mélange de G gaussiennes de densités $h_1, \ldots, h_G$ s'écrit comme la somme pondérée de densités de chaque distribution gaussienne :*

$$g(\boldsymbol{\gamma}|\theta_1, \ldots, \theta_G) = \sum_{k=1}^{G} \tau_k h_k(\boldsymbol{\gamma}|\theta_k), \ \forall \boldsymbol{\gamma} \in \mathbb{R}^p,$$

où θ_k sont les paramètres de la densité h_k et les τ_k sont les proportions du mélange, telles que $\sum_{k=1}^{G} \tau_k = 1$. Pour une densité gaussienne, $\theta_k = \{\mu_k, \Sigma_k\}$, où μ_k et Σ_k sont respectivement la moyenne et la matrice de covariance de la k-ième gaussienne. Les densités h_k s'écrivent

$$h_k(\boldsymbol{\gamma}|\mu_k, \Sigma_k) = \frac{1}{\sqrt{\det(2\pi\Sigma_k)}} \exp\left(-\frac{(\boldsymbol{\gamma} - \mu_k)^T \Sigma_k^{-1} (\boldsymbol{\gamma} - \mu_k)}{2}\right).$$

Modéliser la densité de probabilité par un mélange de gaussiennes revient donc à estimer les paramètres θ_k et τ_k pour $k = 1, \ldots, n$. Pour cela, une méthode d'estimation basée sur l'algorithme *Expectation-Maximization* peut être utilisée.

2.3.3.2 Algorithme Expectation-Maximization

Introduit par Dempster et al. (1977), l'algorithme *Expectation-Maximization* (EM) est une méthode d'estimation de paramètres par maximum de vraisemblance. Elle peut être utilisée notamment dans le cas où les données disponibles seules ne permettent pas d'estimer les paramètres ou quand la vraisemblance est impossible à optimiser analytiquement. Le principe de l'algorithme EM est d'introduire des données supplémentaires ou « manquantes », dépendantes du cas étudié. Ces données combinées aux données disponibles permettent de maximiser plus facilement la vraisemblance. La combinaison des données supplémentaires et disponibles est notée $x^s \subset \mathcal{X}$, et la variable aléatoire associée est notée X. Ces données sont dites « complètes » parce que leur connaissance permet de déterminer les données γ^s, alors que la connaissance de ces dernières ne permet pas de remonter aux données x^s. Les données complètes sont liées aux données γ^s par $\boldsymbol{\gamma} = h(x), \ \forall x \in \mathcal{X}$, où $h : \mathcal{X} \to \mathcal{G}$ est surjective. Seules les données γ^s sont observées, on n'a donc pas accès à x^s directement. La densité de probabilité de X, notée $f(.|\theta)$, est liée à $g(.|\theta)$ par l'expression :

$$g(\boldsymbol{\gamma}|\theta) = \int_{\mathcal{X}(\boldsymbol{\gamma})} f(x|\theta) dx,$$

où $\mathcal{X}(\boldsymbol{\gamma}) = \{x : h(x) = \boldsymbol{\gamma}\}$. Ainsi, la densité de Γ en un point $\boldsymbol{\gamma}$ est égale à la densité de X intégrée sur tous les points de $\mathcal{X}$ antécédents de $\boldsymbol{\gamma}$ par h. La nature de X dépend du cas d'application de l'algorithme, puisque les données observées γ^s dépendent des données X. L'algorithme EM repose sur le fait que l'introduction de ce niveau supplémentaire dans le modèle permet, dans de nombreux cas pratiques, de calculer plus simplement le maximum de vraisemblance.

On définit k la densité conditionnelle de x sachant $\boldsymbol{\gamma}$ et θ :

$$k(x|\boldsymbol{\gamma}, \theta) = \frac{f(x|\theta)}{g(\boldsymbol{\gamma}|\theta)}.$$

En notant $\ell(\Gamma|\theta) = \log g(\Gamma|\theta)$ la log-vraisemblance, on obtient

$$\ell(\Gamma|\theta) = \log g(\Gamma|\theta) = \log f(X|\theta) - \log k(X|\Gamma, \theta).$$

Puis en prenant l'espérance conditionnellement à Γ et θ, on a :

$$\ell(\Gamma|\theta) = Q(\theta, \theta') - H(\theta, \theta'), \tag{2.14}$$

où $Q(\theta, \theta') = E(\log f(X|\theta)|\Gamma, \theta')$ et $H(\theta, \theta') = E(\log k(X|\Gamma, \theta)|\Gamma, \theta')$.

L'algorithme EM se décompose en deux étapes : *expectation step* et *maximization step*. Ces étapes sont répétées jusqu'à convergence de l'algorithme, c'est-à-dire jusqu'à ce que la différence entre deux estimations successives des paramètres soit inférieure à un seuil fixé. A la j-ième étape, on dispose d'une estimation des paramètres, notée $\hat{\theta}^{(j)}$. Dans l'*expectation step*, $Q(\theta, \hat{\theta}^{(j)})$, l'espérance conditionnelle à $\hat{\theta}^{(j)}$ et Γ de la vraisemblance des données complètes, est calculée en fonction des paramètres θ. Le *maximization step* consiste à maximiser la fonction Q de θ estimée à l'étape précédente. Un nouvel estimateur de θ, $\hat{\theta}^{(j+1)}$, est ainsi obtenu : $\hat{\theta}^{(j+1)} = \text{argmax}_\theta(Q(\theta, \hat{\theta}^{(j)}))$. L'algorithme EM est résumé ci-dessous.

Algorithme 2.3 *Expectation-Maximization*

1. Initialiser les paramètres $\theta^{(0)}$.
2. *Expectation Step* : calculer $Q(\theta, \theta^{(j)})$ en fonction de θ.
3. *Maximization Step* : Trouver $\theta^{(j+1)} \leftarrow \text{argmax}_\theta(Q(\theta, \theta^{(j)}))$.
4. Répéter $2 - 3$ jusqu'à convergence.

L'algorithme EM ne maximise pas directement la log-vraisemblance ℓ des données observées mais seulement l'espérance conditionnelle Q de la log-vraisemblance des données complètes. Il faut donc montrer que la maximisation de Q est suffisante. Dempster et al. (1977) prouvent que, $\forall\, (\theta, \theta') \in \Theta^2$,

$$H(\theta, \theta') \leqslant H(\theta, \theta).$$

A l'étape j, comme $\theta^{(j+1)}$ maximise $\theta \mapsto Q(\theta, \theta^{(j)})$, on peut écrire

$$\ell(\Gamma|\theta^{(j+1)}) - \ell(\Gamma|\theta^{(j)}) = \left[Q(\theta^{(j+1)}, \theta^{(j)}) - Q(\theta^{(j)}, \theta^{(j)})\right] - \left[H(\theta^{(j+1)}, \theta^{(j)}) - H(\theta^{(j)}, \theta^{(j)})\right] \geqslant 0. \tag{2.15}$$

Cette inégalité montre que la valeur de la vraisemblance augmente bien à chaque pas de l'algorithme, et que maximiser Q permet bien de maximiser la log-vraisemblance. L'équation (2.15) montre aussi qu'une maximisation de $Q(., \theta^{(j)})$ à chaque étape n'est pas nécessaire. Il suffit de trouver des paramètres $\theta^{(j+1)}$ tels que $Q(\theta^{(j+1)}, \theta^{(j)}) > Q(\theta^{(j)}, \theta^{(j)})$. On peut donc remplacer dans l'algorithme EM le *maximisation step* par la recherche d'un jeu de paramètres qui vérifie cette inégalité. On appelle algorithme EM généralisé (GEM) une telle procédure. Celle-ci peut être utile dans les cas où l'étape de maximisation est coûteuse.

Plusieurs études ont été menées pour déterminer les propriétés théoriques de l'algorithme EM. On présente ici certains résultats de convergence de cette méthode, démontrés par Wu (1983). On introduit d'abord quelques notations. On définit la fonction M qui à un point associe un ensemble, tel que $M(\theta^{(j)}) = \text{argmax}_\theta Q(\theta, \theta^{(j)})$. On dit que M est fermée si elle vérifie la propriété suivante : si $x_k \to x$ et $\boldsymbol{\gamma}_k \to \gamma$, $\boldsymbol{\gamma}_k \in M(x_k)$ implique que $\boldsymbol{\gamma}_k \in M(x)$. De plus, on note $\mathcal{S}$ l'ensemble des points stationnaires de ℓ et M_L l'ensemble des maxima locaux de ℓ.

Proposition 2.10. *Soit une suite $(\theta^{(j)})_{j\in\mathbb{N}}$ avec $\theta^{(j+1)} \in M(\theta^{(j)})$. On suppose que M est fermée sur le complémentaire de $\mathcal{S}$ (respectivement M_L) et que $\ell(\theta^{(j+1)}) > \ell(\theta^{(j)})$, $\forall \theta^{(j)} \notin \mathcal{S}$ (resp. M_L). Alors, toutes les limites θ^* de $(\theta^{(j)})_{j\in\mathbb{N}}$ sont des points stationnaires (resp. des maxima locaux) de ℓ et $(\ell(\theta^{(j)}))_{j\in\mathbb{N}}$ converge de manière monotone vers $\ell(\theta^*)$.*

Comme la continuité de $(\theta, \theta') \mapsto Q(\theta, \theta')$ selon θ et θ' implique que M est fermée, Wu (1983) prouve le théorème suivant, plus utile en pratique puisque ses hypothèses sont plus faciles à vérifier.

Théorème 2.11 (Convergence de l'algorithme EM - Wu 1983)**.** *Si Q est continue selon ses deux variables, alors les limites θ^* d'un algorithme EM sont des points stationnaires de ℓ et $(\ell(\theta^{(j)}))_{j\in\mathbb{N}}$ converge de manière monotone vers $\ell(\theta^*)$.*

2.3.3.3 Estimation des paramètres du mélange de gaussiennes

Pour adapter l'algorithme EM au cas du mélange de gaussiennes, on introduit tout d'abord les données, dites « manquantes », z_{ik} qui définissent l'appartenance de $\boldsymbol{\gamma}_i$ au groupe k :

$$z_{ik} = \begin{cases} 1 & \text{si } \boldsymbol{\gamma}_i \text{ appartient au groupe } k \\ 0 & \text{sinon.} \end{cases}$$

On note Z_{ik} la variable aléatoire associée. A partir des données manquantes, les données complètes x sont définies par $x_i = (z_i, \boldsymbol{\gamma}_i)$. La log-vraisemblance des données complètes se décompose ainsi, en notant $\theta = (\tau_1, \ldots, \tau_G, \theta_1, \ldots, \theta_G)$:

$$\begin{aligned} \log f(x|\theta) &= \sum_{k=1}^{G}\sum_{i=1}^{n} z_{ik} \log \tau_k + \sum_{k=1}^{G}\sum_{i=1}^{n} z_{ik} \log f_k(\boldsymbol{\gamma}_i|\theta_k) \qquad (2.16) \\ &= \sum_{k=1}^{G}\sum_{i=1}^{n} z_{ik} \log \tau_k - \frac{1}{2} nG \log(2\pi) - \frac{1}{2}\sum_{k=1}^{G}\sum_{i=1}^{n} z_{ik} \left[\log\det(\Sigma_k) + (\boldsymbol{\gamma}_i - \mu_k)^T \Sigma_k^{-1} (\boldsymbol{\gamma}_i - \mu_k)\right]. \end{aligned}$$

On remarque que la log-vraisemblance est linéaire en z_{ik}, donc l'*expectation step* ne requiert que le calcul de l'espérance conditionnelle de Z_{ik}. Comme Z_{ik} suit une loi binomiale, l'espérance conditionnelle de $Z_{i,k}$ peut s'écrire ainsi

$$\begin{aligned} z_{ik}^{(j)} &= E(Z_{ik}|\boldsymbol{\gamma}_i, \theta_k^{(j)}) \\ &= P(Z_{ik} = 1|\boldsymbol{\gamma}_i, \theta_k^{(j)}) \\ &= \frac{\tau_k^{(j)} f_k(\boldsymbol{\gamma}_i|\theta_k^{(j)})}{\sum_{g=1}^{G} \tau_g^{(j)} f_g(\boldsymbol{\gamma}_i|\theta_g^{(j)})}. \qquad (2.17) \end{aligned}$$

Le degré d'appartenance de $\boldsymbol{\gamma}_i$ au groupe k dépend donc à la fois de la valeur de la densité de $\boldsymbol{\gamma}_i$ dans le groupe k et de la proportion du groupe dans le mélange. En effet, plus la densité $f_k(\boldsymbol{\gamma}_i|\theta_k^{(j)})$ est grande, plus la probabilité que $\boldsymbol{\gamma}_i$ appartienne au groupe k est forte.

Pour le *maximization step*, il faut déterminer les paramètres θ qui maximisent Q, c'est-à-dire qui maximisent l'équation (2.16) dans laquelle $z_{ik}^{(j)}$ remplace z_{ik}. On dérive la fonction Q en fonction des composantes de $\mu_k^{(j+1)}$ et on cherche à annuler cette dérivée :

$$\frac{\partial Q}{\partial \mu_k^{(j+1)}} = 0 \Rightarrow \left(\Sigma_k^{(j+1)}\right)^{-1} \sum_{i=1}^{n} z_{ik}^{(j)} (\boldsymbol{\gamma}_i - \mu_k^{(j+1)}) = 0.$$

Après simplification, on obtient les valeurs des composantes de $\mu_k^{(j+1)}$:

$$\mu_k^{(j+1)} = \frac{\sum_{i=1}^n z_{ik}^{(j)} \boldsymbol{\gamma}_i}{\sum_{i=1}^n z_{ik}^{(j)}}. \tag{2.18}$$

La moyenne $\mu_k^{(j+1)}$ de chaque groupe k à l'étape j est donc égale à la moyenne de tous les points de γ^s pondérés par leur appartenance au groupe. Ainsi, plus un point est susceptible d'appartenir au groupe, plus il a un poids important dans la moyenne.

Pour la détermination des coefficients des $\Sigma_k^{(j+1)}$, on dérive Q selon $\Sigma_k^{(j+1)}$:

$$\begin{aligned}
\frac{\partial Q}{\partial \Sigma_k^{(j+1)}} = 0 \Rightarrow -\frac{1}{2}\sum_{i=1}^n z_{ik}^{(j)} \left[\frac{\partial \log\det(\Sigma_k^{(j+1)})}{\partial \Sigma_k^{(j+1)}} - \frac{\partial \left((\boldsymbol{\gamma}_i - \mu_k^{(j+1)})^T (\Sigma_k^{(j+1)})^{-1} (\boldsymbol{\gamma}_i - \mu_k^{(j+1)}) \right)}{\partial \Sigma_k^{(j+1)}} \right] &= 0 \\
-\frac{1}{2}\sum_{i=1}^n z_{ik}^{(j)} \left[(\Sigma_k^{(j+1)})^{-1} - (\Sigma_k^{(j+1)})^{-1} (\boldsymbol{\gamma}_i - \mu_k^{(j+1)})(\boldsymbol{\gamma}_i - \mu_k^{(j+1)})^T (\Sigma_k^{(j+1)})^{-1} \right] &= 0.
\end{aligned}$$

On obtient finalement la relation suivante pour la matrice de covariance :

$$\Sigma_k^{(j+1)} = \frac{1}{\sum_{i=1}^n z_{ik}^{(j)}} \sum_{i=1}^n z_{ik}^{(j)} (\boldsymbol{\gamma}_i - \mu_k^{(j+1)})(\boldsymbol{\gamma}_i - \mu_k^{(j+1)})^T. \tag{2.19}$$

De même que dans l'équation (2.18), plus un point $\boldsymbol{\gamma}_i$ a une forte probabilité d'appartenir au groupe, plus il est important dans le calcul de la covariance.

Enfin, par définition, la proportion τ_k du groupe k dans le mélange est égale à la moyenne des probabilités d'appartenance de chaque point de $\boldsymbol{\gamma}^s$ au groupe k : $\tau_k = \frac{1}{n}\sum_{i=1}^n z_{ik}$. En remplaçant z_{ik} par $z_{ik}^{(j)}$ dans l'équation précédente, on obtient l'expression des proportions à l'étape j :

$$\tau_k^{(j+1)} = \frac{1}{n}\sum_{i=1}^n z_{ik}^{(j)}. \tag{2.20}$$

Les équations (2.18), (2.19) et (2.20) donnent l'expression des paramètres $\theta^{(j+1)}$, calculés lors du *maximization step*. L'algorithme 2.4 résume l'application de la méthode EM à l'estimation des paramètres d'un mélange de gaussiennes.

Algorithme 2.4 *Expectation-Maximization*

1. Initialiser les paramètres $\tau_k^{(0)}$, $\mu_k^{(0)}$ et $\Sigma_k^{(0)}$, $\forall k \in \{1, \ldots, G\}$.
2. *Expectation Step* : Calculer $z_{ik}^{(j)}$, $\forall k \in \{1, \ldots, G\}$, $\forall i \in \{1, \ldots, n\}$, par l'équation (2.17).
3. *Maximization Step* : Calculer $\tau_k^{(j+1)}$, $\mu_k^{(j+1)}$ et $\Sigma_k^{(j+1)}$, $\forall k \in \{1, \ldots, G\}$ avec les équations (2.20), (2.18) et (2.19) respectivement.
4. Répéter $2-3$ jusqu'à convergence.

Pour une description plus détaillée de l'algorithme EM et de son application à l'estimation des paramètres d'un mélange de gaussiennes, les livres de Mclachlan et Krishnan (1997) et Gupta et Chen (2011) peuvent être consultés.

2.3.3.4 Choix du nombre de gaussiennes

Le nombre de gaussiennes (ou de groupes) dans le mélange de gaussiennes n'est pas estimé dans l'algorithme EM et doit être déterminé préalablement à partir des données. De nombreuses méthodes

ont été proposées pour sélectionner cette quantité. Une revue très complète de ces critères peut être trouvée dans Oliveira-Brochado et Martins (2005). Le critère présenté ici est basé sur une pénalisation de la log-vraisemblance. En effet, la log-vraisemblance seule ne peut pas être utilisée comme un critère de qualité puisqu'elle augmente quand le nombre de gaussiennes augmente, alors que l'estimation ne s'améliore pas forcément. En effet, pour une densité en dimension d avec G gaussiennes dans le mélange, le nombre total de paramètres à estimer s'écrit :

$$N = G - 1 + Gd + G\frac{d(d+1)}{2}, \tag{2.21}$$

car on doit estimer $G-1$ proportions τ_k, G moyennes $\mu_k \in \mathbb{R}^d$ et G matrices de covariance symétriques. Si le nombre de paramètres dans le mélange est trop élevé, l'efficacité de l'estimation peut diminuer et conduire à du surapprentissage. Le modèle est alors trop sensible aux données d'apprentissage. En effet, la taille des données d'apprentissage ne sera plus assez importante pour permettre d'estimer correctement tous les paramètres. Pénaliser la log-vraisemblance par une fonction du nombre de paramètres permet de trouver un compromis dans le choix du nombre de gaussiennes afin d'éviter le surapprentissage.

On utilise ici le *Bayesian Information Criteria* (BIC) , introduit par Schwarz (1978).

Définition 2.12 (*Bayesian Information Criterion*)**.** *Soit ℓ la log-vraisemblance du modèle, N le nombre de paramètres et n la taille de l'échantillon. Le critère BIC est défini comme suit*

$$\text{BIC} = -2\ell + N \ln n, \tag{2.22}$$

Plus le critère BIC est élevé, meilleur est le modèle. Dans la pratique, des modèles de mélanges de gaussiennes sont estimés pour différents nombres de gaussiennes et le BIC est estimé pour chacun d'eux. Le nombre de gaussiennes pour lequel le BIC est maximal est ensuite retenu.

2.3.4 Amélioration du mélange de gaussiennes par matrices de covariance creuses

Le nombre de paramètres dans le mélange de gaussiennes augmente avec le carré de la dimension (cf. équation (2.21)). Dans le but de limiter le surapprentissage, il peut être intéressant de diminuer le nombre de paramètres en modifiant la forme des matrices de covariance du mélange de gaussiennes. Une première idée serait de considérer que les gaussiennes du mélange ont toutes des covariances diagonales. Le nombre de paramètres est alors réduit à $N = G - 1 + 2Gd$. Cette hypothèse supplémentaire peut être dans certains cas trop forte. Une méthode moins contraignante consiste à considérer que les matrices de covariance du modèle sont creuses. Deux méthodes pour estimer des matrices de covariance creuses sont étudiées dans cette section.

2.3.4.1 Pénalisation sur l'inverse de la matrice de covariance

Krishnamurthy (2012) a proposé un algorithme EM modifié pour estimer de telles matrices de covariance. Dans celui-ci, seul le calcul de la covariance est changé. Le principe de l'algorithme est de maximiser la log-vraisemblance pénalisée par la norme L^1 de l'inverse de la covariance. En effet, la pénalisation L^1 ou Lasso permet d'imposer aux éléments les plus faibles de la matrice de s'annuler. La maximisation pénalisée de la log-vraisemblance s'écrit ainsi :

$$\begin{aligned}
\hat{\Sigma}_k &= \text{argmax}_{\Sigma_k}\left[-\sum_{i=1}^{n} z_{ik}\left(\log\det(\Sigma_k) + (\boldsymbol{\gamma}_i - \mu_k)^T\Sigma_k^{-1}(\boldsymbol{\gamma}_i - \mu_k) + \lambda\|\Sigma_k^{-1}\|_1\right)\right] \\
\hat{\Sigma}_k^{-1} &= \text{argmax}_{C_k}\left[-\sum_{i=1}^{n} z_{ik}\left(-\log\det(C_k) + (\boldsymbol{\gamma}_i - \mu_k)^T C_k(\boldsymbol{\gamma}_i - \mu_k) + \lambda\|C_k\|_1\right)\right],
\end{aligned} \tag{2.23}$$

où Σ_k est la matrice de covariance de la k-ième gaussienne, $\hat{\Sigma}_k$ est la matrice de covariance creuse recherchée, $C_k = \Sigma_k^{-1}$ et $\|M\|_1 = \sum_{i,j=1}^{p} |M_{i,j}|$. On peut réécrire l'équation (2.23) en divisant par $\sum_{i=1}^{n} z_{ik}$:

$$\begin{aligned} \hat{\Sigma}_k^{-1} &= \text{argmax}_{C_k} \left[\log\det(C_k) - \frac{\sum_{i=1}^{n} z_{ik} (\boldsymbol{\gamma}_i - \mu_k)^T C_k (\boldsymbol{\gamma}_i - \mu_k)}{\sum_{i=1}^{n} z_{ik}} - \lambda \|C_k\|_1 \right] \\ \hat{\Sigma}_k^{-1} &= \text{argmax}_{C_k} \left[\log\det(C_k) - \text{tr}(C_k \Sigma_k) - \lambda \|C_k\|_1 \right]. \end{aligned} \tag{2.24}$$

En effet, $a^T C a = \text{tr}(C a a^T)$, pour $C \in \mathbb{R}^{d \times d}$ et $a \in \mathbb{R}^{d \times 1}$.

Plusieurs méthodes ont été développées pour résoudre l'équation (2.24), c'est-à-dire estimer des inverses de matrices de covariance creuses. L'une des plus efficaces est l'algorithme du *graphical Lasso* développé par Friedman et al. (2008). L'algorithme d'estimation de mélange de gaussiennes à covariances creuses, proposé par Krishnamurthy (2012), est détaillé ci-dessous. On note cet algorithme sEM dans la suite.

Algorithme 2.5 sEM (Krishnamurthy, 2012)

1. Initialiser les paramètres $\tau_k^{(0)}$, $\mu_k^{(0)}$ et $\Sigma_k^{(0)}$, $\forall k \in \{1, \ldots, G\}$.
2. *Expectation Step* : Calculer $z_{ik}^{(j)}$, $\forall k \in \{1, \ldots, G\}$, $\forall i \in \{1, \ldots, n\}$, par l'équation (2.17).
3. *Maximization Step* : Calculer $\tau_k^{(j+1)}$, $\mu_k^{(j+1)}$ et $\Sigma_k^{(j+1)}$, $\forall k \in \{1, \ldots, G\}$ avec les équations (2.20), (2.18) et (2.19) respectivement.
4. $\Sigma_k^{(j+1)} \leftarrow \left(\text{argmax}_{C_k} \log\det C_k - \text{tr}(C_k \Sigma_k^{(j+1)}) - \lambda \|C_k\|_1 \right)^{-1}$.
5. Répéter $2-4$ jusqu'à convergence.

Notons qu'un paramètre λ est introduit dans la pénalisation. Krishnamurthy propose de choisir ce paramètre par validation croisée. Les paramètres sont estimés sur une partie des données à λ fixé et la vraisemblance du modèle est calculée sur le reste des données. Le paramètre λ qui maximise cette vraisemblance est retenu.

Le défaut de la méthode présentée par Krishnamurthy (2012) est qu'elle impose à l'inverse de la matrice de covariance d'être creuse mais non à la matrice de covariance. Cependant, si une matrice est creuse, son inverse n'a pas forcément cette propriété. Ainsi, même en diminuant le nombre d'éléments non nuls dans l'inverse de la matrice, la méthode ne garantie pas que le nombre d'éléments non-nuls de la matrice de covariance et donc de paramètres du mélange soit diminué.

2.3.4.2 Pénalisation sur la matrice de covariance

Pour pallier le problème soulevé dans la section précédente, Bien et Tibshirani (2011) cherchent à pénaliser la log-vraisemblance par la norme L^1 de la matrice de covariance et non par son inverse.

Pour un échantillon $\mathbf{x}_1, \ldots, \mathbf{x}_n$ de réalisations d'une variable aléatoire gaussienne multivariée X de moyenne 0 et de matrice de covariance Σ, la log-vraisemblance s'écrit

$$\ell(\Sigma) = -\frac{nd}{2} \log 2\pi - \frac{n}{2} \log\det \Sigma - \frac{n}{2} \text{tr}(\Sigma^{-1} S),$$

avec $S = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T$ la covariance empirique des données. Bien et Tibshirani (2011) introduisent une pénalisation du type lasso et se ramènent au problème d'optimisation suivant :

$$\text{argmin}_{\Sigma} \left[\log\det \Sigma + \text{tr}(\Sigma^{-1} S) + \lambda \|P * \Sigma\|_1 \right], \tag{2.25}$$

où P est une matrice à définir, le paramètre de pénalisation λ est un réel positif, $*$ désigne la multiplication terme à terme de deux matrices et $\|A\|_1 = \sum_{i,j} |A_{i,j}|$. La matrice P, par exemple, peut être choisie telle

que $\forall i, j \in \{1, \ldots, n\}$,

$$P_{ij} = 1,\ P_{ij} = 1 - \delta_{ij} \text{ ou } P_{ij} = \frac{1 - \delta_{ij}}{|S_{ij}|}.$$

La résolution de (2.25) n'est pas aisée puisque la fonction à optimiser, $f : \Sigma \mapsto \log\det\Sigma + \mathrm{tr}(\Sigma^{-1}S) + \lambda \|P * \Sigma\|_1$, n'est pas convexe. Cependant, Bien et Tibshirani (2011) ont prouvé que ce problème se ramène à un problème classique d'optimisation puisque la fonction f est la somme d'une fonction convexe $\Sigma \mapsto \mathrm{tr}(\Sigma^{-1}S) + \lambda \|P * \Sigma\|_1$ et d'une fonction concave $\Sigma \mapsto \log\det\Sigma$. Il existe plusieurs techniques pour résoudre un tel problème d'optimisation parmi lesquelles l'algorithme de *majorization-minimization* utilisé par Bien et Tibshirani (2011). Wang (2014), quant à lui, propose un nouvel algorithme basé sur la descente par coordonnées pour résoudre le problème (2.25). Ce dernier est plus rapide et semble plus stable d'après les tests présentés dans Wang (2014). Les deux algorithmes de Bien et Tibshirani (2011) et Wang (2014) sont détaillés dans l'annexe A.

Il est possible d'utiliser les algorithmes de Bien et Tibshirani (2011) et de Wang (2014) dans le cadre de l'estimation des paramètres d'un mélange de gaussiennes. En pénalisant les problèmes de maximisation du *Maximization Step*, on obtient pour chaque classe $k \in \{1, \ldots, G\}$ le problème suivant :

$$\hat{\Sigma}_k = \mathrm{argmax}_S \left[-\sum_{i=1}^{n} z_{ik} \left(\log\det(S) + (\boldsymbol{\gamma}_i - \mu_k)^T S^{-1} (\boldsymbol{\gamma}_i - \mu_k) \right) + \lambda \|P * S\|_1 \right] \tag{2.26}$$

où Σ_k est la matrice de covariance de la k-ième gaussienne, $\hat{\Sigma}_k$ est la matrice de covariance creuse recherchée, et $\|M\|_1 = \sum_{i,j=1}^{p} |M_{i,j}|$. On peut réécrire l'équation (2.26) en divisant par $\sum_{i=1}^{n} z_{ik}$:

$$\begin{aligned} \hat{\Sigma}_k &= \mathrm{argmin}_S \left[\log\det(S) + \frac{\sum_{i=1}^{n} z_{ik} (\boldsymbol{\gamma}_i - \mu_k)^T S^{-1} (\boldsymbol{\gamma}_i - \mu_k)}{\sum_{i=1}^{n} z_{ik}} + \lambda \|P * S\|_1 \right] \\ \hat{\Sigma}_k &= \mathrm{argmin}_S \left[\log\det(S) + \mathrm{tr}(S^{-1}\Sigma_k) + \lambda \|P * S\|_1 \right]. \end{aligned} \tag{2.27}$$

On propose donc de modifier l'algorithme EM en résolvant à chaque *Maximization Step* le problème de maximisation pénalisée de l'équation (2.27) pour déterminer les matrices de covariance. Ainsi, en pratique, à chaque *Maximization Step*, les G matrices de covariance sont estimées comme dans l'algorithme EM classique par l'équation (2.19). Les valeurs trouvées servent ensuite d'initialisation à l'algorithme de Wang (2014) à l'aide duquel elles sont ré-estimées. On nomme cette méthode sEM2 par la suite. Son algorithme complet est résumé ci-dessous.

Algorithme 2.6 sEM2

1. Initialiser les paramètres $\tau_k^{(0)}$, $\mu_k^{(0)}$ et $\Sigma_k^{(0)}$, $\forall k \in \{1, \ldots, G\}$.
2. *Expectation Step* : Calculer $z_{ik}^{(j)}$, $\forall k \in \{1, \ldots, G\}$, $\forall i \in \{1, \ldots, n\}$, par l'équation (2.17).
3. *Maximization Step* : Calculer $\tau_k^{(j+1)}$, $\mu_k^{(j+1)}$ et $\Sigma_k^{(j+1)}$, $\forall k \in \{1, \ldots, G\}$ avec les équations (2.20), (2.18) et (2.19) respectivement.
4. $\Sigma_k^{(j+1)} \leftarrow \mathrm{argmin}_S \left[\log\det S + \mathrm{tr}(S^{-1}\Sigma_k^{(j+1)}) + \lambda \|P * S\|_1 \right]$.
5. Répéter $2 - 4$ jusqu'à convergence.

Le choix du paramètre de pénalisation λ est important et peut avoir une grande influence sur les résultats obtenus. Comme dans l'article de Krishnamurthy (2012), il est estimé par validation croisée.

2.3.5 Critères proposés pour évaluer la méthodologie

Dans les deux sections précédentes, l'algorithme EM pour l'estimation des paramètres d'un mélange de gaussiennes a été présenté, puis deux améliorations de cette méthode, utilisant des matrices de covariance creuses ont été proposées. Pour évaluer la qualité de ces différents algorithmes et les comparer, trois critères ont été développés. Ces critères remplissent deux objectifs distincts :

– évaluer la qualité des méthodes de modélisation de la densité de probabilité par un mélange de gaussiennes et de la méthodologie globale de quantification des incertitudes de variables aléatoires fonctionnelles ;
– régler certains paramètres de la méthodologie de quantification des incertitudes. En effet, ils peuvent aider à sélectionner la taille d de la base de décomposition fonctionnelle ou le type de pénalisation utilisé dans l'estimation des paramètres du mélange de gaussiennes.

Pour atteindre ces objectifs, on propose trois critères :

Critère 1 basé sur la comparaison des échantillons observés et simulés. A partir de la distribution de probabilité des coefficients estimée, de nouveaux échantillons de coefficients peuvent être simulés. Leur distribution jointe est comparée à celle d'un échantillon de test composé de coefficients obtenus à partir d'observations des variables fonctionnelles à l'aide d'un test d'adéquation multivarié. Le résultat de la comparaison dépend du test multivarié utilisé. Le test développé par Fromont et al. (2012) a été préféré à d'autres tests d'adéquation multivariés (celui de Baringhaus et Franz (2004), par exemple) parce qu'il est exactement de niveau α et non uniquement asymptotiquement de niveau α. Ce test d'adéquation à noyau est décrit plus précisément dans l'annexe B. Le test est appliqué à plusieurs échantillons de coefficients simulés. **Le premier critère proposé, noté C_1^e, est le taux moyen d'acceptation de ces tests.** Plus le critère est élevé, plus la distribution estimée est proche de la distribution à modéliser.

Critère 2 basé sur les corrélations temps par temps. Le deuxième critère proposé a pour but d'évaluer la capacité de la méthodologie à reproduire les corrélations entre les variables fonctionnelles modélisées. Les corrélations étudiées ici sont les corrélations temps par temps calculées en chaque point de la discrétisation $t_1, \dots t_p$. Les $\frac{m(m-1)}{2}$ corrélations temps par temps entre les variables Z_i et Z_j, pour $i, j \in \{1, \dots, m\}$ et $i \neq j$, sont définies comme suit :

$$c_k^{i,j} = \mathrm{Corr}\left(Z_i(t_k), Z_j(t_k)\right), \forall k \in \{1, \dots, p\}. \tag{2.28}$$

Les corrélations calculées sur la base de test sont comparées aux corrélations calculées sur un échantillon de réalisations simulées à l'aide de la densité de probabilité estimée (simulation de la densité des coefficients et reconstruction avec la base fonctionnelle). **L'erreur quadratique moyenne entre les deux vecteurs de corrélations est utilisée comme critère et est notée C_2^e.** Plus précisément, cette erreur est définie de la manière suivante, pour $i, j \in \{1, \dots, m\}$:

$$C_2^e = \frac{1}{p} \sum_{k=1}^{p} \left(c_k^{i,j} - \hat{c}_k^{i,j}\right)^2, \tag{2.29}$$

où $c^{i,j}$ est le vecteur de corrélations calculé sur la base de test et $\hat{c}^{i,j}$ est celui calculé sur l'échantillon simulé.

Critère 3 basé sur la comparaison des covariables observées et simulées. Enfin, dans le cas où il existe une covariable, la capacité de la méthodologie à reproduire la variabilité de la covariable peut aussi être testée. De la même manière que pour le critère C_1^e, un test d'adéquation est utilisé pour comparer la distribution de la covariable observée à celle simulée en utilisant la méthodologie de quantification. L'échantillon de test de réalisations de la covariable est obtenu en appliquant le code $\mathcal{M}$ à l'échantillon de test. De même, un échantillon de réalisations de la covariable approchée est obtenu en appliquant $\mathcal{M}$ aux réalisations simulées à l'aide de la méthodologie de quantification des incertitudes des variables fonctionnelles. Ensuite, un test d'adéquation univarié comme les tests de Kolmogorov-Smirnov ou Anderson-Darling (Conover, 1971) est appliqué entre les deux échantillons. **Le troisième critère C_3^e est alors défini comme le taux d' acceptation moyen calculé en appliquant cette méthode sur plusieurs échantillons de réalisations de la covariable.**

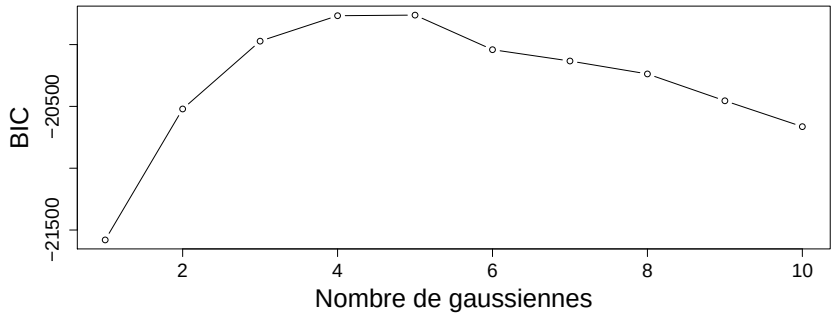

FIGURE 2.22 – Cas analytique : valeurs du critère BIC en fonction du nombre de gaussiennes pour le cas analytique avec $d = 6$.

2.3.6 Application des méthodes d'estimation de densités de probabilité

Dans cette section, les trois cas d'application étudiés dans la section 2.2 sont traités. Pour chacun d'eux, la densité de probabilité jointe des coefficients de la décomposition fonctionnelle obtenue à l'issue de la section 2.2 est modélisée par un mélange de gaussiennes. Les différents modèles étudiés et proposés dans les sections 2.3.3 et 2.3.4 sont testés :

- algorithme EM estimant des matrices de covariance pleines (section 2.3.3.3),
- algorithme sEM estimant des matrices de covariance creuses en utilisant la pénalisation sur l'inverse des matrices de covariance (section 2.3.4),
- algorithme sEM2 estimant des matrices de covariance creuses en utilisant la pénalisation sur les matrices de covariance (section 2.3.4.2). Trois types de pénalisation sont considérées pour cet algorithme :
  - Pénalisation 1 : $P_{ij} = 1$ (algorithme sEM2.1),
  - Pénalisation 2 : $P_{ij} = 1 - \delta_{ij}$ (algorithme sEM2.2),
  - Pénalisation 3 : $P_{ij} = \dfrac{1 - \delta_{ij}}{|S_{ij}|}$ avec S_{ij} la matrice de covariance empirique (algorithme sEM2.3).

2.3.6.1 Application à un modèle analytique

Rappels sur l'exemple analytique

On reprend le modèle analytique défini dans la section 2.2.7.1. Les deux variables fonctionnelles considérées sont les processus gaussiens corrélés Z_1 et Z_2. Ces deux processus sont liés *via* un code $\mathcal{M}$ à une covariable Y dépendant aussi de variables uniformes X_1, X_2 et X_3. Un échantillon de 200 réalisations de Z_1, Z_2, X_1, X_2 et X_3 ainsi que les valeurs de Y correspondantes est disponible.

A l'issue de l'étude réalisée dans la section 2.2.7.1, la décomposition SPLS a été retenue pour décomposer ces deux variables, car elle présentait un bon compromis entre l'approximation des variables fonctionnelles et de leur lien avec la covariable fonctionnelle. La taille d de la décomposition n'a pas été choisie à l'issue de l'étape de décomposition fonctionnelle.

On cherche maintenant à estimer la densité des d coefficients de la décomposition et également à sélectionner un nombre d de fonctions de la base de décomposition à retenir. Tout d'abord, pour une taille de base d fixée, le nombre de gaussiennes dans le mélange est choisi par maximisation du critère BIC défini dans la section 2.3.3.4. Un exemple des valeurs du BIC dans le cas $d = 6$ est représenté sur la figure 2.22. Dans ce cas, le BIC est maximal pour un mélange de $G = 5$ gaussiennes.

Les méthodes d'estimation de paramètres sEM et sEM2. nécessitent l'estimation d'un paramètre de pénalisation λ qui est déterminé par validation croisée. Les données ont été séparées en quatre groupes.

Méthodes	**sEM**	**sEM2.1**	**sEM2.2**	**sEM2.3**
Paramètre de pénalisation	0.01	0.005	0.002	0.01
Pourcentage d'éléments nuls	0%	42 %	35%	33%

TABLE 2.1 – Cas analytique : paramètres de pénalisation estimés (deuxième ligne) et pourcentages d'éléments nuls dans les matrices de covariance estimées (troisième ligne), pour les méthodes sEM, sEM2.1, sEM2.2 et sEM2.3, avec une base de décomposition de taille $d = 6$.

La méthode sEM (resp. sEM2.) est appliquée aux données de trois des quatre groupes à λ fixé puis la vraisemblance du modèle estimé par sEM (resp. sEM2.) est calculée sur les données du quatrième groupe non utilisé pour l'apprentissage du mélange de gaussiennes. Ce calcul est ensuite répété pour les quatre groupes et la moyenne des quatre vraisemblances obtenues est calculée. Cette procédure est ensuite répétée pour plusieurs valeurs de λ. Le paramètre de pénalisation qui maximise la vraisemblance calculée est retenu. La deuxième ligne du tableau 2.1 donne les paramètres λ ainsi retenus pour les méthodes sEM, sEM2.1, sEM2.2 et sEM2.3 pour $d = 6$. Figure aussi dans ce tableau le pourcentage d'éléments nuls dans les matrices de covariance obtenues avec les paramètres de pénalisation retenus. Le modèle de mélange de gaussiennes correspondant possède $N = 139$ paramètres. L'estimation par validation croisée du paramètre de pénalisation de la méthode sEM a conduit à un modèle où aucun élément de la matrice de covariance n'est nul, bien que le paramètre de pénalisation ne soit pas nul. Cela illustre la difficulté de la calibration du paramètre de pénalisation de la méthode sEM.

Pour sélectionner le nombre de composantes à retenir dans la décomposition fonctionnelle ainsi que pour évaluer la qualité de la méthodologie, les critères définis dans la sections 2.3.5 sont calculés pour plusieurs tailles de base avec les cinq méthodes d'estimation.

Le premier critère, C_1^e, quantifie la qualité de la modélisation de la distribution de probabilité des coefficients de la décomposition. Le taux d'acceptation du test d'adéquation multivarié de Fromont et al. (2012) est calculé ici pour plusieurs échantillons d'apprentissage et de test différents. Pour une paire d'échantillons de coefficients de test et d'apprentissage, la densité de probabilité des coefficients est modélisée à l'aide de l'échantillon d'apprentissage, puis plusieurs échantillons sont générés selon la distribution estimée. Ensuite, le test d'adéquation est appliqué entre chaque échantillon généré et la base de test. Le taux d'acceptation du test est calculé à partir des résultats de ces tests sur les différents échantillons générés. Ainsi, un taux d'acceptation est calculé par couple d'échantillons d'apprentissage et de test. Le niveau du test utilisé est $\alpha = 0,05$. Comme les variables fonctionnelles sont simples à générer dans cet exemple, on crée 10 échantillons d'apprentissage de taille $n = 400$ et 10 échantillon de test de taille 1000. Les échantillons d'apprentissage sont obtenus en décomposant n réalisations des variables fonctionnelles par PLS simultanée. Les échantillons de tests sont obtenus en décomposant 1000 réalisations des variables fonctionnelles sur la base obtenue pour l'échantillon d'apprentissage. Ainsi, les coefficients des échantillons d'apprentissage et de test correspondent bien aux mêmes fonctions de base. Les échantillons simulés selon les distributions de probabilité estimées sont aussi de taille 1000. De plus, chaque taux d'apprentissage est calculé à partir des résultats de 100 tests d'adéquation. La Figure 2.23 représente les boxplots de ces taux d'acceptation en fonction de la taille de la base de décomposition. L'évolution du taux d'acceptation en fonction de la taille de la base de décomposition est assez similaire pour les cinq méthodes. Le taux d'acceptation croît en fonction de d jusqu'à environ $d = 10$ puis décroît. Les méthodes d'estimation de matrices de covariance creuses ne donnent pas des résultats meilleurs que l'algorithme EM dans ce cas. Pour les méthodes d'estimation EM, sEM, sEM2.2 et sEM2.3, les taux d'acceptation sont au-dessus de 75% pour des bases de décomposition de plus de deux composantes. Avec la méthode sEM2.1, le taux est moins élevé. Enfin, la taille de base de décomposition $d = 10$ donne les meilleurs résultats.

Le critère C_2^e est ensuite étudié. Il permet d'évaluer si les corrélations entre les variables fonctionnelles sont bien reproduites dans la modélisation des variables fonctionnelles. Pour ce faire, la corrélation temps par temps définie dans l'équation (2.28), est calculée entre les variables Z_1 et Z_2. Ensuite, pour un couple

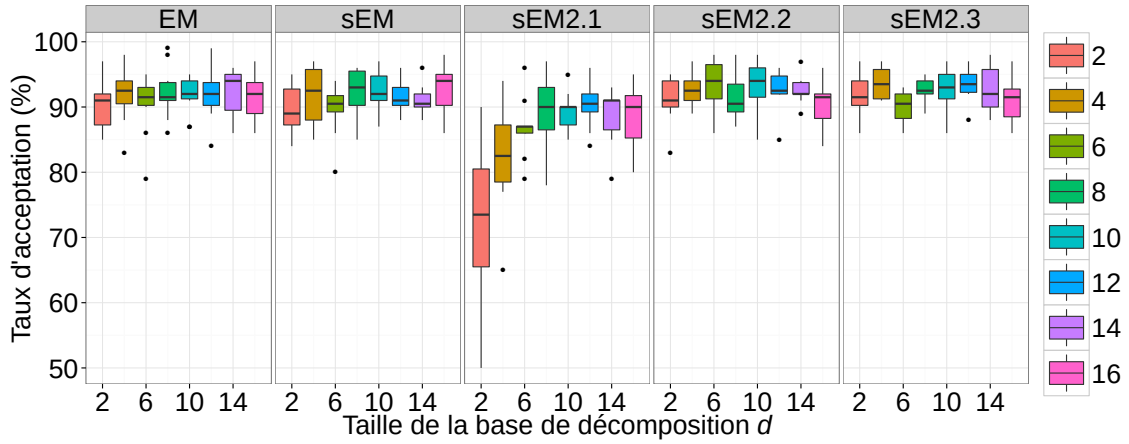

FIGURE 2.23 – Cas analytique : boxplots du critère C_1^e, taux d'acceptation du test d'adéquation de Fromont et al. (2012) entre les coefficients d'échantillons de test et ceux simulés selon la distribution estimée, en fonction de la taille d de la base de décomposition.

d'échantillons d'apprentissage et de test, la corrélation entre les modélisations $\hat{Z}_1$ et $\hat{Z}_2$ puis le critère C_2^e sont calculés. Les mêmes tailles d'échantillons de test et d'apprentissage que pour le critère C_1^e sont utilisées. La Figure 2.24 représente les boxplots du critère C_2^e en fonction de la taille de la base de décomposition. Les valeurs du critère C_2^e sont très proches pour toutes les méthodes. La corrélation temps par temps entre les deux variables fonctionnelles est bien reproduite : l'erreur quadratique moyenne est inférieure à 0,01 pour la plupart des tailles de base de décomposition. L'erreur quadratique sur la corrélation est minimale pour $d = 12$. En effet, pour $d > 12$, la qualité de la modélisation se détériore légèrement comme on le voit sur la Figure 2.23, ce qui peut expliquer que l'erreur quadratique sur les corrélations augmente.

Enfin, le dernier critère étudié C_3^e quantifie la qualité de l'approximation de la variable

$$\hat{Y} = \mathcal{M}\left(X_1, X_2, X_3, \hat{Z}_1, \hat{Z}_2\right). \tag{2.30}$$

Le test d'adéquation de Kolmogorov-Smirnov est appliqué entre des réalisations de la covariable Y et des réalisations de l'estimation de la covariable $\hat{Y}$, obtenue comme dans l'équation (2.30). Le taux d'acceptation du test est calculé selon la même méthodologie que pour le critère C_1^e. Le même nombre et les mêmes tailles d'échantillon et de test sont utilisés. La Figure 2.25 représente le taux d'acceptation du test de Kolmogorov-Smirnov en fonction de la taille de la base de décomposition. Le critère C_3^e est quasiment constant en fonction de la taille de la base. De plus, il est très élevé. En effet, la médiane vaut toujours environ 90%, ce qui atteste de la bonne reproduction de la densité de probabilité de la covariable.

L'étude de ces trois critères montre que la modélisation de la distribution de probabilité des coefficients de la décomposition SPLS et la méthode globale de quantification des incertitudes de variables fonctionnelles donnent de très bons résultats sur ce cas d'application. L'analyse conjointe de ces trois critères peut aussi permettre de sélectionner le nombre d de composantes retenues dans la décomposition. Dans cet exemple, le critère C_1^e est optimal pour $d = 10$ avec toutes les méthodes d'estimation. Le critère C_2^e est aussi optimal pour $d = 10$. Comme le critère C_3^e varie très peu en fonction de d, on peut choisir $d = 10$ comme taille de la base de décomposition. De plus, pour $d = 10$, la méthode d'estimation sEM2.2 donne les meilleurs résultats sur ces critères, elle est donc retenue. Pour illustrer le modèle ainsi obtenu, on représente sur la Figure 2.26 les coefficients observés en noir et le contour de la densité de probabilité estimée avec l'algorithme sEM2.2 en rouge).

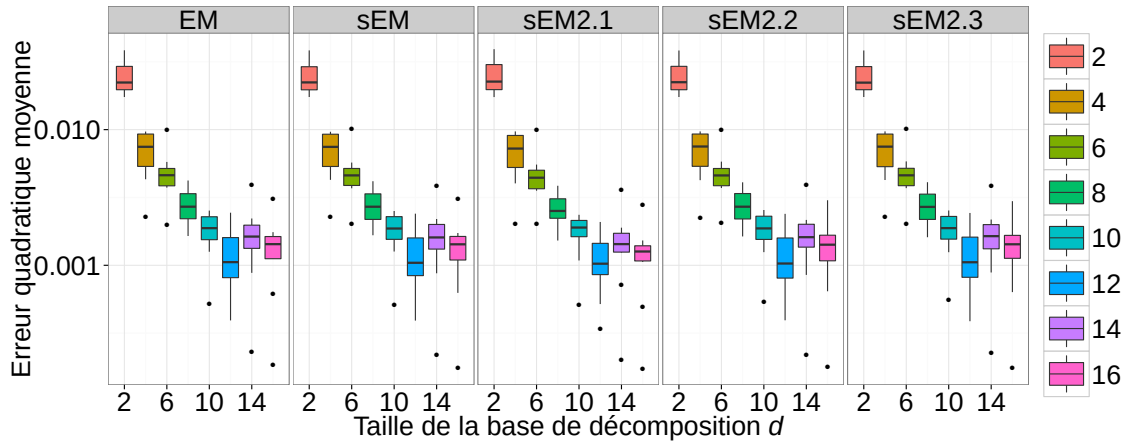

FIGURE 2.24 – Cas analytique : boxplots du critère C_2^e, erreur sur les corrélations temps par temps, en fonction de la taille d de la base de décomposition.

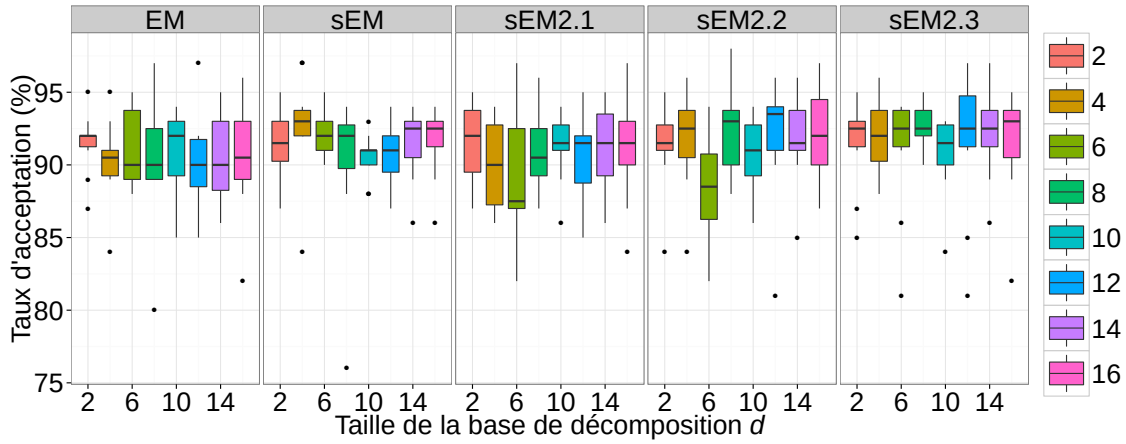

FIGURE 2.25 – Cas analytique : boxplots du critère C_3^e, taux d'acceptation du test de Kolmogorov-Smirnov sur la covariable, en fonction de la taille d de la base de décomposition.

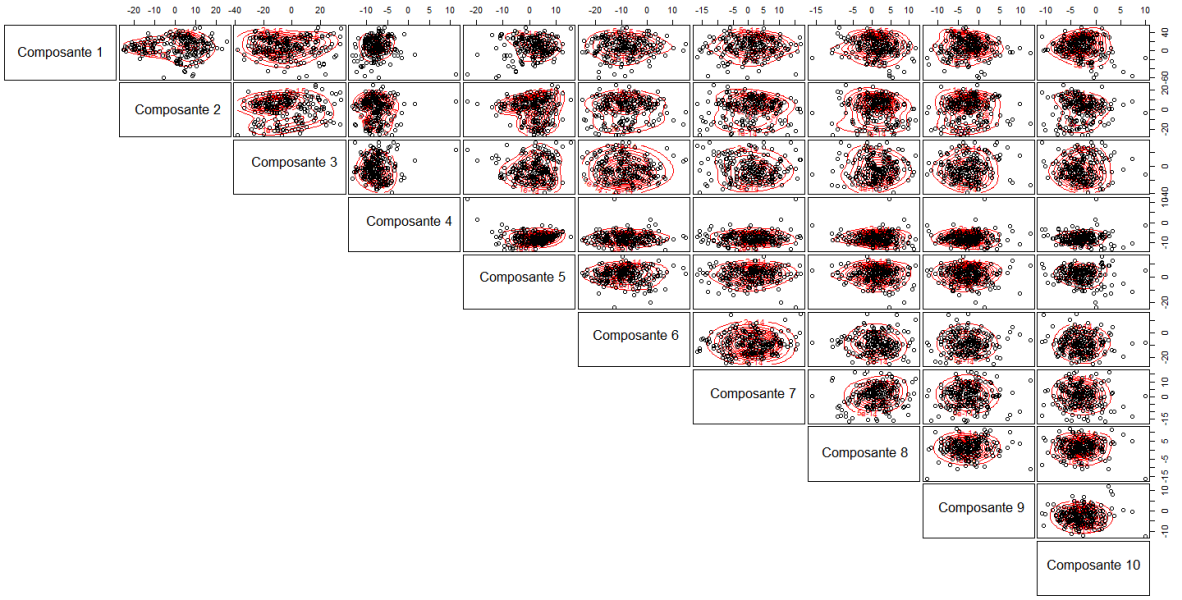

FIGURE 2.26 – Cas analytique : les 200 coefficients de la base d'apprentissage en dimension 10 (en noir) et le contour de la densité de probabilité estimée avec l'algorithme sEM2.2 (en rouge).

Une application possible de cette méthodologie de quantification des incertitudes de variables fonctionnelles est le calcul de probabilités jointes de dépassement de seuil. Pour l'illustrer sur cet exemple analytique, on définit comme suit la probabilité à estimer :

$$p = P\left((t \in I : Z_1(t) > 2) \cup \left(\min_{t \in I}(Z_2(t) < 0) < \frac{1}{2}\right) \cup \left(Y < \frac{1}{2}\right)\right).$$

La valeur de référence de la probabilité p est calculée sur un échantillon de 10^5 réalisations des variables fonctionnelles et de la covariable. La valeur calculée, $p \approx 0,3223$, est considérée, par la suite, comme la valeur de référence. Une estimation $\hat{p}$ de p est estimée avec un échantillon de 10^5 réalisations du mélange de gaussiennes. L'erreur relative d'approximation

$$100\frac{|p - \hat{p}|}{p}$$

est calculée pour 50 échantillons d'apprentissage et pour différentes tailles de la base de décomposition. La Figure 2.27 représente l'erreur relative en fonction de la taille de la base pour chaque algorithme d'estimation des paramètres du mélange de gaussiennes. De bons résultats sont obtenus avec tous les algorithmes, et on observe une décroissance rapide de l'erreur, cette dernière étant aux alentours de 10% pour $d = 10$.

2.3.6.2 Application au cas du choc thermique pressurisé

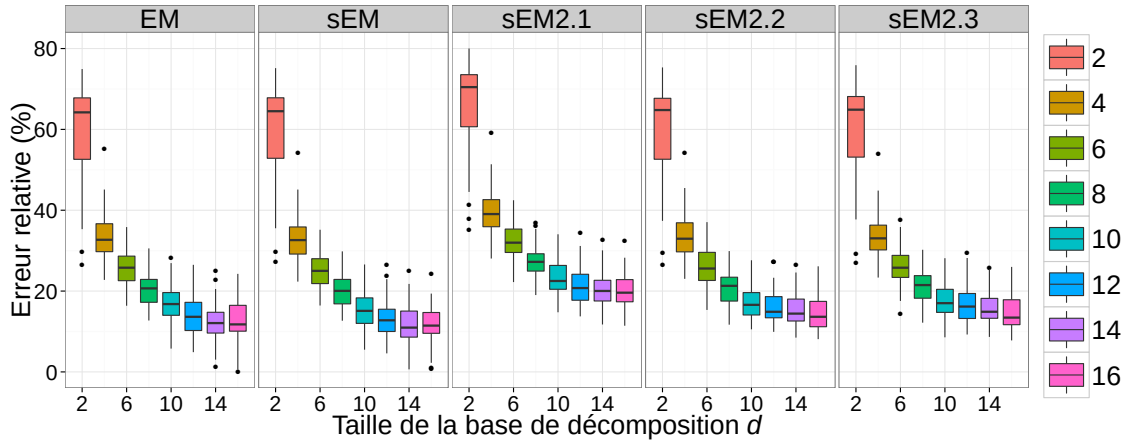

FIGURE 2.27 – Cas analytique : boxplot des erreurs relatives d'approximation entre la probabilité estimée $\hat{p}$ et p en fonction de la taille de la base et pour chaque algorithme d'estimation des paramètres du mélange de gaussiennes.

Rappels sur le cas du choc thermique pressurisé

Dans le cas du PTS, trois variables fonctionnelles ou transitoires T-H sont considérées : la température, la pression et le coefficient d'échange thermique. Ces trois variables sont liées *via* le code CAST3M à une covariable, appelée CS, dépendant aussi de variables scalaires. Un échantillon probabilisé de 1000 réalisations des transitoires T-H et des variables scalaires ainsi que les valeurs de la covariable correspondantes est disponible.

A l'issue de l'étude réalisée dans la section 2.2.7.1, la décomposition SPLS a été retenue pour décomposer les trois variables fonctionnelles. La taille d de la décomposition n'a pas été choisie à l'issue de l'étape de décomposition fonctionnelle.

On cherche dans cette section à choisir le nombre d de composantes de la base de décomposition, et à modéliser la densité de probabilité des d coefficients sélectionnés. Comme dans l'exemple analytique, le nombre de gaussiennes dans le mélange est choisi en maximisant le critère BIC. Le paramètre λ de la pénalisation est sélectionné par validation croisée pour les méthodes sEM et sEM2..

Les trois critères définis dans la section 2.3.5 sont appliqués pour les cinq méthodes étudiées et pour différentes tailles de la base de décomposition. Comme le nombre de réalisations des variables fonctionnelles est limité, contrairement au cas analytique dans la section précédente, on choisit de calculer les critères par validation croisée. L'échantillon de $n = 1000$ réalisations des variables fonctionnelles est partitionné aléatoirement en un échantillon d'apprentissage de taille 600 et un échantillon de test de 400 réalisations. Pour une paire d'échantillons d'apprentissage et de test, la distribution de probabilité des coefficients est estimée sur l'échantillon d'apprentissage et les critères sont calculés à l'aide de l'échantillon de test et d'échantillons générés selon la densité estimée. Pour le critère C_1^e, le taux d'acceptation est calculé à partir de 100 échantillons générés selon la densité estimée, et est représenté en fonction de la taille de la base de décomposition sur la Figure 2.28. L'évolution du critère C_1^e a la même forme que dans le cas analytique. Il augmente puis diminue rapidement. Avec la méthode sEM de Krishnamurthy (2012), la décroissance pour $d > 10$ est plus rapide qu'avec les autres méthodes. Les trois méthodes sEM2. donnent des résultats très proches. La décroissance du taux d'acceptation pour sEM2.2 est légèrement plus lente pour $d > 10$. Les méthodes sEM2. donnent des taux sensiblement meilleurs que EM pour des bases de décomposition de tailles comprises entre 6 et 16. En particulier, pour $d = 10$, la modélisation avec l'algorithme EM a un taux d'acceptation de 82% alors que, pour sEM2., le taux est d'environ 92%. La taille de base qui donne le taux maximal est $d = 8$ pour l'algorithme EM et $d = 10$ avec les méthodes sEM2.1, sEM2.2 et sEM2.3.

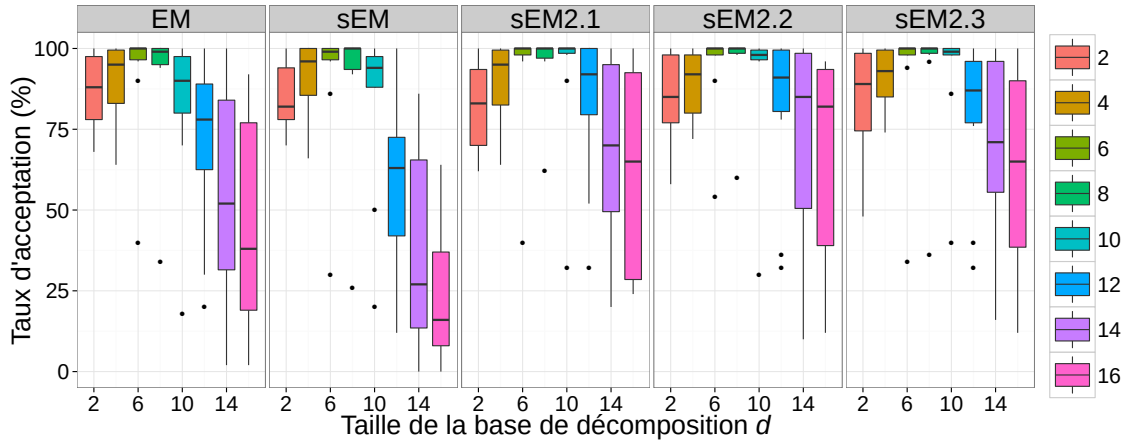

FIGURE 2.28 – Cas PTS : boxplots du critère C_1^e, taux d'acceptation du test d'adéquation sur les coefficients, en fonction de la taille d de la base de décomposition.

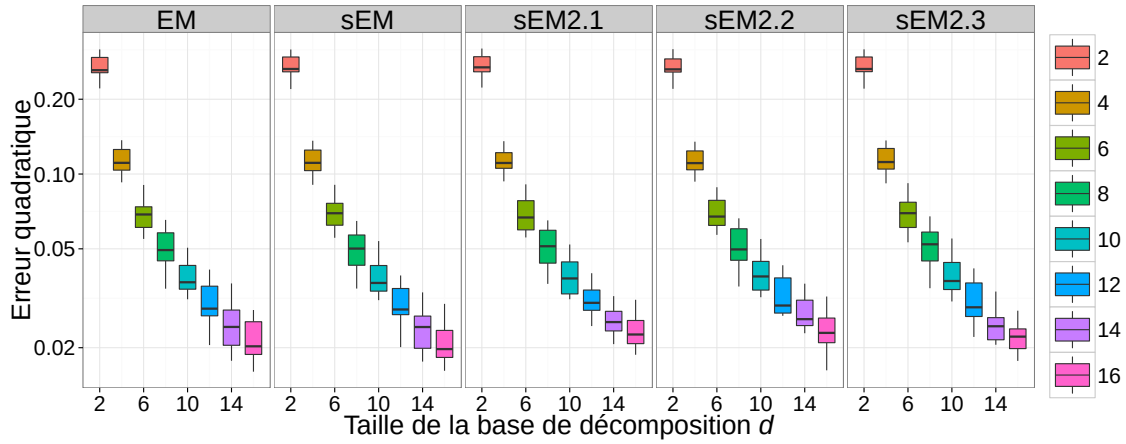

FIGURE 2.29 – Cas PTS : boxplots du critère C_2^e, erreur sur la corrélation temps par temps, en fonction de la taille d de la base de décomposition (échelle logarithmique).

Le critère C_2^e évalue la capacité de la méthodologie globale de quantification des incertitudes à approcher les corrélations temps par temps entre les variables fonctionnelles. La Figure 2.29 représente le critère C_2^e, *i.e.* la somme des erreurs quadratiques sur les corrélations temps par temps entre chaque couple de variables fonctionnelles. Les erreurs quadratiques des différentes méthodes sont relativement proches pour une taille de base donnée et décroissent rapidement quand la taille de la base augmente.

A partir des critères C_1^e et C_2^e, on peut sélectionner une taille de base. Pour l'algorithme sEM2., le critère C_1^e est maximal pour $d = 10$ et, pour $d \geq 10$, l'erreur quadratique moyenne sur la corrélation est en-dessous de 0,05. De plus, pour $d = 10$, C_2^d, la variance expliquée définie dans la section 2.2.6, vaut environ 80%, et le critère C_3^d vaut 90%. Ainsi, les variables fonctionnelles sont assez bien approchées par une décomposition PLS sur 10 composantes, et les caractéristiques des variables fonctionnelles retenues par la décomposition expliquent bien la covariable. **On choisit donc de retenir $d = 10$ composantes dans la décomposition PLS simultanée.**

Le calcul du dernier critère, C_3^e nécessite un grand nombre d'évaluations du code $\mathcal{M}$. Le code CAST3M qui calcule la covariable est assez peu coûteux en temps de calcul, mais son coût d'évaluation limite tout

de même le nombre de calculs possibles à quelques milliers ou dizaines de milliers – moins que le nombre d'évaluations qui seraient nécessaires pour calculer le critère C_3^e entre $d = 2$ et 16. Ainsi, seuls quelques milliers d'appels au code ont été réalisés avec en entrée des fonctions simulées selon la modélisation estimée avec $d = 10$ et la méthode sEM2.3. Pour ces réalisations de la covariable, l'hypothèse d'adéquation est acceptée dans le test de Kolmogorov-Smirnov à 5%.

La méthode d'estimation de matrices de covariances creuses proposée avec la troisième matrice de pénalisation (sEM2.3) donne les meilleurs résultats en matière d'estimation de densité et pour l'approximation de la corrélation entre les transitoires. L'algorithme sEM2. s'avère moins sensible au choix de la matrice de pénalisation que dans l'exemple analytique.

La Figure 2.30 représente à gauche 100 réalisations des variables fonctionnelles et à droite 100 réalisations simulées selon la distribution estimée pour la température, la pression et le coefficient d'échange respectivement. Pour générer ces courbes, la décomposition fonctionnelle utilisée est la PLS simultanée. La densité de probabilité des coefficients de la PLS simultanée a été modélisée par un mélange de 3 gaussiennes estimé par l'algorithme sEM2.3. Dans l'ensemble, les trois variables fonctionnelles sont assez bien modélisées. Pour la température, la forme générale des courbes est bien conservée. L'intervalle $[0; 1500]$ est aussi particulièrement bien reproduit. Cependant, les courbes simulées n'ont pas la même variabilité que les courbes de l'échantillon d'apprentissage. En effet, certains profils de courbes, rares dans l'échantillon d'apprentissage, ne sont pas « capturés » par la décomposition fonctionnelle. La pression semble bien mieux approchée. La variabilité de l'échantillon initial semble bien conservée dans la modélisation. Le transitoire du coefficient d'échange est très bien reproduit jusqu'au pas de temps $t = 500s$ environ. Il est moins bien modélisé au-delà. En effet, les petites fluctuations visibles sur les courbes réelles n'apparaissent pas sur les courbes simulées. Cependant, selon les avis d'experts, cette partie de la courbe aurait *a priori* peu d'effet sur la valeur du critère de sécurité.

2.3.6.3 Application au cas de la rupture LiPoSo

Rappels sur le cas de la rutpure LiPoSo

Dans le cas de la rutpure LiPoSo, trois variables fonctionnelles ou transitoires T-H sont considérées : le débit, la puissance et la température. Aucune covariable n'est considérée dans ce cas d'étude. Un échantillon probabilisé de 200 réalisations des transitoires T-H est disponible.

A l'issue de l'étude réalisée dans la section 2.2.7.1, la décomposition ACPS a été retenue pour décomposer les trois variables fonctionnelles. Avant d'applique la décomposition fonctionnelle, la variable de puissance est translatée pour simplifier sa projection sur la base obtenue par ACPS. La normalisation appliquée aux variables avant l'ACPS est la normalisation optimisée selon la méthode de Perrin et al. (2013) (présentée dans la section 2.2.4.3). La taille d de la décomposition n'a pas été choisie à l'issue de l'étape de décomposition fonctionnelle.

On cherche maintenant à sélectionner la taille d de la base de décomposition et à modéliser la distribution de probabilité des d premiers coefficients de la base ACPS. Seules les méthodes d'estimation des paramètres des mélanges de gaussiennes EM, sEM, sEM2.2 et sEM2.3 sont appliquées dans ce cas. En effet, la méthode sEM2.1 dans laquelle les diagonales des matrices de covariances sont aussi pénalisées, n'est pas illustrée ici car elle a donné de moins bons résultats sur les précédents cas et a montré des difficultés de convergence dans ce cas d'étude. Comme dans les deux autres cas d'application, le nombre de gaussiennes dans le mélange est choisi grâce au critère BIC défini dans la section 2.3.3.4, et le paramètre de la pénalisation est choisi par validation croisée pour les méthodes sEM, sEM2.2 et sEM2.3.

Les critères définis dans la section 2.3.5 sont calculés sur les estimations réalisées avec quatre des algorithmes étudiés et avec différentes tailles de la base de décomposition. Comme le nombre de réalisations des variables fonctionnelles est limité, ces trois critères sont calculés par validation croisée. L'échantillon de $n = 200$ réalisations des variables fonctionnelles est partitionné aléatoirement en un échantillon d'apprentissage de 150 réalisations et un échantillon de test de 50. Pour une paire d'échantillons d'apprentissage et de test, la distribution de probabilité des coefficients est modélisée sur l'échantillon d'apprentissage et les critères sont calculés à l'aide de l'échantillon de test et d'échantillons de

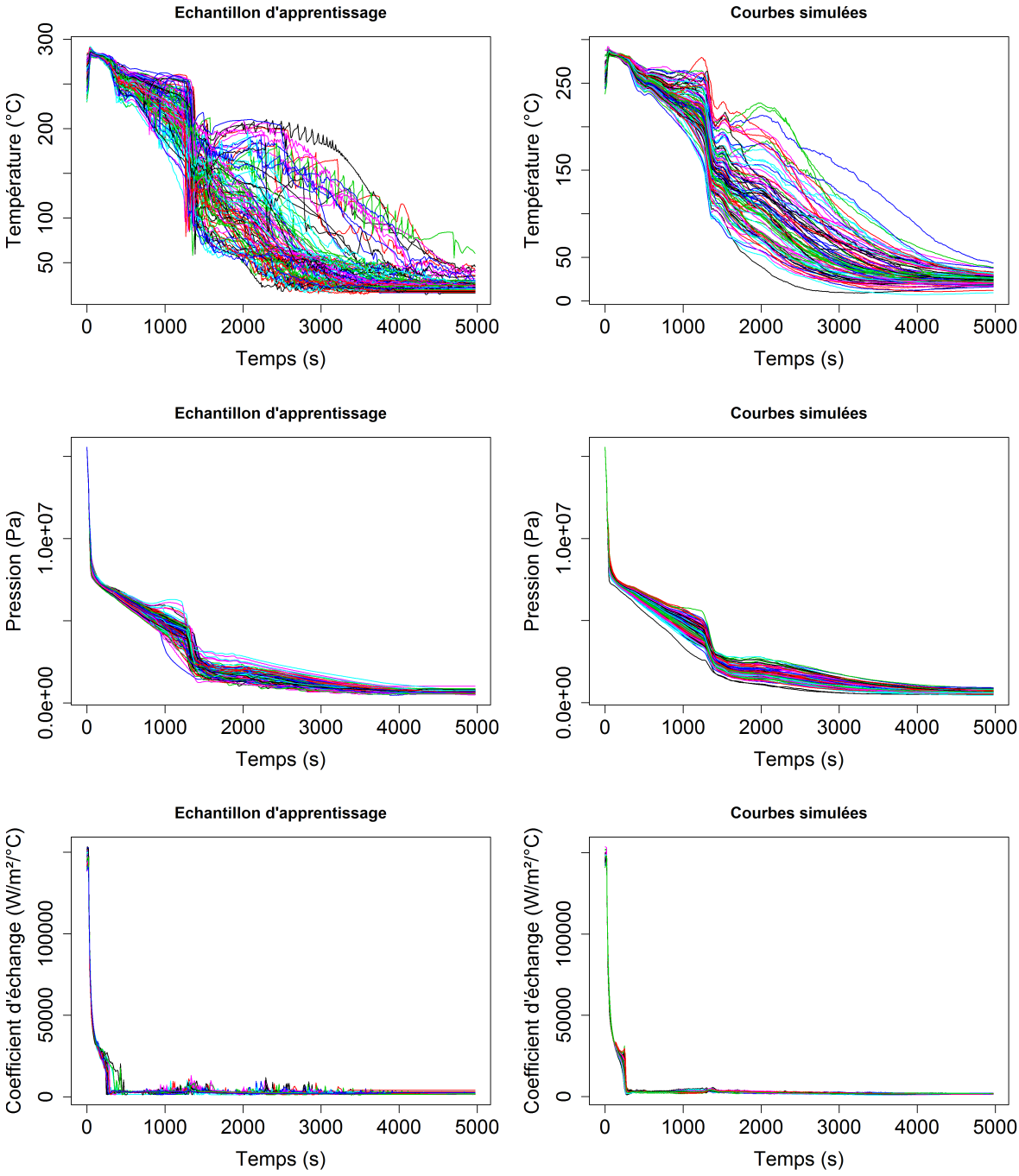

FIGURE 2.30 – Cas PTS : 100 courbes de la base d'apprentissage (à gauche) et simulées selon la densité de probabilité estimée (à droite) pour les variables de température (en haut), pression (au milieu) et coefficient d'échange (en bas).

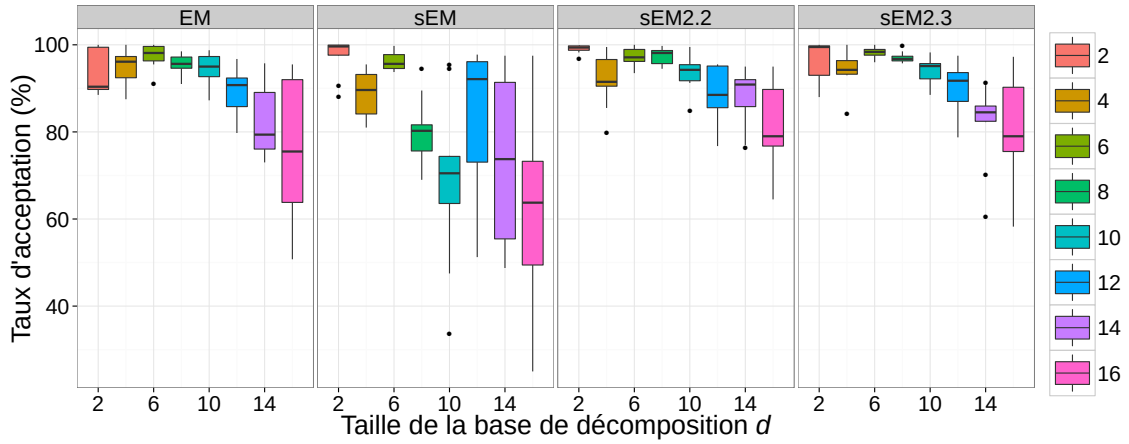

FIGURE 2.31 – Cas LiPoSo : boxplots du critère C_1^e, taux d'acceptation du test d'adéquation sur les coefficients, en fonction de la taille d de la base de décomposition.

taille 100 générés selon la densité estimée. Pour le critère C_1^e, le taux d'acceptation est calculé à partir de 100 échantillons générés, et est représenté en fonction de la taille de la base de décomposition sur la Figure 2.31. Pour la méthode sEM, l'évolution du taux d'acceptation n'est pas régulière. Les trois autres méthodes d'estimation donnent de meilleurs résultats. Pour celles-ci, le critère C_1^e est très élevé jusqu'à $d = 10$: sa médiane est supérieure à 90%. Le mélange de gaussiennes approche donc très bien la distribution des coefficients. Ces trois algorithmes d'estimation des paramètres donnent des résultats très proches, mais la décroissance du taux d'acceptation est légèrement moins rapide avec les algorithmes sEM2. pour $d > 12$.

Le critère C_2^e est calculé par validation croisée. Comme précédemment, l'échantillon de 200 réalisations des variables aléatoires fonctionnelles est partitionné en un échantillon d'apprentissage de taille 150 et un échantillon de test de taille 50. Les corrélations temps par temps entre les modélisations des variables fonctionnelles sont calculées avec des réalisations générées selon cette distribution. Cette corrélation estimée est ensuite comparée à la corrélation calculée sur l'échantillon de test. Ce calcul est répété pour plusieurs échantillons générés selon la distribution estimée et pour 10 paires d'échantillons d'apprentissage et de test. L'erreur absolue sur la corrélation est représentée sur la Figure 2.32. Les erreurs sur les corrélations sont très faibles. De plus, à partir de $d = 10$, l'erreur stagne autour de 5.10^{-3} pour les algorithmes EM et sEM2.. Avec sEM, l'erreur augmente à partir de 12 composantes dans la base de décomposition. Ce comportement différent de ceux des deux autres exemples (cf. Figures 2.24 et 2.29) peut s'expliquer par le fait que la corrélation entre les courbes simulées pourrait être plus dépendante dans cet exemple de la qualité de l'estimation de la densité de probabilité que dans les autres exemples, et que la qualité de l'estimation se dégrade plus vite avec la méthode sEM qu'avec les trois autres algorithmes d'estimation.

Le code Trio_U MCT étant coûteux en temps de calcul, le critère C_3^e n'a pas été calculé ici. Les critères C_1^e, C_2^e ainsi que C_3^d permettent de sélectionner la taille de la base de manière à approcher au mieux les trois variables fonctionnelles. Le critère C_1^e est supérieur à 90% jusqu'à $d = 10$, ce qui indique que la distribution de probabilité des coefficients est très bien modélisée. De plus, le critère C_2^e ne décroît plus pour des bases de taille supérieure ou égale à 10, et ce critère est très faible pour ces tailles de base. Enfin, pour $d = 10$, la variance expliquée par la décomposition est d'environ 99,6%. **Compte tenu de ces différents critères, on choisit donc de retenir $d = 10$ composantes dans la base de décomposition construite par ACP simultanée. On choisit l'algorithme EM car l'approximation est légèrement meilleure à $d = 10$ avec cette méthode.**

Les fonctions générées selon le modèle construit ci-dessus sont comparées aux fonctions issues du code CATHARE2 sur la Figure 2.33. Dans l'ensemble, les trois variables fonctionnelles semblent globalement

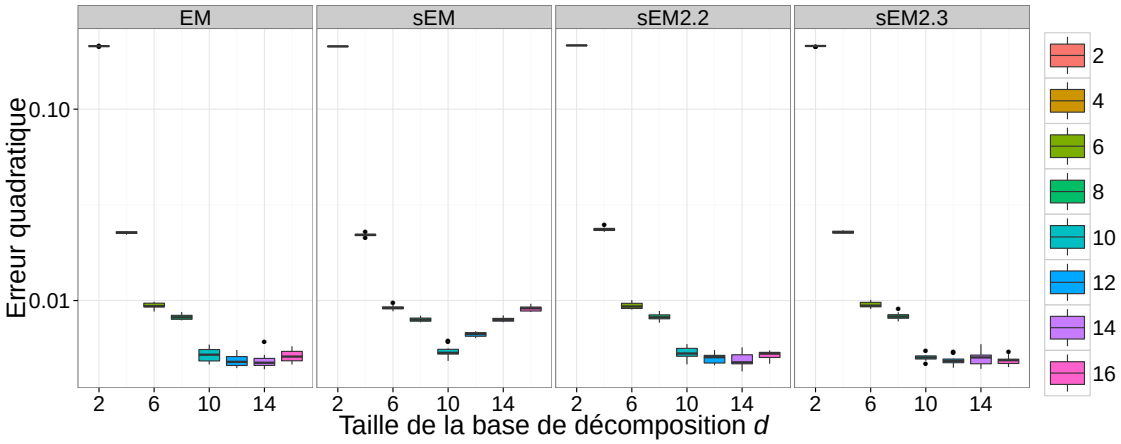

FIGURE 2.32 – Cas LiPoSo : boxplots du critère C_2^e, erreur sur les corrélations temps par temps, en fonction de la taille d de la base de décomposition.

bien modélisées. La variabilité de l'échantillon de courbes disponible est bien reproduite. Pour le débit, on remarque cependant que la décroissance observée dans les courbes de l'échantillon initial vers $t = 1$ ou $2s$ est trop accentuée dans certaines courbes simulées.

Synthèse des tests sur les trois cas d'application

Les méthodes d'estimation des paramètres de mélanges de gaussiennes présentées dans les sections 2.3.3.3 et 2.3.4 ont été appliquées et comparées sur l'exemple analytique introduit dans la section 2.2.7.1, le cas-test du PTS, et celui de la rupture LiPoSo. La densité de probabilité des coefficients des décompositions retenues dans la section 2.2.7 a été estimée pour différentes tailles de la base de décomposition. Les trois critères proposés pour évaluer ces méthodes d'estimation ont été calculés et ont permis de sélectionner une taille d pour la décomposition fonctionnelle. Plusieurs conclusions peuvent être tirées des tests réalisés sur ces trois cas :

- la sélection du paramètre de pénalisation λ de la méthode sEM s'avère difficile en pratique et ne permet pas forcément d'obtenir des matrices de covariance creuses,
- l'efficacité des méthodes sEM2. est dépendante de la matrice de pénalisation choisie,
- les méthodes sEM2.2 et sEM2.3 améliorent l'estimation des paramètres par rapport aux méthodes EM, sEM et sEM2.1,
- la méthodologie proposée donne de bons résultats sur les trois cas d'étude.

2.4 Visualisation de données fonctionnelles

Cette section est consacrée à la visualisation d'un échantillon de données fonctionnelles, *i.e.* d'un ensemble de réalisations de variables aléatoires fonctionnelles, les variables fonctionnelles pouvant être dépendantes entre elles. Ces méthodes ont pour objectif d'aider à l'exploration des données et de mettre en valeur leurs caractéristiques statistiques. Elles sont mises en œuvre sur un échantillon de fonctions $\{z_i\}_{i=1\ldots n}$, $n \in \mathbb{N}$. Tout d'abord, un état de l'art des méthodes de visualisation de données fonctionnelles est présenté. Les sections 2.4.1, 2.4.2 et 2.4.3 présentent trois généralisations du boxplot (Tukey 1977) au cas fonctionnel, à savoir le bagplot fonctionnel, le HDR boxplot et le boxplot fonctionnel. Dans la section 2.4.4, nous proposons une extension du HDR boxplot qui s'appuie sur la méthodologie de quantification des incertitudes que nous avons développée dans les sections précédentes. Enfin, les différentes méthodes présentées sont appliquées aux cas d'étude du choc thermique pressurisé et de la rupture LiPoSo.

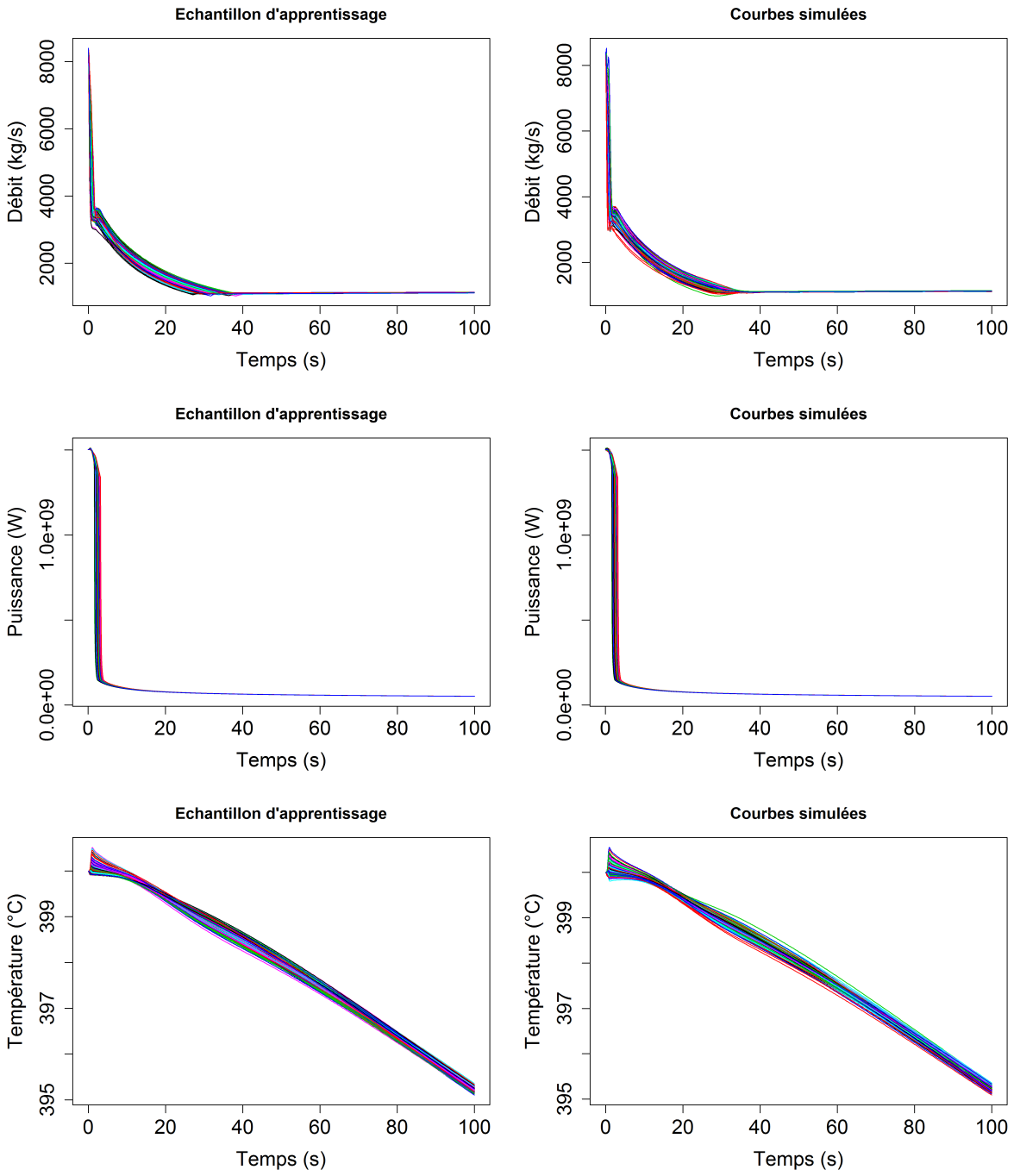

FIGURE 2.33 – Cas LiPoSo : 100 courbes de la base d'apprentissage (à gauche) et simulées selon la densité de probabilité estimée (à droite) pour les variables de débit (en haut), de puissance (au milieu) et de température (en bas).

2.4.1 Bagplot fonctionnel

Rousseeuw et al. (1999) ont adapté la définition du boxplot univarié dans le cas bivarié. Ce nouvel outil de visualisation est appelé bagplot. Celui-ci possède trois principaux composants :

- le sac (*bag*) qui contient la moitié des points,
- l'enveloppe (*fence*) qui sépare les valeurs extrêmes (*outliers*) des autres points,
- la boucle (*loop*) qui contient les points hors du sac mais à l'intérieur de l'enveloppe.

Pour construire le bagplot, les auteurs utilisent la mesure *halfspace location depth* introduite par Tukey (1975). Pour un point x et un ensemble de points Z, cette mesure est définie comme le plus petit nombre de points de Z contenus dans un demi-plan fermé dont la frontière passe par x. La région D_k est l'ensemble des points x ayant une *halfspace location depth* supérieure à k. La médiane est définie comme le point maximisant cette mesure ou, s'il n'y a pas unicité, le centre de l'ensemble des points maximisant cette mesure. Pour construire le sac, on cherche k tel que $|D_k| \leqslant \lfloor \frac{n}{2} \rfloor \leqslant |D_{k-1}|$, où $\lfloor \frac{n}{2} \rfloor$ désigne le plus grand entier inférieur à $\frac{n}{2}$ et $|D_k|$ le cardinal de D_k. Le sac est défini comme étant la région D_{k-1}. Enfin, l'enveloppe est obtenue en augmentant la taille du sac d'un facteur 3 par rapport à la médiane.

Une extension du bagplot de Rousseeuw et al. (1999) au cas fonctionnel est proposée par Hyndman et Shang (2010). Les auteurs appliquent le bagplot bivarié aux coefficients correspondant aux deux premières composantes de la décompositions ACP des fonctions $\{z_i\}_{i=1\ldots n}$. La détermination de l'enveloppe se fait en augmentant le sac d'un facteur 2,58 ce qui permet à l'enveloppe de contenir théoriquement 99% des points si ces points suivent une distribution multi-normale standard. Hyndman et Shang (2010) proposent deux représentations : la représentation bivariée qui correspond au bagplot bivarié des deux premières composantes principales et la représentation fonctionnelle. Dans cette dernière, les éléments du bagplot bivarié sont transposés aux courbes étudiées. Le bagplot fonctionnel représente la courbe médiane, les courbes extrêmes et les enveloppes des courbes contenues respectivement dans le sac et la boucle. Cet outil de visualisation est appliqué aux courbes de mortalité des hommes en France de 1899 à 2005 qui sont étudiées dans Hyndman et Shang (2010). Chaque courbe correspond à une année et représente le taux de mortalité en fonction de l'âge. Les bagplots bivarié et fonctionnel correspondants sont représentés sur la Figure 2.34 qui est extraite de Hyndman et Shang (2010). L'étoile rouge est la médiane, la zone gris foncé est le sac, la zone gris clair est la boucle. Sur le graphique de gauche, la médiane est représentée en noir et les intervalles de confiance à 95% en pointillés bleus. Les courbes et points colorés sont les valeurs extrêmes. Comme on peut le voir sur cette Figure, l'une des limitations du bagplot fonctionnel est qu'il donne de mauvais résultats quand les données ne sont pas unimodales. Dans cet exemple, le point médian est éloigné des coefficients et la courbe médiane correspondante ne peut donc pas être une courbe représentative de l'échantillon.

2.4.2 HDR boxplot

Hyndman et Shang (2010) proposent une autre méthode de visualisation de données fonctionnelles : le HDR (*Highest Density Region*) boxplot fonctionnel. Celle-ci s'appuie sur l'estimation de la densité des deux premières composantes issues de la décomposition ACP des fonctions observées. Pour cela, Hyndman et Shang (2010) utilisent un estimateur à noyau :

$$\hat{f}(z) = \frac{1}{n} \sum_{i=1}^{n} K_{h_i} \left(z - Z_i \right),$$

où $z \in \mathbb{R}^2$ et $Z = \{Z_1, \ldots, Z_n\} \subset \mathbb{R}^2$ est l'ensemble des couples de composantes de l'ACP. Une région de plus haute densité est définie comme $R_\alpha = \{z \in \mathbb{R}^2 : \hat{f}(z) \geqslant f_\alpha\}$, où f_α est tel que $\int_{R_\alpha} \hat{f}(z)dz = 1 - \alpha$. Le HDR boxplot représente les quatre éléments suivants :

- les régions de plus haute densité avec $\alpha = 50\%$ et $\alpha = 99\%$,
- les valeurs extrêmes, définies comme étant les points qui n'appartiennent pas aux deux régions définies précédemment,
- la mode, *i.e.* le point de l'échantillon de plus haute densité.

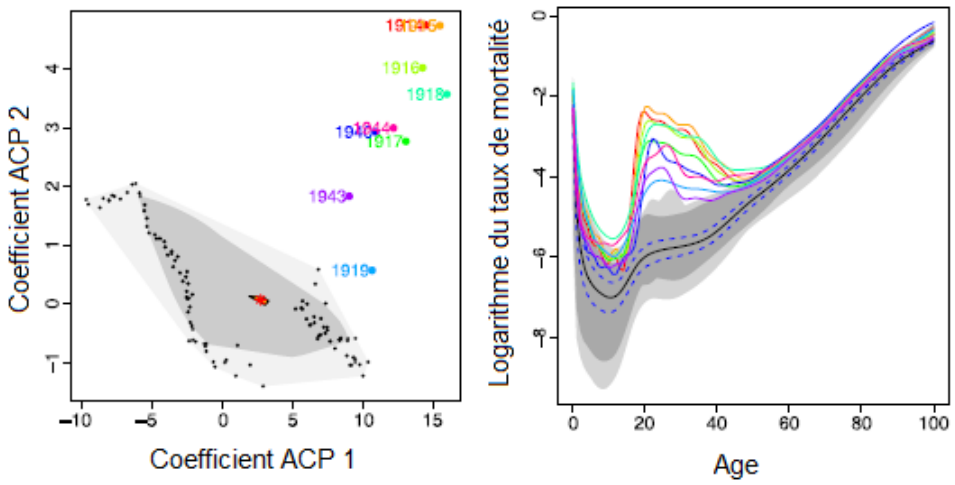

FIGURE 2.34 – Bagplot bivarié des courbes de mortalité (à gauche) et sa représentation fonctionnelle (à droite), extrait de Hyndman et Shang (2010).

Comme dans le bagplot fonctionnel, Hyndman et Shang (2010) définissent à la fois les versions bivariée et fonctionnelle du HDR boxplot. De manière similaire, le HDR boxplot fonctionnel est constitué de la courbe médiane (correspondant au mode), des courbes extrêmes (correspondant aux valeurs extrêmes) et des enveloppes fonctionnelles correspondant aux régions de plus haute densité avec $\alpha = 50\%$ et $\alpha = 99\%$. En appliquant le HDR boxplot au même jeu de données que le bagplot fonctionnel, on obtient les résultats illustrés par la Figure 2.35, extraite de Hyndman et Shang (2010). L'étoile rouge est le mode, la zone gris foncé est la région de plus haute densité à 50%, la zone gris clair est celle à 95%. Sur le graphique de gauche, la courbe médiane est représentée en noir. Les courbes et points colorés sont les données extrêmes. En comparant les Figures 2.34 et 2.35 créées à partir des mêmes données, on remarque que le sac et l'enveloppe du bagplot ont des formes très différentes des ensembles représentés sur le HDR boxplot. Le HDR boxplot bivarié semble présenter deux groupes de données, alors que le bagplot n'en considère qu'un. Le HDR boxplot semble sur cet exemple mieux s'adapter aux données que le bagplot fonctionnel.

2.4.3 Boxplot fonctionnel

Une autre représentation par boxplot est proposée par Sun et Genton (2011). Celle-ci utilise la profondeur de bande (*band depth*) introduite par López-Pintado et Romo (2009) pour ordonner les fonctions. La profondeur de bande mesure la centralité des courbes : plus la profondeur de bande est grande, plus la courbe est centrale.

Soit $J \in \{2, \ldots, n\}$ le nombre de courbes définissant une bande, Z un processus stochastique défini sur D, et $z_1, \ldots, z_n$ des réalisations de ce processus. Le graphe d'une fonction z est définie par $G(z) = \{(t, z(t)) : t \in D\}$ et une bande délimitée par les fonctions $z_{i_1}, \ldots, z_{i_k}$ est définie par :

$$B(z_{i_1}, \ldots, z_{i_k}) = \{(t, z(t)) : t \in D, \min_{r=1\ldots k} z_{i_r}(t) \leqslant z(t) \leqslant \max_{r=1\ldots k} z_{i_r}(t)\}.$$

$B(z_{i_1}, \ldots, z_{i_k})$ est l'ensemble des graphes qui en chaque point de temps sont compris entre le minimum et le maximum des fonctions $z_{i_1}, \ldots, z_{i_k}$. La profondeur de bande de la fonction z relativement à la mesure de probabilité P est définie par

$$BD_J(z, P) = \sum_{j=2}^{J} BD^{(j)}(z, P) = \sum_{j=2}^{J} P(G(z) \subset B(Z_1, \ldots, Z_j)),$$

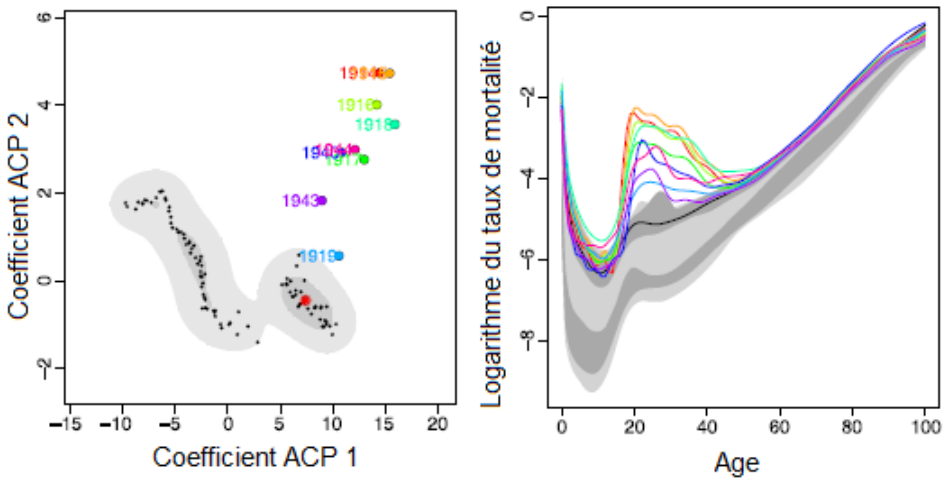

FIGURE 2.35 – HDR boxplot des courbes de mortalité (à gauche) et sa représentation fonctionnelle (à droite), extrait de Hyndman et Shang (2010).

où $B(Z_1, \ldots, Z_j)$ est une bande délimitée par les variables aléatoires $Z_1, \ldots, Z_j$ définies comme des copies indépendantes de Z. Chaque $BD^{(j)}(z, P)$, pour $j \in \{2, \ldots, J\}$, correspond à la probabilité que le graphe G de la fonction z soit compris dans les bandes délimitées par des réalisations de $Z_1, \ldots, Z_j$. López-Pintado et Romo (2009) proposent l'estimateur suivant de la probabilité $BD^{(j)}(z, P)$ d'un graphe d'appartenir à la bande $B(Z_1, \ldots, Z_j)$:

$$BD_n^{(j)}(z) = \binom{n}{j}^{-1} \sum_{1 \leqslant i_1 < \cdots < i_j \leqslant n} I(G(z) \subset B(z_{i_1}, \ldots, z_{i_j})),$$

où I est la fonction indicatrice. Une version plus souple de la profondeur de bande a été proposée par López-Pintado et Romo (2009). Dans cette méthode appelée largeur de bande modifiée, la longueur du domaine dans lequel la courbe z est à l'intérieur de la bande $B(z_{i_1}, \ldots, z_{i_j})$ est calculée.

Les fonctions $z_1, \ldots, z_n$ sont réordonnées selon leur profondeur de bande ou la largeur de bande modifiée : $z_{[1]}, \ldots, z_{[n]}$, où $z_{[1]}$ est la fonction la plus centrale (avec la plus grande profondeur de bande) et $z_{[n]}$ la fonction la plus extrême. On définit alors le sac comme la région à 50%, contenant les $\lceil \frac{n}{2} \rceil$ fonctions les plus centrales :

$$C_{0,5} = \{(t, z(t)) : t \in D, \min_{r=1\ldots\lceil n/2 \rceil} z_{[r]}(t) \leqslant z(t) \leqslant \max_{r=1\ldots\lceil n/2 \rceil} z_{[r]}(t)\},$$

où $\lceil \frac{n}{2} \rceil$ est le plus petit entier supérieur à $\frac{n}{2}$. L'enveloppe est déterminée en augmentant le sac d'un facteur 1,5. Un exemple de boxplot fonctionnel est représenté sur la Figure 2.36, extraite de Sun et Genton (2011). La méthode du boxplot fonctionnel y est appliquée aux températures de surface de l'océan pacifique relevées mensuellement entre 1951 et 2007. Chaque courbe représente une année et est discrétisée en 12 points. La courbe noire est la médiane, la surface en violet représente le sac. La zone entre les courbes bleues représente l'enveloppe. Les courbes extrêmes sont représentées en pointillés rouges.

2.4.4 Extension proposée du HDR boxplot

Comme il a été vu dans les sections 2.4.1 et 2.4.2, le bagplot fonctionnel n'est pas adapté au cas des données multimodales, alors que le HDR boxplot est capable de distinguer les différents modes des données. Il est donc en général préférable d'utiliser ce dernier plutôt que le bagplot fonctionnel. De plus,

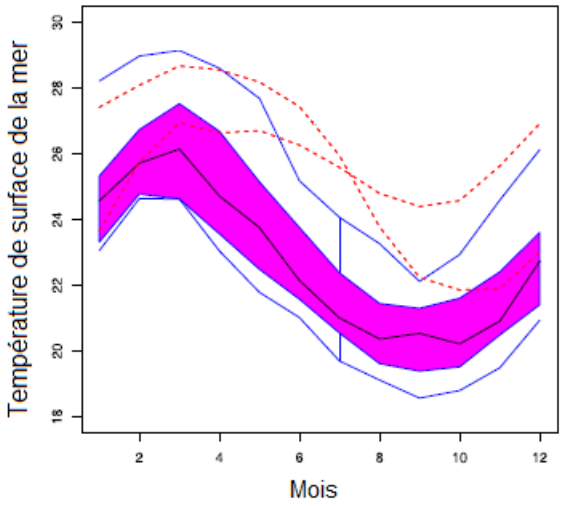

FIGURE 2.36 – Boxplot fonctionnel pour les températures de surface de l'océan pacifique sur une année, extrait de Sun et Genton (2011).

le HDR boxplot et le boxplot fonctionnel ont montré des performances équivalentes dans de nombreux cas d'application (Sun et Genton, 2011; Popelin et Iooss, 2013). Par ailleurs, puisque le HDR boxplot utilise une méthode d'estimation proche de celle proposée dans ce chapitre, on propose d'adapter cette méthode en s'appuyant sur la quantification des incertitudes proposée dans les sections 2.2 et 2.3.

Hyndman et Shang (2010) se limitent dans la définition du HDR boxplot à une décomposition sur les deux premières composantes de l'analyse en composantes principales. Plusieurs améliorations peuvent être proposées pour cette méthode. Tout d'abord, l'utilisation d'autres bases de décompositions, comme la décomposition PLS simultanée étudiée dans la section 2.2, peut se révéler plus appropriée à certaines données, comme illustré dans certains cas-tests traités ici. En effet, si les fonctions étudiées dépendent d'une covariable alors la décomposition basée sur la régression PLS peut permettre de conserver et de visualiser les caractéristiques des fonctions les plus liées à cette covariable. De plus, l'utilisation d'une décomposition simultanée peut permettre de visualiser conjointement plusieurs variables aléatoires fonctionnelles. Ainsi, les courbes médianes ou extrêmes déterminées par la méthode de visualisation sur les coefficients de la décomposition seront des médianes ou des valeurs extrêmes simultanément sur toutes les variables aléatoires visualisées. De plus, il peut être intéressant d'utiliser une base de décomposition de plus de deux fonctions, surtout si la variance expliquée par les deux premières composantes est assez faible. Cependant, l'estimateur à noyau utilisé par Hyndman et Shang (2010) pour estimer la densité des coefficients n'est applicable qu'en petite dimension : l'estimation de la largeur de bande devient coûteuse et difficile dès que la dimension dépasse 6 ou 7. Pour étendre la méthode HDR boxplot avec plus de composantes, on propose d'utiliser l'estimation de la densité par mélange de gaussiennes, présentée en section 2.3.

Ainsi, la nouvelle méthode de visualisation de variables fonctionnelles dépendantes proposée ici consiste donc à décomposer simultanément les variables fonctionnelles sur une base ACP simultanée ou PLS simultanée selon que ces variables sont liées ou non à une covariable. La densité des coefficients issus de cette décomposition est ensuite modélisée par un mélange de gaussiennes. Ensuite, comme dans la méthode de HDR boxplot de Hyndman et Shang (2010), les régions de plus haute densité à 50% et 95%, la courbe médiane et les courbes extrêmes sont déterminées à partir de la densité estimée. Comme un grand nombre de coefficients peut être retenu, seule la version fonctionnelle du HDR boxplot est représentée (pas de représentation multivariée des coefficients de la décomposition). Cette méthode de visualisation sera appelée, par la suite, **HDR boxplot estimé par mélange de gaussiennes**.

2.4.5 Application aux données du choc thermique pressurisé

Les trois méthodes de visualisation existantes, présentées dans les sections 2.4.1, 2.4.2 et 2.4.3, sont tout d'abord appliquées à l'échantillon de 1000 courbes de température qui sont illustrées sur la Figure 1.4 dans l'introduction. Les résultats obtenus sont illustrés par la Figure 2.37. Les trois méthodes produisent des résultats relativement différents. En particulier, les valeurs extrêmes choisies par chaque méthode ne sont pas toutes les mêmes. De plus, même si elles sont proches, les trois médianes estimées présentent des différences. Les médianes des bagplot et HDR boxplot sont assez proches, alors que celle du boxplot fonctionnel décroît plus rapidement autour de 2000 secondes. Ceci est confirmé par le fait que les médianes des représentations multivariées (étoiles rouges) sont très proches pour les deux méthodes. Ceci peut s'expliquer par le fait que ces deux méthodes utilisent la décomposition ACP des fonctions contrairement au boxplot fonctionnel.

Cependant, ces méthodes ne peuvent pas prendre en compte la dépendance entre les données fonctionnelles issues des trois variables de température, pression et coefficient d'échange thermique. C'est pourquoi, on utilise le HDR boxplot estimé par mélange de gaussiennes pour visualiser les trois transitoires du cas-test PTS. On rappelle que la densité de probabilité de ces trois transitoires a été caractérisée par une décomposition sur une base SPLS de 10 composantes combinée à une modélisation de la densité de probabilité des 10 coefficients par un mélange de trois gaussiennes à matrices de covariance creuses, *via* l'algorithme sEM2.3. La part de variance expliquée par cette décomposition est de l'ordre de 80%, alors qu'en prenant une décomposition ACPS de deux composantes, comme proposé dans la méthode de Hyndman et Shang (2010), la variance expliquée est d'un peu moins de 70%. A partir de la densité estimée, on peut déterminer les régions de plus haute densité à 50% et 95%. Comme dans le HDR boxplot introduit par Hyndman et Shang (2010), la courbe médiane correspondant aux coefficients de plus haute densité, ainsi que les enveloppes fonctionnelles contenant les courbes correspondant aux coefficients contenus dans les régions de plus haute densité à 50 et 95% peuvent être représentées. On représente le graphique ainsi obtenu pour chacun des trois transitoires dans la Figure 2.38. Il est composé des éléments suivants :

- la médiane du triplet de variables fonctionnelles (courbe noire),
- l'enveloppe des 50% des courbes les plus probables (zone gris foncé),
- l'enveloppe des 95% des courbes les plus probables (zone gris clair),
- les courbes extrêmes (courbes de couleur).

La médiane de la température est proche des médianes estimées par le bagplot fonctionnel et le HDR boxplot fonctionnel classique (cf. Figure 2.37). A l'inverse, les courbes extrêmes sont assez différentes : sur le HDR boxplot estimé par mélange de gaussiennes, les courbes de température extrêmes sont situées majoritairement dans la partie basse de l'enveloppe fonctionnelle, alors que sur le bagplot fonctionnel et le HDR boxplot, les courbes extrêmes sont pour la plupart concentrées au-dessus de la médiane. Ainsi, les triplets de courbes extrêmes contiennent des courbes de température plutôt basses, alors que les courbes de température extrêmes, considérées seules, sont plus élevées. L'extension proposée du HDR boxplot développé par Hyndman et Shang (2010) permet donc d'une part de visualiser conjointement des variables aléatoires fonctionnelles en tenant compte de leur dépendance et, d'autre part, de gagner en précision en décomposant ces variables sur des bases fonctionnelles plus grandes.

2.4.6 Application aux données de la rupture Liposo

L'adaptation du HDR boxplot proposée dans la section 2.4.4 est appliquée ici aux 200 réalisations disponibles des trois variables fonctionnelles étudiées dans le cas Liposo : le débit, la pression et la température. La quantification des incertitudes de ces variables fonctionnelles dépendantes est décrite dans les sections 2.2.7.3 et 2.3.6.3. Rappelons qu'une décomposition sur une base ACP simultanée de 10 composantes (après translation des courbes de puissance) a été retenue et que la densité de probabilité des coefficients issus de cette décomposition est estimée par un mélange de 3 gaussiennes par l'algorithme EM. La décomposition sur une base de 10 fonctions explique 99,6% de la variance des variables fonctionnelles, alors qu'avec une base de 2 fonctions construite par ACPS, la part de variance expliquée n'est que de 76%. A partir de cette modélisation, les régions de plus haute densité à 50% et 95% sont déterminées,

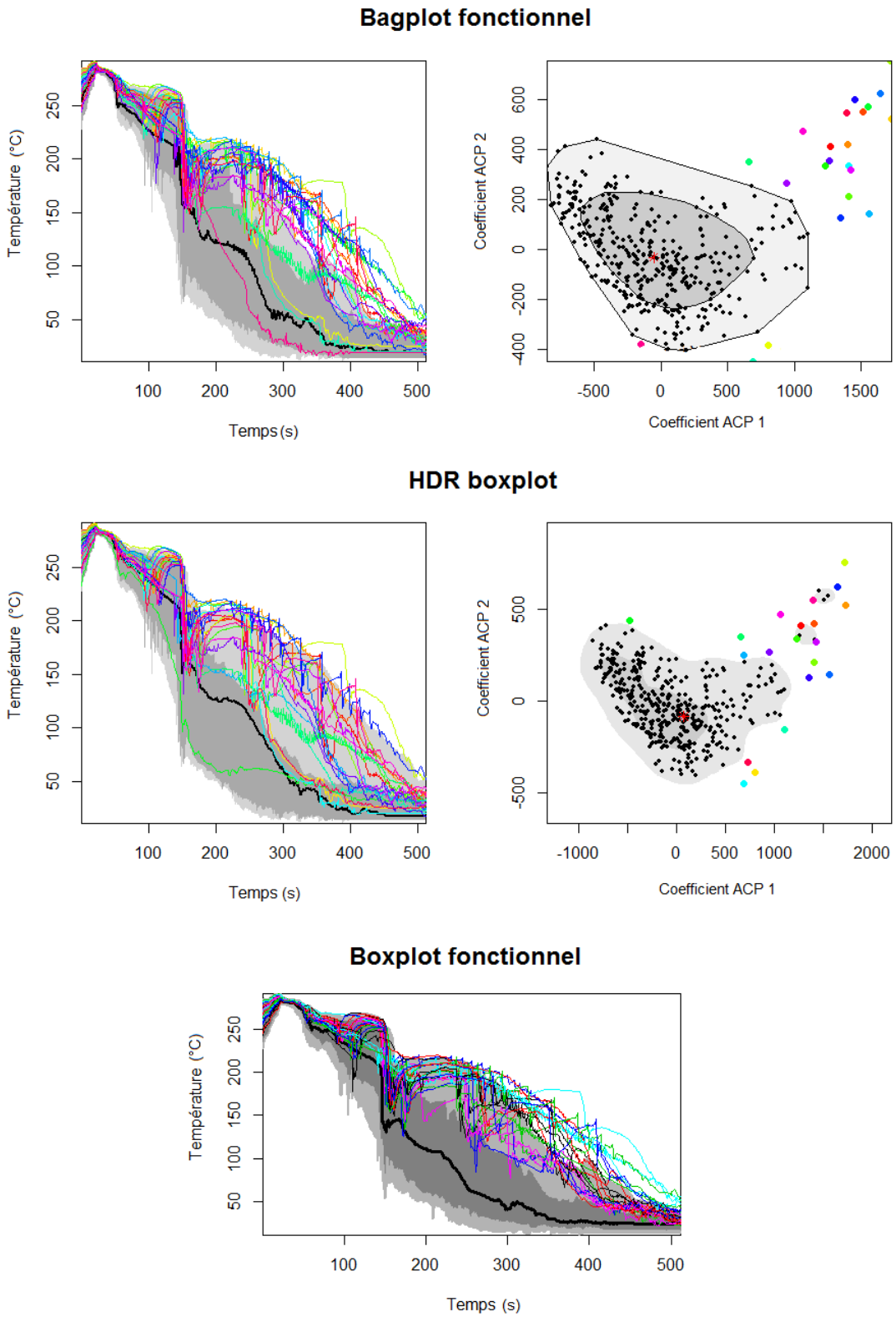

FIGURE 2.37 – Cas PTS : bagplot fonctionnel, HDR boxplot et boxplot fonctionnel de l'échantillon de transitoires de température.

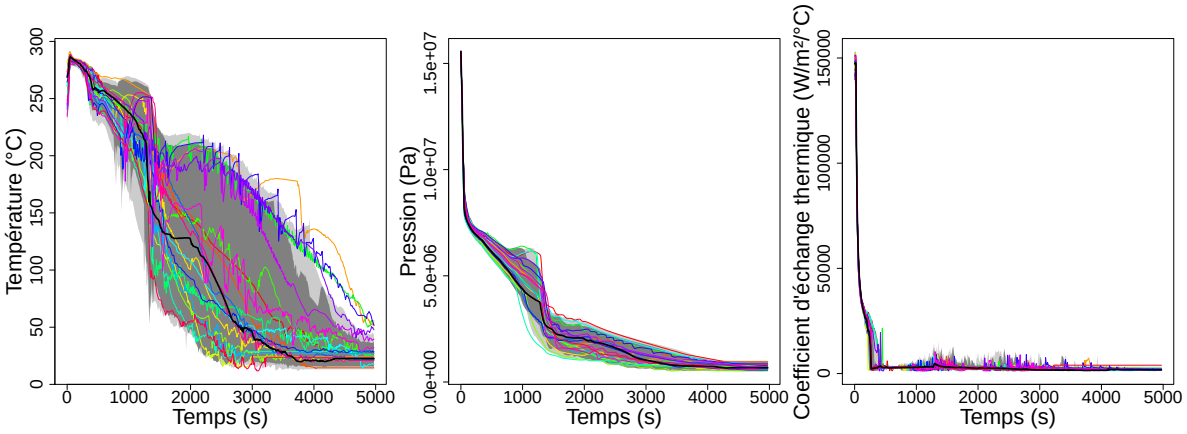

FIGURE 2.38 – Cas PTS : HDR boxplot estimé par mélange de gaussiennes pour les 10 premières composantes de la PLS simultanée des trois transitoires de température, pression et coefficient d'échange.

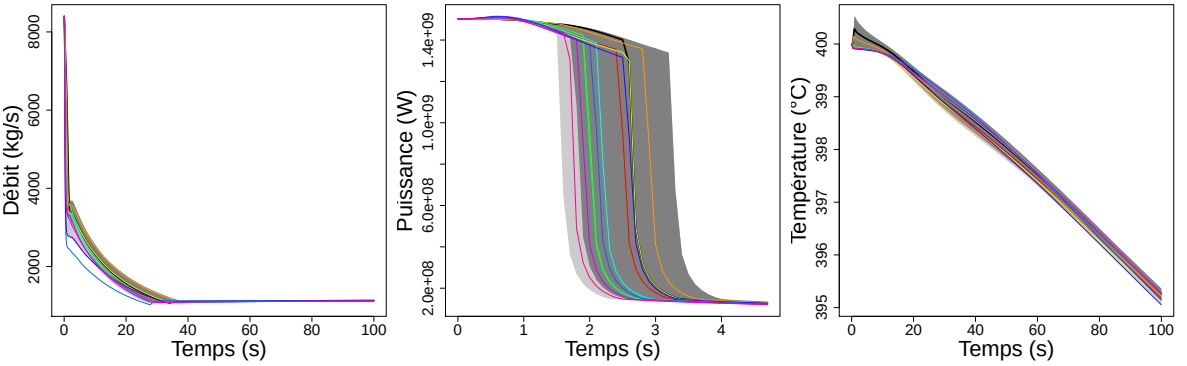

FIGURE 2.39 – Cas LiPoSo : HDR boxplot estimé par mélange de gaussiennes avec les 10 premières composantes de l'ACP simultanée des trois transitoires du cas-test Liposo (débit, pression avec un zoom sur l'intervalle $[0; 5]$ et température).

et l'on obtient le HDR boxplot illustré par la Figure 2.39. Les courbes noires représentent la médiane du groupe de variables fonctionnelles, c'est-à-dire le triplet de courbes correspondant aux coefficients de plus haute densité de probabilité. Les enveloppes gris foncé et gris clair contiennent les courbes correspondant aux coefficients appartenant respectivement aux régions de plus haute densité à 50% et 95%. Les courbes de couleur font partie des triplets de courbes extrêmes. Elles correspondent aux points ayant les plus faibles densités. Pour la pression, au centre, les courbes extrêmes sont pour la plupart en-dessous de la médiane. On peut remarquer que, pour ces courbes, la décroissance plus précoce de la pression est accompagnée par une décroissance plus rapide du débit.

2.5 Synthèse

Une étape essentielle du traitement des incertitudes de codes de calcul est l'identification et la quantification des sources d'incertitudes en entrée du code. Pour ce faire, dans le cadre d'une approche probabiliste, les paramètres d'entrée incertains du code sont modélisés par des variables aléatoires. On s'intéresse plus particulièrement dans ce chapitre à la quantification des incertitudes de plusieurs variables d'entrée fonctionnelles et dépendantes entre elles. De plus, comme on s'inscrit par la suite dans

une démarche d'analyse de sensibilité de la sortie, la quantification des incertitudes doit modéliser avec précision les caractéristiques des variables d'entrée qui expliquent le mieux la sortie, appelée covariable.

Dans ce chapitre, un état de l'art des méthodes de quantification des incertitudes d'une ou de plusieurs variables fonctionnelles a été présenté. Les méthodes décrites sont toutes constituées de deux grandes étapes.

Etape 1 : décomposition fonctionnelle. La variable fonctionnelle est décomposée sur une base de fonctions, et est ensuite représentée par ses coefficients sur la base.

Etape 2 : caractérisation probabiliste des coefficients. Dans cette étape, la distribution de probabilité des coefficients est modélisée.

Après une revue des méthodes utilisées pour réaliser ces deux étapes, une méthodologie complète de quantification des incertitudes a été proposée en améliorant et en adaptant les méthodes existantes, afin de traiter le cas de plusieurs variables fonctionnelles dépendantes entre elles et corrélées à une covariable. De plus, nous avons proposé plusieurs critères pour évaluer l'efficacité des méthodes utilisées à chacune des étapes ainsi que l'efficacité de la méthodologie globale. Ces critères permettent aussi d'ajuster certains paramètres des méthodes.

Deux grandes familles de méthodes ont été présentées pour l'étape 1 de décomposition fonctionnelle. La première, dans laquelle les fonctions de la base de décomposition sont sélectionnées en fonction des données parmi des fonctions fixées *a priori*, regroupe les bases de spline, d'ondelettes ou de paquets d'ondelettes. Dans la deuxième famille de méthodes, les fonctions des bases de décomposition sont construites à partir des données. Dans le cas de l'Analyse en Composantes Principales (ACP), elles sont construites pour minimiser l'erreur quadratique de projection. Dans la décomposition *Partial Least Squares* (PLS), elles sont choisies de manière à réaliser un compromis entre l'approximation de la variable projetée sur la base de décomposition et l'approximation de son lien avec une covariable. Ces deux dernières méthodes ont obtenu de meilleurs résultats que la première famille de décompositions sur les exemples étudiés. Des adaptations des méthodes PLS et ACP ont été proposées dans cette thèse pour prendre en compte la dépendance entre les variables fonctionnelles. Ces décompositions PLS et ACP simultanées (resp. SPLS et ACPS) permettent de décomposer simultanément m variables fonctionnelles sur une base de m-uplets de fonctions. Parmi les méthodes présentées, la SPLS est celle qui répond le mieux aux objectifs d'approximation des variables fonctionnelles et de leur lien avec une covariable. Cependant, dans le cas où on ne considère pas de covariable, on préconise d'utiliser l'ACPS qui permet une bonne approximation des données. Les méthodes de décomposition fonctionnelle présentées dans ce chapitre ont été appliquées et comparées sur un exemple analytique et sur les deux cas industriels décrits dans l'introduction. Les tests effectués ont montré l'efficacité de la SPLS vis-à-vis des différents critères proposés. De plus, nous avons montré sur les exemples traités que les méthodes de décomposition simultanée donnaient à taille de base égale une meilleure approximation que des décompositions séparées quand les variables approchées sont dépendantes.

Concernant la deuxième étape de la quantification, plusieurs méthodes d'estimation de la distribution de probabilité de variables multivariées ont été décrites dans ce chapitre. Nous nous sommes concentrés plus particulièrement sur la modélisation par mélange de gaussiennes, puisque cette méthode simple et rapide à mettre en œuvre permet de modéliser la densité de probabilité pour des dimensions relativement élevées (≥ 6). Plusieurs méthodes basées sur l'algorithme *Expectation-Maximization* (EM) ont été proposées pour estimer les paramètres du mélange de gaussiennes (proportions, moyennes et matrices de covariance de chaque classe). Une des limitations de la modélisation par des mélanges de gaussiennes est que le nombre de leurs paramètres augmente rapidement avec la dimension des variables à modéliser. Pour réduire leur nombre de paramètres, deux méthodes ont été étudiées. La première, notée sEM, estime des matrices de covariance dont les inverses sont creuses. L'inverse d'une matrice creuse n'étant pas forcément creuse, nous avons développé une deuxième méthode (sEM2) qui estime directement des matrices de covariance creuses. Celle-ci est basée sur l'ajout d'une pénalisation Lasso sur les matrices de covariance dans l'étape de maximisation de la vraisemblance de l'algorithme EM. Nous avons présenté trois variantes de la méthode sEM2 dans lesquelles les pondérations accordées à chaque élément de la matrice de covariance dans la pénalisation Lasso diffèrent. Ces méthodes d'estimation ont été appliquées

aux trois cas d'étude. Ces applications ont montré que la méthode sEM2. permettait, dans la plupart des cas, de mieux approcher la densité de probabilité des coefficients que les algorithmes EM ou sEM. Pour la méthode sEM2., les deux pénalisations, dans lesquelles les éléments diagonaux des matrices de covariance ne sont pas pénalisés (sEM2.2 et sEM2.3), ont donné de meilleurs résultats que celle qui pénalise de la même manière l'ensemble des éléments des matrices de covariance. En pratique, on conseille donc d'utiliser les méthodes sEM2.2 et sEM2.3.

Ainsi, la méthodologie globale de quantification des incertitudes retenue à l'issue des différentes études réalisées dans ce chapitre est résumée par la Figure 2.40. Pour chaque étape de la méthodologie, sont représentés les choix de modélisation à effectuer en fonction du cas traité (dans les losanges), les méthodes à appliquer (dans les rectangles), et les paramètres à estimer (dans les ellipses).

Enfin, un état de l'art des méthodes de visualisation de données fonctionnelles a été présenté. Nous avons ensuite proposé une adaptation de la méthode *High Density Region* (HDR) boxplot. La méthode HDR boxplot permet de résumer graphiquement un échantillon de courbes par une médiane fonctionnelle, des enveloppes de fonctions les plus probables et des fonctions extrêmes. L'amélioration proposée consiste à remplacer la méthode de quantification des incertitudes utilisée pour construire le HDR boxplot par celle proposée dans ce chapitre. Ce changement permet tout d'abord d'améliorer la qualité de l'étape de quantification et donc la qualité d'estimation des différentes quantités représentées, en augmentant la taille de la base de décomposition. De plus, cette modélisation permet, dans le cas d'un groupe de variables fonctionnelles, d'appliquer l'outil de visualisation simultanément à toutes ces variables. Par exemple, en appliquant cet outil de visualisation à un groupe de variables fonctionnelles, la médiane représentée correspond à la médiane du groupe de variables fonctionnelles et non au vecteur des médianes de chaque variable fonctionnelle. L'intérêt du nouvel outil de visualisation ainsi obtenu a été illustré sur les cas-test du PTS et de la rupture LiPoSo.

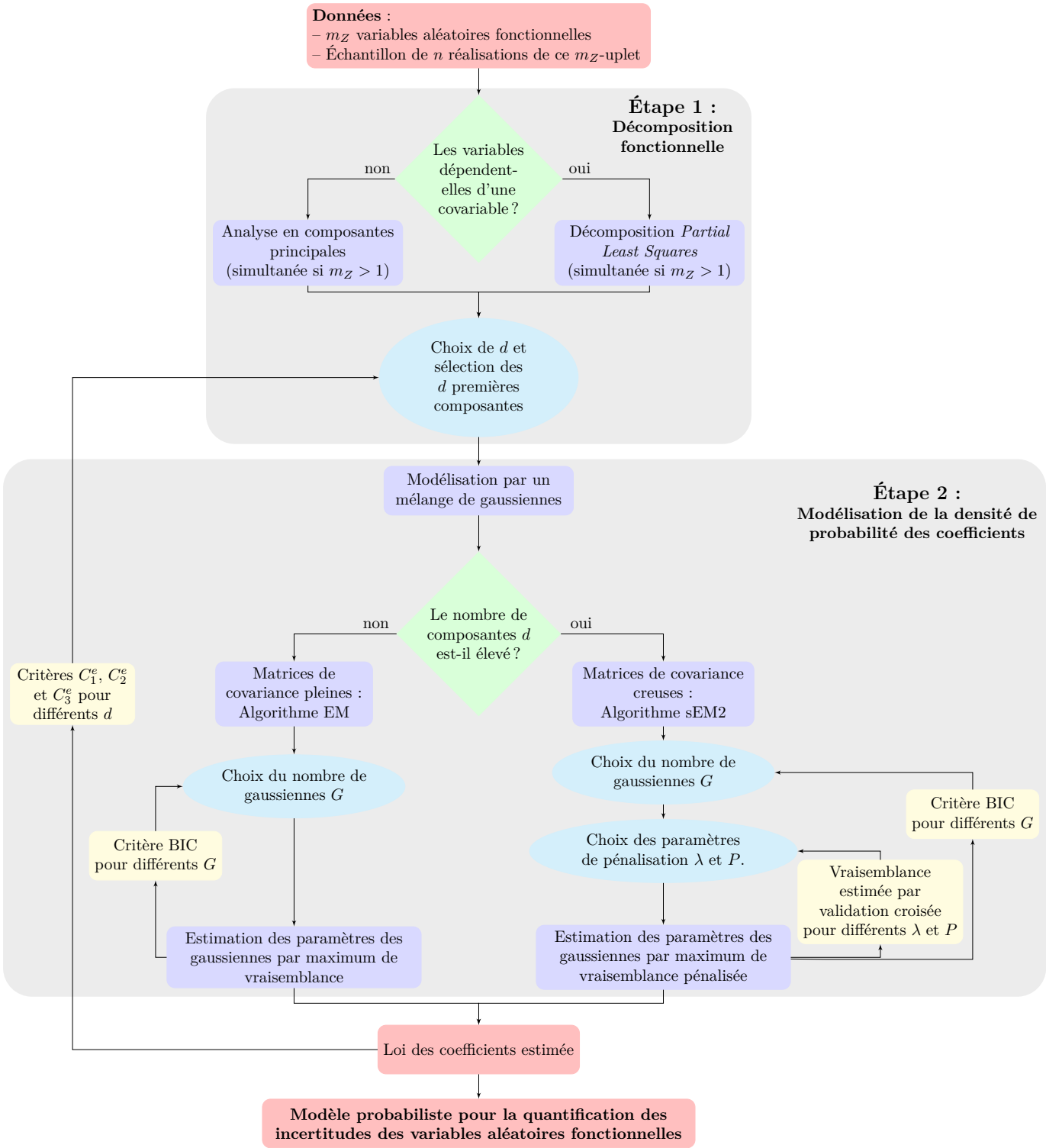

FIGURE 2.40 – Organigramme de la méthode proposée de quantification des incertitudes de variables fonctionnelles dépendantes.

Chapitre 3

Analyse de sensibilité de codes de calcul à entrées fonctionnelles

3.1 Problématiques et objectifs

L'analyse de sensibilité (Saltelli et al., 2000; Helton et al., 2006) est un outil efficace pour identifier comment la variation des paramètres d'entrée d'un code de calcul contribue, qualitativement ou quantitativement, à la variation de la sortie. Ainsi, l'analyse de sensibilité peut aider à valider, simplifier ou mieux comprendre un modèle ainsi qu'à guider les efforts de caractérisation des paramètres d'entrée. De nombreuses revues des méthodes d'analyse de sensibilité (Kleijnen, 1997; Helton et al., 2006; Iooss, 2011) ont été réalisées. Elles distinguent généralement deux catégories de méthode : l'analyse de sensibilité locale et l'analyse de sensibilité globale. La première catégorie étudie l'effet d'une petite perturbation autour de valeurs nominales et utilise souvent, pour cela, les dérivées partielles du code de calcul. La seconde, notée ASG, a pour but de mesurer l'effet des incertitudes des entrées sur l'ensemble de leur domaine de variation. Parmi les nombreuses techniques d'ASG, on s'intéresse plus particulièrement dans ce chapitre à la méthode d'analyse de sensibilité basée sur la variance et introduite par Sobol' (1993). Ces mesures de sensibilité sont appelées indices de Sobol'. En donnant une mesure quantitative de la contribution de l'incertitude des entrées à l'incertitude des sorties du code, ces indices permettent de comparer et d'ordonner les contributions de toutes les entrées et de leurs interactions à l'incertitude de la sortie.

Ce chapitre a pour objectif d'étudier et de proposer des méthodes d'analyse de sensibilité pour les simulateurs numériques utilisés dans les deux cas d'étude définis dans la section 1.2 (cas du choc thermique PTS et de la rupture LiPoSo). Les deux cas étudiés présentent plusieurs des caractéristiques suivantes notées de P_1 à P_3 :

P_1 **:** les entrées du code de calcul sont scalaires et fonctionnelles. De plus, ses entrées fonctionnelles sont possiblement dépendantes,

P_2 **:** les incertitudes liées à ces variables d'entrée fonctionnelles ne sont connues qu'à partir d'un nombre limité de leurs réalisations (elles peuvent être les sorties d'un autre code de calcul, comme c'est le cas dans les applications étudiées ici, ou les résultats de mesures expérimentales),

P_3 **:** le code étudié est coûteux en temps de calcul.

Seul le cas de la rupture LiPoSo possède la caractéristique P_3, alors que les deux autres caractéristiques sont partagées par les deux cas d'étude. **Dans ces deux exemples, les variables fonctionnelles sont toutes dépendantes, et l'objectif de l'analyse de sensibilité est de comparer l'influence du groupe des variables fonctionnelles avec celles des différentes variables scalaires.**

La caractéristique P_1 des codes étudiés a un effet sur l'estimation des indices de Sobol'. En effet, même si la définition des indices de Sobol' s'étend sans difficulté aux variables d'entrée fonctionnelles,

leur estimation peut être plus complexe en pratique. Plusieurs travaux ont été conduits sur ce sujet. Un état de l'art de ceux-ci est réalisé dans la section 3.2.3. De plus, pour prendre en compte la dépendance possible entre les entrées du code de calcul, les indices de Sobol' ont besoin d'être adaptés. Un état de l'art des méthodes développées pour les adapter au cas dépendant est proposé dans la section 3.2.4.

L'analyse de sensibilité globale requiert la connaissance des distributions de probabilité des entrées du code pour être en mesure de générer des réalisations de celles-ci. Cependant, à cause de la caractéristique P_2 du problème, la distribution de probabilité des entrées fonctionnelles du code doit être estimée au préalable. Pour cela, on propose d'utiliser les méthodologies de quantification des incertitudes de variables fonctionnelles proposées dans le chapitre 2.

Enfin, la caractéristique P_3, présente dans le cas d'étude LiPoSo, est une limitation importante, puisque les méthodes d'estimation des indices de Sobol' nécessitent un grand nombre d'appels au code (quelques milliers). Pour réduire ce nombre d'appels et donc le coût de la méthode, une solution peut être de construire un modèle de substitution, ou métamodèle, ayant un coût de calcul négligeable pour approcher le code. L'objectif est ici d'estimer le modèle de prédiction le plus proche possible du code sur l'intégralité du domaine de variation des entrées. Ce métamodèle est ensuite utilisé pour réaliser les études d'analyse de sensibilité. Comme décrit dans le chapitre 4, il existe plusieurs approches pour construire des métamodèles à entrées fonctionnelles, basées, par exemple, sur la décomposition fonctionnelle ou l'utilisation de variables incontrôlables. Pour apprendre un métamodèle approchant le code de calcul étudié, une base d'apprentissage de n évaluations du code (composée des entrées du code et de ses sorties correspondantes) est nécessaire. Comme l'échantillon disponible de réalisations des variables aléatoires fonctionnelles est probabilisé, certaines régions du domaine de variation des entrées sont mieux échantillonnées que les autres, et un métamodèle estimé à partir de ce plan d'expériences représenterait mieux les zones du domaine dans lesquelles le plan possède une plus haute densité de points. Un plan d'apprentissage échantillonné uniformément sur le domaine de variation des entrées serait mieux adapté pour construire un métamodèle représentatif du code de calcul sur l'ensemble du domaine de variation de ses entrées, car tout le domaine est représenté de manière équivalente dans l'échantillon (Santner et al., 2003). On propose donc de construire un plan d'apprentissage uniforme avec des propriétés optimales sur l'espace des entrées fonctionnelles et scalaires du code. Pour les variables scalaires, les plans hypercubes latins (LHS) sont une classe de plans largement utilisée (McKay et al., 1979). La construction de ces plans peut aussi être optimisée (Morris et Mitchell, 1995) selon un critère géométrique ou en évaluant l'uniformité de la répartition des points. Pour cela, la discrépance (centrée ou wrap-around, par exemple, cf. Jin et al. 2005), les critères de distance (maximin, minimax, cf. Johnson et al. 1990) peuvent être utilisés. Récemment, des travaux ont été conduits pour étendre ces techniques d'échantillonnage au cadre fonctionnel. Pebesma et Heuvelink (1999) ont proposé une méthode pour étendre les plans LHS à l'échantillonnage de processus gaussiens. Une adaptation du critère maximin a été proposée par Morris (2012) dans le cadre d'entrées et de sorties fonctionnelles. Enfin, Muehlenstaedt et al. (2014) ont proposé une méthode pour échantillonner des variables scalaires et fonctionnelles, quand les variables fonctionnelles sont bornées. Pour cela, une base de splines est utilisée pour approcher chaque variable fonctionnelle et l'espace des coefficients est échantillonné avec un LHS optimisé selon une variante régularisée du critère maximin. Cet échantillon est ensuite combiné avec un plan LHS construit pour les variables scalaires. Une extension de cette méthode aux deux cas étudiés dans ce chapitre est proposée dans la section 3.3.1.

La méthodologie proposée pour réaliser l'analyse de sensibilité est composée des étapes suivantes :

Etape 1 Modélisation des distributions de probabilité des variables d'entrée fonctionnelles selon la méthode proposée dans le chapitre 2. Les variables sont décomposées sur une base fonctionnelle puis la distribution de probabilité de leurs coefficients est estimée par un mélange de gaussiennes.

Etape 2 Réalisation de l'analyse de sensibilité à partir de la densité de probabilité estimée à l'étape 1 avec l'une des méthodes suivantes, suivant le coût de calcul du code :

Option 2.a Si le code est rapide (cas du PTS), l'analyse de sensibilité est réalisée directement à partir du code.

Option 2.b Si le code est coûteux (cas de la rupture LiPoSo), un échantillon uniforme des variables scalaires et fonctionnelles en entrée du code est généré selon la méthode proposée dans ce chapitre, puis le code est évalué sur les points de cet échantillon. Un métamodèle est ensuite appris en utilisant comme échantillon d'apprentissage l'échantillon uniforme généré et les sorties du code correspondantes. Enfin, l'analyse de sensibilité est réalisée à partir du métamodèle ainsi construit.

Dans la section 3.2, les indices de sensibilité de Sobol' (Sobol', 1993) sont définis, et plusieurs méthodes pour les estimer sont présentées. Les méthodes existantes pour les adapter aux cas d'entrées fonctionnelles et possiblement dépendantes sont ensuite détaillées. La section 3.3 s'intéresse à la construction du métamodèle pour approcher un code de calcul coûteux. Les méthodes d'échantillonnage de variables fonctionnelles et scalaires pour la construction de la base d'apprentissage y sont décrites dans un premier temps, puis la construction du métamodèle est ensuite discutée. Les méthodologies proposées pour l'analyse de sensibilité des deux types de code étudiés sont appliquées à l'exemple analytique défini dans le chapitre 2 et aux deux cas d'étude présentés dans l'introduction. Enfin, une synthèse des méthodes présentées ainsi que des résultats obtenus sur les cas-tests est réalisée dans la section 3.5.

3.2 Analyse de sensibilité globale

On présente dans cette section la définition des indices de sensibilité de Sobol' ainsi que plusieurs méthodes d'estimation. Des états de l'art des méthodes d'analyse de sensibilité dans le cas de paramètres d'entrée fonctionnels et dans celui de paramètres dépendants sont présentés respectivement dans les section 3.2.3 et 3.2.4.

Soit $(\Omega, \mathcal{F}, P)$ un espace de probabilité et $X = (X_1, \ldots, X_{m_X})$ un vecteur de variables aléatoires indépendantes identiquement distribuées selon une loi uniforme sur $[0;1]$. Une distribution uniforme est choisie pour les variables d'entrée afin de simplifier les formules présentées dans cette section, mais l'ensemble des méthodes présentées est généralisable à des variables de distribution quelconque. Dans un premier temps, on étudie la fonction suivante :

$$\mathcal{M}: \left\{ \begin{array}{lll} [0;1]^{m_X} & \to & \mathbb{R} \\ X & \mapsto & Y = \mathcal{M}(X) \end{array} \right.$$

3.2.1 Indices de Sobol'

Les indices de Sobol' (Sobol' 1993) sont basés sur la décomposition fonctionnelle ANOVA (analyse de la variance) de $\mathcal{M}$. Cette décomposition, introduite par Hoeffding (1948) puis reprise notamment dans Efron et Stein (1981), est la somme de 2^p fonctions d'arités croissantes :

$$\begin{aligned} \mathcal{M}(x) &= \mathcal{M}_\emptyset + \sum_{i=1}^{m_X} \mathcal{M}_i(x_i) + \sum_{1 \leq i < j \leq m_X} \mathcal{M}_{i,j}(x_i, x_j) + \cdots + \mathcal{M}_{1,\ldots,m_X}(x) \qquad (3.1) \\ &= \sum_{u \subset \{1,\ldots,m_X\}} \mathcal{M}_u(x_u). \end{aligned}$$

$\mathcal{M}_\emptyset$ est une constante. Les fonctions $\mathcal{M}_i : [0;1] \to \mathbb{R}$, $i \in \{1, \ldots, m_X\}$ sont appelées les effets principaux ou d'ordre 1, les fonctions $\mathcal{M}_{i,j} : [0;1]^2 \to \mathbb{R}$, pour $i, j \in \{1, \ldots, m_X\}$ et $i \neq j$, sont les effets d'ordre 2, etc. Cette décomposition n'étant en général pas unique, il est nécessaire d'imposer les contraintes suivantes pour obtenir l'unicité :

$$\int_{[0;1]} \mathcal{M}_u(x_u) \mathrm{d}x_i = 0, \forall i \in u, \forall u \subset \{1, \ldots, m_X\}. \qquad (3.2)$$

La décomposition ANOVA définie dans l'équation (3.1) est appliquée avec un vecteur de variables aléatoires indépendantes et uniformes $X = (X_1, \ldots, X_{m_X})$ et avec les contraintes (3.2). En prenant la

variance de l'équation obtenue et en tenant compte de l'indépendance entre les variables $X_1, \dots X_{m_X}$, la décomposition de la variance de $\mathcal{M}(X)$ suivante peut être obtenue :

$$\operatorname{Var} Y = \sum_{i=1}^{m_X} \operatorname{Var}\left(\mathcal{M}_i(X_i)\right) + \sum_{1 \le i < j \le m_X} \operatorname{Var}\left(\mathcal{M}_{i,j}(X_i, X_j)\right) + \cdots + \operatorname{Var}\left(\mathcal{M}_{1,\dots,m_X}(X)\right), \tag{3.3}$$

où la variable aléatoire Y est définie comme $Y = \mathcal{M}(X)$.

Définition 3.1 (Indice de Sobol')**.** *L'indice de Sobol' d'ordre $|u|$, $|u|$ étant le cardinal de u, des variables X_u avec $u \subset \{1, \dots, m_X\}$ est défini ainsi :*

$$\begin{aligned} S_u &= \frac{\operatorname{Var}\left(\mathcal{M}_u(X_u)\right)}{\operatorname{Var} Y} \\ S_u &= \frac{\operatorname{Var}\left(\mathbb{E}\left(Y|X_u\right)\right) + \sum_{v \subset u} (-1)^{|u|-|v|} \operatorname{Var}\left(\mathbb{E}(Y|X_v)\right)}{\operatorname{Var} Y}, \end{aligned} \tag{3.4}$$

En particulier, en choisissant $u = \{i\}$, avec $i \in \{1, \dots, m_X\}$, dans la définition précédente, on peut définir l'indice de Sobol' d'ordre 1 de X_i qui s'écrit :

$$S_i = \frac{\operatorname{Var}\left(\mathbb{E}\left(Y|X_i\right)\right)}{\operatorname{Var} Y}, \tag{3.5}$$

L'indice S_u mesure la part de la variance de la sortie qui pourrait être réduite si les entrées X_u étaient fixées. Plus l'indice de Sobol' S_u est proche de 1, plus l'incertitude sur X_u contribue à l'incertitude sur la sortie du modèle. A l'inverse, un indice proche de 0 indique une très faible influence sur la sortie. Ces indices ont plusieurs propriétés intéressantes découlant de la définition des indices (3.4) et de la décomposition de la variance (3.3).

Proposition 3.2. *Soit u un sous-ensemble non-vide de $\{1, \dots, m_X\}$. L'indice de Sobol' S_u vérifie les deux propriétés suivantes :*

$$\begin{gathered} 0 \le S_u \le 1, \\ \sum_{u \subset \{1,\dots,m_X\}, u \neq \emptyset} S_u = 1. \end{gathered}$$

Le nombre d'indices de Sobol' d'ordres 1 à m_X est égal à $2^{m_X} - 1$, si bien qu'en pratique, le calcul de tous les indices n'est pas envisageable quand m_X est élevé. Pour pallier cette difficulté, Homma et Saltelli (1996) proposent d'étudier l'effet total d'une variable, c'est-à-dire l'effet de cette variable et de toutes ses interactions avec les autres variables d'entrée.

Définition 3.3. *Soit $i \in \{1, \dots, m_X\}$. L'indice de Sobol' total d'une variable X_i est défini ainsi :*

$$S_{T_i} = \sum_{u \subset \{1,\dots,m_X\}, i \in u} S_u. \tag{3.6}$$

Les définitions des indices de Sobol' (3.4) et des indices totaux (3.6) ont été présentées dans le cas de variables aléatoires indépendantes et uniformes sur $[0; 1]$ mais sont généralisables sans modification pour tout type de variables aléatoires indépendantes $X_1, \dots, X_{m_X}$. Cela inclut le cas où ces variables sont des processus stochastiques.

3.2.2 Estimation des indices de sensibilité

Dans cette section, on présente deux types d'approche pour estimer les indices de Sobol' : les méthodes basées sur l'estimation Monte Carlo des numérateur et dénominateur des indices de Sobol' et celles reposant sur la décomposition de Fourier du code $\mathcal{M}$.

3.2.2.1 Méthodes Monte Carlo

Soit $D = \left\{\left(X_1^j, \ldots, X_{m_X}^j\right), 1 \leq j \leq n\right\}$ un plan d'expériences de n réalisations de X. Plusieurs estimateurs Monte Carlo (Sobol' 1993; Tarantola et al. 2006b; Monod et al. 2006; Sobol et al. 2007 ; etc.) ont été développés pour calculer les indices de Sobol' du premier ordre. Ces estimateurs découlent du fait que les indices de Sobol' peuvent être réécrits sous la forme suivante :

$$S_i = \frac{\mathrm{Cov}\left(Y, Y_i\right)}{\mathrm{Var}\, Y},$$

où $Y = \left(Y^1, \ldots, Y^n\right)$ est le vecteur des évaluations de $\mathcal{M}$ sur les points de D et $Y_i = \left(Y_i^1, \ldots, Y_i^n\right)$ est le vecteur d'évaluations de $\mathcal{M}$ sur les points du plan D_i ($1 \leq i \leq m_X$), créé en ré-échantillonnant toutes les variables à l'exception de X_i dans le plan D. Parmi ces estimateurs, on peut citer l'estimateur asymptotiquement efficace étudié dans Monod et al. (2006) et Janon et al. (2014) :

$$\hat{S}_i = \frac{\frac{1}{n}\sum_{j=1}^{n} Y^j Y_i^j - \left(\frac{1}{n}\sum_{j=1}^{n} \frac{Y^j + Y_i^j}{2}\right)^2}{\frac{1}{n}\sum_{j=1}^{n} \left(\frac{Y^j + Y_i^j}{2}\right)^2 - \left(\frac{1}{n}\sum_{j=1}^{n} \frac{Y^j + Y_i^j}{2}\right)^2} \tag{3.7}$$

Avec l'estimateur défini dans l'équation (3.7), il est possible de calculer tous les indices de Sobol' du premier ordre avec un coût de $(m_X + 1)n$ appels au code, puisque le code doit être évalué sur les plans $D_1, \ldots, D_{m_X}$ et D. De manière similaire, il est possible de définir des estimateurs Monte Carlo pour les indices de Sobol' d'ordres supérieurs ou pour les indices totaux (Saltelli, 2002; Sobol et al., 2007).

Pour réduire le nombre d'évaluations du code nécessaires au calcul des indices du premier ordre, Mara et Rakoto Joseph (2008) ont proposé d'introduire des plans d'expériences répliqués. Une étude théorique de cette méthode et son extension au calcul des indices du second ordre ont été proposées par Tissot et Prieur (2015). L'idée de cette méthode est d'utiliser les estimateurs Monte Carlo classiques mais de modifier la manière de construire les plans d'expériences $D_1, \ldots, D_{m_X}$ et D afin de diminuer le nombre d'évaluations nécessaires. Pour ce faire, deux plans d'expériences D et D' en dimension m_X et composés de n simulations sont générés sur l'espace de variation des entrées :

$$\begin{aligned} D &= \left\{\left(X_1^j, \ldots, X_{m_X}^j\right), 1 \leq j \leq n\right\} \\ D' &= \left\{\left(X_1^{\pi_1(j)}, \ldots, X_{m_X}^{\pi_{m_X}(j)}\right), 1 \leq j \leq n\right\}, \end{aligned}$$

où $\pi_1, \ldots, \pi_{m_X}$ sont des permutations de $\{1, \ldots, n\}$. Pour estimer l'indice de la variable X_i ($1 \leq i \leq m_X$), les plans D_i sont définis de la manière suivante :

$$\begin{aligned} D_i &= \left\{\left(X_1^{\pi_1 \circ \pi_i^{-1}(j)}, \ldots, X_d^{\pi_{m_X} \circ \pi_i^{-1}(j)}\right), 1 \leq j \leq n\right\} \\ &= \left\{\left(X_1^{\pi_1 \circ \pi_i^{-1}(j)}, \ldots, X_i^j, \ldots, X_d^{\pi_{m_X} \circ \pi_i^{-1}(j)}\right), 1 \leq j \leq n\right\}. \end{aligned}$$

On note Y_D^j (respectivement $Y_{D_i}^j$) la sortie de $\mathcal{M}$ correspondant au point j du plan D (respectivement D_i). Les indices de Sobol' peuvent être estimés en utilisant l'estimateur défini dans l'équation (3.7). L'estimateur proposé par Tissot et Prieur (2015) est donc le suivant :

$$\hat{S}_i = \frac{\frac{1}{n}\sum_{j=1}^{n} Y_D^j Y_{D_i}^j - \left(\frac{1}{n}\sum_{j=1}^{n} \frac{Y_D^j + Y_{D_i}^j}{2}\right)^2}{\frac{1}{n}\sum_{j=1}^{n} \left(\frac{Y_D^j + Y_{D_i}^j}{2}\right)^2 - \left(\frac{1}{n}\sum_{j=1}^{n} \frac{Y_D^j + Y_{D_i}^j}{2}\right)^2}$$

Par construction, les plans d'expériences D_i $(1 \leq i \leq m_X)$ contiennent les mêmes points que le plan D'. Par conséquent, pour calculer tous les indices de Sobol' du premier ordre, il suffit d'évaluer $\mathcal{M}$ sur D et D', et ainsi seules $2n$ évaluations sont nécessaires pour leur estimation. Ce nombre est indépendant du nombre d'entrées du code, ce qui permet de considérablement réduire le nombre d'appels au code.

3.2.2.2 Méthodes FAST

La méthode FAST (Fourier Amplitude Sensitivity Test), introduite par Cukier et al. (1975), repose sur la décomposition du code $\mathcal{M}$ sur une base de Fourier. Cette décomposition spectrale s'écrit comme suit :

$$\mathcal{M}(\mathbf{x}) = \sum_{\mathbf{k} \in \mathbb{Z}^m} c_{\mathbf{k}} \varphi_{\mathbf{k}}(\mathbf{x}),$$

où $(\varphi_{\mathbf{k}})_{\mathbf{k} \in \mathbb{Z}^m}$ est une base de Fourier et les $c_{\mathbf{k}}$ sont les coefficients de $\mathcal{M}$ définis par l'équation suivante :

$$c_{\mathbf{k}} = \int_{[0;1]^{m_X}} \mathcal{M}(\mathbf{x}) \exp\left(-2i\pi \mathbf{k}.\mathbf{x}\right) \mathrm{d}\mathbf{x}. \tag{3.8}$$

Comme les indices de Sobol' peuvent s'écrire comme des fonctions de ces coefficients de Fourier, Cukier et al. (1975) proposent de calculer ces coefficients afin d'estimer les indices de Sobol'. Une égalité permettant de simplifier l'équation (3.8) en remplaçant l'intégrale en dimension m par une intégrale unidimensionnelle a été démontrée par Weyl (1916). Cette égalité s'écrit de la manière suivante pour toute fonction g bornée et intégrable sur $[0;1]^{m_X}$:

$$\int_{[0;1]^{m_X}} g(\mathbf{x}) \mathrm{d}\mathbf{x} = \lim_{T \to +\infty} \frac{1}{2T} \int_{-T}^{T} g(x_1(t), \ldots, x_{m_X}(t)) \mathrm{d}t, \tag{3.9}$$

où $x_i(t) = \omega_i t - \lfloor \omega_i t \rfloor, \forall i \in \{1, \ldots, m_X\}$, et $\lfloor . \rfloor$ est la partie entière. En appliquant le résultat de l'équation (3.9) à la définition des coefficients de Fourier, on obtient l'expression suivante pour $c_{\mathbf{k}}$:

$$c_{\mathbf{k}} = \lim_{T \to +\infty} \frac{1}{2T} \int_{-T}^{T} \mathcal{M}(\mathbf{x}(t)) \exp\left(-2i\pi \mathbf{k}.\mathbf{x}(t)\right) \mathrm{d}\mathbf{x}(t).$$

Pour éviter des problèmes dus à l'intégration sur un ensemble infini, $\mathbf{x}$ est remplacé par la fonction $\mathbf{x}^*$ telle que pour tout $i = 1, \ldots, m_X$:

$$x_i^*(t) = G_i(\sin(2\pi\omega_i t))$$

avec $\omega_i \in \mathbb{N}$ et $G_i : [-1;1] \to [0;1]$. Le coefficient $c_{\mathbf{k}}$ peut alors être approché par

$$c_{\mathbf{k}} \approx \int_0^1 \mathcal{M}(\mathbf{x}^*(t)) \exp\left(-2i\pi \mathbf{k}.\mathbf{x}(t)\right) \mathrm{d}\mathbf{x}(t),$$

qui peut être estimée numériquement sur une grille régulière :

$$\hat{c}_{\mathbf{k}} = \frac{1}{n} \sum_{j=1}^{n} \mathcal{M}\left(\mathbf{x}^*\left(\frac{j}{n}\right)\right) \exp\left(-2i\pi(\mathbf{k}.\boldsymbol{\omega})\frac{j}{n}\right). \tag{3.10}$$

L'estimateur de la méthode FAST de l'indice du premier ordre de la variable X_i est alors défini comme suit :

$$\hat{S}_i^{FAST} = \frac{\sum_{k_i=1}^{N_i} \hat{c}_{k_i}^2}{\sum_{k_1=1}^{N_1} \cdots \sum_{k_{m_X}=1}^{N_{m_X}} \hat{c}_{\mathbf{k}}^2},$$

où $\hat{c}_{k_i}$ est l'estimation du coefficient de Fourier d'ordre $\mathbf{k}$ tel que $k_j = 0$ pour tout $j \neq i$ et $k_j = k_i$ sinon, et $N_1, \ldots, N_{m_X} \in \mathbb{N}$ sont les ordres de troncatures. Les paramètres G_i et ω_i $(i \in \{1, \ldots, m_X\})$

sont ajustables. Le choix de ces paramètres est étudié notamment dans Cukier et al. (1978) et Saltelli et al. (1999).

Plusieurs adaptations et améliorations de cette méthode ont été proposées. Saltelli et al. (1999) constatent que les fonctions G_i peuvent être modifiées en $G_i(. + \phi_i)$ avec $\phi_i \in \mathbb{R}$ une constante, les estimateurs des coefficients de Fourier et des indices de Sobol' demeurant inchangés. Ils proposent une nouvelle méthode, appelée EFAST, utilisant ces déformations pour mieux échantillonner l'espace des entrées. En effet, comme les points sur lesquels le modèle $\mathcal{M}$ est évalué pour estimer les coefficients de Fourier sont tous sur la courbe paramétrée $t \mapsto \mathbf{x}^*(t)$ qui parcourt une zone limitée de l'hypercube $[0;1]^{m_X}$, l'ajout d'un déphasage ϕ_i permet de mieux couvrir l'espace.

Tarantola et al. (2006a) ont ensuite introduit une adaptation de la méthode FAST, appelée RBD (*Random Balance Design*), dans laquelle les difficultés liées au choix des fréquences ω_i sont contournées. En effet, ils proposent de prendre des fréquences ω_i égales, $\forall i \in \{1, \ldots, n\}$. Ce choix entraînant un alignement sur une même courbe des points utilisés dans l'estimation de $\hat{c}_k$ (équation (3.10)), ils rajoutent des permutations aléatoires $\pi_1, \ldots, \pi_{m_X}$ des points afin de mieux couvrir l'espace de entrées. La part de variance $\mathrm{Var}\left[E(Y|X_i)\right]$ expliquée par l'entrée X_i est estimée comme suit :

$$\hat{V}_i = \sum_{k_i=1}^{N_i} |\hat{c}_{k_i}^{\pi_i}|^2,$$

avec

$$\hat{c}_{k_i}^{\pi_i} = \frac{1}{n} \sum_{j=1}^{n} \mathcal{M}(\mathbf{x}^{*,i}(\tfrac{j}{n})) \exp\left(-2i\pi(\mathbf{k}.\boldsymbol{\omega})\tfrac{j}{n}\right),$$
$$\mathbf{x}^{*,i}\left(\tfrac{j}{n}\right) = \left(G_1(\sin(2\pi\omega\tfrac{\pi_1(\pi_i^{-1}(j))}{n})), \ldots, G_i(\sin(2\pi\omega\tfrac{j}{n})), \ldots, G_m(\sin(2\pi\omega\tfrac{\pi_m(\pi_i^{-1}(j))}{n}))\right).$$

La méthode RBD a ensuite été étendue par Mara (2009) au calcul des indices totaux. Enfin, Tissot et Prieur (2012a) montrent que les indices calculés avec cette méthode sont biaisés et proposent une méthode pour corriger ce biais.

Remarque 3.1. *Une construction théorique rigoureuse des approches spectrales d'estimation des indices de Sobol' dont FAST et RBD font partie a été proposée par Tissot et Prieur (2012b).*

3.2.3 Analyse de sensibilité avec des entrées fonctionnelles

Plusieurs approches d'estimation des indices de sensibilité de codes de calcul à entrées fonctionnelles ont déjà été développées. Dans cette revue de l'état de l'art, ces approches sont classées en cinq grandes catégories. Parmi ces travaux, ceux présentés dans les sections 3.2.3.1 à 3.2.3.3 sont liés à des méthodes de caractérisation des incertitudes des paramètres fonctionnels en entrée du code de calcul. Une fois les incertitudes des variables fonctionnelles caractérisées, il est possible d'appliquer une méthode d'estimation Monte Carlo comme présenté dans la section 3.2.2.1. La méthode décrite dans la section 3.2.3.4 ne modélise pas ces incertitudes, et considère plutôt que les variables fonctionnelles sont des paramètres incontrôlables. Pour cela, elle s'appuie sur les méthodes de métamodélisation des codes de calcul stochastiques présentées dans le chapitre 4. La dernière méthode, présentée dans la section 3.2.3.5, ne vise pas à quantifier la sensibilité du code de calcul aux variables fonctionnelles mais la sensibilité aux variables fonctionnelles restreintes à des sous-domaines de leur domaine de définition.

Dans toute cette section, on note $\mathcal{M}$ le code de calcul sur lequel est réalisée l'analyse de sensibilité. Le code $\mathcal{M}$ possède m_Z entrées fonctionnelles $Z_1, \ldots, Z_{m_Z}$ et m_X entrées scalaires $X_1, \ldots, X_{m_X}$. La sortie du code $\mathcal{M}$ est $Y = \mathcal{M}(X_1, \ldots, X_{m_X}, Z_1, \ldots, Z_{m_Z})$.

3.2.3.1 Modélisation par une distribution discrète

Dans cette première approche, on considère que plusieurs réalisations des paramètres d'entrée fonctionnels sont connues. On note $\{z_i^1, \ldots, z_i^{n_i}\}$ l'ensemble des $n_i \in \mathbb{N}$ réalisations connues de la variable fonctionnelle Z_i, pour $1 \leq i \leq m_Z$. La distribution de probabilité de chaque variable Z_i est approchée

par la distribution discrète uniforme prenant ses valeurs dans l'ensemble $\{z_i^1, \ldots, z_i^{n_i}\}$. L'estimation des indices de sensibilité peut alors être réalisée à l'aide d'un tirage Monte Carlo comme présenté dans la section 3.2.2. Cette méthode d'analyse de sensibilité est résumée par l'algorithme 3.1.

Cette méthode a été mise en œuvre dans de nombreux articles parmi lesquels on peut citer Ruffo et al. (2006) et Lilburne et Tarantola (2009). Les premiers l'ont appliquée à l'analyse de sensibilité d'un code de calcul de prédiction de la production d'un réservoir pétrolier dont deux des entrées sont des variables aléatoires spatiales représentant la géométrie du réservoir et le flux de chaleur. Respectivement 8 et 4 réalisations de ces variables ont été utilisées pour l'analyse de sensibilité. Lilburne et Tarantola (2009) ont, quant à eux, utilisé cette méthode pour l'étude d'un code de calcul simulant le transport de nitrates dans les aquifères. Pour les quatre variables spatiales en entrée de leur modèle, les supports des distributions discrètes contenaient seulement de 2 à 4 réalisations.

Algorithme 3.1 Modélisation des variables fonctionnelles par une distribution discrète

1. Générer les réalisations $z_i^1, \ldots, z_i^{n_i}$ pour chaque variable $Z_1, \ldots, Z_{m_Z}$.
2. Créer un échantillon Monte Carlo en simulant selon les distributions discrètes uniformes de supports $\{z_i^1, \ldots, z_i^{n_i}\}$ et selon les distributions des variables scalaires.
3. Evaluer le code de calcul pour chaque point de l'échantillon.
4. Estimer les indices de sensibilité pour les variables scalaires et fonctionnelles.

3.2.3.2 Réduction de la dimension

La deuxième approche étudiée consiste à réduire la dimension des variables fonctionnelles en entrée de $\mathcal{M}$ en les approchant par des variables scalaires. En effet, il est plus simple d'estimer la distribution de variables scalaires et de réaliser ensuite des tirages Monte Carlo de ces variables.

Discrétisation des variables fonctionnelles

Une première manière de résumer une variable fonctionnelle par des variables scalaires consiste à la discrétiser. La densité jointe de la variable discrétisée est ensuite estimée afin de pouvoir réaliser les simulations Monte Carlo nécessaires à l'estimation des indices de Sobol'. Il peut cependant être difficile d'estimer la densité de probabilité jointe des points discrétisés, surtout si le nombre de points de discrétisation est élevé. Pour contourner cette difficulté, une solution grossière peut être de négliger la dépendance entre les points de la discrétisation et d'approcher la densité de probabilité jointe par le produit des densités marginales en chaque point, chacune de ces marginales étant plus facile à estimer que la loi jointe. Cette approche est détaillée dans l'algorithme 3.2. Cette hypothèse est très restrictive et n'est pas adaptée à tous les cas d'application. Heuvelink et al. (2010) ont utilisé cette approche sur un code de calcul d'étude des infiltrations de pesticides dont plusieurs entrées sont des variables spatiales. Ces variables étaient discrétisées sur 258 points. Les points de la grille de discrétisation étaient considérés comme suffisamment éloignés les uns des autres pour que la dépendance entre les différents points soit négligée. Dans cet exemple, les densités de probabilité pour les différents points de la discrétisation étaient modélisées par des lois log-normales indépendantes.

Division des domaines de définition des variables fonctionnelles

Une modélisation assez similaire consiste à séparer le domaine de variation de la variable fonctionnelle en quelques zones et à décrire chacune de ces zones par une ou plusieurs variables scalaires. Les variables scalaires relatives à une même sous-région peuvent être considérées comme dépendantes, alors que des variables de deux sous-régions différentes sont supposées indépendantes. Cette méthode repose sur une hypothèse d'indépendance entre les sous-régions, ce qui peut être une hypothèse très forte dans certains cas. Cette approche a été mise en œuvre notamment par Hall et al. (2005) et Volkova et al. (2008). Ces derniers ont appliqué cette méthode à un modèle d'étude du transport de radionucléides dans les nappes

Algorithme 3.2 Estimation par discrétisation des variables fonctionnelles

1. Générer les réalisations $z_i^1, \ldots, z_i^{n_i}$ des variables Z_i, $i \in \{1, \ldots, m_X\}$.
2. Discrétiser ces réalisations.
3. Modéliser la densité de probabilité des variables fonctionnelles en chaque point de la discrétisation.
4. Créer un échantillon Monte Carlo en simulant selon la densité de probabilité estimée et selon les distributions des variables scalaires.
5. Evaluer le code de calcul pour chaque point de l'échantillon.
6. Estimer les indices de sensibilité pour les variables scalaires et fonctionnelles.

phréatiques. Ils séparent en quatre sous-régions le domaine de variation des variables spatiales en entrée du modèle et décrivent chacune de ces sous-régions par une variable aléatoire : la valeur moyenne de la variable fonctionnelle sur la zone. Les quatre variables ainsi obtenues sont ensuite considérées comme indépendantes pour le calcul des indices de Sobol'.

Décomposition fonctionnelle

Il est aussi possible de décomposer les variables aléatoires fonctionnelles sur une base fonctionnelle. Pour chaque variable fonctionnelle Z_i, d_i fonctions de base sont retenues dans la décomposition. Ainsi, chaque variable est résumée par les d_i coefficients de sa décomposition sur la base. Pour chaque variable fonctionnelle, les coefficients de la décomposition sont des variables scalaires dépendantes. La distribution jointe de ces variables scalaires est ensuite modélisée. Plusieurs méthodes de décomposition fonctionnelle et de modélisation de densités jointes peuvent être trouvées dans le chapitre 2. L'indice de sensibilité d'une variable fonctionnelle est alors approché par l'indice de sensibilité du groupe de ses coefficients sur la base (Jacques et al. 2006). L'algorithme 3.3 décrit les différentes étapes de cette méthodologie. Anstett-Collin et al. (2015) proposent une application de cette approche sur un code de calcul modélisant les flux thermiques d'un bâtiment. Parmi les entrées de ce simulateur, six sont des variables aléatoires temporelles. Anstett-Collin et al. (2015) proposent une décomposition fonctionnelle sur une base de Karhunen-Loève (Loève 1955).

Algorithme 3.3 Estimation par décomposition fonctionnelle

1. Générer les réalisations $z_i^1, \ldots, z_i^{n_i}$ des variables Z_i, $i \in \{1, \ldots, m\}$.
2. Appliquer une méthode de décomposition fonctionnelle aux réalisations des variables fonctionnelles et tronquer la base estimée.
3. Modéliser la densité de probabilité des coefficients de la décomposition.
4. Créer un échantillon Monte Carlo en simulant selon la densité de probabilité estimée des coefficients de la décomposition et selon les distributions des variables scalaires. Construire les fonctions correspondant aux jeux de coefficients simulés.
5. Evaluer le code de calcul pour chaque point de l'échantillon.
6. Estimer les indices de sensibilité pour les variables scalaires et fonctionnelles.

Décomposition fonctionnelle et modèles fonctionnels linéaires

Enfin, Fort et al. (2013) étudient le cas particulier où $\mathcal{M}$ peut être approché par le modèle fonctionnel linéaire suivant :

$$Y = \mu + \sum_{k=1}^{m_Z} \langle Z_k, \beta_k \rangle + \varepsilon,$$

où $\mu \in \mathbb{R}$, β_k, $1 \leq k \leq m_Z$, sont des fonctions, $\langle , \rangle$ un produit scalaire et ε est un bruit centré et indépendant des variables $Z_1, \ldots, Z_{m_Z}$. Pour ce faire, chaque variable fonctionnelle Z_k, pour $k \in \{1, \ldots, m_X\}$, est décomposée selon la transformation de Karhunen-Loève (Loève 1955) :

$$Z_k \approx \sum_{l=1}^{d} \sqrt{\lambda_l^k} \xi_i^k \phi_l^k,$$

où $\lambda_1^k \ldots, \lambda_d^k$ et $\phi_1^k, \ldots, \phi_d^k$ sont respectivement les valeurs propres et les fonctions propres de l'opérateur de covariance de Z_k défini par $f \mapsto E\left(\langle Z_k, f \rangle Z_k\right)$, et $\xi_1^k, \ldots, \xi_d^k$ sont des variables aléatoires indépendantes, centrées et de variance 1. Pour l'échantillon de réalisations i.i.d. $\left(Z_i^1, \ldots, Z_i^{m_X}, Y_i\right)_{1 \leq i \leq n}$, Fort et al. (2013) définissent l'estimateur suivant de l'indice de Sobol' de Z_k :

$$\hat{S}_k = \frac{\sum_{l=1}^{d} \frac{1}{\lambda_l^k} \frac{1}{n(n-1)} \sum_{1 \leq i \neq j \leq n} \langle Z_i^k, \phi_l^k \rangle Y_i \langle Z_j^k, \phi_l^k \rangle Y_j}{\frac{1}{n-1} \sum_{i=1}^{n} \left(Y_i - \bar{Y}_n\right)^2},$$

où $\bar{Y}_n$ est la moyenne des $Y_1, \ldots, Y_n$. Le numérateur de cet estimateur est biaisé et son biais est égal à $\sum_{l=d+1}^{\infty} \lambda_l^k \gamma_l$, avec $\beta = \sum_{l=1}^{\infty} \gamma_l \phi_l$. La limitation principale de cette méthode est qu'elle suppose la linéarité du code $\mathcal{M}$ par rapport à ses entrées fonctionnelles.

Parmi les méthodes basées sur la réduction de dimension des variables fonctionnelles, celle utilisant la décomposition fonctionnelle semble être la plus prometteuse, car elle nécessite moins d'hypothèses sur les variables d'entrée, comme dans les deux premières méthodes, ou sur le code, comme dans la méthode proposée par Fort et al. (2013).

3.2.3.3 Sensibilité à la présence de la variable fonctionnelle

Crosetto et Tarantola (2001) proposent de modifier les variables fonctionnelles considérées en introduisant une variable aléatoire « switch », notée ξ et suivant une distribution de Bernoulli de probabilité $\frac{1}{2}$. Pour chaque variable fonctionnelle Z_i, une valeur nominale Z_i^0 est choisie et une nouvelle variable aléatoire $\tilde{Z}_i$ conditionnelle à ξ est définie comme suit. Pour simuler une réalisation $\tilde{z}_i$ de $\tilde{Z}_i$, on tire tout d'abord une réalisation de ξ. Si celle-ci vaut 0, la réalisation $\tilde{z}_i$ est fixée à la valeur nominale z_i^0, sinon z_i est simulée selon la loi de Z_i. Crosetto et Tarantola (2001) proposent d'utiliser l'indice de Sobol' de la variable ξ, S_ξ, comme indice de sensibilité de la variables Z_i. Les étapes de la procédure sont précisées dans l'algorithme 3.4.

Algorithme 3.4 Estimation de la sensibilité à la présence de la variable fonctionnelle

1. Pour chaque variable fonctionnelle Z_i, définir une variable aléatoire, ξ_i, suivant une distribution de Bernoulli de probabilité $\frac{1}{2}$, une valeur nominale, z_i^0 et une variable $\tilde{Z}_i = \mathbb{1}_{\{\xi_i = 0\}} z_i^0 + \mathbb{1}_{\{\xi_i = 1\}} Z_i$.
2. Créer un échantillon Monte Carlo en simulant selon les variables $\tilde{Z}_1, \ldots \tilde{Z}_{m_Z}$ et selon les distributions des variables scalaires.
3. Evaluer le code de calcul pour chaque point de l'échantillon.
4. Estimer les indices de sensibilité pour les variables scalaires et les variables switch $\xi_1, \ldots, \xi_{m_Z}$.

Par ce procédé, les auteurs estiment seulement la sensibilité de $\mathcal{M}$ au fait que Z_i soit variable ou non ; ils quantifient uniquement la sensibilité de $\mathcal{M}$ à la présence de variabilité dans Z_i. S_ξ ne quantifie donc pas la contribution de l'incertitude de Z_i à l'incertitude de Y et n'est pas égal à l'indice de Sobol' de Z_i. Cependant, S_ξ constitue un indicateur qualitatif de la sensibilité de $\mathcal{M}$ à Z_i. Dans Crosetto et Tarantola (2001), cette méthode est appliquée à l'étude d'un modèle hydrologique de prédiction d'inondations, et permet de prendre en compte plusieurs variables aléatoires spatiales en entrée du modèle.

3.2.3.4 Métamodèles joints

Récemment, Iooss et Ribatet (2009) ont proposé d'utiliser des métamodèles joints pour réaliser l'analyse de sensibilité dans le cas de variables fonctionnelles. Cette méthode, décrite plus en détail dans le chapitre 4, avait été précédemment introduite pour approcher les codes de calcul stochastiques. Dans celle-ci, deux métamodèles approchent la moyenne et la dispersion de la sortie du code. Pour cela, Zabalza et al. (1998), Iooss et Ribatet (2009) et Marrel et al. (2012) emploient respectivement des modèles linéaires généralisés, des modèles additifs généralisés et des métamodèles processus gaussiens. Iooss et Ribatet (2009) proposent d'étendre cette méthode aux entrées fonctionnelles, en considérant ces entrées comme des paramètres incontrôlables, c'est-à-dire régis par une graine aléatoire ε. Les variables fonctionnelles sont alors notées W_ε, et les métamodèles joints de la moyenne et de la dispersion s'écrivent

$$\begin{aligned} Y_m(\mathbf{X}) &= \mathbb{E}(Y|\mathbf{X}) \\ Y_d(\mathbf{X}) &= \mathrm{Var}(Y|\mathbf{X}) \;=\; \mathbb{E}\left[(Y - Y_m(\mathbf{X}))^2|\mathbf{X}\right], \end{aligned}$$

avec $\mathbf{X} = (X_1, \ldots, X_{m_X})$ et où Y_d et Y_m sont respectivement les composantes de la moyenne et de la dispersion. A l'aide de la formule de la variance totale, la variance de Y peut être déduite des deux métamodèles :

$$\begin{aligned} \mathrm{Var}(Y) &= \mathrm{Var}\left[\mathbb{E}\left(Y|\mathbf{X}\right)\right] + \mathbb{E}\left[\mathrm{Var}\left(Y|\mathbf{X}\right)\right] \\ &= \mathrm{Var}\left[Y_m(\mathbf{X})\right] + \mathbb{E}\left[Y_d(\mathbf{X})\right]. \end{aligned} \tag{3.11}$$

De plus, les auteurs montrent que tous les indices de Sobol' de sous-ensembles de $\{X_1, \ldots, X_{m_X}\}$ peuvent être calculés avec la décomposition de l'équation (3.11) et le métamodèle de la moyenne Y_m. L'indice total de W_ε, noté S_{T_ε}, dépend du métamodèle de dispersion de la manière suivante :

$$S_{T_\varepsilon} = \frac{\mathbb{E}[Y_d(\mathbf{X})]}{\mathrm{Var}(Y)}. \tag{3.12}$$

Si la variable W_ε représente une (ou plusieurs) variable fonctionnelle, l'indice total S_{T_ε} peut être interprété comme l'indice de Sobol' total de l'entrée fonctionnelle. Seule la part de variance totale est estimée. L'effet de la variable seule (ou du groupe de variables) ne peut être distingué de ses interactions avec les paramètres $X_1, \ldots, X_{m_X}$. Toutefois, il est possible, dans certains cas, de déduire les potentielles interactions entre les variables fonctionnelles et scalaires en comparant les indices d'ordre un et totaux de tous les paramètres (Marrel et al., 2012). La méthode complète d'analyse de sensibilité avec des métamodèles joints est résumée dans l'algorithme 3.5.

Algorithme 3.5 Méthode des métamodèles joints

1. Simuler un échantillon de réalisations de variables scalaires et calculer les sorties Y correspondantes en tirant aléatoirement les variables fonctionnelles.
2. Construire les joints métamodèles Y_m et Y_d de la moyenne et de la dispersion.
3. Créer un échantillon Monte Carlo en simulant selon les distributions des variables scalaires.
4. Evaluer les deux métamodèles Y_m et Y_d pour chaque point de l'échantillon.
5. Estimer les indices de sensibilité pour les variables scalaires avec Y_m. Estimer l'indice total du groupe des variables fonctionnelles à l'aide de l'équation (3.12).

3.2.3.5 Sensibilité sur des sous-intervalles du domaine de définition

Dans de récents travaux, Fruth et al. (2015) proposent d'étudier la sensibilité de la sortie Y par rapport aux restrictions des variables fonctionnelles sur différents sous-domaines de leur domaine de définition. Le domaine de définition D_i de chaque variable fonctionnelle Z_i, $1 \leq i \leq m_Z$, est partitionné en p_i intervalles disjoints, notés $D_i^1, \ldots D_i^{p_i}$. Au lieu de calculer les indices de sensibilité de la variable

fonctionnelle sur tout son intervalle de variation, p_i indices de sensibilité sont calculés pour les restrictions de la variable Z_i aux sous-intervalles D_i^j, $j \in \{1, \ldots, p_i\}$. Pour calculer ces indices, Fruth et al. (2015) font l'hypothèse que les variables fonctionnelles sont constantes sur chaque sous-intervalle. Par conséquent, l'indice de sensibilité de la variable Z_i restreint à l'intervalle D_i^j représente l'influence de la moyenne de la variable Z_i sur D_i^j. La restriction de la variable Z_i à un certain sous-intervalle est considérée comme indépendante de la restriction de Z_i à un autre sous-intervalle. Fruth et al. (2015) définissent de nouveaux indices de sensibilité, à savoir les coefficients de la régression linéaire entre les entrées du code de calcul et Y, normalisés par la taille de l'intervalle D_i^j.

De plus, une stratégie itérative est utilisée pour construire la partition des domaines de variation D_i. A chaque étape, les intervalles les plus intéressants sont partitionnés en deux intervalles égaux, de nouveaux points sont ajoutés à la base d'apprentissage du modèle linéaire selon la méthode des bifurcations séquentielles (Bettonvil 1995) et une nouvelle estimation des indices est réalisée. Le choix des intervalles à séparer en deux prend en compte notamment la valeur des indices calculés à l'étape précédente.

Par rapport aux autres méthodes présentées dans cette section, ces indices de sensibilité ont l'avantage d'indiquer plus précisément sur quelles parties de leur domaine de définition les variables fonctionnelles sont influentes. Cependant, à l'intérieur des sous-intervalles, de fortes influences peuvent s'annuler si elles sont de signe contraire, et il n'est alors pas possible de les détecter. Une deuxième limite de cette méthode est qu'elle ne tient compte que des comportement linéaires et peut ainsi ignorer les influences non-linéaires des paramètres d'entrée.

Fruth et al. (2015) ont appliqué ces nouveaux indices de sensibilité à un code de calcul modélisant le processus de profilage des plaques de métal. Les paramètres d'entrée du modèle sont deux variables temporelles. Trois étapes de l'algorithme de partitionnement itératif ont été appliquées et le domaine de variation de chaque variable est partitionné en six intervalles.

3.2.4 Analyse de sensibilité avec des entrées dépendantes

Dans la méthode d'analyse de sensibilité globale présentée dans le chapitre 3.2.1, il est supposé que les variables d'entrée de la fonction $\mathcal{M}$ étudiée sont indépendantes. En effet, dans le cas de variables dépendantes, la décomposition de la variance décrite dans l'équation (3.3) n'est plus vérifiée. Toutefois, dans de nombreux cas industriels, l'hypothèse d'indépendance ne peut pas être faite et il est nécessaire d'adapter les mesures de sensibilité existantes. De nombreux travaux ont été conduits pour adapter les indices de Sobol' au cas dépendant ou pour développer de nouveaux indices de sensibilité pour lesquels la dépendance entre entrées ne serait pas un problème. Dans cette section, un état de l'art de ces différents travaux est proposé.

Dans la section 3.2.4.1, sont présentés les travaux de Jacques et al. (2006) qui proposent de calculer les indices de groupes de variables dépendantes plutôt que les indices de chaque variable. En choisissant ces groupes de variables indépendants les uns des autres, le problème se ramène ainsi au cas indépendant. Le but de la deuxième famille d'approche, présentée dans la section 3.2.4.2, est d'adapter les méthodes d'estimation des indices de Sobol'. Pour ce faire, Xu et Gertner (2007) modifient la méthode *Random Balance Design* (Tarantola et al. 2006a). Une méthode d'estimation des indices de Sobol' à l'aide de polynômes locaux est développée par Da Veiga et al. (2009). Enfin, Mara et Tarantola (2012) utilisent une orthogonalisation des variables d'entrée pour se ramener au cas classique des variables indépendantes.

Cependant, la pertinence des indices de Sobol' pour des paramètres d'entrée dépendants peut être remise en question. En effet, la décomposition de la variance n'est plus vérifiée et la somme des indices de Sobol' n'est plus assurée d'être égale à un. La perte de cette propriété peut rendre plus difficile l'interprétation des indices. Dans le cas d'une corrélation positive entre deux variables d'entrée, l'effet de la première variable est comptabilisé dans les indices de sensibilité du premier ordre des deux variables et dans leurs indices d'ordre supérieur. Ainsi, une valeur élevée d'un indice de Sobol' peut être due à sa contribution à la variabilité de la sortie et/ou à sa corrélation avec une autre variable influente. Il n'est pas possible de discerner ces deux effets. Au vu de ce problème, plusieurs méthodes proposent d'adapter la décomposition ANOVA pour tenir compte de la dépendance entre les variables et définissent

de nouveaux indices de sensibilité. On peut citer par exemple Xu et Gertner (2008) et Li et al. (2010) qui proposent des décompositions basées sur une approximation par métamodèles afin de définir des indices de sensibilité différents. Enfin, Chastaing et al. (2012) définissent, sous certaines conditions, une décomposition ANOVA dont les termes sont hiérarchiquement orthogonaux. Les indices issus de ces adaptations de la décomposition ANOVA présentent le défaut d'être difficiles à interpréter. En particulier, ils ne se somment pas tous à 1 et peuvent être négatifs

3.2.4.1 Analyse de sensibilité de variables d'entrée groupées

Jacques et al. (2006) proposent de grouper les variables de sorte qu'à l'intérieur d'un groupe, les variables soient dépendantes, mais que des variables de deux groupes différents soient indépendantes. On note $X_1, \dots X_{m_X}$ les m_X variables aléatoires en entrée de $\mathcal{M}$. Ces variables sont donc réparties en g groupes, et g nouvelles variables sont définies :

$$X = \left(\underbrace{X_1, \dots, X_{s_1}}_{\vec{X}_1}, \dots, \underbrace{X_{s_{g-1}+1}, \dots, X_{s_g}}_{\vec{X}_g} \right), \tag{3.13}$$

où $s_j = \sum_{i=1}^{j} l_i$ et l_j est le nombre de variables dans le groupe $j \in \{1, \dots, g\}$. De plus, $s_g = \sum_{i=1}^{g} l_i = m_X$. Les variables vectorielles $\vec{X}_1, \dots, \vec{X}_g$ sont par définition indépendantes.

L'indice de Sobol' du j-ème groupe de variables $\left(X_{s_{j-1}}, \dots, X_{s_j}\right)$ est défini comme l'indice de sensibilité du vecteur aléatoire $\vec{X}_j$, pour $j \in \{1, \dots, g\}$:

$$S_j = \frac{\operatorname{Var}\left(\mathbb{E}\left(Y|\vec{X}_j\right)\right)}{\operatorname{Var} Y}. \tag{3.14}$$

Si $l_j = 1$, alors $\vec{X}_j = X_j$ et l'indice de Sobol' ainsi défini est égal à l'indice de Sobol' du premier ordre défini pour une variable scalaire (équation 3.5). Ces indices mesurent les effets de toutes les variables du groupe ainsi que de leurs interactions. Ils sont appelés indices multidimensionnels dans Jacques et al. (2006) et correspondent aux indices *closed* définis notamment dans Janon et al. (2014).

Avec cette méthode, l'influence de chaque variable dépendante ne peut pas être quantifiée individuellement. Cela pose d'autant plus problème si les groupes contiennent un grand nombre de variables ou ont un indice élevé (ce qui traduit une grande influence du groupe). Une autre limitation de cette méthode est qu'elle n'est pas applicable quand toutes les variables sont dépendantes, car il n'y aurait alors qu'un seul groupe et donc un seul indice de sensibilité.

Remarque 3.2. *Il est à noter que Gilquin et al. (2015) ont proposé d'étendre la méthode d'estimation basée sur les plans d'expériences répliqués décrite dans la section 3.2.2.1 à l'estimation des indices de sensibilité de groupes de variables.*

3.2.4.2 Techniques d'estimation des indices de Sobol' dans le cas d'entrées dépendantes

Les méthodes présentées dans cette section ont toutes pour but de définir des méthodes de calculs des indices de Sobol' pour des variables corrélées. La définition des indices de Sobol' donnée dans la section 3.2.1 n'est pas modifiée. Seules sont modifiées les méthodes d'estimation et les méthodes d'échantillonnage utilisées pour calculer ces indices.

Tout d'abord, Xu et Gertner (2007) adaptent l'échantillonnage des points de la méthode RBD (*Random Balance Design*) introduite par Tarantola et al. (2006a) et décrite dans la section 3.2.2.2. Dans cette méthode basée sur la décomposition du code sur une base de Fourier, des permutations $\pi_1, \dots, \pi_{m_X}$ sont appliquées aux valeurs des variables d'entrée utilisées dans l'estimation. Comme les variables d'entrée sont corrélées, permuter aléatoirement les valeurs des variables d'entrée ferait perdre la structure de

corrélation entre les variables. C'est pourquoi, Xu et Gertner (2007) utilisent, pour permuter les coordonnées, une procédure introduite par Iman et Conover (1982) qui permet de générer des données corrélées à partir de données simulées indépendamment. Les indices sont ensuite estimés comme dans la méthode RBD.

Une autre méthode développée par Da Veiga et al. (2009) repose sur l'ajustement polynomial local. Cet outil permet aux auteurs d'obtenir deux estimateurs des indices de Sobol' S_i, $1 \leq i \leq m_X$, basés sur le fait que l'indice S_i peut s'écrire des deux façons suivantes :

$$S_i = \frac{\mathrm{Var}(E(Y|X_i)}{\mathrm{Var}\, Y} \text{ et } S_i = 1 - \frac{E(\mathrm{Var}(Y|X_i)}{\mathrm{Var}\, Y},$$

puisque la variance de Y se décompose en $\mathrm{Var}\, Y = \mathrm{Var}(E(Y|X_i) + E(\mathrm{Var}(Y|X_i)$. La méthode comporte deux étapes. Tout d'abord, un modèle de régression hétéroscédastique est utilisé pour décrire Y en fonction de la variable X_i, à partir d'un échantillon d'entrées et de sorties $\{(x_i^j, y^j)\}_{1\leq j\leq n}$:

$$y^j = m(x_i^j) + \sigma(x_i^j)\varepsilon_j,$$

où $m(x_i^j) = E(Y|X_i = x_i^j)$ et $\sigma^2(x_i^j) = \mathrm{Var}(Y|X_i = x_i^j)$, et $\varepsilon_1, \ldots, \varepsilon_n$ sont des variables aléatoires indépendantes de moyennes nulles et de variances unitaires. Les fonctions m et σ^2 sont déterminées par ajustement polynomial local. Ce modèle consiste à approcher localement m par un polynôme : $\hat{m}(z) = \sum_{k=0}^{p} \beta_k (z - x_i)^k$ pour un z « proche » de x. Les coefficients β_k, $k \in \{1, \ldots, m\}$, sont déterminés par la résolution du problème d'ajustement polynomial suivant :

$$\min_{\beta} \sum_{j=1}^{n} \left(Y^j - \sum_{k=0}^{p} \beta_k (z - x_i)^k \right)^2 K_1 \left(\frac{X_i^j - x_i}{h_1} \right).$$

De même, on approche localement σ^2 par le polynôme $\hat{\sigma}^2(z) = \sum_{k=0}^{p} \gamma_k (z - x_i)^k$ en résolvant :

$$\min_{\gamma} \sum_{j=1}^{n} \left((Y^j - \hat{m}(X_i^j))^2 - \sum_{k=0}^{p} \gamma_k (z - x_i)^k \right)^2 K_2 \left(\frac{X_i^j - x_i}{h_2} \right).$$

$K_1, K_2 : \mathbb{R} \to \mathbb{R}$ sont des noyaux, et $h_1, h_2 \in \mathbb{R}$.

Enfin, il reste à déterminer $\mathrm{Var}(\hat{m}(X_i))$ et $E(\hat{\sigma}^2(X_i))$. Pour ce faire, les estimateurs classiques de la variance et de l'espérance sont utilisés à partir d'un deuxième échantillon $(\tilde{X}_i^j)_{j=1,\ldots,n'}$ de même distribution que le premier, ce qui conduit aux deux estimateurs suivants de $\mathrm{Var}(Y|X_i)$, $1 \leq i \leq m_X$:

$$\hat{S}_i^1 = \frac{1}{n'-1} \sum_{j=1}^{n'} (\hat{m}(\tilde{X}_i^j) - \bar{\hat{m}})^2 \text{ et } \hat{S}_i^2 = \frac{1}{n'} \sum_{j=1}^{n'} \hat{\sigma}^2(\tilde{X}_i^j).$$

Enfin, Mara et Tarantola (2012) proposent une troisième méthode dans laquelle ils calculent les indices de sensibilité à partir de l'orthogonalisation des X_i. Les auteurs proposent d'orthogonaliser les paramètres d'entrée de la manière suivante :

$$\begin{aligned} \bar{X}_1 &= X_1 \\ \bar{X}_i &= X_i - E(X_i|\bar{X}_1, ..., \bar{X}_{i-1}), \forall i \in \{1, \ldots, m_X\}. \end{aligned} \tag{3.15}$$

Ainsi, chaque variable $\bar{X}_i$, pour $i \in \{1, \ldots, m_X\}$, n'est corrélée qu'avec les variables $\bar{X}_j$, pour $j \geq i$. Les indices proposés par Mara et Tarantola (2012) s'écrivent comme suit :

- $\bar{S}_1 = \frac{Var(E(Y|\bar{X}_1))}{Var(Y)} = S_1$ est la contribution totale de X_1 à Y,
- $\bar{S}_2 = \frac{Var(E(Y|\bar{X}_2))}{Var(Y)} = S_{2-1}$ est la contribution de X_2 à Y sans la partie liée à la corrélation avec X_1,

– ...
– $\bar{S}_{m_X} = \frac{Var(E(Y|\bar{X}_{m_X}))}{Var(Y)} = S^u_{m_X}$ est la contribution non-corrélée de X_{m_X} à Y.

Comme ces indices de sensibilité dépendent de l'ordre des facteurs, on peut réaliser les étapes d'orthogonalisation et de calcul des indices de sensibilité, sur m_X ordonnancements des facteurs ayant des indices finaux différents. De cette manière, les indices de sensibilité $S^u_i, \forall i \in \{1, \ldots, m_X\}$, donnant la contribution non-corrélée de la variable i, sont obtenus pour chaque entrée. L'interprétation des autres indices est plus difficile.

La transformation proposée par Mara et Tarantola (2012) n'est une orthogonalisation que si les entrées suivent des lois normales. Dans le cas général où les variables aléatoires ne sont pas gaussiennes, la transformation de Nataf (Lebrun et Dutfoy, 2009) peut être utilisée à la place de l'orthogonalisation. Elle a cependant le désavantage de rendre plus complexe la relation entre les nouvelles entrées $\bar{X}_i$ et la sortie Y, ainsi que l'interprétation des indices de sensibilité.

Les trois méthodes présentées dans cette section permettent de calculer les indices de Sobol' de variables dépendantes. Cependant, la principale limitation de ces indices réside dans le fait que la plupart des propriétés des indices de Sobol' ne s'appliquent pas dans le cas dépendant et qu'il est donc plus difficile d'interpréter les résultats obtenus. C'est pourquoi, plusieurs auteurs ont proposé de nouvelles définitions d'indices de sensibilité adaptés au cas dépendant.

3.2.4.3 Nouveaux indices de sensibilité dans le cas d'entrées dépendantes

Xu et Gertner (2008) présentent une méthode basée sur la régression linéaire et qui ne s'applique donc qu'aux modèles linéaires ou pouvant être approchés par un modèle linéaire. La variance de Y conditionnellement à X_i, $1 \leq i \leq m_X$, est décomposée en deux termes $V_i = V_i^U + V_i^C$, V_i^U étant la variance partielle sans les corrélations et V_i^C étant la variance ajoutée par les facteurs corrélés avec la variable X_i. Un estimateur de V_i est obtenu par régression de Y par rapport à X_i seul : $Y = \theta_0 + \theta_i X_i + \varepsilon$. A partir d'un échantillon $(X_i, Y_i)_{i=1,\ldots,n}$, l'estimateur s'écrit :

$$\hat{V}_i = \frac{1}{n-1} \sum_{j=1}^{n} (\theta_0 + \theta_i X_i - \bar{Y})^2.$$

Un estimateur de V_i^U est obtenu par régression de Y par rapport aux résidus de la régression de X_i par rapport aux $(X_j)_{j \neq i}$: $Y = \alpha_0 + \alpha_i \hat{Z}_i + \varepsilon'$, où $\hat{Z}_i = X_i - \left(\beta_0 + \sum_{j \neq i} \beta_j X_j\right)$. On définit donc :

$$\hat{V}_i^U = \frac{1}{n-1} \sum_{j=1}^{n} (\alpha_0 + \alpha_i \hat{Z}_i - \bar{Y}).$$

On détermine enfin $\hat{V}_i^C = \hat{V}_i - \hat{V}_i^U$.

Pour la construction de ces indices de sensibilité, Xu et Gertner (2008) font l'hypothèse que le code $\mathcal{M}$ est linéaire. Cette hypothèse n'étant pas toujours vérifiée en pratique, Li et al. (2010) ont proposé de nouveaux indices de sensibilité s'affranchissant de l'hypothèse de linéarité. Li et al. (2010) proposent d'approcher Y par une somme de $n_p << 2^n - 1$ fonctions : $Y = f_0 + \sum_{j=1}^{n_p} f_{p_j}(x_{p_j}) + \varepsilon$, avec $f_0 \in \mathbb{R}$ et $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ l'erreur du modèle. On suppose que ε est orthogonal à $f_{p_k}, \forall k \in \{1, \ldots, n_p\}$. Alors, la variance de Y se décompose ainsi :

$$\begin{aligned} \mathrm{Var}(Y) &= E((Y - E(Y))^2) \\ &= \sum_{j=1}^{n_p} \mathrm{Cov}(f_{p_j}, Y) + E(\varepsilon^2) \\ &= \sum_{j=1}^{n_p} \left(\mathrm{Var}(f_{p_j}) + \mathrm{Cov}(f_{p_j}, \sum_{k=1, k \neq j}^{n_p} f_{p_k}) \right) + E(\varepsilon^2), \end{aligned} \tag{3.16}$$

On définit les indices de sensibilité suivants :

$$S_{p_j} = \frac{\mathrm{Cov}(f_{p_j}, Y)}{\mathrm{Var}(Y)} \ ; \ S^a_{p_j} = \frac{\mathrm{Var}(f_{p_j})}{\mathrm{Var}\, Y} \ ; \ S^b_{p_j} = \frac{\mathrm{Cov}(f_{p_j}, \sum_{k=1,k\neq j}^{n_p} f_{p_k})}{\mathrm{Var}\, Y}.$$

D'après l'équation (3.16), on a $S_{p_j} = S^a_{p_j} + S^b_{p_j}$. Les indices $S^a_{p_j}$, $S^b_{p_j}$ et S_{p_j} sont appelés respectivement contribution structurelle, corrélée et totale.

En pratique, la détermination des f_{p_j} se fait en utilisant un modèle RS-HDMR (Random Sampling-High Dimensional Model Representation). Les fonctions sont décomposées sur des bases de B-splines et leurs coefficients sont déterminés par *backfitting*. Un test de Fisher est utilisé pour déterminer, pour chaque f_{p_j}, si celle-ci doit être incluse dans la décomposition. Ainsi, le calcul des indices de sensibilité est fait à partir d'un métamodèle, le code de calcul n'étant appelé que pour la construction du métamodèle.

Cependant, puisque dans les méthodes de Xu et Gertner (2008) et Li et al. (2010), la décomposition de $\mathcal{M}$ est déterminée par des métamodèles, elle dépend grandement de leur qualité et elle peut changer selon le métamodèle utilisé. Pour s'affranchir de cette problématique, Chastaing et al. (2012) proposent une généralisation de la décomposition ANOVA de la variance décrite dans l'équation (3.1) et inspirée de celle de Hooker (2007). Sous certaines conditions sur la densité de probabilité jointe des variables d'entrée et en imposant des contraintes de hiérarchie entre les composantes de la décomposition, Chastaing et al. (2012) démontrent que, comme dans le cas indépendant, il existe une unique décomposition de la variance sous la forme de l'équation (3.1). A partir de cette décomposition, de nouveaux indices de sensibilité sont proposés. L'indice de la variable X_i, $i \in \{1, \ldots, m_X\}$, s'écrit comme suit :

$$S_i = \frac{\mathrm{Var}(\mathcal{M}_i(X_i)) + \sum_{v\subset\{1,\ldots,m_X\}, i\notin v} \mathrm{Cov}\left(\mathcal{M}_i(X_i), \mathcal{M}_v(X_v)\right)}{\mathrm{Var}\, Y}.$$

où les fonctions $\mathcal{M}_v$, $v \subset \{1, \ldots, m_X\}$, sont les composantes de la décomposition ANOVA. Comme pour les indices de Sobol', la somme des indices proposés est égale à 1, mais les indices ne sont pas bornés entre 0 et 1, puisque les covariances peuvent être négatives dans la formule. L'interprétation de ces indices est donc plus difficile que celle des indices de Sobol'. En pratique, une méthode d'estimation des nouveaux indices proposés est développée dans Chastaing et al. (2014) et Champion et al. (2015). Celle-ci consiste à construire récursivement pour chaque composante de la décomposition ANOVA une base de décomposition. Les coefficients correspondant à chaque composante sont ensuite estimés par moindres carrés. Une méthode de sélection de variable basée sur la pénalisation Lasso est utilisée pour diminuer le nombre de coefficients à estimer.

3.2.5 Méthodes retenues

Dans cette section, les indices de sensibilité de Sobol' basés sur la décomposition de la variance sont présentés ainsi que plusieurs méthodes pour les estimer. Une méthode d'estimation efficace, basée sur les plans répliqués, a notamment été présentée. En effet, cette méthode a l'avantage de requérir un nombre moins important d'évaluations du code étudié que la plupart des méthodes existantes, et qui est, de plus, indépendant du nombre d'entrées du code. On s'est aussi intéressé au cas où certains des paramètres d'entrée du code sont fonctionnels et au cas où ces paramètres sont dépendants.

Dans le cas des codes à entrées fonctionnelles, cinq familles de méthodes sont étudiées. **Parmi celles-ci, on choisit de se concentrer sur la méthode s'appuyant sur la décomposition fonctionnelle des entrées résumée dans l'algorithme 3.3.** Dans cette méthode, les variables fonctionnelles sont approchées par leurs coefficients sur la base de décomposition (des variables aléatoires) dont la distribution de probabilité est ensuite modélisée. Cette méthode offre l'avantage d'imposer moins d'hypothèses aux variables fonctionnelles et au code étudié que les autres méthodes basées sur la réduction de dimension, présentées dans la section 3.2.3.2. De plus, la modélisation de la distribution des variables fonctionnelles utilisée semble en général permettre de mieux approcher leur distribution que la modélisation par une

distribution discrète, utilisée dans la méthode présentée dans la section 3.2.3.1. De plus, la méthode des métamodèles joints ne permet que de calculer les indices totaux des variables fonctionnelles, et n'est donc pas retenue dans la suite. Enfin, la méthode de Crosetto et Tarantola (2001) n'est pas utilisée dans la suite puisqu'elle ne permet pas de quantifier l'influence des variables fonctionnelles, et la méthode de Fruth et al. (2015) ne répond pas aux objectifs fixés puisqu'elle calcule l'influence des variables restreintes à des sous-intervalles et non leur influence globale.

Dans le cas d'entrées dépendantes, les indices de Sobol' peuvent toujours être calculés mais ils perdent une partie des propriétés qu'ils possèdent dans le cas indépendant, et deviennent difficiles à interpréter. C'est pourquoi, plusieurs indices basés sur la variance ont été définis pour étendre les indices de Sobol' au cas dépendant. Néanmoins, ces indices se révèlent, eux aussi, assez difficiles à interpréter. **Pour éviter ces problèmes d'interprétation, la technique retenue dans la suite est détaillée dans la section 3.2.4.1, et consiste à regrouper les variables dépendantes dans des groupes indépendants entre eux et à calculer leurs indices de Sobol' multidimensionnels.** Les indices calculés dans cette méthode sont faciles à interpréter. De plus, ils présentent l'avantage d'être aussi simples à calculer que les indices de Sobol' du premier ordre. Cependant, leur principale limitation est qu'ils ne permettent pas de distinguer les influences de deux variables dépendantes, ce qui n'est pas nécessaire ici, puisque, comme précisé dans l'introduction (cf. section 3.1), l'objectif dans les deux cas traités est de déterminer l'influence de l'ensemble des paramètres corrélés et non les influences de chacun.

Pour le cas du PTS, la méthodologie d'analyse de sensibilité retenue dans ce chapitre est la suivante. La densité de probabilité des variables fonctionnelles est estimée à partir des réalisations disponibles et à l'aide de la méthode présentée dans le chapitre 2. Ensuite, les indices de sensibilité multidimensionnels (cf. section 3.2.4.1) des groupes de variables dépendantes sont calculés en utilisant la méthode des plans répliqués (cf. section 3.2.2.1).

Dans le cas de la rupture LiPoSo, dans lequel le code étudié est coûteux en temps de calcul, la même méthode d'analyse de sensibilité est retenue, mais elle ne peut pas être appliquée directement au code. Pour pallier cette difficulté, on propose de construire un métamodèle qui pourra être substitué au code pour réaliser l'analyse de sensibilité. La construction d'un plan d'expériences des entrées du code et la construction de ce métamodèle à partir du plan d'expériences est l'objet de la section suivante.

3.3 Échantillonnage et métamodélisation pour un code à entrées fonctionnelles

Cette section est consacrée à la construction d'un métamodèle pour l'analyse de sensibilité et la propagation des incertitudes sur un code de calcul à entrées fonctionnelles et scalaires. La première étape de la méthode consiste à réaliser un plan d'expériences uniforme sur l'espace des entrées pour l'apprentissage du métamodèle. L'utilisation d'un plan d'expériences uniforme permet d'apprendre le métamodèle sans privilégier de régions de l'espace et d'assurer ainsi que ses capacités de prédiction seront approximativement les mêmes sur tout l'espace, ce qui ne serait pas le cas avec un plan d'expériences probabilisé. En effet, l'objectif est d'obtenir un métamodèle avec de bonnes capacités de prédiction sur l'ensemble du domaine de variation des paramètres d'entrée (indépendamment de la densité de probabilité des entrées).

Cette problématique d'échantillonnage uniforme d'entrées scalaires et vectorielles, présentée dans la section 3.3.1, peut être séparée en plusieurs parties. Tout d'abord, l'échantillonnage uniforme est réalisé séparément pour les variables scalaires et pour chaque variable fonctionnelle. Ensuite, les différents échantillons générés sont combinés pour créer un unique plan d'expériences sur l'espace des entrées. Enfin, le métamodèle est construit à partir des échantillons générés et des sorties du code de calcul correspondantes. Dans la section 3.3.2, la construction d'un métamodèle processus gaussien est présentée.

3.3.1 Plans d'expériences uniformes de variables fonctionnelles et scalaires

L'objectif de cette section est de proposer une méthode pour générer un échantillon uniforme sur l'espace des entrées. Dans les sections 3.3.1.1 et 3.3.1.2, la construction de plans d'expériences pour les variables scalaires indépendantes puis pour les variables fonctionnelles est étudiée. Dans le cas des variables fonctionnelles, la méthode d'échantillonnage proposée est basée sur la méthode de quantification des incertitudes proposée dans le chapitre 2. L'objectif de la section 3.3.1.3 est de proposer une méthode pour combiner de manière optimale les plans d'expériences construits pour les variables scalaires et fonctionnelles. Dans les sections 3.3.1.2 et 3.3.1.3, les variables fonctionnelles sont supposées indépendantes. L'adaptation au cas dépendant de la méthode proposée est étudiée dans la section 3.3.1.3.

Soit $\mathcal{M}$ un code de calcul. Soit $X = (X_1, \ldots, X_{m_X})$ les variables aléatoires scalaires en entrée de $\mathcal{M}$, et $Z = (Z_1, \ldots, Z_{m_Z})$ ses paramètres d'entrée fonctionnels. La variable scalaire Y est la sortie du code $\mathcal{M}$. La distribution de probabilité des variables scalaires est supposée connue. La densité de probabilité des variables fonctionnelles est estimée à l'aide de la méthode de quantification des incertitudes proposée dans le chapitre 2 en utilisant un échantillon d'apprentissage probabilisé. Plus précisément, les variables fonctionnelles sont décomposées sur une base construite par ACP simultanée et la densité de probabilité des coefficients dans cette base est modélisé par un mélange de gaussiennes. Ainsi, chaque variable fonctionnelle Z_k $(1 \leq k \leq m_Z)$ est approchée par ses d_k coefficients notés $\Gamma_k = (\Gamma_{k,1}, \ldots, \Gamma_{k,d_k})^T$.

3.3.1.1 Échantillonnage de variables scalaires

Dans cette section, l'échantillonnage uniforme de variables scalaires continues et indépendantes est étudié. Les méthodes proposées dans la suite sont définies pour des variables scalaires ayant un domaine de variation borné. Dans le cas contraire, on propose de se ramener au cas borné en limitant le domaine d'échantillonnage aux quantiles d'ordre $\alpha/2$ et $(1-\alpha/2)$, où $\alpha \in]0;1[$. Le domaine ainsi défini sur lequel l'échantillonnage doit être réalisé est un hypercube.

De nombreuses méthodes ont été développées pour générer des points dans un hypercube H. Leur objectif est de répartir les points d'un échantillon de manière homogène. On se propose ici de faire un court état de l'art de ces méthodes. Pour une étude plus approfondie et plus complète de ces méthodes, on peut consulter le livre de Fang et al. (2006).

Sans perte de généralité, on suppose dans la suite que les intervalles de variation des variables $X_1, \ldots, X_{m_X}$ sont $[0;1]$. L'échantillon que l'on cherche à construire est noté $X_u = \left\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\right\}$, avec $\mathbf{x}^{(i)} = \left(x_1^{(i)}, \ldots, x_n^{(i)}\right)^T$ pour $i = 1, \ldots, n$.

Hypercubes Latins (LHS)

L'échantillonnage Hypercube Latin (LHS), introduit par McKay et al. (1979), est un type de plan d'expériences dont les projections sur chaque dimension ont des propriétés de répartition intéressantes. Chacun de ces intervalles est partitionné en n intervalles équiprobables $I_k = [\frac{k-1}{n}; \frac{k}{n}]$, pour $k = 1, \ldots, n$. On définit m_X permutations $\pi_1, \ldots, \pi_{m_X}$ de l'ensemble $\{1, \ldots, n\}$. La composante j du i-ème point d'un plan LHS de n points de $[0;1]^{m_X}$ est tiré selon une loi uniforme dans l'intervalle $I_{\pi_j(i)}$. Des exemples de plans LHS sont donnés sur la Figure 3.1.

Cette construction assure que la projection du plan sur chaque axe donne un plan en dimension 1 avec de bonnes qualités de répartition. En effet, après projection sur l'axe j, le plan possède un et un seul point dans chaque intervalle I_k $(1 \leq k \leq n)$ de la partition. Plus généralement, toute projection d'un plan LHS sur un espace de taille $m \leq m_X$ est aussi un LHS. La structure LHS évite la redondance d'information sur les marginales. En effet, dans un plan quelconque, deux points peuvent avoir des composantes égales sur une dimension et être confondus après projection sur cet axe. Le plan de dimension 1 ainsi obtenu possède moins d'information que dans le cas où les deux points n'auraient pas été confondus.

LHS optimisés

Les plans LHS n'assurent pas une bonne occupation de l'espace, comme l'illustre le premier plan de la Figure 3.1. Il a donc été proposé de construire des plans LHS optimisés pour obtenir une bonne

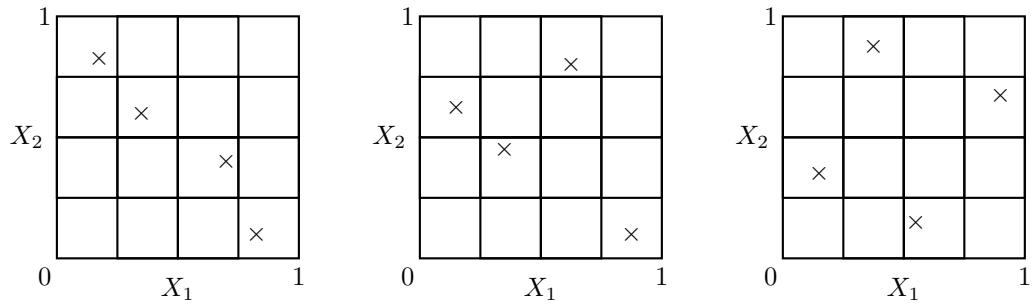

FIGURE 3.1 – Exemples d'échantillonnage par hypercube latin de 4 points en dimension 2.

répartition des points dans l'espace. La qualité de répartition de points dans un espace hypercubique peut être caractérisée par de nombreux critères. On peut citer trois catégories de critères : les critères géométriques basés sur les distances entre points de l'échantillon, ceux basés sur la notion de discrépance et ceux basés sur les arbres couvrant minimaux.

Johnson et al. (1990) ont introduit deux critères basés sur la distance. Le premier repose sur l'idée que pour avoir une bonne répartition des points dans l'hypercube, la distance entre un point quelconque du domaine et le point le plus proche de celui-ci dans l'échantillon doit être la plus petite possible. Pour cela, Johnson et al. (1990) proposent de minimiser le critère suivant, appelé minimax :

$$\phi_{mM}(X_u) = \max_{\mathbf{x}\in H} \min_{1\leq i\leq n} \|\mathbf{x} - \mathbf{x}^{(i)}\|_2,$$

où $\|.\|_2$ est la norme euclidienne. Cependant, le coût calcul de ce critère le rend difficile à utiliser en pratique (Pronzato et Müller, 2012). Le second critère proposé par Johnson et al. (1990), appelé maximin, consiste à maximiser la distance minimale entre tous les points de l'échantillon, et est défini ainsi :

$$\phi_{Mm}(X_u) = \min_{1\leq i<j\leq n} \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2. \tag{3.17}$$

Bien que aisé à calculer, ce critère peut être assez difficile à maximiser en pratique. Pour pallier ce problème, Morris et Mitchell (1995) ont proposé une variante régularisée du critère maximin définie ainsi :

$$\phi_q(X_u) = \left(\sum_{1\leq i<j\leq n} \|\mathbf{x}^{(i)} - \mathbf{x}^{(j)}\|_2^{-q} \right)^{1/q}. \tag{3.18}$$

Il a été montré (Morris et Mitchell, 1995; Pronzato et Müller, 2012) que minimiser ϕ_q est équivalent à maximiser ϕ_{Mm} pour q suffisamment grand. En pratique, utiliser la valeur $q = 50$, comme proposé dans Morris et Mitchell (1995), semble suffisant dans de nombreux cas.

La discrépance mesure l'écart entre la distribution spatiale des points de la loi uniforme sur l'hypercube et l'échantillon considéré, en comparant la fonction de répartition empirique de l'échantillon à la fonction de répartition théorique de la loi uniforme. Une discrépance faible indique une bonne répartition des points dans l'espace. Il existe différentes mesures de la discrépance. Les discrépances basées sur la distance L^2 sont les plus utilisées car elles possèdent des expressions analytiques et sont plus faciles à calculer. De plus, Jin et al. (2005) ont montré que les discrépances L^2 centrées et wrap-around avaient des propriétés intéressantes comme l'invariance par rotation et des garanties d'uniformité des

sous-projections. Les formules des discrépances centrée (C^2) et wrap-around (W^2) sont respectivement :

$$\begin{aligned} C^2(X_u) &= \left(\frac{13}{12}\right)^{m_X} - \frac{2}{n}\sum_{i=1}^{n}\prod_{k=1}^{m_X}\left(1+\frac{1}{2}\left|x_k^{(i)}-\frac{1}{2}\right|-\frac{1}{2}\left|x_k^{(i)}-\frac{1}{2}\right|^2\right) \\ &\quad +\frac{1}{n^2}\sum_{i,j=1}^{n}\prod_{k=1}^{m_X}\left(1+\frac{1}{2}\left|x_k^{(i)}-\frac{1}{2}\right|+\frac{1}{2}\left|x_k^{(j)}-\frac{1}{2}\right|-\frac{1}{2}\left|x_k^{(i)}-x_k^{(j)}\right|\right) \\ W^2(X_u) &= \left(\frac{4}{3}\right)^{m_X} + \frac{1}{n^2}\sum_{i,j=1}^{n}\prod_{k=1}^{m_X}\left(\frac{3}{2}-\left|x_k^{(i)}-x_k^{(j)}\right|\left(1-\left|x_k^{(i)}-x_k^{(j)}\right|\right)\right). \end{aligned}$$

L'avantage de la discrépance wrap-around est qu'elle prend en compte l'espace en l'enroulant sur lui-même, considérant ainsi que deux points sur des bords opposés sont proches. Ce critère défavorise donc l'accumulation de points sur les bords de l'hypercube, puisque ces points sont considérés comme proches et pénalisent le critère. De plus, Damblin et al. (2013) ont montré que ces deux mesures de discrépance donnent de bons résultats en pratique. En particulier, les plans optimisés selon ces mesures ont montré une plus grande robustesse aux projections en dimension deux que les plans LHS simples ou LHS maximin.

Enfin, Franco et al. (2009) ont introduit un concept issu de la théorie des graphes, l'arbre couvrant minimal, pour l'étude de plans d'expériences. Les points du plan sont considérés comme les sommets d'un graphe. Chaque paire de sommets du graphe est relié par une arête de poids égal à la distance entre les deux points correspondant aux deux sommets. L'arbre couvrant minimal est défini comme l'arbre passant par tous les sommets du graphe ainsi défini et dont la somme des poids des arrêtes est minimale. Un plan avec de bonnes propriétés de répartition correspond à une moyenne des valeurs des arêtes élevée et un écart-type faible. Cependant, ces deux critères sont assez difficiles à optimiser en pratique (Franco et al., 2009).

Les plans LHS optimisés ont montré dans de nombreuses études de meilleurs résultats que les plans LHS simples (Pronzato et Müller, 2012; Damblin et al., 2013). Parmi les critères utilisés pour optimiser les plans LHS, ceux basés sur les arbres couvrant minimaux se révèlent difficiles à optimiser et, par conséquent, peu adaptés. Enfin, Damblin et al. (2013) ont montré que l'utilisation des critères de discrépance centré et wrap-around était préférable en pratique. En particulier, ces auteurs conseillent d'utiliser la discrépance centrée puisqu'elle a dans leur étude montré de meilleurs résultats que la discrépance wrap-around pour les critères basés sur les arbres couvrant minimaux. On utilisera donc dans la suite des plans LHS optimisés selon la discrépance centrée pour échantillonner dans les espaces de variation de variables scalaires.

3.3.1.2 Proposition d'un méthode d'échantillonnage de variables fonctionnelles

Plusieurs travaux récents ont été consacrés à l'adaptation de méthodes d'échantillonnage uniforme dans des espaces fonctionnels. Un état de l'art des méthodes existantes est présenté dans un premier temps, puis la méthode proposée est détaillée.

Pebesma et Heuvelink (1999) ont proposé un algorithme pour étendre l'échantillonnage LHS, présenté dans la section 3.3.1.1, pour l'échantillonnage de processus gaussiens. Dans leur méthode, le processus gaussien est discrétisé en un vecteur gaussien. Ils appliquent alors l'algorithme développé par Stein (1987) pour générer un plan LHS de variables dépendantes. En effet, la distribution de probabilité marginale d'un vecteur gaussien en chaque point est une variable gaussienne corrélée avec les distributions marginales aux autres points. La méthode proposée pour simuler un échantillon de réalisations d'un processus gaussien en p points est résumée par l'algorithme 3.6.

Remarque 3.3. *La méthode précédente a été proposée pour des processus gaussiens, mais elle peut être appliquée avec très peu de modifications à tout type de variable fonctionnelle dont on connaît la fonction de répartition marginale en chaque point. Il suffit en effet d'utiliser dans l'étape 3 de l'algorithme 3.6 la*

fonction de répartition de la distribution marginale de la variable fonctionnelle considérée. Cette méthode permet ainsi de générer un échantillon de réalisations d'une variable fonctionnelle selon sa distribution de probabilité et non uniformément sur son domaine de variation.

Algorithme 3.6 Création d'un plan LHS pour processus gaussiens

1. Créer un échantillon de taille n du processus gaussien, noté $\left(z^{(i)}\right)_{1\leq i\leq n}$ en p points $x_1, \ldots, x_p$.
2. Noter $s(x_j) = \left(z^{(1)}(x_j), \ldots, z^{(n)}(x_j)\right)$ les valeurs des réalisations de l'échantillon au point x_j. $r_j = (r_{j1}, \ldots, r_{jn})$ est le vecteur des rangs de $s(x_j)$.
3. Soit F_{x_j} la fonction de répartition de la distribution marginale de la variable fonctionnelle au point x_j. Partitionner $\mathbb{R}$ en n domaines I_l définis comme suit pour $l \in \{1, \ldots, n\}$:

$$I_l = \left[F_{x_j}^{-1}\left(\frac{l-1}{n}\right); F_{x_j}^{-1}\left(\frac{l}{n}\right)\right]$$

4. La nouvelle valeur de $z^{(i)}$ au point x_j est obtenue en tirant selon une loi uniforme sur l'intervalle $I_{r_{ji}}$.

Une méthode plus générale a été proposée par Morris (2012) pour l'échantillonnage uniforme de variables fonctionnelles. Il a introduit une extension du critère maximin, défini dans l'équation (3.17), au cas de variables fonctionnelles. Dans le cas traité par Morris (2012), le code de calcul $\mathcal{M}$ étudié a des entrées et des sorties fonctionnelles. Soit Z et Y respectivement les variables aléatoires fonctionnelles d'entrée et de sortie du code, dont les réalisations sont des fonctions définies sur l'intervalle $[0, T]$. On définit un plan d'expériences $D = \left\{z^{(1)}(t), \ldots, z^{(n)}(t)\right\}$ de réalisations de Z, et la distance entre deux réalisations $z^{(i)}$ et $z^{(j)}$, pour $i, j \in \{1, \ldots, n\}$ est définie de la manière suivante :

$$d^s(z^{(i)}, z^{(j)}) = \int_0^s w(s-u)\left|z^{(i)}(u) - z^{(j)}(u)\right|^2 \mathrm{d}u,$$

où w est une fonction de pondération et s est un point de $[0; T]$. On note S l'ensemble des points de $[0; T]$ pour lesquels la sortie Y doit être prédite, et on définit le critère maximin fonctionnel suivant :

$$\phi_S(D) = \min_{1\leq i<j\leq n, s\in S} d_{i,j}^s. \tag{3.19}$$

Morris (2012) fait l'hypothèse que seules les valeurs des réalisations de Z sur l'intervalle $[0; s]$ ont une influence sur la sortie au temps s, donc seules les distances aux temps de S interviennent dans le critère ϕ_S. Comme pour le critère maximin présenté dans la section 3.3.1.1, Morris (2012) propose de maximiser ce critère pour construire un plan de réalisations de Z. Dans Morris (2014), une borne supérieure est donnée pour ce critère, et il est montré que cette borne est atteinte par certains plans d'expériences. L'inconvénient de ce critère est qu'il ne peut être appliqué directement que dans le cas où la variable à échantillonner est liée à une sortie fonctionnelle définie sur le même domaine de variation, mais pas dans le cas d'une sortie scalaire.

Plus récemment, une autre méthode d'échantillonnage dans le cas d'entrées fonctionnelles a été proposée par Muehlenstaedt et al. (2014). Pour simuler selon des variables fonctionnelles, Muehlenstaedt et al. (2014) les décomposent sur une base fonctionnelle pour se ramener à un problème d'échantillonnage de variables scalaires. Dans le cas d'application traité dans Muehlenstaedt et al. (2014), l'espace d'échantillonnage est celui des fonctions définies sur $[0; 1]$ et bornées entre 0 et 1. Une base de splines cubiques est utilisée pour la décomposition, et le domaine de variation des coefficients est alors un hypercube. Ils proposent de construire, dans l'espace des coefficients, un plan LHS optimisé selon le critère ϕ_q, défini dans l'équation (3.18). Par rapport aux méthodes de Pebesma et Heuvelink (1999) et de Morris (2012),

cette méthode présente l'avantage d'être applicable quelle que soit la nature des entrées fonctionnelles et de la sortie du code.

La méthode proposée ici pour échantillonner uniformément une variable scalaire Z_k utilise des idées issues des méthodes de Morris (2012) et Muehlenstaedt et al. (2014). En effet, il est possible d'adapter le critère maximin fonctionnel défini dans l'équation (3.19) au cas d'une sortie scalaire. On propose donc le nouveau critère, noté ϕ_T et défini ainsi :

$$\phi_T(D) = \min_{1\leq i<j\leq n} d^T(z^{(i)}, z^{(j)}). \tag{3.20}$$

Dans celui-ci, seule l'intégrale des fonctions de l'échantillon sur l'ensemble de leur domaine de définition est utilisée. De plus, sous l'hypothèse d'une décomposition ACP, les variables aléatoires fonctionnelles Z_k, $1 \leq k \leq m_Z$, sont complètement décrites par les coefficients $\Gamma_1^k, \ldots, \Gamma_d^k$ de la décomposition. La distance d^T entre deux réalisations $z_k^{(i)}$ et $z_k^{(j)}$ est égale à la distance d_A définie ainsi :

$$d_A(z_k^{(i)}, z_k^{(j)}) = (\boldsymbol{\gamma}_k^{(i)} - \boldsymbol{\gamma}_k^{(j)})^T A(\boldsymbol{\gamma}_k^{(i)} - \boldsymbol{\gamma}_k^{(j)}),$$

où A est une matrice de taille $d_k \times d_k$ telle que $A_{kl} = \int_0^T w_T(T-t)\phi_k(t)\phi_l(t)\mathrm{d}t$, et où $\boldsymbol{\gamma}_k^{(i)}$ est le vecteur des coefficients de la décomposition ACP de $z_k^{(i)}$. Ainsi, la maximisation du critère de l'équation (3.20), adapté de celui proposé par Morris (2012) est équivalente à la maximisation du critère maximin sur l'espace des coefficients avec la distance d_A. De plus, en prenant la fonction de pondération w constante et égale à 1, la matrice A est égale à la matrice identité puisque la base ACP est orthonormale, et la distance d_A est alors la distance euclidienne sur $\mathbb{R}^d$.

Cependant, puisque la distribution de probabilité des coefficients est modélisée par un mélange de gaussiennes avec un support non borné, il n'est pas possible d'échantillonner uniformément sur leur domaine de variation. On propose de réaliser au préalable une troncature de leur distribution, comme dans le cas scalaire. Pour ce faire, on propose de définir une restriction du domaine de variation des coefficients inspirée de celle proposée dans la section 3.3.1.1 pour les variables scalaires. Pour chaque variable Z_k, le domaine des coefficients $\Gamma_1^k, \ldots, \Gamma_d^k$ est réduit à R_α^k défini par :

$$R_\alpha^k = \left\{x \in \mathbb{R}^d : f_k(x) \geq f_\alpha^k\right\}, \tag{3.21}$$

où f_k est la densité de probabilité jointe des coefficients $\boldsymbol{\gamma}_k$ et f_α^k est le seuil défini par

$$f_\alpha^k \in [0;1] : \int_{\mathbb{R}^d} f_k(x)\mathbb{1}_{(f_k(x)\geq f_\alpha^k)}\mathrm{d}x = 1-\alpha.$$

R_α^k est ainsi l'ensemble des points possédant une densité de probabilité plus élevée que le seuil f_α^k. Une illustration de cet ensemble dans le cas d'un mélange de deux gaussiennes dans $\mathbb{R}^2$ est donné par la Figure 3.2. L'ensemble R_α^k avec $\alpha = 5\%$ est l'espace contenu dans la ligne d'iso-probabilité (en gras), qui correspond à la valeur $f_\alpha^k = 0,006$. Le choix du paramètre α, définissant le volume de R_α^k est dépendant du cas étudié. Sa sélection est discutée au travers d'un cas d'étude analytique dans la section 3.4.

Une fois les supports des distributions fonctionnelles bornés, il est possible d'échantillonner uniformément ces variables. Contrairement au cas étudié dans Muehlenstaedt et al. (2014), l'espace d'échantillonnage n'est pas un hypercube et il n'est pas possible d'utiliser les méthodes classiques présentées dans la section 3.3.1.1 pour générer le plan de coefficients. Pour un espace non-hypercubique, peu de méthodes d'échantillonnage ont été proposées. Auffray et al. (2012) ont notamment développé une méthode d'échantillonnage de plans maximisant le critère maximin pour des espaces de forme quelconque. Leur algorithme de recuit simulé fait peu d'hypothèse sur le domaine d'échantillonnage. Il doit être borné, connexe et sa fonction indicatrice doit être connue. Le principe de l'algorithme est de sélectionner une paire de points très proches l'un de l'autre et de remplacer un de ces deux points par un point tiré aléatoirement. A chaque étape de l'algorithme, l'acceptation ou le rejet du plan contenant le nouveau point dépend d'une probabilité calculée à partir du critère maximin. On propose d'échantillonner avec l'algorithme développé par Auffray et al. (2012) dans les espaces R_α^k des coefficients en utilisant le critère maximin présenté précédemment avec la distance euclidienne sur les coefficients.

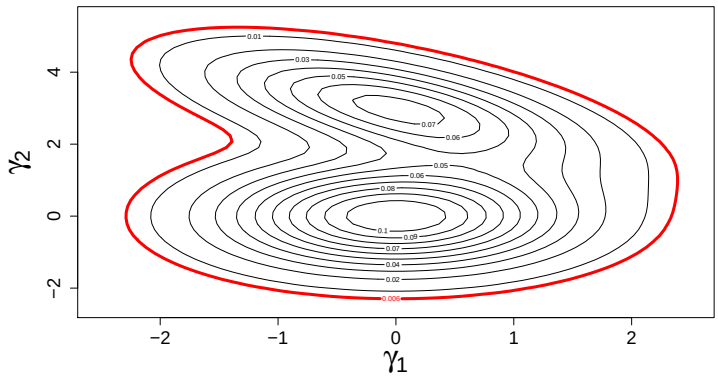

FIGURE 3.2 – Courbes d'iso-probabilité d'un mélange de deux gaussiennes dans $\mathbb{R}^2$. La courbe en gras est la frontière de l'ensemble $R_{5\%}$.

3.3.1.3 Algorithme développé pour la combinaison optimale des plans d'expériences

Les méthodes décrites dans les sections 3.3.1.1 et 3.3.1.2 peuvent être utilisées pour créer des plans d'expériences pour les variables scalaires et fonctionnelles, notés respectivement D^X et $D^{Z_1}, \dots, D^{Z_{m_Z}}$. Pour obtenir un plan d'expériences des variables $(X_1, \dots, X_{m_X}, Z_1, \dots, Z_{m_Z})$, il est ensuite nécessaire de combiner les $m_Z + 1$ plans construits, c'est-à-dire d'associer à chaque réalisation du plan D_X, une réalisation de chaque plan $D^{Z_1}, \dots, D^{Z_{m_Z}}$. Pour combiner de manière optimale ces plans, on propose dans ce qui suit une adaptation de la méthode de Muehlenstaedt et al. (2014).

On définit tout d'abord une combinaison des plans définis plus haut comme suit :

$$D(\pi_1, \dots, \pi_{m_Z}) = \left(\left(x^{(1)}, z_1^{(\pi_1(1))}, \dots, z_{m_Z}^{(\pi_{m_Z}(1))} \right), \dots, \left(x^{(n)}, z_1^{(\pi_1(n))}, \dots, z_{m_Z}^{(\pi_{m_Z}(n))} \right) \right), \tag{3.22}$$

où $\pi_1, \dots \pi_{m_Z}$ sont des permutations de $\{1, \dots, n\}$. Dans la combinaison de l'équation (3.22), seuls les indices des réalisations sont permutés ; les plans D^X et $D^{Z_1}, \dots, D^{Z_{m_Z}}$ restent inchangés. Afin de trouver la combinaison optimale $D^* = D(\pi_1^*, \dots, \pi_{m_Z}^*)$, Muehlenstaedt et al. (2014) proposent de minimiser le critère ϕ_q défini par l'équation (3.18). Pour le plan $D(\pi_1, \dots, \pi_{m_Z})$, le problème d'optimisation s'écrit de la manière suivante :

$$\min_{\pi_1, \dots, \pi_{m_Z}} \left[\sum_{1 \leq i < j \leq n} d\left(\left(x^{(i)}, z_1^{(\pi_1(i))}, \dots, z_{m_Z}^{(\pi_{m_Z}(i))} \right), \left(x^{(j)}, z_1^{(\pi_1(j))}, \dots, z_{m_Z}^{(\pi_{m_Z}(j))} \right) \right)^{-q} \right]^{1/q}.$$

En pratique, Muehlenstaedt et al. (2014) optimisent le critère ϕ_q sur les permutations $\pi_1, \dots, \pi_{m_Z}$ à l'aide d'un algorithme de recuit simulé proposé initialement par Morris et Mitchell (1995). De plus, ils utilisent dans le critère ϕ_q la distance suivante :

$$d\left(\left(x^{(i)}, z_1^{(i)}, \dots, z_{m_Z}^{(i)} \right), \left(x^{(j)}, z_1^{(j)}, \dots, z_{m_Z}^{(j)} \right) \right) = \left[\|x^{(i)} - x^{(j)}\|_2^2 + \sum_{k=1}^{m_Z} \int_I \left(z_k^{(i)}(t) - z_k^{(j)}(t) \right)^2 \mathrm{d}t \right]^{1/2}.$$

Cette distance est la somme de distances L^2 respectivement sur les espaces scalaires et fonctionnels. Dans le cas étudié par Muehlenstaedt et al. (2014), les m_X variables scalaires appartiennent à l'intervalle $[0, 1]$ et les réalisations des variables fonctionnelles sont définies sur $[0, 1]$ et bornées entre 0 et 1. Sous ces hypothèses, le carré de la distance L^2 sur chaque variable est compris entre 0 et 1. Par conséquent, Muehlenstaedt et al. (2014) utilisent la même pondération pour toutes les distances dans la distance d.

Dans un cadre plus général (si, par exemple les variables fonctionnelles ne sont pas bornées entre 0 et 1), on propose de pondérer la contribution de chaque variable afin que chaque entrée contribue de la même manière à la distance d. Comme la distance L^2 entre les variables fonctionnelles est égale à la distance euclidienne entre les coefficients (puisque la base ACP est orthonormale), la distance proposée est la suivante :

$$d_w\left(\left(x^{(i)},z^{(i)}\right),\left(x^{(j)},z^{(j)}\right)\right)=\left[\sum_{k=1}^{m_X} w_{x_k}\|x_k^{(i)}-x_k^{(j)}\|_2^2+\sum_{k=1}^{m_Z} w_{z_k}\|\boldsymbol{\gamma}_k^{(i)}-\boldsymbol{\gamma}_k^{(j)}\|_2^2\right]^{1/2},$$

où $\boldsymbol{\gamma}_k^{(i)}$ est le vecteur de coefficients de la décomposition ACP de $z_k^{(i)}$, et

$$\begin{aligned} w_{x_k} &= 1/\max_{x,x'\in I_\alpha^k}\|x-x'\|_2^2, \text{ pour } k=1,\ldots,m_X \\ w_{z_k} &= 1/\max_{\boldsymbol{\gamma}_k,\boldsymbol{\gamma}_k'\in R_\alpha^k}\|\boldsymbol{\gamma}_k-\boldsymbol{\gamma}_k'\|_2^2, \text{ pour } k=1,\ldots,m_Z, \end{aligned}$$

où I_α^k est le domaine borné de la variable X_k. Ces poids sont choisis de manière à ce que les termes $w_{x_k}\|x_k^{(i)}-x_k^{(j)}\|_2^2$ (avec $1\leq k\leq m_X$) et $w_{z_k}\|\boldsymbol{\gamma}_k^{(i)}-\boldsymbol{\gamma}_k^{(j)}\|_2^2$ (avec $1\leq k\leq m_Z$) prennent ses valeurs dans $[0,1]$. En pratique, le calcul de $\{w_{x_k}\}_{k=1,\ldots,m_X}$ est direct, puisque le domaine de variation de chaque variable scalaire est un intervalle de $\mathbb{R}$, et $\{w_{z_k}\}_{k=1,\ldots,m_Z}$ peut être estimé par une méthode de Monte Carlo, car le domaine de variation des coefficients est connu.

Dans le cas de variables fonctionnelles dépendantes, les méthodologies d'échantillonnage de variables fonctionnelles et de combinaison de plans d'expériences proposées peuvent être appliquées au groupe des variables fonctionnelles dépendantes avec quelques modifications. Comme la quantification des incertitudes est appliquée au groupe des variables fonctionnelles, ce dernier est résumé par ses coefficients $\boldsymbol{\gamma}=(\gamma_1,\ldots,\gamma_d)^T\in\mathbb{R}^d$. Le plan d'expériences D^Z du groupe peut être construit en utilisant ces coefficients avec la méthode développée par Auffray et al. (2012), comme décrit dans la section 3.3.1.2. Contrairement au cas dépendant, un seul plan D^Z est construit pour toutes les m_Z variables dépendantes, au lieu de m_Z plans. La méthodologie proposée ci-dessus peut être appliquée aux deux plans D^X et D^Z. Contrairement au cas indépendant où chaque variable fonctionnelle a son poids, dans le cas dépendant, une unique pondération w_z est associée au groupe des variables fonctionnelles.

3.3.2 Métamodélisation

Le code de calcul que l'on cherche à approcher est évalué sur les points du plan d'expériences. La base d'apprentissage constituée du plan d'expériences et des sortie du code correspondantes est utilisée ici pour construire un métamodèle entre les paramètres scalaires et fonctionnels en entrée du code $\mathcal{M}$ et sa sortie.

On utilise ici le métamodèle processus gaussien (Oakley et O'Hagan, 2002; Rasmussen et Williams, 2006), une méthode de régression couramment utilisée pour la métamodélisation de codes de calcul. Celle-ci a montré de bons résultats (Welch et al., 1992; Marrel, 2008) dans le cadre de l'approximation de codes de calcul, même en dimension élevée. A noter cependant que d'autres types de métamodèle auraient pu être utilisés, la technique d'échantillonnage n'étant pas spécifiquement adaptée au processus gaussien. Dans le cadre du métamodèle processus gaussien, la réponse Y d'un code de calcul est considérée comme la réalisation d'un processus gaussien stochastique Y_{Gp}, dépendant des entrées X du code comme suit :

$$Y_{Gp}(X)=f(X)+Z(X), \tag{3.23}$$

où f est une fonction déterministe, et Z est un processus gaussien centré, caractérisé par sa fonction de covariance C. Pour une entrée $x^{(0)}$, on montre que la distribution conditionnelle du processus gaussien sachant un échantillon d'apprentissage $(X_s,Y_s)=\left((x^{(1)},y^{(1)}),\ldots,(x^{(n)},y^{(n)})\right)$ est aussi une gaussienne dont la moyenne et la variance s'écrivent ainsi (cf. Rasmussen et Williams 2006) :

$$\mathbb{E}\left(Y_{Gp}(x^{(0)})\,|X_s,Y_s\right)=f(x^{(0)})+c(x^{(0)})^tC_s^{-1}(Y_s-F_s), \tag{3.24}$$

$$\mathrm{Var}\left(Y_{Gp}(x^{(0)})\,|X_s,Y_s\right)=C(x^{(0)},x^{(0)})-c(x^{(0)})^tC_s^{-1}c(x^{(0)}), \tag{3.25}$$

où $F_s = f(X_s)$, $c(x^{(0)}) = \left(C(x^{(0)}, x^{(1)}), \ldots, C(x^{(0)}, x^{(n)})\right)^t$ est le vecteur de covariance entre $x^{(0)}$ et les points de X_s, et $C_s = \left(C(x^{(i)}, x^{(j)})\right)_{1 \leq i,j \leq n}$ est la matrice de covariance de l'échantillon d'apprentissage.

Le prédicteur du métamodèle est la moyenne conditionnelle (3.24). La variance conditionnelle (3.25) représente l'erreur quadratique moyenne de prédiction du métamodèle. En pratique, la tendance f et la fonction de covariance sont paramétrées. Leurs paramètres peuvent être estimés par maximum de vraisemblance (Rasmussen et Williams, 2006; Marrel et al., 2008) ou par validation croisée (Bachoc, 2013).

3.4 Applications

3.4.1 Application à un modèle analytique

On reprend le modèle analytique proposé dans les sections 2.2.7.1 et 2.3.6.1. On rappelle brièvement sa définition. Le modèle $\mathcal{M}$ considéré dépend de trois variables scalaires X_1, X_2, X_3 et de deux variables fonctionnelles Z_1, Z_2. Les variables scalaires ont une distribution uniforme sur $[0; 1]$ et les variables fonctionnelles sont des processus gaussiens corrélés entre eux. La sortie de $\mathcal{M}$ est une variable scalaire notée Y. Dans le chapitre 2, la distribution de probabilité du couple de variables fonctionnelles a été modélisée par la méthode de quantification des incertitudes proposée. Ainsi, un échantillon de $n = 100$ réalisations de Z_1 et Z_2 a été décomposé simultanément sur une base SPLS de taille $d = 10$, puis la distribution de probabilité de ces coefficients a été modélisée par un mélange de gaussiennes, dont les paramètres ont été estimés par l'algorithme sEM2.2 (cf. section 2.3.4.2). On suppose dans cette section que les sorties de $\mathcal{M}$ correspondantes ne sont pas disponibles, et donc qu'il n'y a pas de covariable. L'échantillon de 100 réalisations des variables fonctionnelles est décomposé sur une base ACP simultanée. La distribution de probabilité de leurs coefficients sur la base ACPS est modélisée par un mélange de 3 gaussiennes estimé par l'algorithme sEM2.2. L'objectif de cette section est d'approcher le code $\mathcal{M}$ par un métamodèle pour réaliser son analyse de sensibilité.

Pour apprendre ce métamodèle, on échantillonne uniformément les variables d'entrées du code. Pour ce faire, on utilise la méthode d'échantillonnage de variables scalaires et fonctionnelles présentée dans la section 3.3.1. Les domaines de variation des trois variables scalaires sont bornés et ne nécessitent donc pas de troncature préalable. Un plan LHS, noté D^X, optimisé selon le critère de discrépance centrée (Jin et al., 2005) et de taille 100 est généré sur l'hypercube $[0; 1]^3$. Pour générer les réalisations du couple de variables fonctionnelles, leur domaine de variation doit tout d'abord être borné. Comme décrit dans la section 3.3.1.2, un sous-ensemble R_α, avec $\alpha \in]0; 1[$, est estimé dans l'espace des coefficients, et l'échantillon D^Z est généré en simulant uniformément des coefficients sur R_α. Les deux plans d'expériences D^X et D^Z sont ensuite combinés selon la méthode présentée dans la section 3.3.1.3. Le code $\mathcal{M}$ est évalué sur le plan ainsi obtenu pour constituer la base d'apprentissage. Un métamodèle processus gaussien est ensuite appris comme décrit dans la section 3.3.2. Sa tendance f (cf. équation (3.23)) est un polynôme de degré 1 des variables d'entrée, et la fonction de covariance du processus gaussien est un noyau de Matèrn 5/2 stationnaire, anisotrope et tensorisé (Rasmussen et Williams, 2006). Les paramètres de la tendance et du noyau sont estimés par maximum de vraisemblance.

Dans cet exemple, cette opération est répétée avec plusieurs valeurs de α comprises entre 0,001 et 0,5 pour comparer l'effet de α sur la qualité du métamodèle construit. De plus, cette méthode est répétée avec 50 plans D^X et D^Z différents pour une même valeur de α. Pour évaluer l'efficacité de la méthode, on compare la prédictivité des métamodèles obtenus avec celle des métamodèles obtenus sans échantillonnage uniforme mais à partir d'échantillons probabilisés des variables d'entrée. Pour ce faire, on calcule le coefficient de prédictivité Q^2 de chacun de ces métamodèles sur une base de test probabilisée. La comparaison des deux approches par le coefficient Q^2 est biaisée et avantage le métamodèle appris sur un plan probabilisé puisque la base de test est aussi probabilisée. Les Q^2 calculés sont représentés sur la Figure 3.3 en noir pour les métamodèles appris sur des échantillons uniformes et en rouge (trait gras) pour ceux appris sur des échantillons probabilisés. Les Q^2 sont tous élevés, donc les métamodèles estiment bien le code de calcul. Pour la plupart des paramètres α testés, les Q^2 calculés sur les métamodèles appris

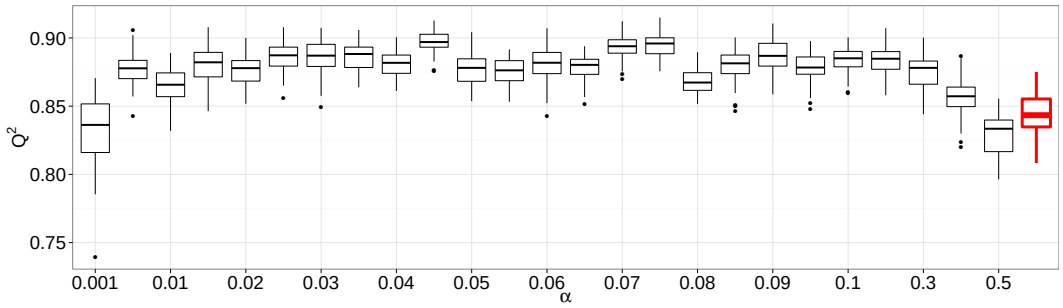

FIGURE 3.3 – Exemple analytique : coefficient de prédictivité Q^2 des métamodèles construits sur un plan uniforme en fonction de α (en noir) et sur des échantillons probabilisés (en trait gras et rouge).

Variables	X_1	X_2	X_3	(Z_1, Z_2)
Indices de référence	0,033	0,083	0,12	0,74
Méthodologie proposée	0,036	0,10	0,11	0,73

TABLE 3.1 – Exemple analytique : indices de Sobol' des trois variables scalaires et du groupe de variables fonctionnelles calculés directement sur $\mathcal{M}$ et avec la méthodologie proposée.

sur les échantillons uniformes sont plus élevés que ceux des métamodèles appris sur les échantillons probabilisés. La prédictivité du métamodèle se dégrade pour de très faibles valeurs de α ($\alpha \leq 0,005$). En effet, à nombre de points constants, la distance entre les points de l'échantillon augmente quand le volume de R_α augmente (*i.e.* α diminue), puisque le plan d'expériences est uniforme. Cette augmentation des distances dans la base d'apprentissage entraîne une détérioration de l'apprentissage du métamodèle. A l'inverse, quand les valeurs de α sont élevées, le volume de R_α diminue et l'information non prise en compte dans le métamodèle augmente, ce qui peut expliquer sa moins bonne prédictivité pour $\alpha \geq 0,2$. Les Q^2 sont à peu près constants pour un large spectre de α (approximativement de 0,005 à 0,2). Par conséquent, dans cet intervalle, α semble avoir une influence limitée sur la qualité du métamodèle. **Dans cet exemple, on choisit par la suite de prendre $\alpha = 0,05$.**

L'analyse de sensibilité est réalisée sur le métamodèle construit sur un échantillon uniforme généré avec $\alpha = 0,05$. Les indices de Sobol' du premier ordre des trois variables scalaires et du groupe de variables fonctionnelles sont calculés à l'aide du métamodèle et de la méthode des plans répliqués décrite dans la section 3.2.2. 2.10^5 appels au métamodèle sont effectués. Les indices de référence sont calculés directement sur le modèle $\mathcal{M}$ avec le même nombre d'évaluations du code et en utilisant la distribution réelle des variables fonctionnelles. Les indices de Sobol' de référence et ceux estimés sont donnés par la Table 3.1, et l'erreur absolue entre les indices calculés selon ces deux méthodes est inférieure à 2.10^{-2}. **La méthodologie proposée donne donc ici une bonne estimation des indices de Sobol' avec seulement 100 évaluations du code $\mathcal{M}$.**

On compare maintenant la méthodologie d'échantillonnage que nous proposons à trois autres méthodes :

- l'échantillonnage probabilisé, méthode notée M_1,
- l'échantillonnage uniforme par la méthode de rejet, méthode notée M_2,
- la méthodologie proposée sans l'étape de combinaison optimisée des deux plans, *i.e.* en combinant aléatoirement les plans D^X et D^Z, méthode notée M_3.

La différence entre les méthodes M_2 et M_3 est que dans la seconde les plans d'expériences D^X et D^Z sont optimisés alors que dans la première le tirage est aléatoire. La différence entre la méthode M_3 et la méthodologie proposée réside uniquement dans le fait que, dans la seconde, la combinaison entre D^X

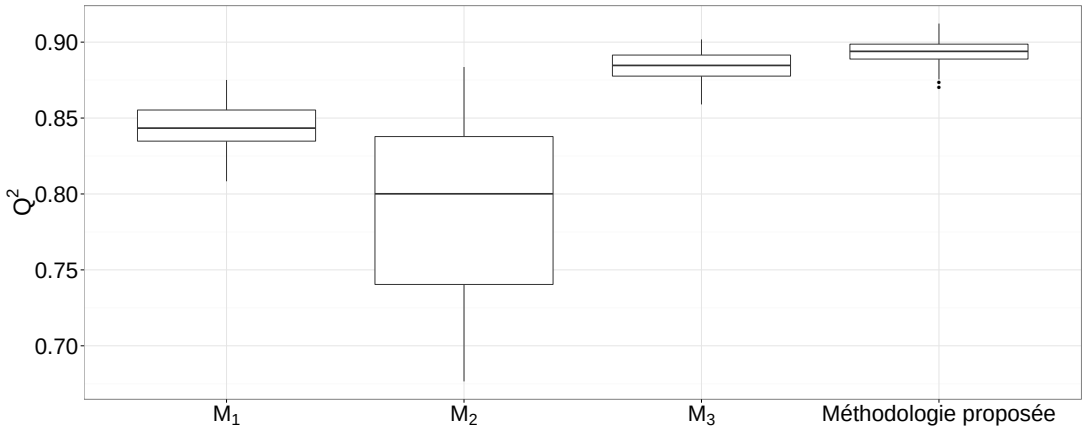

FIGURE 3.4 – Exemple analytique : coefficient de prédictivité Q^2 des métamodèles construits sur les plans d'expériences obtenus avec les méthodes M_1, M_2, M_3 et la méthodologie proposée.

et D^Z est optimisée selon la méthode décrite dans la section 3.3.1.3. Tout d'abord, les coefficients de prédictivité Q^2 des métamodèles construits sur les plans obtenus avec ces quatre méthodes sont calculés sur une base de test probabilisée de 1000 points pour 50 plans d'expériences différents. Les boxplots de ces coefficients sont représentés sur la Figure 3.4. Les Q^2 des méthodes M_1, M_3 et de la méthodologie proposée sont élevés : ils sont toujours supérieurs à 0,8. La méthodologie d'échantillonnage proposée donne les coefficients Q^2 les plus élevés et les moins variables. La méthode M_3 a une variance plus élevée et une médiane plus faible que la méthode proposée, mais donne de meilleurs résultats que les méthodes M_1 et M_2. La méthode de rejet, M_2, donne les moins bons résultats et a une variance bien plus élevée que les trois autres méthodes. **Ainsi, au vu des résultats des méthodes M_2 et M_3, l'utilisation de plans optimisés pour échantillonner D^X et D^Z améliore grandement la qualité des métamodèles sur cet exemple, et l'optimisation de la combinaison des plans D^X et D^Z a une influence plus faible sur la qualité de l'échantillonnage mais réduit quand même l'erreur de prédiction et sa variance.**

Enfin, on compare les estimations des indices de Sobol' obtenues avec ces quatre méthodes. Les erreurs absolues entre les indices de référence, donnés dans la Table 3.1, et les indices obtenus en réalisant l'analyse de sensibilité à partir des 50 métamodèles construits précédemment avec les quatre méthodes sont représentées sur la Figure 3.5. Les erreurs sont faibles pour toutes les méthodes sauf M_2. Pour la méthodologie proposée, par exemple, elles sont toutes inférieures à 0,05. Comme pour le coefficient Q^2, la méthode de rejet, M_2, donne les moins bons résultats. Les erreurs obtenues avec l'échantillonnage probabilisé, M_1, sont élevées pour les indices de X_2 et du groupe (Z_1, Z_2), alors qu'elles sont faibles pour ceux de X_1 et X_3. La méthode M_1 donne notamment les plus faibles erreurs pour l'indice de X_3. La méthodologie proposée donne les plus faibles erreurs pour les indices de X_1 et (Z_1, Z_2). La méthode M_3 a des erreurs et une variance bien plus élevées pour les indices de X_3 et de (Z_1, Z_2) que la méthodologie proposée, alors qu'elle est meilleure pour l'indice de X_2. La méthodologie proposée donne donc globalement de meilleurs résultats que les trois autres méthodes. La hiérarchie entre les quatre méthodes observée avec le coefficient Q^2 est donc à peu près conservée sur les erreurs d'estimation des indices de Sobol', bien que les différences entre les méthodes soient moins tranchées.

3.4.2 Application au cas-test du choc thermique pressurisé

L'objectif de cette section est de réaliser une analyse de sensibilité sur le code de calcul CAST3M dans le cas du choc thermique pressurisé présenté dans l'introduction (cf. section 1.2.1). Le code CAST3M correspond au premier cas considéré dans la section 3.1. Son temps d'évaluation est assez court (environ 2

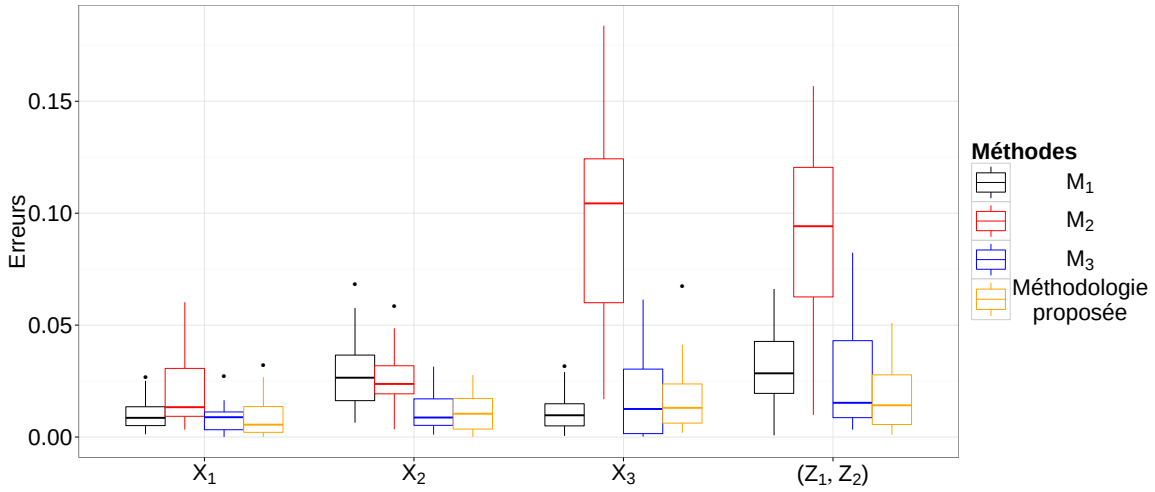

FIGURE 3.5 – Exemple analytique : boxplots des erreurs absolues entre les indices de référence et les indices obtenus en utilisant les métamodèles construits sur les plans d'expériences échantillonnés selon les méthodes M_1, M_2, M_3 et la méthodologie proposée.

minutes), et l'analyse de sensibilité peut être réalisée directement sans utiliser de métamodèle. La sortie de CAST3M considérée est le critère de sécurité, une variable scalaire notée Y. Le code CAST3M possède dix entrées scalaires incertaines, définies dans la Table 1.2, et trois entrées fonctionnelles incertaines (température, pression, coefficient d'échange thermique), notées Z_1, Z_2, Z_3. Les dix variables scalaires sont indépendantes entre elles et indépendantes des entrées fonctionnelles. Leur distribution de probabilité est connue. Les trois entrées fonctionnelles sont dépendantes entre elles. Un échantillon probabilisé de 1000 réalisations de (Z_1, Z_2, Z_3) est disponible. Pour chacune de ces réalisations, un tirage aléatoire probabilisé des variables scalaires a été réalisé et le code CAST3M a été évalué pour le jeu de paramètre d'entrée ainsi constitué. Ainsi, une réalisation du CS a été associée à chaque réalisation de (Z_1, Z_2, Z_3). A partir des réalisations des trois variables fonctionnelles et des sorties de CAST3M correspondantes, la distribution de probabilité jointe des variables fonctionnelles a été estimée dans les sections 2.2.7.2 et 2.3.6.2. Les trois variables fonctionnelles ont été décomposées sur une base SPLS de dix composantes, et les coefficients correspondants ont été modélisés, dans la section 2.3.6.2, par un mélange de gaussiennes, dont les paramètres ont été estimés par la méthode sEM2.3.

Les incertitudes des variables scalaires et fonctionnelles étant modélisées, une analyse de sensibilité peut alors être réalisée à partir de ces coefficients, comme décrit dans la section 3.2.3.2. Pour calculer les indices de Sobol' du premier ordre, la méthode basée sur les plans d'expériences répliqués est mise en œuvre avec 2.10^4 appels au code CAST3M. Les indices du premier ordre sont représentés sur la Figure 3.6. Les intervalles de confiance asymptotiques à 95% des variables scalaires sont aussi représentés. Ces intervalles de confiance asymptotiques proposés par Tissot et Prieur (2015) ne sont définis que pour des variables scalaires et ne peuvent donc pas être calculés pour le groupe de variables fonctionnelles. La hauteur H du défaut est la variable la plus influente, puisque son indice vaut environ 0,6. Le groupe de variables fonctionnelles est le deuxième facteur le plus important avec un indice du premier ordre de 0,1. Le troisième paramètre le plus influent est K_{IC}, la ténacité de l'acier. Les autres indices sont assez faibles. La somme des indices du premier ordre représente environ 0,95 : les interactions entre variables d'entrée ont très peu d'influence sur l'incertitude du critère de sécurité.

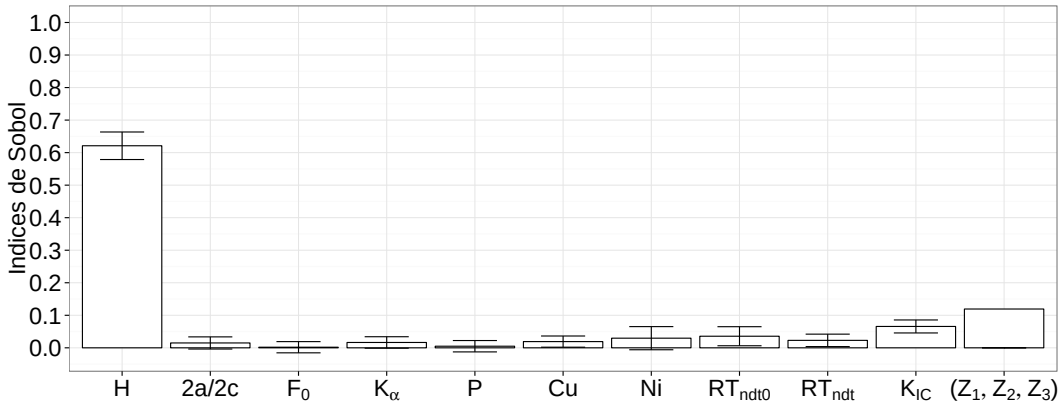

FIGURE 3.6 – Cas PTS : indices de Sobol' du premier ordre des entrées scalaires du code CAST3M et du groupe de ses entrées fonctionnelles, ainsi que les intervalles de confiance asymptotiques à 95% des indices des entrées scalaires.

3.4.3 Application au cas-test de la rupture LiPoSo

Le cas-test de la rupture LiPoSo présenté dans l'introduction (cf. section 1.2.2) est étudié dans cette section. L'objectif est de réaliser l'analyse de sensibilité du code de calcul Trio_U MCT qui calcule les températures du sodium et des gaines des aiguilles dans l'ensemble du réacteur. On s'intéresse plus particulièrement à quatre sorties du code. Ces quatre sorties d'intérêt sont décrites dans l'introduction. Deux d'entre elles sont des fonctions du temps définies sur l'intervalle $[0, 20]$ et notées respectivement Y_{Na} et Y_{NG}. Les deux autres sont les maxima des fonctions précédentes, notés respectivement $Y_{Na}^{max} = \max_{t\in[0,20]} Y_{Na}(t)$ et $Y_{NG}^{max} = \max_{t\in[0,20]} Y_{NG}(t)$. En entrée du code Trio_U MCT, trois paramètres fonctionnels et sept paramètres scalaires sont considérés comme incertains. Les trois variables fonctionnelles Z_1, Z_2, Z_3 sont les évolutions temporelles du débit, de la puissance et de la température. Leur distribution de probabilité a été modélisée dans la section 2.3.6.3 à partir d'un échantillon disponible de 200 réalisations. Dans la modélisation retenue, une translation est tout d'abord appliquée à la puissance. Ensuite, les variables sont décomposées sur une base ACPS de 10 composantes. La distribution de probabilité des coefficients de la décomposition est ensuite modélisée par un mélange de trois gaussiennes dont les paramètres sont estimés par l'algorithme EM. Les sept entrées scalaires $X_1, \ldots, X_7$ de Trio_U MCT sont décrites dans la Table 1.3. Les paramètres X_6 et X_7 sont des coefficients multiplicatifs appliqués respectivement aux fonctions de puissance (Z_2) et de débit (Z_1) dans le code Trio_U MCT. Comme on s'intéresse ici à l'influence des transitoires de débit, puissance et température ainsi transformés par rapport à celle des autres variables, on cherche à estimer les indices des variables $X_1, \ldots, X_5$ et l'indice du groupe de variables thermo-hydrauliques $(X_6, X_7, Z_1, Z_2, Z_3)$ qui représente l'incertitude sur les transitoires.

Compte tenu du temps cpu nécessaire à chaque simulation du code Trio_U MCT (environ 2 heures), il est nécessaire d'approcher ce dernier par un métamodèle pour en réaliser l'analyse de sensibilité. Le métamodèle est appris sur un échantillon uniforme des variables d'entrée. On restreint dans un premier temps les intervalles de variation des variables non-bornées (X_1, X_2, X_4, X_7) aux intervalles dont les bornes sont les quantiles $\alpha/2$ et $1 - \alpha/2$ de leurs distributions respectives définies dans la Table 1.3. Les domaines de variation des trois autres variables scalaires (bornées) sont inchangés. Pour les variables fonctionnelles, le domaine R_α est construit dans l'espace des coefficients de la base ACPS en utilisant la densité de probabilité modélisée dans la section 2.3.6.3. Comme il a été illustré dans la section 3.4.1 que le paramètre α pris dans $[0, 005; 0, 2]$ avait une influence limitée sur la qualité du métamodèle, on choisit ici de fixer α à 0,05. On construit alors un échantillon uniforme de 100 jeux de paramètres d'entrée. Un LHS optimisé selon la discrépance centrée, noté D^X, est construit pour les variables scalaires, et

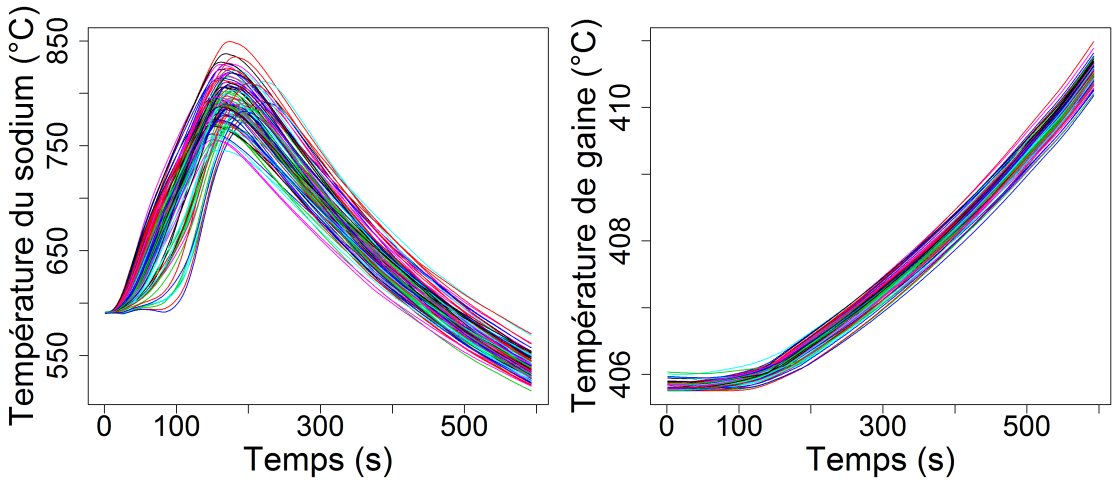

FIGURE 3.7 – Cas LiPoSo : échantillon de 100 réalisations des courbes de température du sodium (gauche) et de gaine (droite) dans la dérivation la plus chaude.

un échantillon D^Z est simulé sur $R_{0,05}$ avec la méthode de Auffray et al. (2012). Les 100 réalisations de l'échantillon D^Z sont représentées sur la Figure 3.7. Les deux échantillons D^X et D^Z sont combinés suivant la méthode proposée dans la section 3.3.1.3, puis le code est appelé sur les 100 jeux de paramètres obtenus.

Deux métamodèles de type processus gaussien sont ensuite construits sur l'échantillon combiné et sur les réalisations de Y_{Na}^{max} et Y_{NG}^{max} correspondantes, comme décrit dans la section 3.3.2. Les coefficients de prédictivité Q^2 des deux métamodèles sont calculés par leave-one-out. Ils valent respectivement 0,98 et 0,99. Les deux métamodèles approchent donc avec précision les deux sorties scalaires étudiées.

Pour réaliser l'analyse de sensibilité des deux sorties temporelles Y_{Na} et Y_{NG}, il est nécessaire de construire tout d'abord un métamodèle approchant ces deux sorties. Etant donné la discrétisation de ces variables, il n'est pas possible de les considérer comme des sorties vectorielles et de construire un métamodèle pour chaque point de leur discrétisation. On utilise ici une méthode employée avec succès par Shi et al. (2007), Bayarri et al. (2007) et Marrel et al. (2011) qui consiste à décomposer la sortie sur une base fonctionnelle puis à construire un métamodèle pour chaque coefficient de la décomposition. Ici, on utilise la décomposition ACP, car elle minimise l'erreur quadratique d'approximation. Pour chacune des sorties, 10 composantes de l'ACP sont retenues. Les variances expliquées par les deux bases de décomposition, définies dans l'équation (2.11), sont respectivement 0,99987 et 0,99998. Ces deux sorties sont donc très bien approchées par leur projection sur la base ACP. Un métamodèle processus gaussien est appris entre les variables d'entrée et chaque coefficient de la décomposition. La prédictivité du métamodèle est évaluée par le coefficient de prédictivité Q^2 en chaque temps, défini comme suit pour une variable fonctionnelle $Y(t)$, $t \in I$:

$$Q^2(t) = \frac{\sum_{i=1}^{n} \left(y^{(i)}(t) - \hat{y}^{(i)}(t)\right)^2}{\sum_{i=1}^{n} \left(y^{(i)}(t) - \bar{y}(t)\right)^2},$$

où $y^{(1)}, \ldots, y^{(n)}$ sont les réalisations de Y, $\bar{y}(t) = \frac{1}{n}\sum_{i=1}^{n} y^{(i)}(t)$ est leur moyenne, et $\hat{y}^{(i)}$ leurs prédictions par le métamodèle. Ce coefficient calculé par leave-one-out est représenté respectivement en ligne continue noire et en pointillés rouge pour Y_{Na} et Y_{NG} sur la Figure 3.8. Pour Y_{NA}, le Q^2 est très élevé sauf pour des temps très faibles (inférieurs à 0,1 seconde). Pour Y_{NG}, le coefficient $Q^2(t)$ est élevé la plupart du temps. Il est inférieur à 0,8 sur l'intervalle $[0; 0,89]$. Les deux métamodèles donnent donc une très bonne approximation sur presque tout l'intervalle de définition de la sortie. néanmoins, la prédictivité des deux métamodèles est moins bonne sur le tout début de l'intervalle. Le deuxième critère calculé pour évaluer

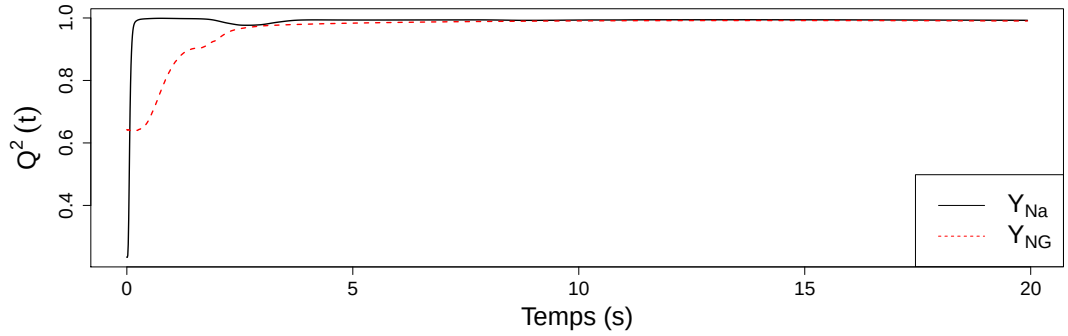

FIGURE 3.8 – Cas LiPoSo : coefficient de prédictivité $Q^2(t)$ calculés sur les métamodèles de Y_{Na} (ligne noire) et Y_{NG} (pointillés rouges).

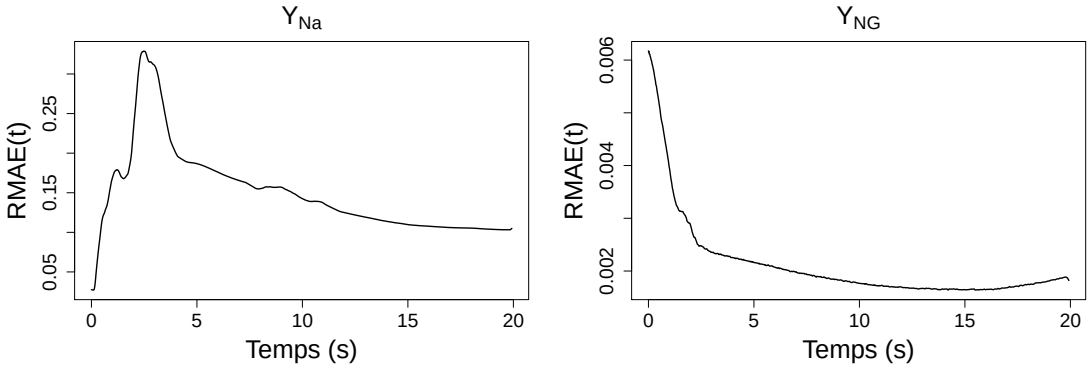

FIGURE 3.9 – Cas LiPoSo : RMAE(t) calculés sur les métamodèles de Y_{Na} (à gauche) et Y_{NG} (à droite).

la qualité des deux métamodèles fonctionnels est l'erreur moyenne relative absolue en fonction du temps, calculée par leave-one-out, et définie ainsi :

$$\mathrm{RMAE}(t) = \frac{100}{n} \sum_{i=1}^{n} \frac{\left|y^{(i)}(t) - \hat{y}^{(i)}(t)\right|}{\left|y^{(i)}(t)\right|}.$$

Les RMAE des métamodèles de Y_{Na} et Y_{NG} sont représentées sur les parties gauche et droite de la Figure 3.9. Les erreurs obtenues sont faibles, en particulier pour la température de gaine. On peut conclure de l'étude de ce critère que les deux métamodèles donnent une bonne approximation des températures de gaine et du sodium.

L'analyse de sensibilité peut ensuite être réalisée à partir des quatre métamodèles ainsi construits. La méthode d'estimation utilisée est celle basée sur les plans d'expériences répliqués présentée en section 3.2.2. Pour chaque analyse de sensibilité, 10^6 appels au métamodèle sont réalisés. Les indices de Sobol' des deux sorties scalaires sont donnés dans la Table 3.2. Pour Y_{Na}, le groupe de variables T-H a une influence prédominante. Les autres variables ont une influence nulle ou quasi nulle. Pour Y_{NG}, l'influence du groupe de variables T-H est grande, mais la variable X_2, l'épaisseur des gaines, a aussi une influence non négligeable puisqu'elle représente environ 14% de la variabilité de la sortie. Les autres variables sont, comme pour Y_{Na}, très peu influentes.

Pour réaliser l'analyse de sensibilité d'un code à sortie fonctionnelle, une méthode classique est de traiter la sortie comme un vecteur en la discrétisant et de réaliser une analyse de sensibilité pour chaque élément du vecteur. Une valeur des indices de Sobol' est alors obtenue pour chaque point de la discrétisa-

Variables	X_1	X_2	X_3	X_4	X_5	$(X_6, X_7, Z_1, Z_2, Z_3)$
Indices pour Y_{Na}^{max}	0	0,003	0	0	0,001	0,996
Indices pour Y_{NG}^{max}	0,002	0,136	0,001	0,004	0,001	0,858

TABLE 3.2 – Cas LiPoSo : indices de Sobol' calculés pour les sorties Y_{Na}^{max} et Y_{NG}^{max}.

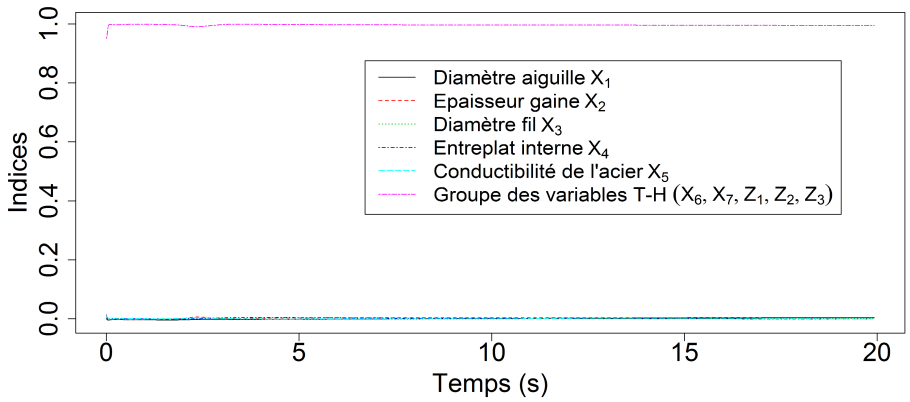

FIGURE 3.10 – Cas LiPoSo : indices de Sobol' du premier ordre pour la température du sodium sur la dérivation la plus chaude en fonction du temps.

tion, et une courbe d'indices de sensibilité peut être représentée. Cette méthode est appliquée ici pour les variables Y_{Na} et Y_{NG}, et les indices de Sobol' à chaque temps de discrétisation pour ces deux variables sont représentés sur les Figures 3.10 et 3.11. Pour Y_{Na}, le groupe de variables T-H est prédominant sur tout l'intervalle. Pour $t = 0$, l'indice du groupe de variables T-H diminue très légèrement mais reste de loin le plus élevé. Les autres variables ont une influence très faible. Pour la température de gaine Y_{NG}, le groupe de variables T-H a toujours la plus grande influence, mais comme dans le cas de Y_{NG}^{max}, l'épaisseur de gaine a une influence significative sur presque tout l'intervalle. Au début de l'intervalle de temps, l'indice du diamètre du fil est plus élevé, alors que sur le reste de l'intervalle il est à peu près nul. Notons que les indices de Y_{NG} à $t = 20s$ sont égaux aux indices de Y_{NG}^{max}, puisque toutes les courbes de Y_{NG} sont maximales à $t = 20s$ (comme on peut le voir sur la partie droite de la Figure 3.7).

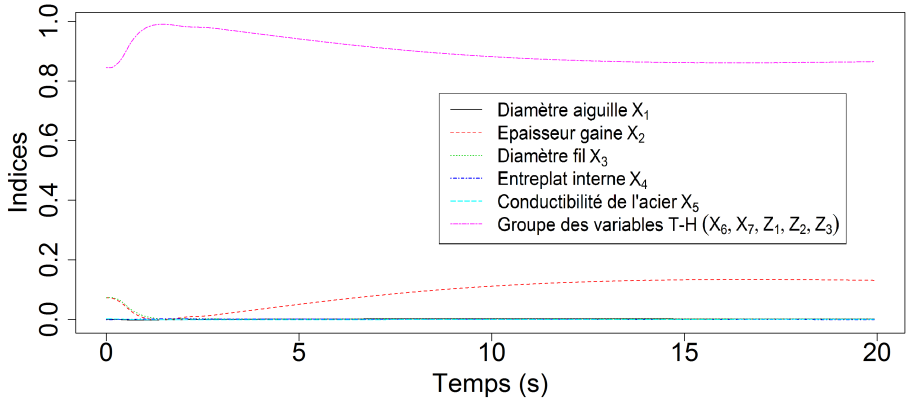

FIGURE 3.11 – Cas LiPoSo : indices de Sobol' du premier ordre pour la température de gaine sur la dérivation la plus chaude en fonction du temps.

3.5 Synthèse

Dans ce chapitre, l'analyse de sensibilité globale des codes de calcul à entrées fonctionnelles a été étudiée. Dans la section 3.2, les indices de sensibilité globale basés sur la variance, appelés indices de Sobol' puis des méthodes d'estimation de ces indices ont été présentés. Un état de l'art des méthodes d'estimation des indices de Sobol' dans le cas d'entrées fonctionnelles a ensuite été proposé. Enfin, plusieurs adaptations de ces indices au cas d'entrées dépendantes ont été présentées.

Dans les cas d'étude du choc thermique pressurisé (PTS) et de la rupture LiPoSo, les simulateurs utilisés pour modéliser et prédire les variables caractéristiques du phénomène ont en commun deux caractéristiques principales. Leurs entrées sont fonctionnelles (transitoires thermo-hydrauliques) et scalaires, et elles sont possiblement dépendantes. De plus, la distribution de probabilité des variables d'entrée fonctionnelles n'est connue qu'à travers un nombre limité de leurs réalisations (obtenues en sortie du code CATHARE2), alors que la distribution des variables scalaires est connue. La différence entre les deux applications réside dans le temps de calcul nécessaire à chaque évaluation du simulateur : le code utilisé pour prédire le critère de sécurité (CS) dans le cas du PTS a un temps d'exécution rapide (quelques minutes), alors que celui utilisé pour calculer les températures de gaine et du sodium dans le cas LiPoSo est beaucoup plus lent (quelques heures). **Dans chacune de ces deux configurations, une méthode a été proposée pour réaliser l'analyse de sensibilité.**

Pour le cas d'étude du PTS, les incertitudes des variables fonctionnelles en entrée du calcul du CS sont modélisées en utilisant la méthode de quantification des incertitudes développée dans le chapitre 2. Celle-ci permet notamment de prendre en compte la dépendance des variables fonctionnelles entre elles. De plus, l'utilisation d'une décomposition *Partial Least Squares* (PLS) simultanée avec comme covariable la sortie du code de calcul (cf. section 2.2.5) permet de prendre en compte le lien entre le CS et les variables d'entrée fonctionnelles. Ainsi, les caractéristiques des variables fonctionnelles expliquant le mieux la sortie du code sont conservées dans la quantification des incertitudes, ce qui contribue à améliorer les résultats de l'analyse de sensibilité. Ensuite, une analyse de sensibilité est réalisée sur le code de calcul à l'aide de la méthode des plans répliqués. Pour prendre en compte la dépendance entre variables d'entrée fonctionnelles, des indices de groupe (cf. section 3.2.4) sont utilisés dans cette analyse de sensibilité. Cette méthodologie, appliquée au cas d'étude du PTS, a permis d'obtenir des estimations assez précises des indices de Sobol' du premier ordre des entrées du code. Ces résultats ont révélé la très forte influence de la hauteur du défaut sur l'incertitude du CS. Les transitoires thermo-hydrauliques et la ténacité de la cuve du réacteur ont aussi une influence non négligeable (de l'ordre de 10% chacune).

Pour le cas d'étude de la rupture LiPoSo, la méthode de quantification proposée dans le chapitre 2 a aussi été utilisée pour modéliser l'incertitude des transitoires T-H en entrée du code Trio_U MCT. En raison du temps de calcul important du code Trio_U MCT, le lien entre les variables fonctionnelles et les sorties du code n'a pas été pris en compte dans la caractérisation des incertitudes. En l'absence de covariable, une décomposition Analyse en Composantes Principales (ACP) simultanée a été utilisée pour réaliser la caractérisation statistique des transitoires T-H, comme préconisé dans le schéma 2.40. Dans un deuxième temps, un métamodèle a été construit pour approcher le code de calcul. Pour construire un métamodèle approchant le code sur tout le domaine de variation des entrées indépendamment de leur distribution de probabilité, il a été proposé de générer un échantillon d'apprentissage uniformément sur ce domaine. **Une méthodologie a été développée afin de générer un échantillon uniforme de variables scalaires et fonctionnelles.** Pour cela, le domaine de variation des variables scalaires non bornées est restreint en excluant les zones de faible densité. Sur ce domaine, les variables scalaires sont ensuite échantillonnées à partir d'un plan LHS optimisé selon la discrépance centrée (Fang et al., 2006). Pour les variables aléatoires fonctionnelles, leur support n'étant pas borné, il a été proposé de le restreindre en ne gardant que les régions de plus hautes densités déterminées en utilisant la distribution de probabilité estimée lors de l'étape de caractérisation (troncature de niveau α). Un échantillon de réalisations optimisé selon le critère *maximin* a ensuite été généré sur le support borné à partir des coefficients de la décomposition fonctionnelle (Auffray et al., 2012). Pour combiner de manière optimale les plans des variables fonctionnelles et scalaires, une méthode adaptée de celle de Muehlenstaedt et al.

(2014) a été proposée. Celle-ci optimise le plan combiné selon une version régularisée du critère *maximin* (Morris et Mitchell, 1995). Le code de calcul est ensuite évalué sur le plan d'expériences ainsi construit. A partir de ce plan d'expériences et des sorties correspondantes du code, il a été proposé de construire un métamodèle de type processus gaussien expliquant la sortie du code en fonction de ses entrées scalaires et des coefficients de la décomposition des variables fonctionnelles. Une analyse de sensibilité globale a pu ensuite être réalisée en estimant directement les indices de sensibilité à partir du métamodèle (approche de type Monte Carlo, basée sur des plans répliqués). La méthodologie utilisée dans ces deux configurations est résumée par le schéma de la Figure 3.12. Les méthodes d'échantillonnage et d'analyse de sensibilité utilisées sont indépendantes du type de métamodèle construit, et d'autres types de métamodèle auraient pu être utilisés.

Cette méthodologie a été mise en œuvre sur le cas analytique introduit dans la section 2.2.7.1 et sur le cas de la rupture LiPoSo. Dans l'exemple analytique, l'influence du niveau de troncature α a été étudiée. Dans ce cas, pour un grand intervalle de valeurs de α (entre 0,005 et 0,2), la prédictivité du métamodèle est quasiment constante. Ainsi, le paramètre α, pris dans cet intervalle, semble avoir très peu d'influence. La méthodologie d'échantillonnage proposée a ensuite été comparée à trois autres méthodes : l'échantillonnage Monte Carlo probabilisé, l'échantillonnage Monte Carlo uniforme (non optimisé), et l'échantillonnage optimisé avec une combinaison aléatoire des plans des variables scalaires et fonctionnelles. La méthode proposée a permis de construire un métamodèle avec une meilleure prédictivité que les autres méthodes et a donné globalement une meilleure approximation des indices de Sobol' du premier ordre. En particulier, l'amélioration de la métamodélisation et de l'analyse de sensibilité est très importante entre l'échantillonnage uniforme non optimisé et la méthodologie proposée, ce qui montre que l'optimisation du plan d'échantillonnage améliore grandement la qualité du métamodèle. La différence entre les résultats obtenus avec la méthodologie proposée et avec la méthode combinant aléatoirement les plans des variables scalaires et fonctionnelles montre que l'optimisation de la combinaison des deux plans améliore aussi la métamodélisation mais dans une moindre mesure. Dans le cas LiPoSo, à partir de la méthode d'échantillonnage proposée, des métamodèles ont été construits pour les quatre sorties d'intérêt du code et les analyses de sensibilité de ces sorties ont été réalisées, dont l'analyse temps par temps des deux sorties fonctionnelles (les températures du sodium et de gaine au cours du temps). Elles ont révélé la prédominance de l'influence des variables fonctionnelles par rapport à l'influence des autres variables.

Afin d'améliorer la méthode d'échantillonnage proposée, il pourrait être intéressant d'assouplir la troncature du support des distributions non bornées en autorisant l'échantillonnage d'une certaine proportion de points en dehors du support tronqué. Une autre méthode à envisager pourrait être d'échantillonner directement sur l'espace de l'ensemble des entrées au lieu de combiner les échantillons réalisés sur les espaces des entrées scalaires et fonctionnelles. Une troisième voie d'amélioration de la méthodologie serait d'adapter l'algorithme d'échantillonnage, proposé par Auffray et al. (2012), utilisé pour générer les coefficients de la décomposition des variables fonctionnelles pour optimiser la discrépance plutôt que le critère *maximin*. En effet, dans le cas scalaire, ce critère donne de meilleurs résultats en pratique (Damblin et al., 2013).

Dans le cadre d'un code coûteux, la méthodologie proposée s'appuie sur l'utilisation d'un métamodèle, en l'occurrence un métamodèle de type processus gaussien. Cependant, celle-ci est indépendante du type de métamodèle utilisé et pourrait être mise en œuvre avec d'autres techniques de métamodélisation, telles que celle proposée dans le chapitre qui suit.

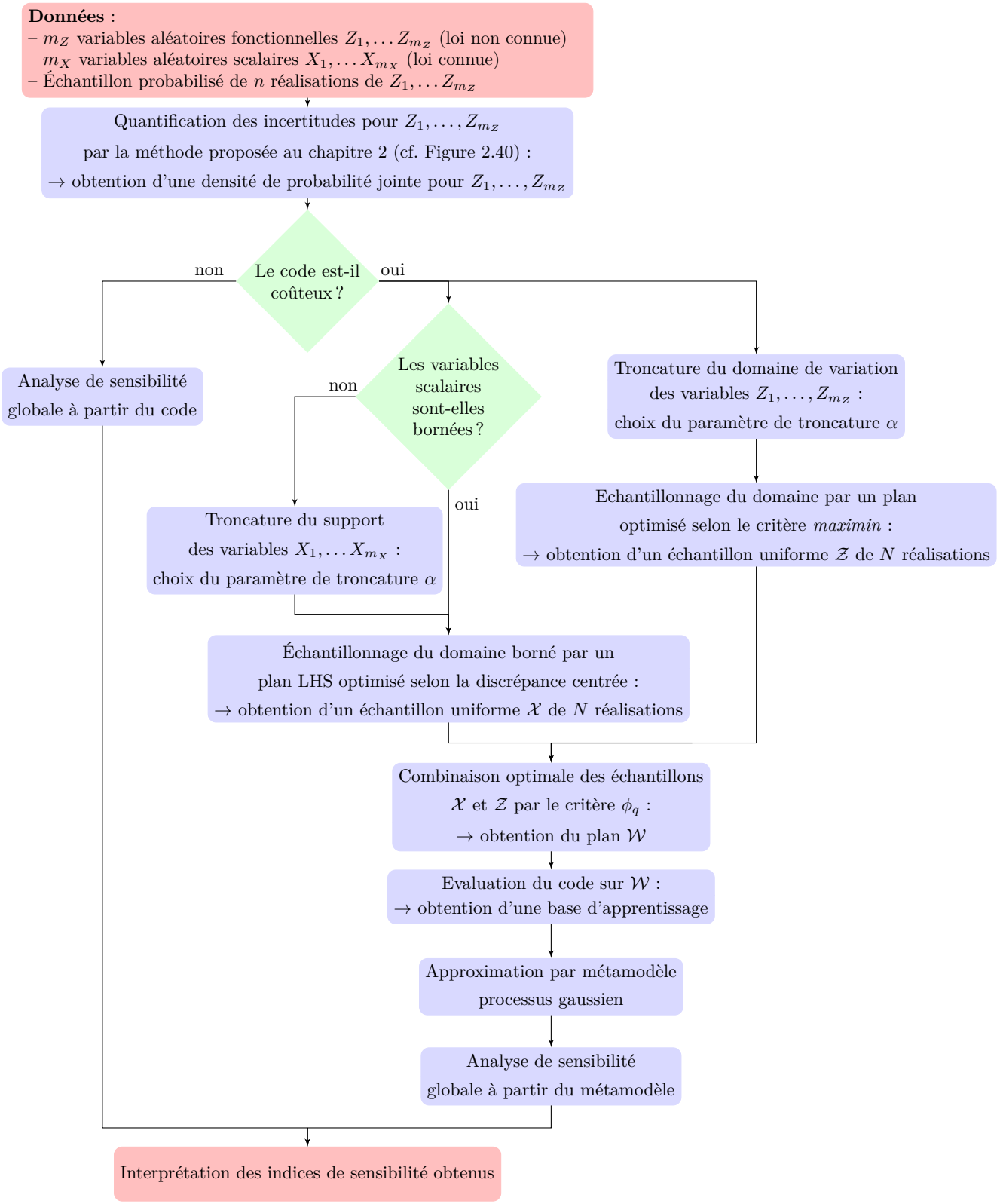

FIGURE 3.12 – Organigramme de la méthode proposée d'analyse de sensibilité d'un code à entrées fonctionnelles.

Chapitre 4

Développements autour de la métamodélisation de codes à entrées fonctionnelles *

4.1 Problématique et objectifs

Pour des codes coûteux en temps de calcul, la principale limitation des méthodes d'analyse de sensibilité globales quantitatives est le grand nombre d'évaluations nécessaires à leur mise en œuvre. Comme proposé et illustré dans le chapitre précédent, une solution possible pour pallier ce problème est de construire un modèle simplifié du code coûteux qui peut alors être utilisé à la place du code pour réaliser l'analyse de sensibilité. Ce modèle de substitution, aussi appelé métamodèle, surface de réponse ou émulateur, a pour objectif d'approcher le plus précisément possible le code de calcul sur son domaine de définition, tout en ayant un temps d'évaluation le plus faible possible. Dans le cas où les paramètres d'entrée et de sortie du code sont scalaires, de très nombreuses méthodes de construction de métamodèles ont été proposées, parmi lesquelles on peut citer les modèles de régression linéaire généralisée (McCullagh et al., 1989), de régression non-paramétrique (Nadaraya, 1964), les modèles additifs généralisés (Hastie et al., 2001), les réseaux de neurones (Bishop, 1995), les polynômes de chaos (Xiu et Karniadakis, 2002) ou encore les métamodèles processus gaussiens (Rasmussen et Williams, 2006).

Dans ce chapitre, l'objectif est d'explorer de nouvelles approches pour construire un métamodèle adapté à un code de calcul dont une ou plusieurs entrées sont fonctionnelles. Plusieurs approches ont été proposées pour construire un tel métamodèle. Une première famille de méthodes consiste à utiliser des métamodèles basés sur des distances entre les paramètres et à les adapter au cas des entrées fonctionnelles en introduisant des distances fonctionnelles. Notamment, Ferraty et Vieu (2006) proposent d'adapter la régression non-paramétrique en utilisant des semi-métriques entre les paramètres fonctionnels. De la même manière, les métamodèles processus gaussiens peuvent être adaptés en utilisant des distances fonctionnelles dans les fonctions de covariance du processus gaussien utilisé (Morris, 2012; Ginsbourger et al., 2013).

Un autre type d'approche consiste à réduire la dimension des paramètres d'entrée fonctionnels avant la métamodélisation. Cette approche est celle qui a été utilisée pour l'analyse de sensibilité du cas test LiPoSo dans le chapitre précédent. Les entrées sont décomposées sur une base fonctionnelle, puis un métamodèle est construit en prenant comme entrées les coefficients des variables fonctionnelles sur la base de décomposition. Par exemple, Muehlenstaedt et al. (2014) décomposent les paramètres fonctionnels du code étudié sur une base de splines (De Boor, 2001), puis construisent un métamodèle processus gaussien.

∗. Les travaux présentés dans ce chapitre ont été réalisés en collaboration avec deux autres doctorants (Vincent Moutoussamy d'EDF R&D et Benoît Pauwels de l'IFPEN) dans le cadre d'un projet de recherche réalisé au cours du CEMRACS 2013 (Centre d'Eté Mathématique de Recherche Avancée en Calcul Scientifique).

Une troisième approche, mise en œuvre dans le contexte de l'analyse de sensibilité, propose de considérer les paramètres fonctionnels comme des variables « incontrôlables » (Iooss et Ribatet, 2009). Les autres paramètres d'entrée sont dits « contrôlables ». Les variables incontrôlables sont considérées comme un aléas du code de calcul, qui est alors vu comme stochastique, c'est-à-dire que plusieurs évaluations de ce code pour un même jeu de paramètres contrôlables donnent des résultats différents. Sa sortie est donc une variable aléatoire, conditionnellement aux paramètres contrôlables. La métamodélisation de codes stochastiques a été moins étudiée que celle de codes déterministes. Une première solution, proposée par Iooss et Ribatet (2009), est d'estimer la moyenne et la variance de la sortie du code stochastique (conditionnellement aux variables contrôlables) en construisant des métamodèles joints sur ces deux quantités. Ces deux métamodèles sont des fonctions ayant pour entrée uniquement les paramètres contrôlables. Une analyse de sensibilité globale peut ensuite être réalisée à partir de ces deux estimations. Cependant, la méthode de Iooss et Ribatet (2009) n'apporte qu'une information limitée sur la variable aléatoire en sortie, puisque seuls ses deux premiers moments sont modélisés. **Dans ce quatrième chapitre, on propose d'approfondir et d'améliorer ce type d'approche afin d'obtenir une description plus complète de la sortie du code.** Pour cela, on construit un métamodèle entre les entrées contrôlables et la densité de probabilité de la sortie. En effet, cette dernière caractérise complètement la distribution de probabilité de la variable. Ainsi, avec cette méthode, il est possible de prédire non seulement les moments de la sortie mais aussi d'autres quantités comme des quantiles ou des probabilités de dépassement de seuil. **De cette manière, on se ramène à la construction d'un métamodèle sur un code dont les entrées sont scalaires et dont la sortie est une fonction, cette fonction devant respecter plusieurs contraintes afin d'être une densité de probabilité. En effet, d'une part, elle doit être positive et d'autre part, son intégrale sur son domaine de variation doit être égale à un.**

Dans la section 4.2, plusieurs familles de méthodes de métamodélisation d'un code stochastique sont présentées. Dans un premier temps, une revue de méthodes basées sur la métamodélisation des moments de la sortie du code stochastique est présentée. Dans une deuxième partie, plusieurs méthodes de métamodélisation de la densité de probabilité de la sortie du code stochastique sont proposées. Ces méthodes sont ensuite appliquées à plusieurs cas d'application dans la section 4.3 : deux exemples analytiques de codes stochastiques sont étudiés, puis le cas du choc thermique pressurisé (cas PTS) est traité. Enfin, une synthèse des méthodes de métamodélisation présentées ainsi que des résultats obtenus est réalisée dans la section 4.4.

4.2 Métamodélisation de codes stochastiques

Le code à entrées fonctionnelles que l'on cherche à approcher par un métamodèle est le suivant

$$\mathcal{M}\colon \left\{ \begin{array}{ll} \boldsymbol{\mathcal{X}} \times \mathcal{Z} & \to \mathbb{R} \\ (x_1, \ldots, x_d, z_1, \ldots, z_{d'}) & \mapsto y, \end{array} \right.$$

où $\boldsymbol{\mathcal{X}}$, le domaine de définition des entrées scalaires de $\mathcal{M}$, est un sous domaine de $\mathbb{R}^d$, et $\mathcal{Z}$, celui de ses entrées fonctionnelles, est le produit cartésien d'espaces de fonctions. Ce code possède d entrées scalaires $\mathbf{x} = (x_1, \ldots, x_d)$, dites contrôlables et d' entrées fonctionnelles $(z_1, \ldots, z_{d'})$, considérées comme incontrôlables.

On note $\mathcal{M}_s$ le code stochastique défini dans la section précédente qui prend en entrée les variables contrôlables de $\mathcal{M}$ et dont la sortie est une variable aléatoire conditionnellement aux paramètres contrôlables, notée Y. Elle est définie comme suit, pour $\mathbf{x} \in \boldsymbol{\mathcal{X}}$:

$$Y = \mathcal{M}_s(\mathbf{x}) = \mathcal{M}(\mathbf{x}, Z),$$

où Z est une variable aléatoire uniforme sur l'ensemble $\mathcal{Z}$.

4.2.1 Métamodèles basés sur les deux premiers moments de la sortie

Une première approche possible pour construire un métamodèle pour un code stochastique est de résumer la variable aléatoire en sortie du code par plusieurs quantités scalaires associées comme par exemple ses deux premiers moments. Cette approche a été mise en œuvre de différentes manières.

Tout d'abord, dans le contexte de la modélisation de données expérimentales, Vining et Myers (1990) introduisent une méthode dans laquelle deux régressions polynomiales sont construites séparément pour prédire la moyenne et la variance de la sortie. Pour la construction de ces deux métamodèles, il est nécessaire de répéter pour un même jeu de paramètres les évaluations du code. Ce type de méthode a été utilisé dans de nombreux contextes et notamment dans le cadre de l'optimisation robuste (Dellino et al., 2010). Une description plus détaillée de ces méthodes peut être trouvée dans Myers et al. (2009).

Une autre approche, développée par Nelder et Lee (1991) et suggérée initialement par Pregibon (1984), consiste à construire deux métamodèles joints ou liés permettant d'approcher simultanément la moyenne et la variance de la sortie du code. Dans le modèle introduit par Nelder et Lee (1991) et repris par Zabalza et al. (1998), les métamodèles joints utilisés sont des modèles linéaires généralisés (GLM pour *Generalized Linear Model*). Le premier des deux métamodèles joints modélisant la moyenne μ de Y, la sortie du code stochastique $\mathcal{M}_s$, est défini comme suit :

$$\begin{aligned} E(Y) &= \mu, \text{ avec } \eta = g(\mu) = \beta^T.\mathbf{x}, \\ \text{Var}(Y) &= \phi v(\mu), \end{aligned} \tag{4.1}$$

où g est une fonction de lien monotone et différentiable, ϕ est la dispersion du code, β sont les paramètres de la régression entre $\mathbf{x}$ et η, et v est la fonction de la variance. Les fonctions g et v doivent être choisies en fonction de la forme des données. La fonction de lien peut, par exemple, être l'identité, la racine carrée ou le logarithme, alors que des choix classiques pour la fonction v sont la fonction constante, l'identité ou la fonction carré. Le couple identité et fonction constante pour g et v correspond au modèle de régression linéaire. Dans un modèle linéaire généralisé classique, le paramètre de dispersion ϕ est considéré comme constant. Dans la méthode proposée par Nelder et Lee (1991), ϕ n'est plus constante mais dépend de variables explicatives $\mathbf{u}$, généralement choisies parmi les paramètres $\mathbf{x}$. La dispersion ϕ est la moyenne des résidus centrés réduits au carré, notés d. Le même type de métamodèle que celui de l'équation (4.1) est utilisé pour modéliser la dispersion de Y :

$$\begin{aligned} E(d) &= \phi, \text{ avec } \xi = g_d(\phi) = \gamma^T.\mathbf{u}, \\ \text{Var}(d) &= \tau v_d(\phi), \end{aligned}$$

où g_d et v_d sont la fonction de lien et la fonction de la variance de la dispersion, et τ et γ sont des paramètres du GLM.

Par la suite, des modèles additifs généralisés (GAM) joints (Rigby et Stasinopoulos, 1996; Iooss et Ribatet, 2009) puis des métamodèles processus gaussien joints (Marrel et al., 2012) ont été introduits dans le but de métamodéliser la moyenne et la variance. L'algorithme proposé par Marrel et al. (2012) consiste à construire un métamodèle processus gaussien, noté Gp_1, de la sortie en considérant que sa dispersion est homoscédastique. Puis un métamodèle processus gaussien Gp_2 est construit sur les résidus au carré du métamodèle Gp_1 avec une variance homoscédastique. Dans une troisième étape, un nouveau métamodèle Gp_3 de la sortie est construit sur la sortie du code avec une variance hétéroscédastique estimée par le métamodèle Gp_2. Enfin, un métamodèle Gp_4 est appris pour les résidus au carré de Gp_3, de la même manière que Gp_2. Dans le cadre de l'analyse de sensibilité, Marrel et al. (2012) ont montré sur plusieurs exemples analytiques que les métamodèles processus gaussiens et GAM joints pouvaient améliorer considérablement la qualité de la métamodélisation par rapport aux métamodèles GLM joints. L'avantage des méthodes utilisant les métamodèles joints est que le nombre d'appels au code de calcul est réduit par rapport aux métamodèles séparés, présentés précédemment. En effet, il a été montré par Zabalza et al. (1998) que répéter l'évaluation du code de calcul sur un même jeu de paramètres d'entrée n'apporte pas une meilleure approximation du code, et qu'il est préférable d'utiliser tous les appels au code disponibles pour mieux couvrir le domaine des variables d'entrée. De plus, les métamodèles joints donnent de meilleurs résultats en pratique que la construction séparée des métamodèles.

Enfin, une dernière approche a été étudiée par Reich et al. (2012). Ces derniers font l'hypothèse que la réponse du code est la somme d'un modèle de régression linéaire et d'une variable aléatoire modélisée par un mélange de gaussiennes composé d'une infinité de classes, appelé *stick-breaking model*. Ainsi, la moyenne et la variance de la sortie du code sont modélisés en fonction des paramètres β du modèle de régression linéaire et de ceux du mélange de gaussiennes comme suit :

$$\begin{aligned} E(Y|x) &= x^T\beta + \sum_{k=1}^{\infty} p_k(x)\mu_k \\ \mathrm{Var}(Y|x) &= \sum_{k=1}^{\infty} p_k(x)\left[\sigma_k^2 + \left(\mu_k - \sum_{l=1}^{\infty} p_l(x)\mu_l\right)^2\right], \end{aligned}$$

où μ_k, σ_k^2 et p_k désignent respectivement la moyenne, la variance et la proportion de la k-ième gaussienne du mélange. La moyenne conditionnelle est la somme du modèle de regression linéaire et de la moyenne du mélange de gaussiennes, et la variance conditionnelle est la variance du mélange. Reich et al. (2012) considèrent que seules les proportions dépendent des entrées du code. Elles sont modélisées par un produit de noyaux gaussiens. Une méthode bayésienne de sélection de variables est ensuite mise en œuvre pour déterminer les composantes non-nulles du mélange de gaussiennes infini, ainsi que les paramètres du mélange et pour estimer des modèles pour les proportions p_k.

4.2.2 Métamodèles de la densité de probabilité de la sortie

Contrairement aux approches présentées dans la section précédente, on cherche ici à prédire la distribution complète de la sortie du code stochastique et non uniquement ses premiers moments. Pour ce faire, on propose de construire un métamodèle pour approcher le code $\mathcal{M}_d$ possédant les mêmes entrées que $\mathcal{M}_s$ et dont la sortie pour un jeu de paramètres d'entrée $\mathbf{x}$ donné est la densité de probabilité de la variable aléatoire $\mathcal{M}_s(\mathbf{x})$. On définit donc $\mathcal{M}_d$ comme suit :

$$\begin{aligned} \mathcal{M}_d : \boldsymbol{\mathcal{X}} \subset \mathbb{R}^d &\to \mathcal{F} \\ \mathbf{x} &\mapsto f_{\mathcal{M}_s(\mathbf{x})}, \end{aligned}$$

où $f_{\mathcal{M}_s(\mathbf{x})}$ est la densité de probabilité de la variable aléatoire $\mathcal{M}_s(\mathbf{x})$, et $\mathcal{F}$ est l'espace des densités de probabilité dont le support est un intervalle I de $\mathbb{R}$:

$$\mathcal{F} = \left\{ f : I \to \mathbb{R}_+ : \int_I f(t)\mathrm{d}t = 1 \right\}.$$

Soit $\mathbf{X}_N = \{(\mathbf{x}_i, f_i) : i = 1, \ldots, N\} \subset (\boldsymbol{\mathcal{X}} \times \mathcal{F})^N$ un échantillon d'apprentissage ($N \in \mathbb{N} \setminus \{0\}$). On suppose avoir accès à la densité de probabilité f_i associée à une entrée $\mathbf{x}_i$ pour N jeux de paramètres d'entrée, *i.e.*

$$f_i = \mathcal{M}_d(\mathbf{x}_i, .) \qquad (i = 1, \ldots, N).$$

En pratique, pour un i donné dans $\{1, \ldots, N\}$, seules des réalisations de la sortie du code stochastique $\mathcal{M}_s$ sont disponibles et non celles de $\mathcal{M}_d$. Donc, seul un échantillon fini de valeurs de la sortie de $\mathcal{M}_s$ est disponible pour un jeu d'entrées donné. On note $y_{i1}, \ldots, y_{iM}$ les valeurs de la sortie de $\mathcal{M}_s$ obtenues pour l'entrée $\mathbf{x}_i$. On peut utiliser, pour obtenir la densité de probabilité f_i, la méthode d'estimation à noyau présentée dans la section 2.3.1.

Soit $\mathbf{x}_0 \in \boldsymbol{\mathcal{X}}$ et $f_0 = \mathcal{M}_d(\mathbf{x}_0, .)$. L'objectif est ici d'estimer f_0 par $\hat{f}_0 \in \mathcal{F}$, sachant $\mathbf{X}_N$. La construction d'un métamodèle pour un code de calcul stochastique se ramène donc ici à la construction d'un métamodèle pour un code à sorties fonctionnelles sous contrainte. En effet, la principale difficulté associée à cette approche est d'imposer à l'approximation $\hat{f}_0$ d'appartenir à $\mathcal{F}$, *i.e.* d'être une densité de probabilité.

Dans la suite de cette section, on étudie plusieurs solutions pour répondre à ce problème :

– la méthode de régression à noyau classique,
– une extension de la régression à noyau basée sur la distance de Hellinger,
– un métamodèle utilisant la décomposition fonctionnelle sous contrainte des densités de probabilité.

4.2.2.1 Régression à noyau classique

On s'intéresse tout d'abord à un estimateur pour lequel il est simple de garantir que l'approximation de la densité de probabilité appartienne à $\mathcal{F}$ et qui s'écrit de la manière suivante :

$$\hat{f}_0 = \sum_{i=1}^{N} \alpha_i(\mathbf{x}_0) f_i$$

où $\alpha_i(\mathbf{x}_0) \in \mathbb{R}$ $(i = 1, \ldots, N)$. En effet, pour imposer que $\hat{f}_0$ soit dans $\mathcal{F}$, il suffit d'imposer que

$$\alpha_i(\mathbf{x}_0) \quad \geq 0 \quad (i = 1, \ldots, N), \tag{4.2}$$

$$\sum_{i=1}^{N} \alpha_i(\mathbf{x}_0) \qquad = 1. \tag{4.3}$$

La régression non-paramétrique à noyau, introduite par Nadaraya (1964) et Watson (1964), vérifie les deux contraintes (4.2) et (4.3).

Définition 4.1 (Estimateur à noyau classique)**.** *Soit $K_H : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ un noyau, et H une matrice positive de taille $d \times d$, appelée largeur de bande. L'estimateur à noyau de la densité de probabilité f_0 correspondant au point $\mathbf{x}_0$ est le suivant :*

$$\hat{f}_0 = \sum_{i=1}^{N} \frac{K_H(\mathbf{x}_i, \mathbf{x}_0)}{\sum_{j=1}^{N} K_H(\mathbf{x}_j, \mathbf{x}_0)} f_i. \tag{4.4}$$

La méthode d'estimation à noyau repose sur l'intuition que, si le point $\mathbf{x}_0$ est proche d'un point $\mathbf{x}_i$ de l'échantillon d'apprentissage, alors la densité de probabilité f_0 sera plus influencée par f_i. La Figure 4.1 illustre cette intuition dans le cas de densités gaussiennes. Dans cet exemple, la sortie du code $\mathcal{M}_d$ est une densité de probabilité gaussienne définie comme suit : $\mathcal{M}_d\left((\mu, \sigma), .\right) = \frac{e^{-\frac{1}{2}\left(\frac{\cdot - \mu}{\sigma}\right)^2}}{\sqrt{2\pi\sigma^2}}$, avec μ et σ sa moyenne et son écart-type. Les paramètres d'entrée de $\mathcal{M}_d$ sont donc la moyenne et l'écart-type $\mathbf{x} = (\mu, \sigma) \in \boldsymbol{\mathcal{X}} = \mathbb{R} \times \mathbb{R}_+^*$ de la sortie. La partie gauche de la Figure 4.1 représente un échantillon de paramètres d'entrée du code et sa partie droite les sorties correspondantes. La densité de probabilité associée à la croix noire (ligne pointillée) sur la partie gauche de la Figure 4.2.2 semble bien plus proche de la densité de probabilité représentée en rouge et correspondant au point rouge que des autres densités.

Cet estimateur de régression est très proche de l'estimateur de densité de probabilité introduit dans la section 2.3.1. Comme il a été précisé dans cette section, le choix du noyau a peu d'influence sur la qualité de l'estimation. On propose donc d'utiliser par la suite le noyau gaussien, très répandu en pratique :

$$K_H(\mathbf{x}, \mathbf{y}) = \frac{1}{\sqrt{2\pi \det(H)}} e^{-(\mathbf{x}-\mathbf{y})^T H^{-1} (\mathbf{x}-\mathbf{y})} \quad \left(\mathbf{x}, \mathbf{y} \in \mathbb{R}^d\right).$$

L'estimation de la matrice de largeur de bande a bien plus d'impact sur la qualité de l'estimateur à noyau. On choisit de prendre une matrice H diagonale pour diminuer le nombre d'éléments de la matrice à estimer. Plus les éléments de sa diagonale sont élevés, plus le noyau est étendu dans la direction j et plus de points de l'échantillon sont pris en compte dans l'estimation pour un nouveau point $\mathbf{x}_0$. On

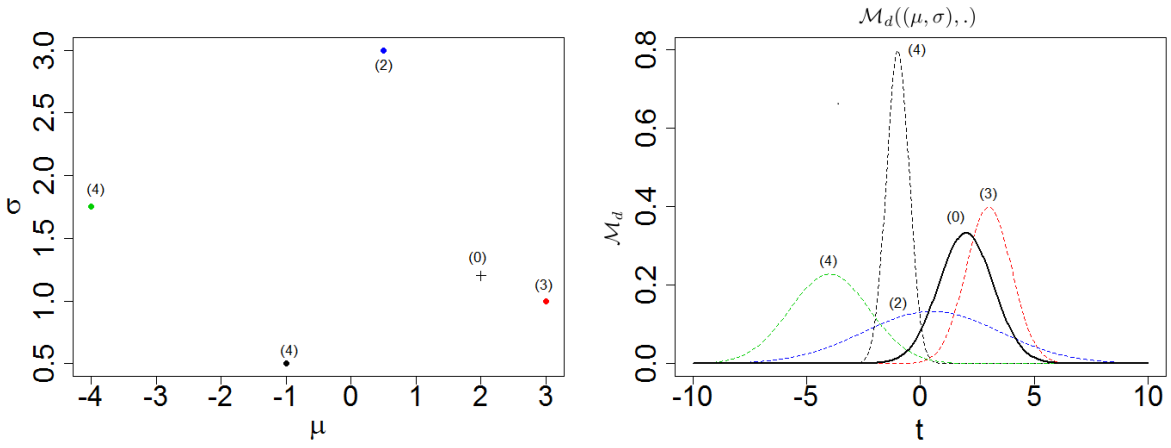

FIGURE 4.1 – A gauche : les points de l'échantillon $\boldsymbol{\mathcal{X}}$ (points) et un point x_0 (croix). A droite : les densités de probabilité associées données par le code $\mathcal{M}_d$ défini dans la section 4.2.2.

cherche la largeur de bande H^* qui minimise l'erreur globale d'estimation. Cela signifie que l'on cherche $H^* = \text{diag}(h_1^*, \ldots, h_d^*)$ telle que

$$\text{diag}(h_1^*, \ldots, h_d^*) \in \underset{h_1,\ldots,h_d \in \mathbb{R}_+}{\arg\min} \int_{\boldsymbol{\mathcal{X}}} \left\| \hat{f}_0 - f_0 \right\|_{L^2}^2 \mathrm{d}\mathbf{x}_0.$$

Pour obtenir une approximation de H^*, on utilise la méthode de validation croisée (Hastie et Tibshirani, 1990), comme proposé par Hardle et al. (1985). Ainsi, on écrit

$$H^* \in \underset{h_1,\ldots,h_d \in \mathbb{R}_+}{\arg\min} \sum_{i=1}^{N} \left\| \hat{f}_{-i} - f_i \right\|_{L^2}^2,$$

avec

$$\hat{f}_{-i} = \sum_{\substack{j=1 \\ j\neq i}}^{N} \alpha_j^{-i} f_j$$

$$\alpha_j^{-i} = \frac{e^{-(\mathbf{x}_i - \mathbf{x}_j)^t H^{-1} (\mathbf{x}_i - \mathbf{x}_j)}}{\sum_{\substack{l=1 \\ l\neq i}}^{N} e^{-(\mathbf{x}_i - \mathbf{x}_l)^t H^{-1} (\mathbf{x}_i - \mathbf{x}_l)}}.$$

Pour résoudre ce problème d'optimisation, on propose d'utiliser l'algorithme L-BFGS-B, développé par Byrd et al. (1995). Une fois H^* estimé, le métamodèle par régression à noyau est donné par la formule suivante

$$\hat{f}_0 = \sum_{i=1}^{N} \alpha_{i,\hat{H}^*} f_i, \tag{4.5}$$

avec

$$\alpha_{i,\hat{H}^*} = \frac{e^{-(\mathbf{x}_0 - \mathbf{x}_i)^T (\hat{H}^*)^{-1} (\mathbf{x}_0 - \mathbf{x}_i)}}{\sum_{j=1}^{N} e^{-(\mathbf{x}_0 - \mathbf{x}_j)^T (\hat{H}^*)^{-1} (\mathbf{x}_0 - \mathbf{x}_j)}}.$$

4.2.2.2 Proposition d'une régression à noyau basée sur la distance de Hellinger

La régression à noyau, définie dans l'équation (4.4), peut être considérée comme un problème de minimisation.

Proposition 4.2. *L'estimateur à noyau défini dans l'équation* (4.4) *est équivalent au problème de minimisation suivant :*

$$\hat{f}_0 = \underset{f \in \mathcal{F}}{\arg\min} \sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0) \int_I (f_i - f)^2 (t) \mathrm{d}t. \tag{4.6}$$

Démonstration de la proposition 4.2. L'objectif est de montrer que l'expression classique de l'estimateur à noyau de l'équation (4.4) est la solution du problème de minimisation suivant :

$$\hat{f}_0 = \underset{f \in \mathcal{F}}{\arg\min} \sum_{i=1}^{n} K_H(\mathbf{x}_i, \mathbf{x}_0) \int_I (f_i(t) - f(t))^2 \, \mathrm{d}t.$$

D'après le théorème d'Euler-Lagrange (Gelfand et Fomin, 2000), une condition nécessaire pour que la fonction $\hat{f}_0$ soit solution du problème d'optimisation (4.6) est qu'elle vérifie l'équation suivante :

$$\frac{\partial J}{\partial f} - \frac{d}{dt} \frac{\partial J}{\partial f'} = 0, \tag{4.7}$$

où $J(t, f(t), f'(t)) = \sum_{i=1}^{n} K_H(\mathbf{x}_i, \mathbf{x}_0) (f_i(t) - f(t))^2$, $\forall t \in I$.

La solution $\hat{f}_0$ de l'équation (4.7) vérifie l'expression suivante $\forall t \in I$:

$$\begin{aligned} & 2 \left[\hat{f}_0(t) \left(\sum_{i=1}^{n} K_H(\mathbf{x}_i, \mathbf{x}_0) \right) - \sum_{i=1}^{n} K_H(\mathbf{x}_i, \mathbf{x}_0) f_i(t) \right] = 0 \\ \iff \quad & \hat{f}_0(t) = \sum_{i=1}^{n} \frac{K_H(\mathbf{x}_i, \mathbf{x}_0)}{\sum_{j=1}^{n} K_H(\mathbf{x}_j, \mathbf{x}_0)} f_i(t). \end{aligned} \tag{4.8}$$

Il est ensuite aisé de vérifier que la fonction de l'équation (4.8) est solution du problème (4.6). □

La régression à noyau est donc une méthode de régression locale, qui peut être estimée en minimisant l'erreur des moindres carrés. On peut observer que les distances L^2 entre les fonctions de l'échantillon et la fonction inconnue apparaissent dans la fonction objectif du problème de minimisation (4.6). Par la suite, on note $\hat{f}_{0,L^2}$ l'estimateur donné par l'équation (4.6).

Parmi les mesures classiques de similarité entre densités de probabilité, on s'intéresse à la distance de Hellinger, un cas particulier des f-divergences (Csiszár, 1967), définie de la manière suivante pour deux densités de probabilité f et g :

$$d_H(f, g) = \int_I \left(\sqrt{f(t)} - \sqrt{g(t)} \right)^2 \mathrm{d}t. \tag{4.9}$$

Cette distance permet en particulier de placer plus de poids sur les queues des densités de probabilité que la distance L^2. On propose ici un nouvel estimateur de régression à noyau basé sur la distance de Hellinger en remplaçant la distance L^2 par cette distance dans l'équation (4.6).

Définition 4.3 (Estimateur à noyau utilisant la distance de Hellinger)**.** *Le nouvel estimateur à noyau proposé est la solution du problème de minimisation suivant :*

$$\hat{f}_{0,He} = \underset{f \in \mathcal{F}}{\arg\min} \sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0) \int_I \left(\sqrt{f_i(t)} - \sqrt{f(t)} \right)^2 \mathrm{d}t. \tag{4.10}$$

Comme dans le cas de l'estimateur à noyau utilisant la distance L^2, on peut déduire une expression analytique de la solution du problème (4.10).

Proposition 4.4. *L'estimateur suivant est la solution du problème de minimisation* (4.10) *et est donc l'estimateur de régression à noyau basé sur la distance de Hellinger :*

$$\hat{f}_{0,He} = \frac{\left(\sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0)\sqrt{f_i}\right)^2}{\int_I \left(\sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0)\sqrt{f_i(t)}\right)^2 \mathrm{d}t}. \tag{4.11}$$

Démonstration de la proposition 4.4. L'objectif est de montrer que les expressions (4.10) et (4.11) sont équivalentes. La contrainte que l'intégrale de l'estimateur $\hat{f}_0$ soit égale à 1 dans le problème d'optimisation (4.10) est transformée à l'aide du multiplicateur de Lagrange associé. Le problème devient alors :

$$\hat{f}_0 = \arg\min_{f,\lambda} L(f, \lambda), \tag{4.12}$$

avec

$$L(f, \lambda) = \sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0) \int_I \left(\sqrt{f_i(t)} - \sqrt{f(t)}\right)^2 \mathrm{d}t - \lambda \left(\int_I f(t)\mathrm{d}t - 1\right).$$

D'après le théorème d'Euler-Lagrange (Gelfand et Fomin, 2000), une condition nécessaire pour que la fonction $\hat{f}_0$ soit solution du problème d'optimisation (4.12) est qu'elle vérifie l'équation suivante :

$$\frac{\partial J}{\partial f} - \frac{d}{dt}\frac{\partial J}{\partial f'} = 0 \tag{4.13}$$

où $J(t, f(t), f'(t)) = \sum_{i=1}^{n} K_H(\mathbf{x}_i, \mathbf{x}_0)\left(\sqrt{f_i(t)} - \sqrt{f(t)}\right)^2 - \lambda f(t)$, $\forall t \in I$, et avec $\lambda \in \mathbb{R}$. L'équation (4.13) peut se réécrire ainsi :

$$\begin{aligned} &\sum_{i=1}^{n} K_H(\mathbf{x}_i, \mathbf{x}_0) - \sum_{i=1}^{n} K_H(\mathbf{x}_i, \mathbf{x}_0)\sqrt{\frac{f_i(t)}{\hat{f}_0(t)}} - \lambda = 0 \\ \iff \quad &\hat{f}_0(t) = \left(\sum_{i=1}^{N} \frac{K_H(\mathbf{x}_i, \mathbf{x}_0)}{\sum_{j=1}^{N} K_H(\mathbf{x}_j, \mathbf{x}_0) - \lambda}\sqrt{f_i(t)}\right)^2, \end{aligned} \tag{4.14}$$

La valeur du paramètre λ est déterminée en imposant à $\hat{f}_0$ de respecter la contrainte que son intégrale soit égale à 1.

$$\begin{aligned} &\int_I \hat{f}_0(t)\mathrm{d}t = 1 \\ \iff \quad &\int_I \left(\sum_{i=1}^{N} \frac{K_H(\mathbf{x}_i, \mathbf{x}_0)}{\sum_{j=1}^{N} K_H(\mathbf{x}_j, \mathbf{x}_0) - \lambda}\sqrt{f_i(t)}\right)^2 \mathrm{d}t = 1 \\ \iff \quad &\lambda = \sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0) - \left[\int_I \left(\sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0)\sqrt{f_i(t)}\right)^2 \mathrm{d}t\right]^{1/2}. \end{aligned} \tag{4.15}$$

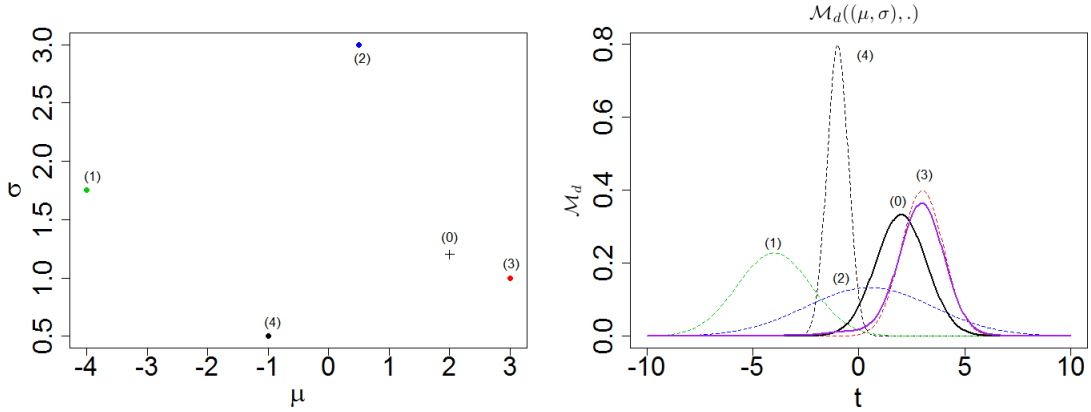

FIGURE 4.2 – A gauche : les points de l'échantillon $\boldsymbol{\mathcal{X}}$ (points) et un point x_0 (croix). A droite : estimateur à noyau basé sur la distance de Hellinger (en trait plein mauve) de la courbe inconnue (trait plein noir) correspondant au point x_0, construit à partir des densités de probabilité (courbes bleu, rouge, vert et noir en pointillés) de l'échantillon $\mathcal{X}$.

La combinaison des équations (4.14) et (4.15) donne la formule de l'équation (4.11) :

$$\hat{f}_0 = \frac{\left(\sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0)\sqrt{f_i}\right)^2}{\int_I \left(\sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0)\sqrt{f_i(t)}\right)^2 \mathrm{d}t}.$$

Enfin, la fonction $\hat{f}_0$ vérifiant l'équation d'Euler-Lagrange est bien solution du problème d'optimisation (4.11).

□

Ce nouvel estimateur basé sur la distance de Hellinger est illustré sur l'exemple des densités de probabilité gaussiennes défini dans la section 4.2.2.1. L'échantillon d'apprentissage de l'estimateur de régression à noyau basé sur la distance de Hellinger est constitué de quatre jeux de paramètres et des sorties de $\mathcal{M}_d$ correspondantes. Les paramètres sont représentés par des points sur la partie gauche de la Figure 4.2, et les densités de probabilité sont en ligne pointillé sur la partie droite. Le modèle est utilisé pour prédire la densité de probabilité $x^{(0)}$ représenté par une croix. La courbe correspondant à ces paramètres et son approximation $\hat{f}_{0,\mathrm{He}}$ sont représentées respectivement en ligne continue noire et mauve. On peut observer que, même si la sortie n'est pas bien approchée, la plus grande partie du poids est donnée à la courbe verte correspondant aux paramètres les plus proches.

Remarque 4.1. *Comme précisé précédemment, dans les équations* (4.6) *et* (4.10)*, un modèle de régression locale est défini. Localement, la fonction $\mathcal{M}_s$ est approchée par une constante. Une perspective d'amélioration de la méthode présentée dans cette section pourrait être d'utiliser une régression polynomiale locale (Fan et Gijbels, 1996) plutôt qu'une régression constante locale. Le problème de minimisation à résoudre serait alors le suivant pour un polynôme d'ordre $q \in \mathbb{N}$:*

$$\min_{\beta_1,\ldots,\beta_q \in \mathcal{F}} \sum_{i=1}^{N} K_H(\mathbf{x}_i, \mathbf{x}_0) \int_I \left(\sqrt{f_i(t)} - \left(\sum_{j=0}^{q} \beta_j(t)\,(\mathbf{x}_i - \mathbf{x}_0)^j\right)^{1/2}\right)^2 \mathrm{d}t.$$

4.2.2.3 Décompositions fonctionnelles sous contrainte

Dans cette section, on cherche à construire une base de fonctions $w_1, \ldots, w_q$ pour approcher dans $\mathcal{F}$ les fonctions d'un échantillon $\{f_1, \ldots, f_N\}$ de densités de probabilité. L'objectif est de réduire la dimension de l'échantillon des données en l'approchant sur un espace de dimension q inférieure et engendré par la base $W = (w_1, \ldots, w_q)$. La projection $\hat{f}_i$ d'une fonction f_i sur la base W s'écrit de la manière suivante :

$$\hat{f}_i = \sum_{k=1}^{q} \psi_{ik} w_k,$$

où les $\psi_{i1}, \ldots, \psi_{iq} \in \mathbb{R}$ sont les coefficients de la fonction sur la base W. On cherche ici à imposer à l'approximation $\hat{f}_i$ d'appartenir à l'ensemble $\mathcal{F}$ des densités de probabilité. Pour ce faire, on cherche à imposer les trois conditions suivantes $\forall i \in \{1, \ldots, N\}$:

$$\begin{aligned} & w_k \in \mathcal{F} \\ & \psi_{ik} \geq 0 \qquad (k = 1, \ldots, q) \\ & \sum_{k=1}^{q} \psi_{ik} = 1. \end{aligned} \tag{4.16}$$

Trois approches pour construire la base de décomposition sont étudiées : une adaptation de l'analyse en composantes principales fonctionnelles proposée par Kneip et Utikal (2001), une modification de l'algorithme des *Magic Points*, et une méthode basée sur la minimisation sous contrainte de l'erreur L^2 de projection sur la base.

Analyse en Composantes Principales Contrainte (ACPC)

Parmi toutes les techniques de réduction de dimension, une des plus utilisées est l'Analyse en Composantes Principales Fonctionnelle (ACPF), introduite dans la section 2.2.4.2.

Soit $\bar{f}$ la moyenne empirique de l'échantillon $f_1, \ldots, f_N$:

$$\bar{f} = \frac{1}{N} \sum_{i=1}^{N} f_i.$$

Comme précisé dans la section 2.2.4.2, l'ACPF permet de construire une base orthonormale $w_1, \ldots, w_q$ pour les fonctions centrées $f_i^c = f_i - \bar{f}$, pour $i = 1, \ldots, N$, telle que la variance projetée de chaque fonction sur la base $w_1, \ldots, w_q$ soit maximale. Pour assurer cette propriété, le problème suivant est considéré :

$$\begin{aligned} \underset{w_1,\ldots,w_q : I \to \mathbb{R}}{\text{Minimiser}} \quad & \sum_{i=1}^{N} \left\| f_i^c - \sum_{k=1}^{q} \langle f_i^c, w_k \rangle_{L^2} \, w_k \right\|_{L^2}^2 \\ \text{soumis à} \quad & \langle w_k, w_l \rangle_{L^2} = \delta_{kl} \qquad (k, l = 1, \ldots, q), \end{aligned}$$

où δ_{kl} est 1 quand $k = l$ et 0 sinon. La décomposition ACPF assure que $\int_I \hat{f}_i = \int_I f_i$ for $i = 1, \ldots, N$ si bien que les intégrales des approximations $\hat{f}_i$ sont égales à 1. Cependant, elle n'assure pas la positivité des approximations.

Pour ce faire, Delicado (2011) propose d'appliquer l'ACPF aux fonctions $g_i = \log f_i$ pour conserver la positivité des prédictions $\hat{f}_i = \exp \hat{g}_i$, pour $i = 1, \ldots, N$. Toutefois, cette approche ne permet pas d'assurer que les intégrales des fonctions projetées soient égales à 1.

Affleck-Graves et al. (1979) ont proposé d'ajouter une contrainte de positivité sur la première fonction de base w_1 et de laisser les autres fonctions de base sans contrainte. En effet, ils affirment qu'en général il n'est pas possible d'ajouter des contraintes sur les fonctions $w_2, \ldots, w_q$. Comme ces fonctions de base ne sont pas contraintes, la positivité de l'approximation n'est pas garantie.

Une autre méthode a été proposée par Kneip et Utikal (2001) pour imposer la contrainte de positivité. La décomposition ACPF est appliquée à l'échantillon. Ensuite, les valeurs négatives des densités de probabilité approchées sont mises à 0 et les approximations sont normalisées à 1. Les études réalisées dans Kneip et Utikal (2001) ont montré que la méthode donnait des résultats intéressants en pratique. On note ACPC pour ACP contrainte cette dernière méthode.

Modified Magic Points (MMP)

La méthode *Empirical Interpolation Method* (EIM) ou *Magic Points* (MP) (Maday et al., 2007; Bebendorf et al., 2014) est un algorithme glouton qui construit des interpolateurs pour un échantillon de fonctions. Ici, l'échantillon de fonctions étudié est $\mathcal{M}_d(\boldsymbol{\mathcal{X}},.) = \{\mathcal{M}_d(\mathbf{x},.) : I \to \mathbb{R} : \mathbf{x} \in \boldsymbol{\mathcal{X}}\}$. L'algorithme choisit itérativement une base fonctionnelle dans $\mathcal{M}_d(\boldsymbol{\mathcal{X}},.)$ et un échantillon de points d'interpolation dans I pour construire un interpolateur en ces points des fonctions de l'échantillon. Soit $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ et $f_{\mathbf{x}} = \mathcal{M}_d(\mathbf{x},.)$. A l'étape $q-1$, les bases de fonctions et de points d'interpolation sont notées respectivement $w_1, \ldots, w_{q-1} : I \to \mathbb{R}$ et $t_{i_1}, \ldots, t_{i_{q-1}} \in I$. L'interpolateur $I_{q-1}[f_{\mathbf{x}}]$ de $f_{\mathbf{x}}$ est défini comme une combinaison linéaire de ces fonctions de base :

$$I_{q-1}[f_{\mathbf{x}}] = \sum_{k=1}^{q-1} \psi_k(\mathbf{x}) w_k,$$

où les valeurs des coefficients $\psi_1, \ldots, \psi_{q-1} : \boldsymbol{\mathcal{X}} \to \mathbb{R}$ en $\mathbf{x}$ sont définies de manière unique par l'équation suivante, dont une preuve de l'existence et l'unicité peut être trouvée dans Maday et al. (2007) :

$$I_{q-1}[f_{\mathbf{x}}](t_{i_k}) = f_{\mathbf{x}}(t_{i_k}) \qquad (k = 1, \ldots, q-1).$$

A l'étape q, on sélectionne le paramètre $\mathbf{x}_{i_q}$ tel que $f_{\mathbf{x}_{i_q}} \left(= \mathcal{M}_d\left(\mathbf{x}_{i_q},.\right)\right)$ est l'élément de $\mathcal{M}_d(\boldsymbol{\mathcal{X}},.)$ pour lequel la distance L^2 à l'estimateur est la plus élevée. De la même manière, le prochain point d'interpolation t_{i_q} est celui qui maximise l'écart entre $f_{\mathbf{x}_{i_q}}$ et $I_{q-1}[f_{\mathbf{x}_{i_q}}]$. Enfin, la nouvelle fonction de base w_q est définie comme une combinaison linéaire de l'interpolateur à l'étape précédente et de la fonction $f_{\mathbf{x}_{i_q}}$ la moins bien approchée par cet interpolateur. L'algorithme est résumé ci-dessous (Bebendorf et al., 2014).

Algorithme 4.1 Magic Points - MP

– Définir $q \leftarrow 1$ et $I_0 \leftarrow 0$.

– Pendant que $\varepsilon > tol$

1. Choisir $\mathbf{x}_{i_q}$ dans $\boldsymbol{\mathcal{X}}$:

$$\mathbf{x}_{i_q} \leftarrow \arg\sup_{\mathbf{x} \in \boldsymbol{\mathcal{X}}} \|f_{\mathbf{x}} - I_{q-1}[f_{\mathbf{x}}]\|_{L^2}$$

et le point d'interpolation associé t_{i_q} :

$$t_{i_q} \leftarrow \arg\sup_{t \in I} \left| f_{\mathbf{x}_{i_q}}(t) - I_{q-1}[f_{\mathbf{x}_{i_q}}](t) \right|.$$

2. Définir le prochain élément de la base :

$$w_q \leftarrow \frac{f_{\mathbf{x}_{i_q}}(.) - I_{q-1}[f_{\mathbf{x}_{i_q}}](.)}{f_{\mathbf{x}_{i_q}}(t_{i_q}) - I_{q-1}[f_{\mathbf{x}_{i_q}}](t_{i_q})}.$$

3. Calculer l'erreur d'interpolation :

$$\varepsilon \leftarrow \|err_q\|_{L^\infty(\boldsymbol{\mathcal{X}})}, \text{ où } err_q : \mathbf{x} \mapsto \|f_{\mathbf{x}} - I_q[f_{\mathbf{x}}]\|_{L^2}.$$

4. Imposer $q \leftarrow q + 1$.

En général, les interpolateurs donnés par la méthode MP ne sont pas des densités de probabilité. On propose donc une version modifiée de l'algorithme MP pour la décomposition de densités de probabilité qui ne possède plus la propriété d'interpolation mais qui produit des approximations non négatives et d'intégrales égales à 1. Dans la nouvelle méthode, on note $A_{q-1}[f_{\mathbf{x}}]$ l'approximation à l'étape $q-1$ de la fonction $f_{\mathbf{x}}$ associée au paramètre $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ et qui s'écrit ainsi :

$$A_{q-1}[f_{\mathbf{x}}] = \sum_{k=1}^{q-1} \psi_k(\mathbf{x}) w_k,$$

où les valeurs des coefficients en $\mathbf{x}$ sont maintenant définies comme les solutions du problème d'optimisation quadratique convexe suivant :

$$\begin{aligned} \underset{\psi_1(\mathbf{x}),\ldots,\psi_{q-1}(\mathbf{x})\in\mathbb{R}}{\text{Minimiser}} \quad & \left\| f_{\mathbf{x}} - \sum_{k=1}^{q-1} \psi_k(\mathbf{x}) w_k \right\|_{L^2}^2 \\ \text{soumis à} \quad & \begin{cases} \psi_k(\mathbf{x}) \geq 0 & (k = 1, \ldots, q-1) \\ \displaystyle\sum_{k=1}^{q-1} \psi_k(\mathbf{x}) = 1. \end{cases} \end{aligned} \tag{4.17}$$

A l'étape q, le paramètre $\mathbf{x}_{i_q}$ est choisi de la même manière que dans l'algorithme MP. La différence est que la nouvelle fonction de base w_q est alors $f_{\mathbf{x}_{i_q}}$. L'algorithme *Modified Magic Points* (MMP) ainsi obtenu est détaillé ci-dessous.

Algorithme 4.2 Modified Magic Points - MMP

– Définir $q \leftarrow 1$ et $I_0 \leftarrow 0$.

– Pendant que $\varepsilon > tol$

1. Choisir $\mathbf{x}_{i_q}$ dans $\boldsymbol{\mathcal{X}}$:
$$\mathbf{x}_{i_q} \leftarrow \arg\sup_{\mathbf{x}\in\boldsymbol{\mathcal{X}}} \|f_{\mathbf{x}} - A_{q-1}[f_{\mathbf{x}}]\|_{L^2}.$$
2. Définir le prochain élément de la base :
$$w_q \leftarrow f_{\mathbf{x}_{i_q}}.$$
3. Calculer l'erreur d'approximation :
$$\varepsilon \leftarrow \|err_q\|_{L^\infty(\boldsymbol{\mathcal{X}})}, \text{ où } err_q : \mathbf{x} \mapsto \|f_{\mathbf{x}} - A_q[f_{\mathbf{x}}]\|_{L^2}.$$
4. Imposer $q \leftarrow q+1$.

A l'étape (1) de l'algorithme MMP, la norme L^2 peut être remplacée par la distance de Hellinger. Ainsi, les fonctions de base sont choisies en fonction de l'erreur d'approximation calculée avec la distance de Hellinger plutôt que la distance L^2.

En pratique, l'algorithme MMP (tout comme l'algorithme MP) n'est pas appliqué à l'ensemble $\mathcal{M}_d(\boldsymbol{\mathcal{X}}, .)$ tout entier puisqu'il n'est pas possible de calculer l'erreur d'approximation sur tous les points de l'espace. Il est appliqué à un échantillon de densités de probabilité $\{f_1, \ldots, f_N\}$. Alors, l'étape (1) de l'algorithme 4.2.2.3 n'est plus qu'une maximisation sur un ensemble fini de N fonctions. De plus, une grille régulière de M points $t_1, \ldots, t_M$ est construite sur l'intervalle I :

$$\begin{aligned} & t_j = t_1 + (j-1)\Delta t \qquad (j = 1, \ldots, M) \\ & \text{avec } \Delta t \in \mathbb{R}_+, \end{aligned} \tag{4.18}$$

et les fonctions f_i $(i = 1, \ldots, N)$ et w_k $(k = 1, \ldots, q)$ sont discrétisées sur cette grille :

$$\begin{aligned} \mathbf{f}_{ij} &= f_i(t_j) \\ \mathrm{w}_{kj} &= w_k(t_j) \end{aligned} \qquad (j = 1, \ldots, M).$$

Ainsi, le problème (4.17) se réécrit $\forall i \in \{1, \ldots, N\}$ comme suit :

$$\begin{aligned} \underset{\psi_{i1},\ldots,\psi_{iq}}{\text{Minimiser}} \quad & \sum_{j=1}^{M} \left[\mathbf{f}_{ij} - \sum_{k=1}^{q} \psi_{ik} \mathrm{w}_{kj} \right]^2 \Delta t \\ \text{soumis à} \quad & \begin{cases} \psi_{i1}, \ldots, \psi_{iq} \geq 0 \\ \displaystyle\sum_{k=1}^{q} \psi_{ik} = 1. \end{cases} \end{aligned} \tag{4.19}$$

Ces N problèmes quadratiques convexes ont q inconnues et $q+1$ contraintes linéaires, et sont assez faciles à résoudre pour des tailles de base inférieures à 10^4.

Minimisation Quadratique Alternée (MQA)

La troisième décomposition fonctionnelle sous contrainte étudiée consiste à résoudre directement le problème de minimisation de l'erreur L^2 d'approximation avec les contraintes de l'équation (4.16). Cette décomposition diffère de l'ACP car aucune condition d'orthonormalité n'est imposée, et la décomposition est réalisée directement sur les données non centrées. Le problème de minimisation fonctionnelle à résoudre peut s'écrire de la manière suivante :

$$\begin{aligned} \underset{\substack{w_k, \psi_{ik} \\ k=1,\ldots,q, i=1,\ldots,N}}{\text{Minimiser}} \quad & \frac{1}{2} \sum_{i=1}^{N} \left\| f_i - \sum_{k=1}^{q} \psi_{ik} w_k \right\|_{L^2}^2 \\ \text{soumis à} \quad & \begin{cases} w_k \in \mathcal{F} & (k = 1, \ldots, q) \\ \psi_{ik} \geq 0 & (i = 1, \ldots, N)(k = 1, \ldots, q) \\ \displaystyle\sum_{k=1}^{q} \psi_{ik} = 1 & (i = 1, \ldots, N). \end{cases} \end{aligned} \tag{4.20}$$

Comme précédemment, au lieu de travailler directement sur les fonctions, on en considère une version discrétisée. Elles sont discrétisées sur le même ensemble de points que dans l'équation (4.18) et on définit les vecteurs suivant :

$$\begin{aligned} \mathbf{f}_i &= \begin{bmatrix} f_i(t_1) & \cdots & f_i(t_M) \end{bmatrix} \qquad (i = 1, \ldots, N) \\ &= \begin{bmatrix} \mathrm{f}_{i1} & \cdots & \mathrm{f}_{iM} \end{bmatrix} \\ \mathbf{w}_k &= \begin{bmatrix} w_k(t_1) & \cdots & w_k(t_M) \end{bmatrix} \qquad (k = 1, \ldots, q) \\ &= \begin{bmatrix} \mathrm{w}_{k1} & \cdots & \mathrm{w}_{kM} \end{bmatrix}. \end{aligned}$$

Le problème de minimisation (4.20) est remplacé par le problème vectoriel suivant (dans cette section

$\|.\|_{\mathbb{R}^M}$ désigne la norme euclidienne sur $\mathbb{R}^M$) :

$$\begin{aligned}
&\underset{\substack{\mathbf{w}_k,\psi_{ik}\\ k=1,\dots,q,i=1,\dots,N}}{\text{Minimiser}} \quad \frac{1}{2}\sum_{i=1}^{N}\left\|\mathbf{f}_i-\sum_{k=1}^{q}\psi_{ik}\mathbf{w}_k\right\|_{\mathbb{R}^M}^2\\
&\text{soumis à} \quad \begin{cases}
\mathrm{w}_{kj}\geq 0 & (k=1,\dots,q)(j=1,\dots,M)\\
\displaystyle\sum_{j=1}^{M}\mathrm{w}_{kj}\Delta t=1 & (k=1,\dots,q)\\
\psi_{ik}\geq 0 & (i=1,\dots,N)(k=1,\dots,q)\\
\displaystyle\sum_{k=1}^{q}\psi_{ik}=1 & (i=1,\dots,N).
\end{cases}
\end{aligned} \tag{4.21}$$

On introduit les notations suivantes pour reformuler la fonction objectif et les contraintes d'une manière plus dense :

$$\mathbf{\Psi}=[\psi_{ik}]_{i=1,\dots,N;k=1,\dots,q}=\begin{bmatrix}\boldsymbol{\psi}_1\\ \vdots\\ \boldsymbol{\psi}_N\end{bmatrix} \text{ et } \mathbf{W}=[\mathrm{w}_{kj}]_{k=1,\dots,q;\ j=1,\dots,M}=\begin{bmatrix}\mathbf{w}_1\\ \vdots\\ \mathbf{w}_q\end{bmatrix}.$$

On rappelle que la norme de Frobenius d'une matrice $\mathbf{A}=[a_{ij}]_{i=1,\dots,N;j=1,\dots,M}\in\mathbb{R}^{N\times M}$ est définie ainsi :

$$\|\mathbf{A}\|_{\mathrm{F}}=\sqrt{\sum_{i=1}^{N}\sum_{j=1}^{M}a_{ij}^2}.$$

Le problème de minimisation vectorielle (4.21) peut être réécrit sous forme matricielle :

$$\begin{aligned}
&\underset{\mathbf{\Psi}\in\mathbb{R}_+^{N\times q},\mathbf{W}\in\mathbb{R}_+^{q\times M}}{\text{Minimiser}} \quad \mathrm{O}\left(\mathbf{\Psi},\mathbf{W}\right)=\frac{1}{2}\left\|\mathbf{F}-\mathbf{\Psi}\mathbf{W}\right\|_{\mathrm{F}}^2\\
&\text{soumis à} \quad \begin{cases}
\boldsymbol{\psi}_i\mathbf{1}=1 & (i=1,\dots,N)\\
\mathbf{w}_k\mathbf{1}=1/\Delta t & (k=1,\dots,q).
\end{cases}
\end{aligned} \tag{4.22}$$

La fonction objectif du problème (4.21) est un polynôme du second degré, et donc deux fois continûment différentiable (mais pas nécessairement convexe). Ses dérivées du premier ordre s'expriment ainsi :

$$\begin{aligned}
\partial_{\psi_{ik}}\mathrm{O}\left(\mathbf{\Psi},\mathbf{W}\right)&=-\mathbf{w}_k\left(\mathbf{f}_i-\boldsymbol{\psi}_i\mathbf{W}\right)^T && (i=1,\dots,N)(k=1,\dots,q),\\
\partial_{w_{kj}}\mathrm{O}\left(\mathbf{\Psi},\mathbf{W}\right)&=-\mathbf{\Psi}_{.,k}^T\left(\mathbf{F}_{.,j}-\mathbf{\Psi}\mathbf{W}_{.,j}\right) && (k=1,\dots,q)\,(j=1,\dots,M),
\end{aligned}$$

où $\mathbf{\Psi}_{.,k}$ est la k^{e} colonne de $\mathbf{\Psi}$, les $\mathbf{F}_{.,j}$ et $\mathbf{W}_{.,j}$ sont les j^{e} colonnes de $\mathbf{F}$ et $\mathbf{W}$ respectivement. Les dérivées secondes de la fonction objectif sont les suivantes :

$$\begin{aligned}
\partial^2_{\psi_{ik}\psi_{ik'}}\mathrm{O}\left(\mathbf{\Psi},\mathbf{W}\right)&=\mathbf{w}_k\mathbf{w}_{k'}^T && (i=1,\dots,N)(k,k'=1,\dots,q),\\
\partial^2_{w_{kj}w_{kj'}}\mathrm{O}\left(\mathbf{\Psi},\mathbf{W}\right)&=\delta_{jj'}\left\|\mathbf{\Psi}_{.,k}\right\|_{\mathbb{R}^N}^2 && (k=1,\dots,q)\,(j,j'=1,\dots,M).
\end{aligned}$$

Le problème (4.22) a $q(M+N)$ ($\approx 10^4$ dans les cas étudiés) variables, $q(M+N)+q+N$ contraintes, et peut ne pas être convexe. Par conséquent, sa résolution peut être très coûteuse en temps de calcul. Pour contourner cette limitation, on décompose (4.22) en plusieurs minimisations quadratiques convexes. Cette décomposition du problème (4.22) peut se formuler ainsi :

$$\inf_{\substack{(\mathbf{\Psi},\mathbf{W})\in\left(\mathbb{R}_+^{N\times M}\right)^2\\ \boldsymbol{\psi}_i\mathbf{1}=1\ (i=1,\dots,N)\\ \mathbf{w}_k\mathbf{1}=1/\Delta t\ (k=1,\dots,q)}}\mathrm{O}\left(\mathbf{\Psi},\mathbf{W}\right)=\inf_{\substack{\mathbf{w}_1\in\mathbb{R}_+^M\\ \mathbf{w}_1\mathbf{1}=1/\Delta t}}\cdots\inf_{\substack{\mathbf{w}_q\in\mathbb{R}_+^M\\ \mathbf{w}_q\mathbf{1}=1/\Delta t}}\inf_{\substack{\boldsymbol{\psi}_1\in\mathbb{R}_+^q\\ \boldsymbol{\psi}_1\mathbf{1}=1}}\cdots\inf_{\substack{\boldsymbol{\psi}_N\in\mathbb{R}_+^q\\ \boldsymbol{\psi}_N\mathbf{1}=1}}\mathrm{O}\left(\mathbf{\Psi},\mathbf{W}\right).$$

L'idée est de minimiser le critère $\mathrm{O}(\boldsymbol{\Psi}, \mathbf{W})$ en travaillant sur un vecteur ligne à la fois : $\boldsymbol{\psi}_1$ puis $\boldsymbol{\psi}_2$, ..., $\boldsymbol{\psi}_N$ puis $\mathbf{w}_1$, $\mathbf{w}_2$, ..., $\mathbf{w}_N$. Ce processus est répété jusqu'à convergence.

Pour $i \in \{1, \ldots, N\}$, on remarque que les deux problèmes suivants sont équivalents, dans le sens où leurs solutions optimales sont les mêmes :

$$\begin{array}{ll} \underset{\boldsymbol{\psi}_i}{\text{Minimiser}} & \mathrm{O}(\boldsymbol{\Psi}, \mathbf{W}) \\ \text{soumis à} & \begin{cases} \boldsymbol{\psi}_i \in \mathbb{R}_+^q \\ \boldsymbol{\psi}_i \mathbf{1} = 1 \end{cases} \end{array} \iff \begin{array}{ll} \underset{\boldsymbol{\psi}_i}{\text{Minimiser}} & \frac{1}{2} \left\| \mathbf{f}_i - \boldsymbol{\psi}_i \mathbf{W} \right\|_{\mathbb{R}^M}^2 \\ \text{soumis à} & \begin{cases} \boldsymbol{\psi}_i \in \mathbb{R}_+^q \\ \boldsymbol{\psi}_i \mathbf{1} = 1. \end{cases} \end{array}$$

L'algorithme proposé pour résoudre le problème (4.21) est le suivant. La convergence de cet algorithme n'a pas été démontrée.

Algorithme 4.3 Minimisation Quadratique Alternée - MQA

– Initialiser $\boldsymbol{\Psi}$ et $\mathbf{W}$ avec des valeurs uniformes.

$$\begin{array}{ll} \psi_{ik} = 1/q & \text{pour } i = 1, \ldots, N \text{ et } k = 1, \ldots, q \\ \mathrm{w}_{kj} = 1/(M\Delta t) & \text{pour } k = 1, \ldots, q \text{ et } j = 1, \ldots, M. \end{array}$$

– Tant que le critère d'arrêt n'est pas atteint (*i.e.* un nombre maximal $iter_{\max} \approx 20$ d'itérations),

1. Pour $i = 1, \ldots, N$, résoudre

$$\begin{array}{ll} \underset{\boldsymbol{\psi}_i}{\text{Minimiser}} & \frac{1}{2} \left\| \mathbf{f}_i - \boldsymbol{\psi}_i \mathbf{W} \right\|_{\mathbb{R}^M}^2 \\ \text{soumis à} & \begin{cases} \boldsymbol{\psi}_i \in \mathbb{R}_+^q \\ \boldsymbol{\psi}_i \mathbf{1} = 1. \end{cases} \end{array}$$

2. Pour $k = 1, \ldots, q$, résoudre

$$\begin{array}{ll} \underset{\mathbf{w}_k}{\text{Minimiser}} & \mathrm{O}(\boldsymbol{\Psi}, \mathbf{W}) \\ \text{soumis à} & \begin{cases} \mathbf{w}_k \in \mathbb{R}_+^M \\ \mathbf{w}_k \mathbf{1} = 1/\Delta t. \end{cases} \end{array}$$

Chacun des problèmes de minimisation résolus dans cet algorithme ont des fonctions objectifs quadratiques et convexes dépendant de M variables avec autant de contraintes d'inégalité et une contrainte d'égalité. On propose de les résoudre avec la méthode *primal-Dual Active-Set*, décrite en détail dans Goldfarb et Idnani (1983) et spécifiquement construite pour les problèmes quadratiques convexes.

4.2.2.4 Proposition d'un métamodèle basé sur la décomposition fonctionnelle des sorties

Dans la section précédente, plusieurs méthodes de décomposition fonctionnelle sous contrainte ont été proposées pour approcher les réalisations d'un échantillon $\{f_1, \ldots, f_N\}$ de la manière suivante $\forall i \in \{1, \ldots, N\}$:

$$\mathcal{M}_d(\mathbf{x}_i, .) = \sum_{k=1}^{q} \psi_k(\mathbf{x}_i) w_k, \tag{4.23}$$

où $(w_1, \ldots, w_q)$ est la base de décomposition, et $\psi_k(\mathbf{x}_i)$ est le coefficient de la fonction $\hat{f}_i = \mathcal{M}_d(\mathbf{x}_i)$ sur la fonction de base w_k. L'objectif est ici de faire le lien entre les coefficients des fonctions sur la base de décomposition utilisée et les entrées x. On cherche dans cette section des métamodèles $\hat{\psi}_1, \ldots, \hat{\psi}_q$ des fonctions $\psi_1, \ldots, \psi_q : \boldsymbol{\mathcal{X}} \to \mathbb{R}$. A partir de ces métamodèles, on obtient un métamodèle de $\mathcal{M}_d$ sous la forme suivante :

$$\hat{\mathcal{M}}_d(\mathbf{x}, .) = \sum_{k=1}^{q} \hat{\psi}_k(\mathbf{x}) w_k. \tag{4.24}$$

On cherche de plus à ce que les sorties du métamodèle soient des densités de probabilité, *i.e.* $\hat{\mathcal{M}}_d : \boldsymbol{\mathcal{X}} \to \mathcal{F}$.

Pour les décompositions MMP et MQA, par exemple, la recherche de tels métamodèles est difficile. En effet, les coefficients de ces décompositions sont contraints, comme décrit dans les équations (4.19) et (4.20). Les métamodèles $\hat{\psi}_1, \ldots, \hat{\psi}_q$ doivent donc être à valeurs dans le simplexe suivant

$$\left\{ (x_1, \ldots, x_q) \in \mathbb{R}^q : x_1, \ldots, x_q \geq 0, \text{ et } \sum_i^q x_i = 1 \right\},$$

pour que $\mathcal{M}_d$ aient ses valeurs dans $\mathcal{F}$. Par conséquent, il est nécessaire de construire un métamodèle conjointement sur tous les coefficients.

Afin de simplifier la construction du métamodèle, on propose de construire un métamodèle pour chaque fonction ψ_k ($1 \leq k \leq q$), et d'utiliser ensuite la technique proposée par Kneip et Utikal (2001) dans l'ACPC pour contraindre les projections sur la base construite par ACP à être des densités de probabilité. Plus précisément, on suppose qu'une décomposition ACPC, MQA, ou MMP a été appliquée aux densités de probabilité. Un métamodèle, noté ψ_k est construit pour chaque coefficient ψ_k indépendamment des autres coefficients et sans contrainte. Le métamodèle non contraint $\hat{\mathcal{M}}_d$ est construit à partir des $\hat{\psi}_k$ comme dans l'équation 4.24. Si une prédiction $\hat{f}_{\mathbf{x}} = \hat{\mathcal{M}}_d(\mathbf{x}, .)$ du métamodèle n'est pas une densité de probabilité, les valeurs négatives de $\hat{f}_{\mathbf{x}}$ sont prises égales à 0, puis la fonction ainsi modifiée est normalisée à 1. Le nouveau métamodèle contraint, noté $\tilde{\mathcal{M}}_d$, ainsi obtenu s'écrit de la manière suivante pour tout $t \in I$:

$$\tilde{\mathcal{M}}_d(\mathbf{x}, t) = \frac{\max\left\{\hat{\mathcal{M}}_d(\mathbf{x}, t), 0\right\}}{\displaystyle\int_I \max\left\{\hat{\mathcal{M}}_d(\mathbf{x}, s), 0\right\} \mathrm{d}s}.$$

Puisque les métamodèles des fonctions $\psi_1, \ldots, \psi_q$ ne sont pas contraints, un grand nombre de méthodes classiques de métamodélisation peuvent être utilisées (modèles additifs généralisés, polynômes de chaos, réseaux de neurones, métamodèles processus gaussien, etc.). Dans les applications traitées dans la section 4.3, les métamodèles processus gaussiens (Rasmussen et Williams, 2006) sont utilisés. Leur efficacité a été illustrée dans de nombreux exemples de métamodélisation de codes de calcul (Marrel, 2008). La méthode proposée peut néanmoins s'étendre à n'importe quel type de métamodèle (régression linéaire, polynômes de chaos, etc.).

Pour prendre en compte les contraintes, on a proposé d'utiliser un métamodèle non contraint puis d'appliquer une transformation à ses prédictions pour qu'elles respectent ces contraintes. Une autre possibilité pourrait être d'étendre les travaux de Da Veiga et Marrel (2012) et Maatouk et Bay (2014) sur les métamodèles processus gaussien soumis à des contraintes d'inégalité au cas d'une sortie vectorielle contrainte.

4.3 Applications

Les méthodes présentées dans la section 4.2 sont appliquées à deux exemples analytiques de dimensions respectives $d = 1$ et $d = 5$, puis au cas-test du choc thermique pressurisé.

La qualité des méthodes proposées est évaluée à l'aide de neuf quantités d'intérêt. Les trois premières quantités sont les distances L^1, L^2 et de Hellinger entre les densités de probabilité f et leur prédiction

ou approximation $\hat{f}$ définies comme suit :

$$E_{rel,L^1} = 100\frac{\int\limits_I \left|f(t) - \hat{f}(t)\right| \mathrm{d}t}{\int_I |f(t)| \,\mathrm{d}t}$$
$$E_{rel,L^2} = 100\frac{\int\limits_I \left|f(t) - \hat{f}(t)\right|^2 \mathrm{d}t}{\int_I f(t)^2 \mathrm{d}t}$$
$$E_{rel,He} = 100\frac{\int\limits_I \left|\sqrt{f(t)} - \sqrt{\hat{f}(t)}\right|^2 \mathrm{d}t}{\int_I \sqrt{f(t)}^2 \mathrm{d}t}.$$

Les autres quantités d'intérêt sont la moyenne, la variance, et les quantiles à 1%, 25%, 75% et 99% des densités de probabilité. Pour ces quantités, les critères étudiés sont les erreurs relatives entre ces quantités calculées pour une densité de probabilité et sa prédiction ou son approximation :

$$E_{rel,u} = 100\frac{|u - \hat{u}|}{|u|},$$

où u désigne la moyenne, la variance ou un des quantiles étudiés définis comme suit $\inf\{q \in \mathbb{R},\ \hat{F}(q) > p\}$ avec $p \in \mathbb{R}$ et $F(q) = \int_{-\infty}^{q} f(t)\mathrm{d}t$.

Dans chaque cas d'application, l'étude est séparée en trois parties. Tout d'abord, les métamodèles de régression à noyau basés sur la distance L^2 et de Hellinger sont étudiés. L'erreur de prédiction des deux estimateurs est comparée sur une base de test ou par validation croisée. Dans un deuxième temps, les qualités d'approximation des méthodes de décomposition présentées dans la section 4.2.2.3 sont comparées. Dans cette partie, les erreurs étudiées sont des erreurs moyennes d'approximation et non de prédiction de densités de probabilité d'un échantillon de test. Elles ne peuvent donc pas être comparées aux erreurs obtenues avec les régressions à noyau. Enfin, dans une troisième partie, ces méthodes de décomposition sont couplées au métamodèle proposé en section 4.2.2.4. Les erreurs de prédiction de ce métamodèle sont comparées aux erreurs obtenues avec les deux estimateurs de régression à noyau (classique et basé sur la distance de Hellinger).

4.3.1 Exemple analytique en dimension 1

Le code de calcul $\mathcal{M}_s$ du premier exemple analytique est défini de la manière suivante :

$$\mathcal{M}_s(x, \xi_1, \xi_2, U) = (\sin(x(\xi_1 + \xi_2)) + U)\mathbb{1}_{\{\sin(x(\xi_1+\xi_2))+U \geq -1\}}$$

où

$$\begin{aligned} x &\in \boldsymbol{\mathcal{X}} = [0;1] \\ \xi_1 &\sim \mathcal{N}(1,1) \\ \xi_2 &\sim \mathcal{N}(2,1) \\ U &\sim \mathcal{U}([0,1]). \end{aligned}$$

$\mathcal{M}_s$ est donc un code stochastique d'une variable réelle x. L'aléa du code est dû aux trois variables aléatoires ξ_1, ξ_2 et U. Soit $x_0 \in [0,1]$, on cherche ici à estimer la densité de probabilité f_0 de la variable aléatoire $\mathcal{M}_d(x_0, \xi_1, \xi_2, U)$. On définit comme dans la section 4.2.2 le code de calcul $\mathcal{M}_d$ qui à une entrée $x \in \boldsymbol{\mathcal{X}}$ de $\mathcal{M}_s$ associe la densité de probabilité de la variable aléatoire $\mathcal{M}_s(x)$. Les densités de probabilité sont obtenues en utilisant la méthode d'estimation à noyau comme décrit dans la section 4.2.2, et sont discrétisées en 512 points. La Figure 4.3 représente $N = 50$ réalisations de ces densités de probabilité correspondant à N valeurs des entrées x tirées uniformément et indépendamment sur $\boldsymbol{\mathcal{X}}$.

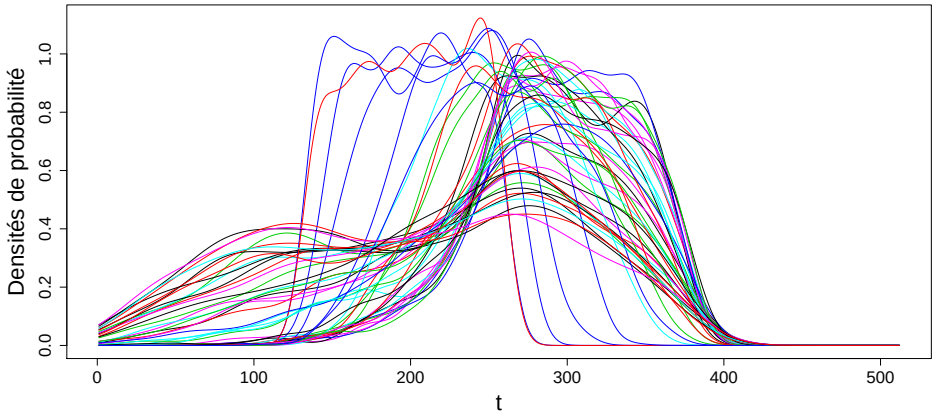

FIGURE 4.3 – Cas analytique 1 : $N = 50$ sorties du code $\mathcal{M}_d$.

Régression à noyau. Dans cette section, on compare les estimateurs basés sur la norme L^2 et la distance de Hellinger définis respectivement par les équations (4.5) et (4.11). La Figure 4.10 représente pour quatre valeurs de $\mathbf{x}_0$ la densité de probabilité $\mathcal{M}_d(\mathbf{x}_0, .)$ (en pointillés noirs) et la prédiction obtenue avec les deux estimateurs basés sur la norme L^2 (ligne continue rouge) et sur la distance de Hellinger (pointillés bleus). Les deux estimateurs sont proches des densités à estimer. Cependant, l'estimateur à noyau basé sur la distance de Hellinger donne de meilleurs résultats dans les quatre cas illustrés.

La qualité des deux métamodèles est ensuite évaluée à l'aide des neuf indicateurs définis au début de la section 4.3 pour des tailles N d'échantillons d'apprentissage différentes. Pour chaque taille N, les indicateurs sont calculés sur une base de test de 1000 densités de probabilité et cette opération est répétée pour 25 bases d'apprentissage indépendantes de taille N. Les boxplots des résultats sont représentés sur la Figure 4.5 en fonction de la taille de l'échantillon d'apprentissage. **L'estimateur basé sur la distance de Hellinger donne des résultats légèrement meilleurs que celui basé sur la norme L^2, notamment pour la variance ou le quantile à 99%.** L'erreur est assez faible pour la plupart des quantités représentées sauf pour les quantiles à 1% et 25%. De plus, l'erreur ainsi que sa variance décroît en fonction de la taille de l'échantillon pour la plupart des quantités étudiées.

Décomposition fonctionnelle contrainte. On compare à présent les quatre décompositions fonctionnelles présentées dans la section 4.2.2.3 : MMP avec la norme L^2 et la distance de Hellinger, MQA et ACPC. Elles sont appliquées à des échantillons d'apprentissage de tailles $N = 50$ et 100. Les résultats obtenus en retenant de $q = 1$ à 20 composantes dans la base de décomposition sont donnés par la Figure 4.6 : les erreurs relatives moyennes sur la base d'apprentissage pour chacune des neuf quantités d'intérêt sont représentées. Tout d'abord, les erreurs relatives sont faibles pour toutes les quantités d'intérêt et en particulier pour les trois distances, la moyenne et les quantiles à 75% et 99%. Pour les trois autres quantités d'intérêt, la variance et les faibles quantiles, les erreurs relatives sont très élevées pour de petites bases de décomposition, mais pour des bases de taille plus élevées, les erreurs sont plus faibles. La décroissance de l'erreur est régulière pour toutes les quantités d'intérêt sauf pour les quantiles à 1% et 25%. **La décomposition ACPC donne de meilleurs résultats que les autres méthodes pour toutes les quantités d'intérêt sauf le quantile à 99%.** Pour ce quantile, les décompositions MMP donnent de meilleurs résultats. Par ailleurs, quelles que soient la quantité d'intérêt et la méthode décomposition, l'erreur relative pour un échantillon d'apprentissage de 100 densités de probabilité est plus élevée que l'erreur pour un échantillon de taille 50. En effet, pour un même nombre de composantes q, il est plus difficile d'approcher un nombre N plus élevé de fonctions, si bien que l'erreur relative augment avec la taille de la décomposition pour un q constant.

On cherche dans la suite de cette section à étudier la construction de l'algorithme MMP. Pour ce faire, on compare la base choisie par la méthode MMP et composée de fonctions de l'échantillon d'apprentissage

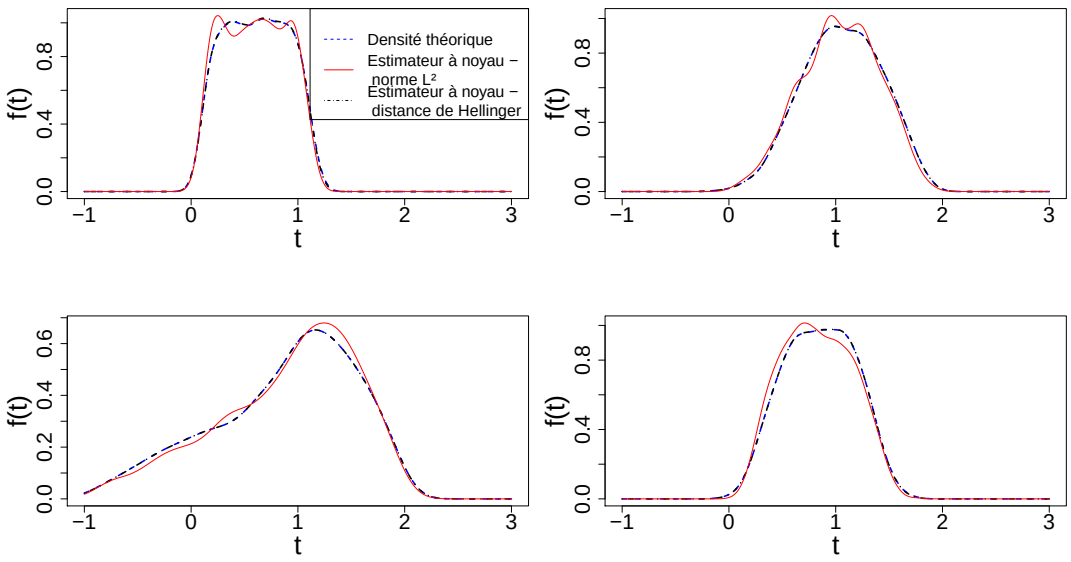

FIGURE 4.4 – Cas analytique 1 : densité de probabilité $\mathcal{M}_d(\mathbf{x}_0)$ à estimer (en pointillés bleu), estimations par les régressions à noyau basées sur la norme L^2 (en ligne continue rouge) et sur la distance de Hellinger (en pointillés noirs).

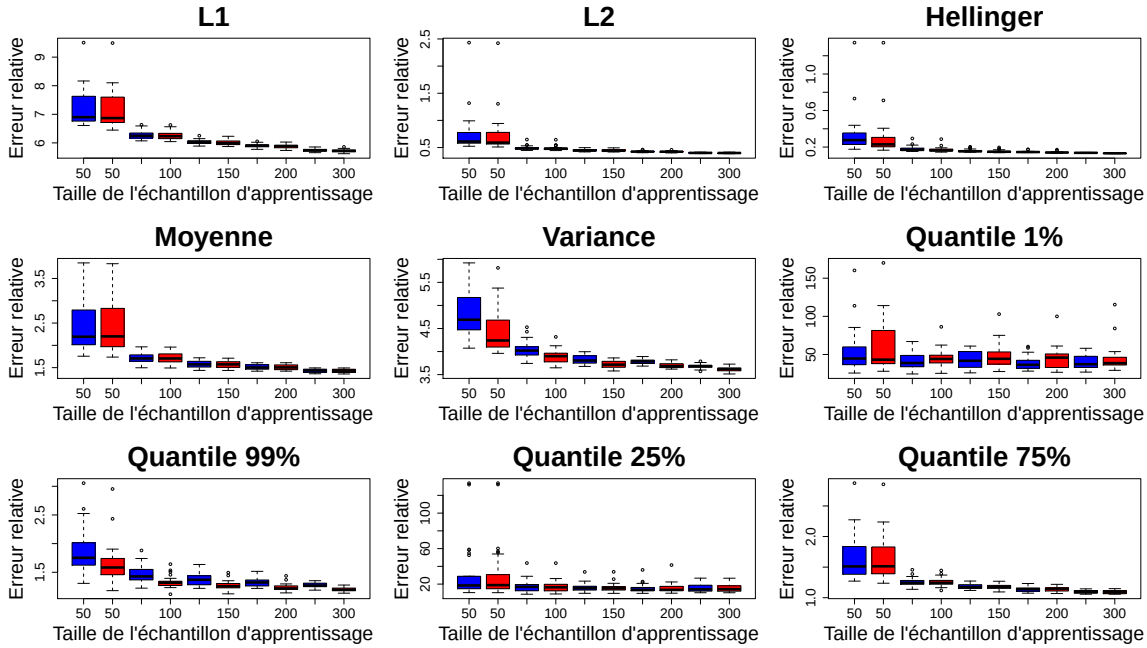

FIGURE 4.5 – Cas analytique 1 : boxplots des erreurs pour différentes tailles de base d'apprentissage. En bleu : l'estimateur à noyau basé sur la norme L^2. En rouge : l'estimateur à noyau basé sur la distance de Hellinger.

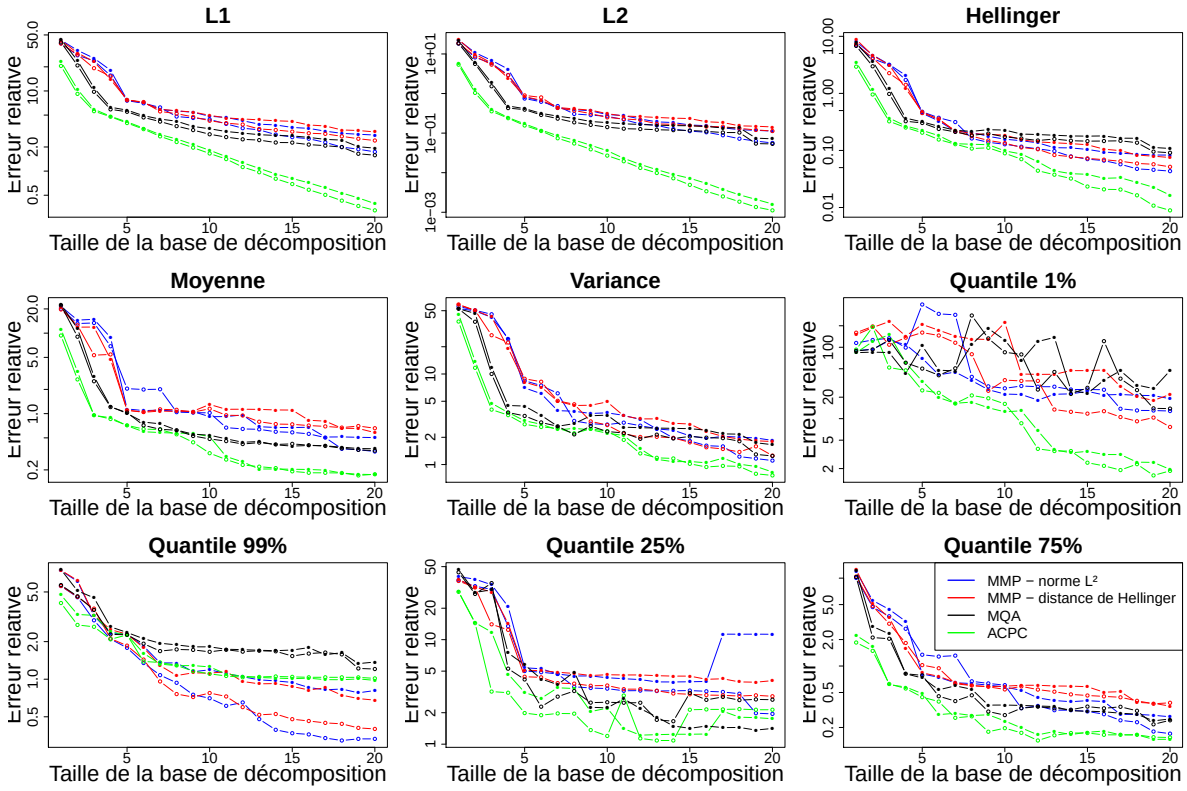

FIGURE 4.6 – Cas analytique 1 : comparaison des erreurs relatives sur les neuf quantités d'intérêt obtenues avec les décompositions MMP avec la norme L^2 (bleu), MMP avec la distance de Hellinger (rouge), MQA (noir) et ACPC (vert) avec un échantillon d'apprentissage de taille $N = 50$ (cercles) et 100 (cercles pleins) en fonction de la taille de la base de décomposition.

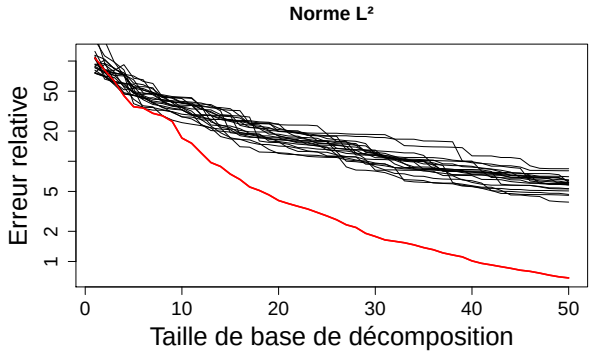

FIGURE 4.7 – Cas analytique 1 : erreur relative sur la norme L^2 avec la méthode MMP (en rouge) et 20 bases choisies aléatoirement parmi les fonctions de l'échantillon initial d'apprentissage (en noir) en fonction de la taille de la base de décomposition.

et des bases composées de fonctions de l'échantillon d'apprentissage choisies aléatoirement. La Figure 4.7 représente l'erreur relative moyenne sur la norme L^2 calculée sur la base d'apprentissage obtenue avec la méthode MMP (en rouge) et celle obtenue avec 20 bases choisies aléatoirement parmi les fonctions de la base d'apprentissage (en noir). La base d'apprentissage comporte ici 200 densités de probabilité. Il apparaît que la méthode MMP est plus performante que les stratégies aléatoires. Ce résultat montre la pertinence de la technique de construction de la base de décomposition de MMP sur cet exemple, puisque ce choix est meilleur qu'une sélection purement aléatoire.

Métamodélisation et décomposition fonctionnelle. La méthode de métamodélisation proposée dans la section 4.2.2.4 est mise en œuvre avec les décompositions ACPC, MMP et MQA. La norme L^2 est utilisée dans la méthode MMP. En effet, la Figure 4.6 montre que la méthode MMP avec les distances L^2 et de Hellinger donne des résultats très proches. Les différents métamodèles sont appris avec une taille de décomposition $q = 10$. Leur prédictivité est ensuite évaluée sur un échantillon de test de 1000 densités de probabilité, comme pour les régressions à noyau. La Figure 4.8 donne, sous forme de boxplots, les erreurs relatives de prédiction des métamodèles calculées sur l'échantillon de test pour les neuf quantités d'intérêt étudiées. Les erreurs des deux estimateurs à noyau sont aussi représentées sur la Figure 4.8 pour comparaison. **Les deux estimateurs à noyau et le métamodèle basé sur la décomposition ACPC donnent les meilleurs résultats.** Ce dernier est meilleur que les estimateurs à noyau pour les distances L^1 et L^2, la moyenne et pour les quantiles 25% et 75%. L'estimateur à noyau basé sur la distance de Hellinger donne de meilleurs résultats pour la variance, le quantile à 99% et la distance de Hellinger. Pour le quantile à 1%, les résultats des trois méthodes sont très proches. Les meilleurs résultats de la méthode ACPC sont dus au fait que la méthode de décomposition donne la meilleure approximation des densités de probabilité parmi les trois méthodes (cf. Figure 4.6). La méthode MMP a les moins bons résultats. Cela s'explique par le fait que de nombreux coefficients de la décomposition sont nuls. En effet, si, par exemple, une densité de l'échantillon est choisie comme fonction de la base de décomposition, alors tous ses coefficients sont nuls sauf un. Cela rend plus difficile l'apprentissage du métamodèle.

4.3.2 Exemple analytique en dimension 5

Soit $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5) \in \boldsymbol{\mathcal{X}} = [0;1]^5$. Le code de calcul stochastique $\mathcal{M}_s$ étudié est défini de la manière suivante :

$$\mathcal{M}_s(\mathbf{x}, N, U_1, U_2, B) = (x_1 + 2x_2 + U_1)\sin(3x_3 - 4x_4 + N) + U_2 + 10x_5 B + \sum_{i=1}^{5} i x_i$$

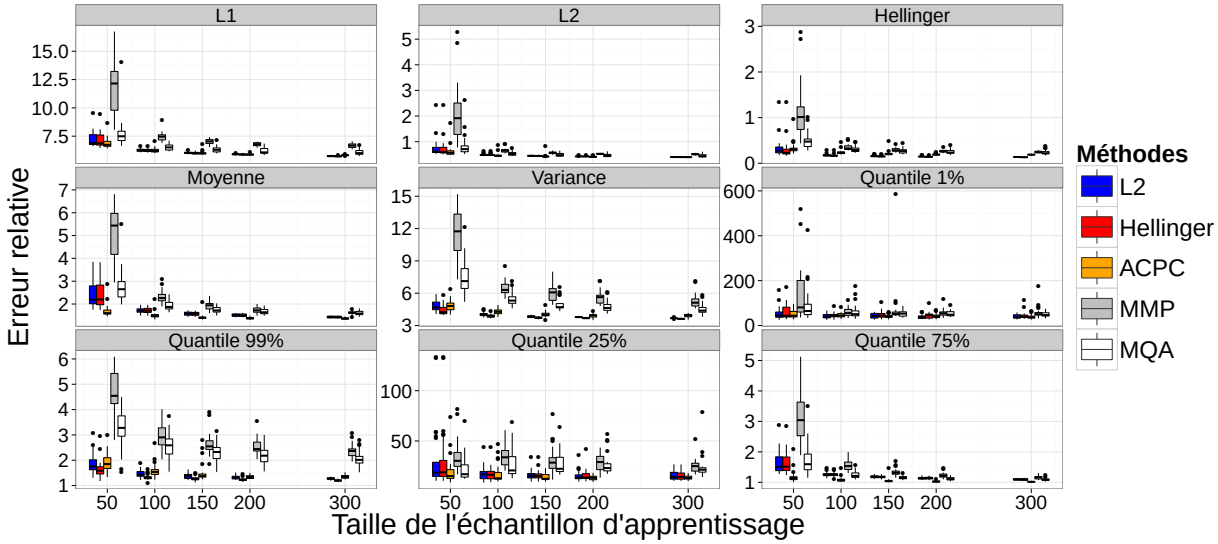

FIGURE 4.8 – Cas analytique 1 : boxplots des erreurs relatives en fonction de la taille N de l'échantillon d'apprentissage pour la régression à noyau basée sur la norme L^2 (bleu), baséz sur la distance de Hellinger (rouge) et les métamodèles basés sur les décompositions ACPC (orange), MMP (gris) et MQA (blanc).

où

$$\begin{aligned} N &\sim \mathcal{N}(0,1) \\ U_1 &\sim \mathcal{U}([0,1]) \\ U_2 &\sim \mathcal{U}([1,2]) \\ B &\sim \mathcal{B}ern(1/2). \end{aligned}$$

L'aléa du code $\mathcal{M}_s$ est dû aux quatre variables aléatoires N, U_1, U_2 et B. Soit $\mathbf{x}_0 \in [0,1]^5$, la quantité à estimer est ici la densité de probabilité f_0 de la variable aléatoire $\mathcal{M}_s(\mathbf{x}_0, N, U_1, U_2, B)$. On définit comme dans la section 4.2.2 le code de calcul $\mathcal{M}_d$ qui à une entrée $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ de $\mathcal{M}_s$ associe la densité de probabilité de la variable aléatoire $\mathcal{M}_s(\mathbf{x})$. Les densités de probabilité sont obtenues en utilisant la méthode d'estimation à noyau comme décrit dans la section 4.2.2. Elles sont discrétisées en 512 points. La Figure 4.9 représente $N = 50$ de ces densités de probabilité correspondant à N valeurs des entrées x tirées sur $\boldsymbol{\mathcal{X}}$ de manière indépendante et uniforme.

Régression à noyau. On commence par évaluer la qualité des deux métamodèles de régression à noyau proposés en utilisant des largeurs de bande isotrope et anisotrope †. Comme dans le premier cas analytique, la Figure 4.10 représente pour quatre valeurs de $\mathbf{x}_0$ la densité de probabilité $\mathcal{M}_d(\mathbf{x}_0, .)$ (pointillés noirs) et la prédiction obtenue avec les deux estimateurs basés sur la norme L^2 (ligne continue rouge) et sur la distance de Hellinger (pointillés bleus) avec une largeur de bande isotrope. Dans cet exemple, les deux estimateurs donnent des résultats différents. L'estimateur $\hat{f}_{0,\mathrm{He}}$ semble bien plus proche de la densité à estimer (en bleu) que l'estimateur $\hat{f}_{0,\mathrm{L}^2}$. Notamment, pour la densité en bas à droite de la figure, la prédiction avec l'estimateur à noyau basé sur la distance L^2 présente deux maxima alors que la densité en bleu n'en présente qu'un seul.

La qualité des deux métamodèles est évaluée sur les neuf quantités d'intérêt définies précédemment pour des tailles N d'échantillons d'apprentissage différentes. Pour chaque taille N, les indicateurs sont

†. La largeur de bande H est isotrope si tous ses éléments diagonaux sont égaux. S'ils sont différents, la largeur de bande est dite anisotrope.

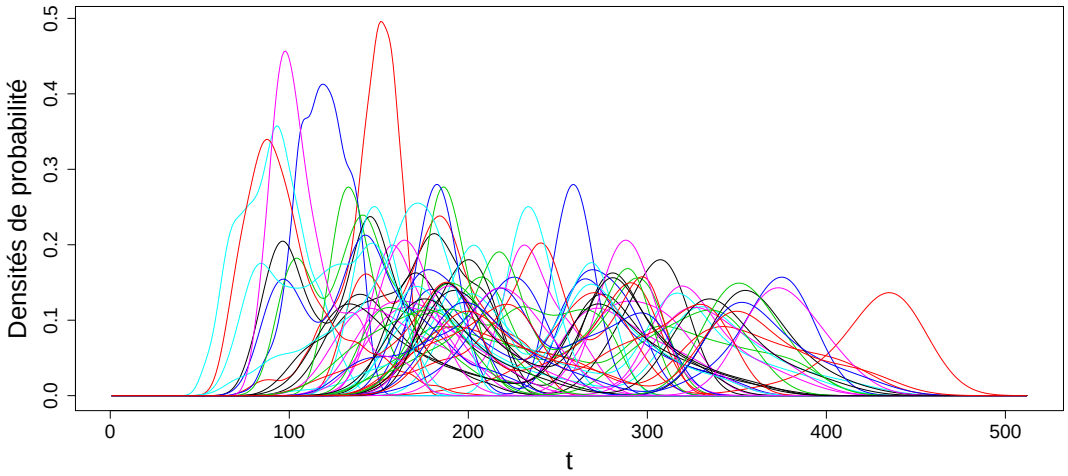

FIGURE 4.9 – Cas analytique 2 : $N = 50$ sorties du code $\mathcal{M}_d$.

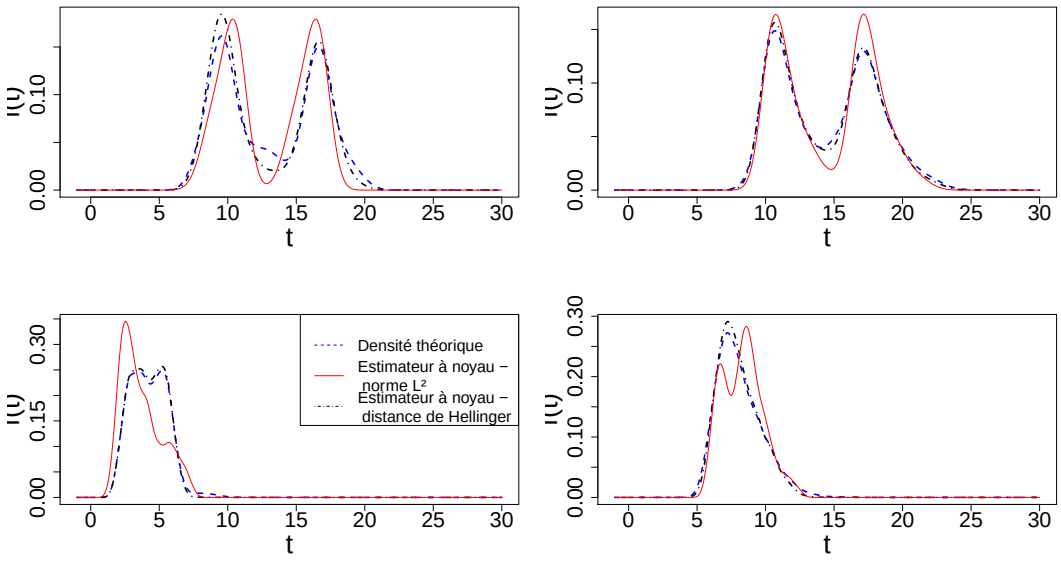

FIGURE 4.10 – Cas analytique 2 : densité de probabilité $\mathcal{M}_d(\mathbf{x}_0)$ à estimer (en pointillés bleu), estimations par les régressions à noyau basées sur la norme L^2 (en ligne continue rouge) et sur la distance de Hellinger (en pointillés noirs).

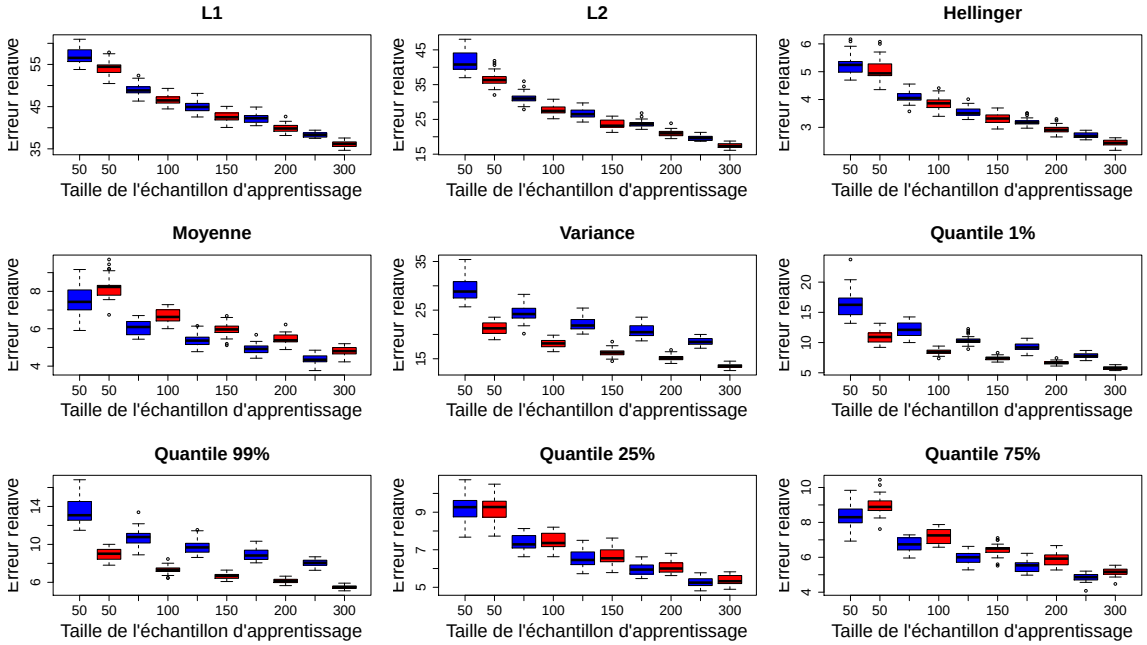

FIGURE 4.11 – Cas analytique 2 : boxplots des erreurs relatives sur les neuf quantités d'intérêt, pour différentes tailles N de l'échantillon d'apprentissage, obtenues avec les estimateurs à noyau basés sur la norme L^2 (en bleu) et sur la distance de Hellinger (en rouge) avec une largeur de bande isotrope.

calculés sur une base de test de 1000 densités de probabilité et cette opération est répétée pour 25 bases d'apprentissage indépendantes de taille N, comme dans le premier cas analytique. Les résultats obtenus avec des largeurs de bande H isotrope et anisotrope sont donnés dans les Figures 4.11 et 4.12. **Dans le cas isotrope, l'estimateur basé sur la distance de Hellinger (en rouge) donne une meilleure prédiction de la plupart des quantités d'intérêt sauf la moyenne et le quantile à 75%.** Les erreurs sont plus élevées que dans le premier cas analytique, en particulier, les erreurs relatives sur les normes L^1 et L^2 ainsi que sur la variance. Cela peut s'expliquer par le fait que les profils de densités de probabilité varient beaucoup selon la valeur de $\mathbf{x}$, comme on peut le voir sur la Figure 4.9. Enfin, les erreurs ainsi que leurs variances décroissent très régulièrement. **L'utilisation d'une largeur de bande anisotrope n'améliore pas la qualité de l'estimation.** Les erreurs sont approximativement les mêmes pour les deux types de noyau. La plus grande précision apportée par l'utilisation d'une largeur de bande anisotrope semble être contrebalancée par la difficulté à l'estimer.

Décomposition fonctionnelle contrainte. Les quatre décompositions fonctionnelles présentées dans la section 4.2.2.3, MMP avec la norme L^2 et la distance de Hellinger, MQA et ACPC, sont appliquées au cas analytique et leurs performances sont comparées. Elles sont mises en œuvre sur des échantillons de tailles $N = 50$ et 100. Les erreurs relatives entre les fonctions de l'échantillon d'apprentissage et leurs projections sur les bases de décomposition sont représentées sur la Figure 4.13 en fonction du nombre de composantes retenues dans la base de décomposition. La décroissance en fonction de q est moins régulière que dans le premier exemple analytique. Les courbes d'erreurs sur les quantiles à 1% et 99% sont par exemple particulièrement instables. Les erreurs sont cependant assez faibles pour la plupart des quantités d'intérêt. La variance et le quantile à 1% ne sont cependant pas bien approchés. **Les décompositions ACPC et MQA donnent de meilleurs résultats que MMP pour la plupart des quantités étudiées. La décomposition ACPC est meilleure que la décomposition MQA pour la plupart des quantités à l'exception de la variance et des quantiles à 1% et 99%.**

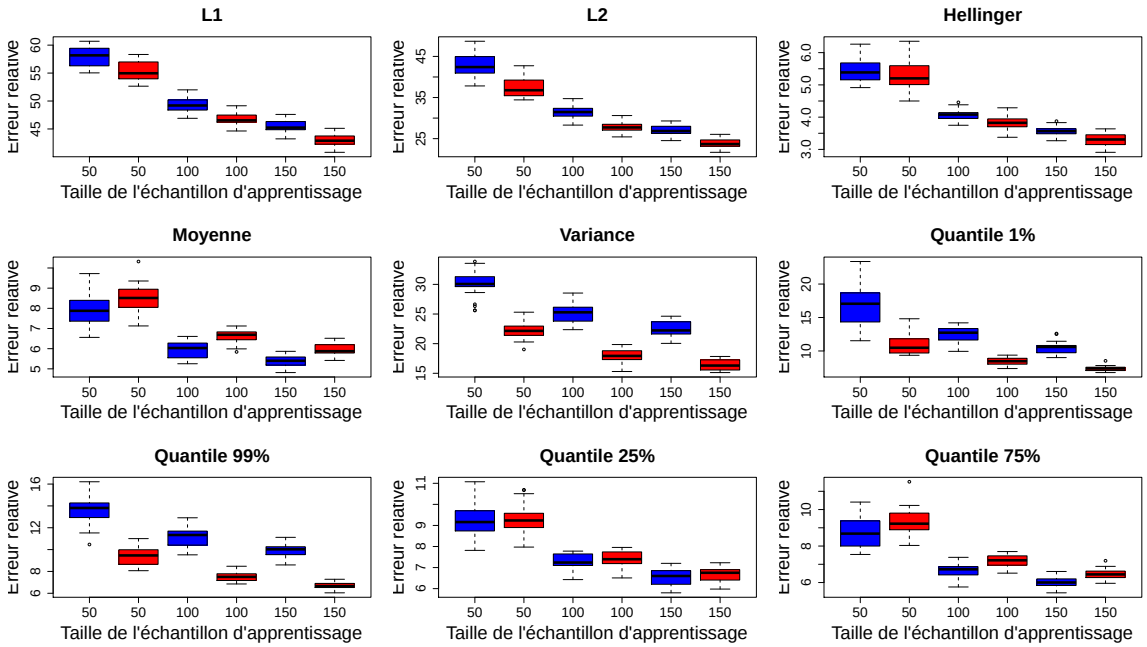

FIGURE 4.12 – Cas analytique 2 : boxplots des erreurs relatives sur les neuf quantités d'intérêt, pour différentes tailles N de l'échantillon d'apprentissage, obtenues avec les estimateurs à noyau basés sur la norme L^2 (en bleu) et sur la distance de Hellinger (en rouge) avec une largeur de bande anisotrope.

Métamodélisation et décomposition fonctionnelle. La méthode de métamodélisation proposée dans la section 4.2.2.4 est ensuite mise en œuvre avec les décompositions ACPC, MMP et MQA. Comme précédemment, la norme L^2 est utilisée dans la méthode MMP, puisque les approximations avec la décomposition MMP en utilisant la distance L^2 ou de Hellinger donnent des résultats très proches. Les différents métamodèles sont appris avec une taille de décomposition $q = 10$. Leur prédictivité est ensuite évaluée sur un échantillon de test de 1000 densités de probabilité. La Figure 4.14 donne les erreurs relatives calculées sur l'échantillon de test pour les neuf quantités d'intérêt étudiées. **Pour la plupart des quantités d'intérêt (distances, variance et quantiles à 1% et 99%), l'estimateur à noyau basé sur la distance de Hellinger donne les meilleurs résultats. Pour les trois autres quantités d'intérêt (moyenne et quantiles à 25% et 75%), la métamodèle utilisant la décomposition ACPC donne les erreurs les plus faibles.** En général, les deux estimateurs à noyau et le métamodèle basé sur l'ACPC prédisent le mieux les densités de probabilité. Il est cependant à noter que ce dernier donne de beaucoup moins bons résultats pour la variance et les quantiles à 1% et 99%. Pour ces trois quantités, les erreurs moyennes d'approximation (cf. Figure 4.13) sont en effet assez élevées pour $q = 10$. Finalement, comme dans le premier cas analytique, le métamodèle basé sur la décomposition MMP donne les moins bons résultats pour toutes les quantités d'intérêt étudiées.

4.3.3 Application au cas du choc thermique pressurisé

Les méthodes présentées dans la section 4.2 sont utilisées ici pour construire un métamodèle pour le code CAST3M. Ce code de calcul possède 3 entrées fonctionnelles et 10 entrées scalaires. Les entrées fonctionnelles sont considérées comme incontrôlables tandis que les 10 variables scalaires sont dites contrôlables. On définit le code $\mathcal{M}_s$ qui pour un jeu des 10 paramètres scalaires $\mathbf{x} \in \mathbb{R}^{10}$ renvoie la sortie de CAST3M correspondant à ce jeu de paramètres scalaires et à un jeu de paramètres fonctionnels choisi aléatoirement. La sortie de $\mathcal{M}_s$ est donc une variable aléatoire conditionnellement aux variables contrôlables. Comme décrit dans la section 4.2.2, on note $\mathcal{M}_d$ le code de calcul ayant les mêmes entrées

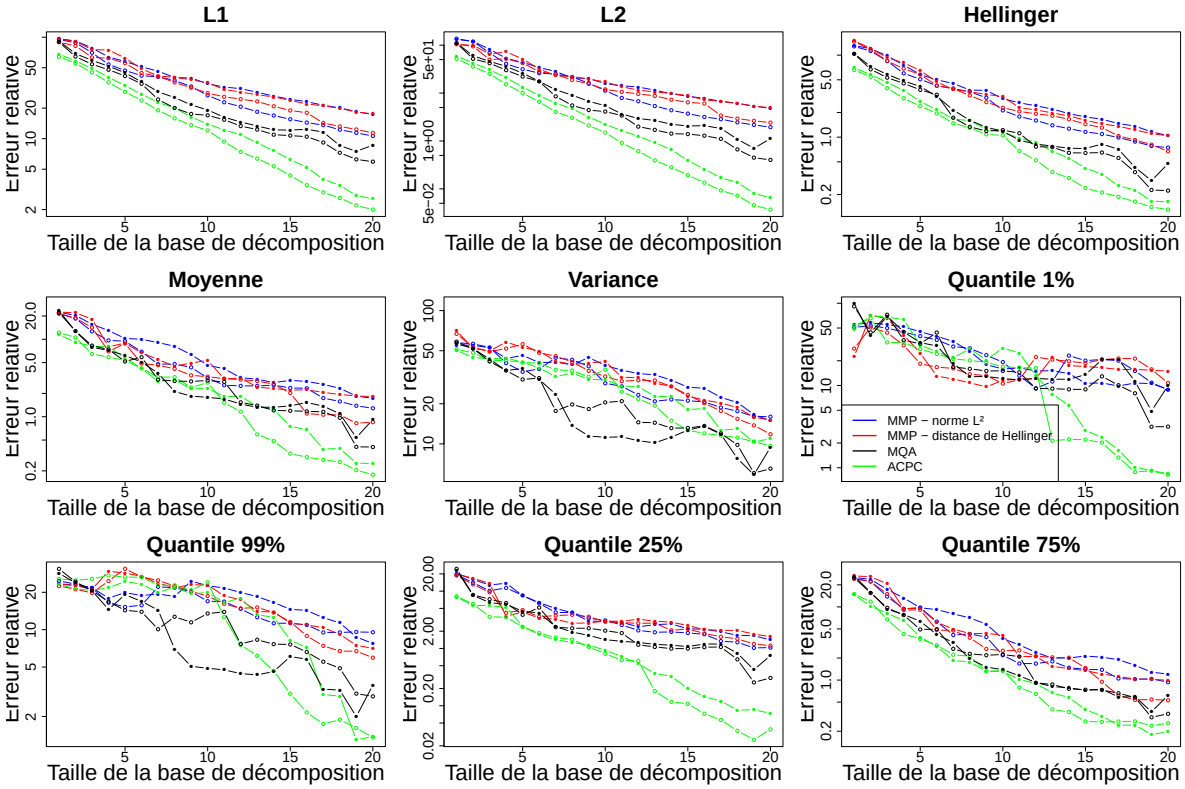

FIGURE 4.13 – Cas analytique 2 : comparaison des erreurs relatives sur les neuf quantités d'intérêt obtenues avec les décompositions MMP avec la norme L^2 (bleu), MMP avec la distance de Hellinger (rouge), MQA (noir) et ACPC (vert) avec un échantillon d'apprentissage de taille $N = 50$ (cercles) et 100 (cercles pleins) en fonction de la taille de la base de décomposition.

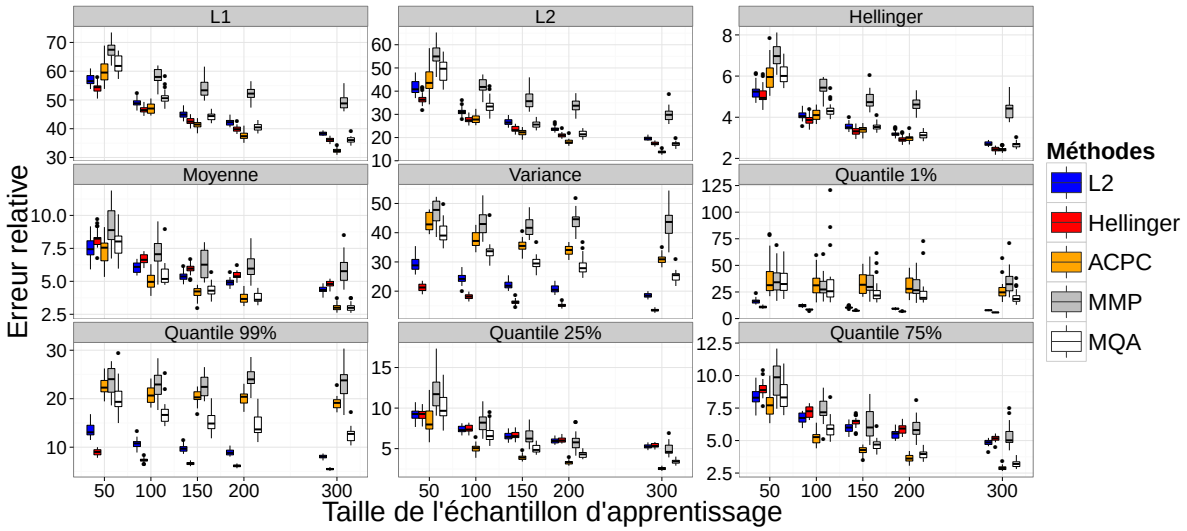

FIGURE 4.14 – Cas analytique 2 : boxplots des erreurs relatives sur les neuf quantités d'intérêt en fonction de la taille N de l'échantillon d'apprentissage pour la régression à noyau basé sur la norme L^2 (bleu), basé sur la distance de Hellinger (rouge) et les métamodèles basés sur les décompositions ACPC (orange), MMP (gris) et MQA (blanc).

que $\mathcal{M}_s$ et dont la sortie pour un jeu d'entrée $\mathbf{x} \in \mathbb{R}^{10}$ est la densité de probabilité de la variable aléatoire $\mathcal{M}_s(\mathbf{x})$.

Un hypercube latin optimisé selon la discrépance centrée (McKay et al., 1979) a été utilisé pour construire un échantillon d'apprentissage des paramètres contrôlables (*i.e.* des 10 paramètres scalaires), constitué de 500 points en dimension $d = 10$. Pour chaque jeu de paramètres contrôlables, CAST3M a été évalué 400 fois avec différentes valeurs des paramètres incontrôlables (*i.e.* des 3 entrées fonctionnelles), tirées aléatoirement parmi un échantillon de réalisations disponibles (tirage avec remise). La densité de probabilité f_i $(i = 1, .., 500)$ de la sortie de CAST3M a été estimée par estimation à noyau à l'aide des 400 sorties de CAST3M pour chacun des 500 jeux de paramètres contrôlables. Un exemple des densités de probabilité obtenues est représenté sur la Figure 4.15.

Régression à noyau. Les deux estimateurs de régression à noyau sont appliqués au cas d'étude avec des largeurs de bande isotrope et anisotrope. Un exemple des résultats obtenus avec ces deux estimateurs est donné dans la Figure 4.16. Sur celle-ci, sont représentés pour quatre jeux de paramètres $\mathbf{x}_0$ différents la densité de probabilité à estimer (pointillés bleus) et les prédictions obtenues avec les deux estimateurs $\hat{f}_{L^2}$ et $\hat{f}_{\text{He}}$ (respectivement en traits pleins rouges et en pointillés orange). La largeur de bande est isotrope sur cet exemple. Les deux estimateurs sont éloignés de la fonction à estimer dans ces quatre exemples. En particulier, en bas à gauche de la figure, la moyenne de la densité de probabilité prédite est très éloignée de la moyenne réelle. L'estimateur basé sur la distance de Hellinger semble donner des résultats légèrement meilleurs.

Pour confirmer cette première analyse graphique, la qualité des deux estimateurs a été évaluée sur les neuf quantités d'intérêt définis précédemment, par validation croisée. La largeur de bande du modèle de régression est estimée en utilisant tous les points de l'ensemble d'apprentissage sauf un noté $\mathbf{x}_i$. La densité de probabilité correspondant au point $\mathbf{x}_i$ est estimée avec le métamodèle construit sans le point $\mathbf{x}_i$, et les erreurs sur les quantités d'intérêt sont calculées entre $\mathcal{M}_d(\mathbf{x}_i, .)$ et son estimation. Cette procédure est répétée pour chaque densité de probabilité. Les moyennes des erreurs relatives des deux estimateurs sont données dans la Table 4.1 pour des largeurs de bande isotrope et anisotrope. **Les erreurs des**

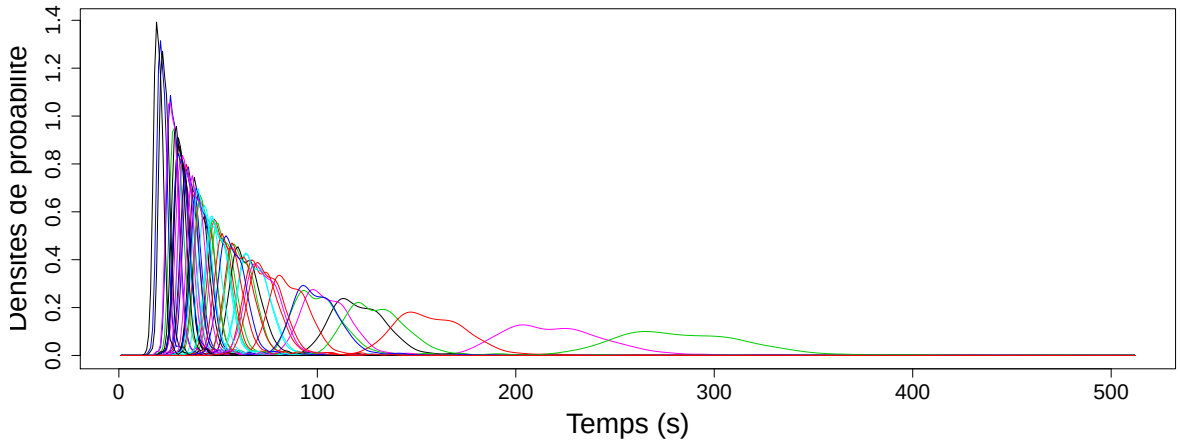

FIGURE 4.15 – Cas du PTS : $N = 50$ sorties du code $\mathcal{M}_d$.

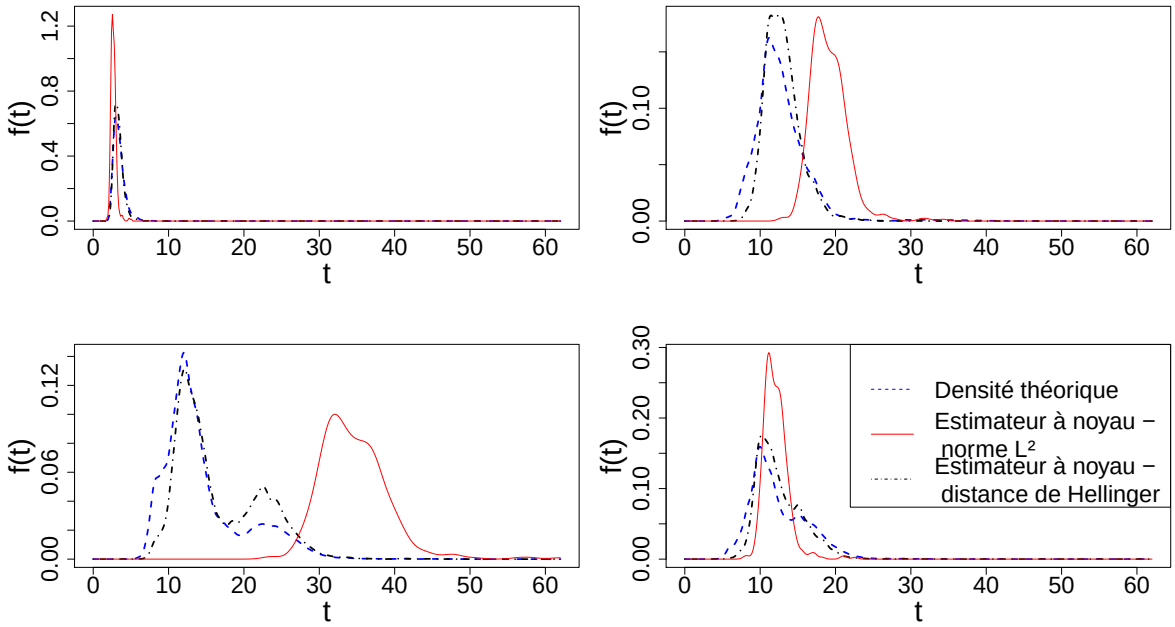

FIGURE 4.16 – Cas du PTS : densité de probabilité $\mathcal{M}_d(\mathbf{x}_0)$ à estimer (en pointillés bleu), estimations par les régressions à noyau basées sur la norme L^2 (en ligne continue rouge) et sur la distance de Hellinger (en pointillés noirs).

Largeur de bande	**Isotrope**		**Anisotrope**	
Estimateur	**distance de Hellinger**	**norme L^2**	**distance de Hellinger**	**norme L^2**
Norme L^1	114%	115%	94%	95%
Norme L^2	168%	172%	108%	109%
Distance de Hellinger	17%	17%	14%	14%
Moyenne	22%	22%	15%	16%
Variance	86%	86%	72%	72%
Quantile à 1%	72%	74%	43%	43%
Quantile à 99%	40%	41%	23%	24%
Quantile à 25%	31%	32%	20%	20%
Quantile à 75%	24%	24%	16%	16%

TABLE 4.1 – Cas du PTS : Erreurs relatives moyenne sur les quantités d'intérêt calculées par leave-one-out pour les estimateurs à noyau basés sur la norme L^2 et la distance de Hellinger avec des largeurs de bande isotrope et anisotrope.

deux estimateurs sont élevées, ce qui confirme les mauvais résultats obtenus sur la Figure 4.16. Les erreurs sur les normes L^1 et L^2 sont particulièrement élevées. Ceci pourrait s'expliquer par l'importante influence des paramètres contrôlables par rapport à celle des paramètres incontrôlables, comme le montre l'analyse de sensibilité faite dans le chapitre 3. **Cependant, l'utilisation d'une largeur de bande anisotrope améliore significativement les résultats pour toutes les quantités d'intérêt.**

Décomposition fonctionnelle contrainte. Les quatre décompositions fonctionnelles présentées dans la section 4.2.2.3 sont utilisées pour approcher les densités de probabilité de la base d'apprentissage. La Figure 4.17 représente les erreurs relatives sur les neuf quantités d'intérêt étudiées en fonction de la taille de la base de décomposition. Les erreurs sont élevées pour des petites bases de décomposition mais décroissent vite pour toutes les méthodes de décomposition. La décroissance est particulièrement rapide et régulière pour les distances L^1, L^2 et de Hellinger. L'erreur sur la variance est plus élevée que les autres puisqu'elle ne décroît pas en dessous de 10%. **La décomposition ACPC est, pour la plupart des quantités d'intérêt et des tailles de base, meilleure que les autres décompositions.** Pour la variance et les quantiles à 1% et 99%, la décomposition MQA est beaucoup moins performante que les autres méthodes et décroît lentement. Les méthodes MQA et ACPC semblent plus stables que la méthode MMP pour la plupart des quantités d'intérêt et en particulier pour les moments et les quantiles. De manière générale, les résultats sont assez bons pour les quatre méthodes.

Métamodélisation et décomposition fonctionnelle. La méthode de métamodélisation des coefficients de la décomposition fonctionnelle de la densité de probabilité, proposée dans la section 4.2.2.4, est ensuite mise en œuvre avec les décompositions ACPC, MMP et MQA. Comme précédemment, la norme L^2 est utilisée dans la méthode MMP, puisque les approximations avec la décomposition MMP en utilisant la distance L^2 ou de Hellinger donnent des résultats très proches. La prédictivité des métamodèles est évaluée par leave-one-out. La Figure 4.18 représente les erreurs relatives sur les neuf quantités d'intérêt obtenues avec les trois métamodèles en fonction de la taille de la base de décomposition utilisée. Les erreurs relatives des trois métamodèles sur les coefficients des décompositions fonctionnelles sont pour la plupart plus élevées que les celles obtenues avec la régression à noyau à largeur de bande anisotrope mais plus basses que celles de la régression à largeur de bande isotrope. Les erreurs du métamodèle sur les coefficients de la décomposition MMP ont un comportement plus erratiques que celles des deux autres métamodèles. Elles augmentent avec la taille de la base de décomposition pour la variance et les quantiles à 1%, 25% et 99%. Les deux autres métamodèles donnent de meilleurs résultats, notamment pour les quantiles.

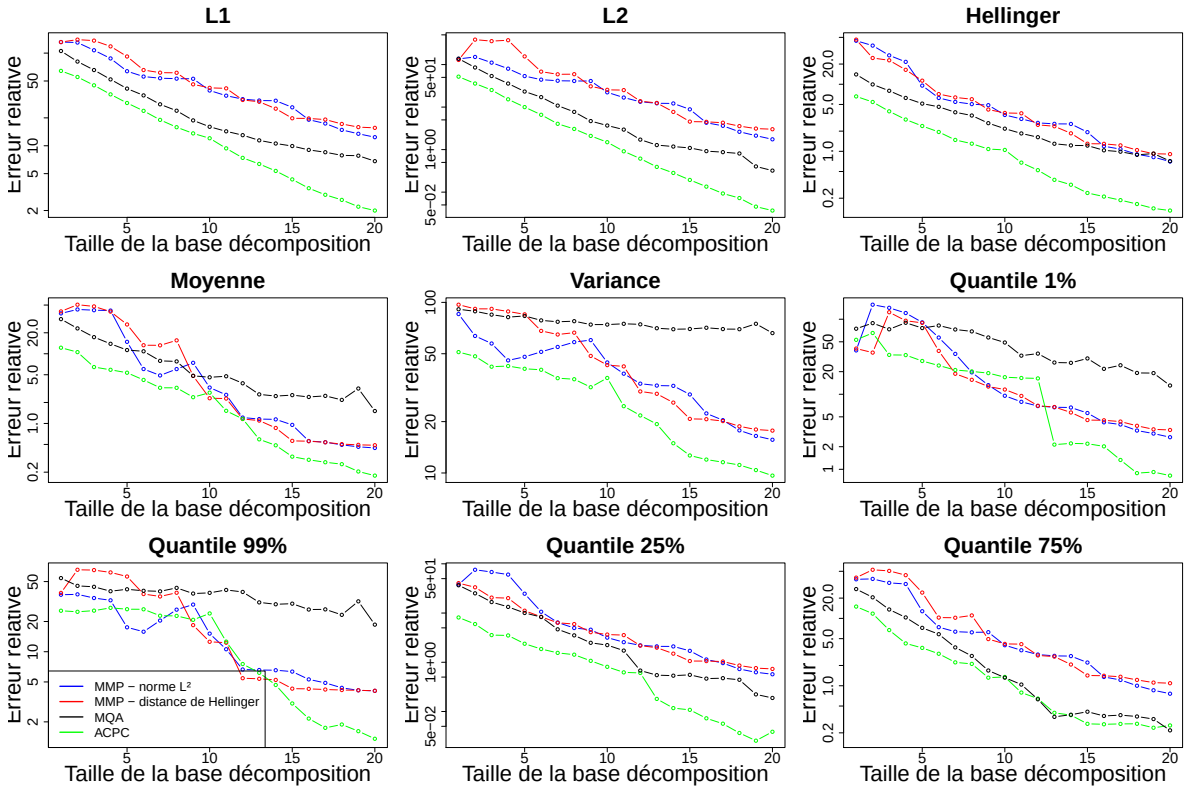

FIGURE 4.17 – Cas du PTS : comparaison des erreurs relatives sur les neuf quantités d'intérêt obtenues avec les décompositions MMP avec la norme L^2 (bleu), MMP avec la distance de Hellinger (rouge), MQA (noir) et ACPC (vert) avec un échantillon d'apprentissage de taille $N = 500$ en fonction de la taille de la base de décomposition.

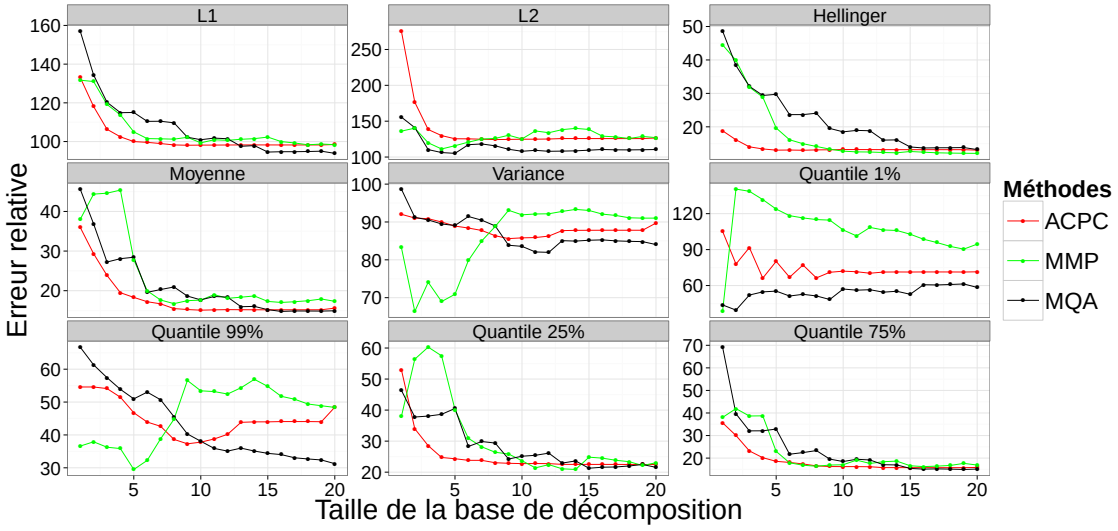

FIGURE 4.18 – Cas du PTS : erreurs relatives moyennes sur les neuf quantités d'intérêt en fonction de la taille d de la base de décomposition pour les métamodèles basés sur les décompositions ACPC (triangles rouges), MMP (croix vertes) et MQA (cercles noirs).

Synthèse des tests sur les trois cas d'application

Dans cette section, on a appliqué à deux exemples analytiques et au cas du PTS différentes méthodes pour modéliser une fonction de densité en sortie d'un code, à savoir les régressions à noyau utilisant la distance L^2 et celle de Hellinger, et les métamodèles basés sur les décompositions fonctionnelles sous contraintes. A l'issue des tests et des comparaisons réalisés, les conclusions suivantes peuvent être faites :

- la régression à noyau utilisant la distance de Hellinger donne globalement de meilleurs résultats que la régression à noyau classique,
- la décomposition ACPC approche en général mieux les densités de probabilité que les décompositions MQA et MMP selon les critères proposés,
- les métamodèles basés sur la décomposition MMP donnent de bien moins bons résultats que ceux basés sur les décompositions MQA et ACPC,
- les deux régressions à noyau et le métamodèle utilisant la décomposition ACPC donnent des résultats assez proches en général.

4.4 Synthèse

Dans ce chapitre l'approximation d'un code de calcul à entrées fonctionnelles par un métamodèle, moins coûteux en temps de calcul, a été étudiée. Pour ce faire, l'approche, proposée dans Iooss et Ribatet (2009) a été utilisée. Celle-ci consiste à considérer les paramètres fonctionnels en entrée du code de calcul comme des variables aléatoires incontrôlables, les autres paramètres (scalaires) étant dits contrôlables. Les paramètres incontrôlables sont alors vus comme un aléas du code de calcul qui varie à chacune de ses évaluations. Dans ce cadre, le code étudié est vu comme un code stochastique, c'est-à-dire un code dont la sortie pour un jeu de paramètres contrôlables fixé est une variable aléatoire caractérisée par sa densité de probabilité. Pour construire un métamodèle sur le code stochastique ainsi défini, plusieurs approches ont été proposées dans la littérature. Celles-ci reposent sur la construction de deux métamodèles interdépendants ou non qui approchent respectivement la moyenne et la variance du code stochastique, conditionnellement aux paramètres contrôlables.

Afin d'obtenir une description plus complète de la sortie du code stochastique, nous avons proposé dans ce chapitre une nouvelle approche basée sur l'approximation de la densité de probabilité de la sortie conditionnellement aux paramètres d'entrée contrôlables. Le problème considéré consiste alors à construire un métamodèle pour un code à entrées scalaires et dont la sortie est une fonction devant respecter plusieurs contraintes. Ces contraintes sont que la sortie doit être positive et que son intégrale doit être égale à un. Deux approches ont été proposées pour construire ce type de métamodèle.

La première est d'utiliser la régression à noyau (Nadaraya, 1964) avec des sorties fonctionnelles. Par sa construction, ce métamodèle permet d'assurer que ses sorties respectent les contraintes imposées. La régression à noyau peut être vue comme une régression locale basée sur la minimisation de la distance L^2. Une modification de cette méthode dans laquelle la distance L^2 est remplacée par la distance de Hellinger a été proposée, et un nouvel estimateur a été obtenu en résolvant le problème de minimisation modifié.

La deuxième approche consiste à décomposer les densités de probabilité sur une base de fonctions puis à construire un métamodèle entre les entrées du code et le coefficient de chaque fonction de la base de décomposition. Pour ce faire, nous avons proposé d'utiliser un métamodèle processus gaussien (Rasmussen et Williams, 2006). Afin d'assurer que la sortie prédite par le métamodèle est une densité de probabilité, ses parties négatives sont prises nulles puis elle est normalisée par son intégrale sur son domaine de variation. Pour réaliser la première étape de la méthode, plusieurs décompositions fonctionnelles sous contraintes ont été étudiées. Tout d'abord, une adaptation de la décomposition par analyse en composantes principales fonctionnelle, proposée par Kneip et Utikal (2001) et notée ACPC (Analyse en Composantes Principales Contrainte), a été présentée. Ensuite, deux nouvelles méthodes de construction de la base de décomposition assurant que les approximations soient des densités de probabilité ont été

proposées. La première, notée MMP (*Modified Magic Points*) est l'adaptation d'une méthode de décomposition introduite par Maday et al. (2007), dans laquelle les fonctions de la base de décomposition sont choisies parmi les fonctions de l'échantillon d'apprentissage sur lequel est appliquée la décomposition. Dans la deuxième méthode, notée MQA (Minimisation Quadratique Alternée), la base de décomposition est construite en minimisant sous contraintes l'erreur quadratique d'approximation des fonctions de l'échantillon sur la base. En pratique, le problème d'optimisation initial est remplacé par une série de problèmes quadratiques plus simples qui sont résolus itérativement jusqu'à convergence de la solution.

Les différentes méthodes de métamodélisation ainsi proposées ont été appliquées dans ce chapitre à deux exemples analytiques de codes stochastiques et au cas du choc thermique pressurisé. Pour évaluer la prédictivité des métamodèles et la qualité d'approximation des décompositions fonctionnelles, neuf quantités ont été étudiées : les distances L^1, L^2 et de Hellinger entre les densités observées et leur approximation par le métamodèle, ainsi que les erreurs relatives entre la moyenne, la variance, les quantiles à 1%, 25%, 75% et 99% des densités de probabilité et de leurs approximations. Les codes étudiés dans les deux exemples analytiques ont respectivement un et cinq paramètres d'entrée, et le code CAST3M étudié dans le cas du choc thermique pressurisé possède dix entrées contrôlables. Les deux approches de métamodélisation proposées (celle par régression à noyau et celle utilisant une décomposition fonctionnelle) ont donné d'assez bons résultats sur les exemples analytiques. Concernant la première approche, la régression à noyau basée sur la distance de Hellinger a donné pour la plupart des quantités étudiées de meilleurs résultats que la régression à noyau classique. Dans la deuxième approche, les métamodèles basés sur la décomposition ACPC ont fourni de meilleurs résultats que ceux basés sur les décompositions MQA et MMP. Cela s'explique en partie par le fait que la décomposition ACPC approche mieux les densités de probabilité que les deux autres décompositions dans ces deux cas analytiques. La régression à noyau basée sur la distance de Hellinger et le métamodèle basé sur la décomposition ACPC ont donné des résultats équivalents dans les trois exemples étudiés. Cependant, de moins bons résultats ont été obtenus dans le cas du PTS. Plusieurs voies d'amélioration sont envisagées pour améliorer ces résultats.

Les méthodes étudiées dans ce chapitre pour la métamodélisation d'un code dont la sortie est une densité de probabilité ou pour la décomposition fonctionnelle de densités de probabilité sont basées sur la minimisation de distances fonctionnelles (la distance L^2 ou la distance de Hellinger). Une amélioration possible de ces méthodes pourrait être d'utiliser une distance fonctionnelle donnant plus de poids aux queues des densités de probabilité. Cela pourrait permettre d'améliorer la prédiction des quantiles qui sont sur les exemples étudiés moins bien prédits que les autres quantités considérées. La construction d'un métamodèle basé sur la distance de Wasserstein (Gibbs et Su, 2002), une distance entre distributions de probabilité, pourrait être une autre piste intéressante. Une troisième possibilité d'amélioration des approches proposées pourrait être d'étudier les fonctions de répartition et non les densités de probabilité de la sortie du code stochastique. En effet, pour estimer les densités de probabilité à partir des réalisations de la sortie, la méthode d'estimation à noyau utilisée nécessite l'estimation de paramètres comme la largeur de bande, alors que l'estimation classique de la fonction de répartition empirique ne nécessite pas de paramètre et converge uniformément vers la fonction de répartition. Les sorties du métamodèle devraient dans ce cas respecter d'autres contraintes, à savoir que la sortie approchée est bornée entre 0 et 1 et qu'elle est croissante. Enfin, les méthodes de métamodélisation développées dans ce chapitre pourraient être utilisées en support à la méthode d'analyse de sensibilité globale d'un code de calcul coûteux proposée dans le chapitre 3, et leurs performances pourraient être comparées à celle du métamodèle initialement utilisé dans le chapitre 3.

Conclusion et perspectives

Ces travaux de thèse s'inscrivent dans le cadre du traitement des incertitudes dans les simulateurs numériques utilisés pour modéliser et prédire des phénomènes physiques. Les deux cas d'application étudiés dans cette thèse sont liés aux études de sûreté pour les réacteurs nucléaires. Chaque cas modélise une situation accidentelle : le choc thermique pressurisé (PTS) sur les réacteurs à eau pressurisée du parc nucléaire actuel et la rupture de la Liaison Pompe-Sommier (LiPoSo) sur les réacteurs rapides refroidis au sodium. Ces deux applications ont des caractéristiques assez proches. Les phénomènes physiques mis en jeu sont modélisés par des workflows de deux codes de calcul (ou groupes de codes) chaînés. Un premier code prend en entrée des paramètres incertains thermo-hydrauliques (T-H) et simule des transitoires T-H, fonctions du temps. Ces variables fonctionnelles associées à d'autres paramètres incertains scalaires sont ensuite en entrée d'un second code qui calcule une ou plusieurs quantités d'intérêt. Les seconds codes des workflows étudiés ont donc dans chaque cas des entrées scalaires et des entrées temporelles potentiellement dépendantes entre elles. La distribution de probabilité de ces dernières n'est pas connue directement et seul un échantillon probabilisé de ces variables, obtenu par simulation du premier code, est disponible. Dans les deux cas étudiés, ces réalisations sont issues d'un code de calcul, mais un des objectifs de la méthodologie proposée est d'être applicable quelle que soit leur origine (simulateur ou mesure expérimentale). Le second code, quant à lui, peut être rapide à évaluer comme dans le cas du PTS ou coûteux en temps de calcul comme dans le cas de la rupture LiPoSo.

L'objectif principal des travaux de recherche présentés dans ce mémoire était de développer une méthodologie complète de traitement des incertitudes de simulateurs numériques pour les deux cas étudiés. Plus précisément, la méthodologie proposée devait permettre de quantifier les incertitudes des paramètres fonctionnels en entrée du code de calcul étudié, de propager ces incertitudes pour quantifier leur effet sur la sortie du code, et de déterminer l'influence des paramètres d'entrée scalaires et fonctionnels sur la sortie. En complément de la quantification des incertitudes, la problématique de la visualisation des incertitudes des variables fonctionnelles a aussi été étudiée. Les caractéristiques des deux cas d'étude soulèvent plusieurs problématiques qui ont été traitées dans ces travaux. La méthodologie développée peut ainsi être séparée en trois grandes parties.

La première est l'étape de quantification des incertitudes des paramètres d'entrée du code. En particulier, nous avons proposé une méthode pour quantifier les incertitudes de variables aléatoires fonctionnelles potentiellement dépendantes entre elles à partir d'un échantillon de réalisations probabilisées de ces variables. Cette méthode permet de modéliser et d'estimer la densité de probabilité jointe des variables fonctionnelles. En plus de tenir compte de la dépendance entre les variables fonctionnelles, cette méthode peut permettre de prendre en compte le lien entre ces variables et la sortie du code étudié ou toute autre variable d'intérêt, appelée covariable, si des réalisations de cette covariable sont disponibles. La première étape de la méthode consiste à réduire la dimension des variables étudiées. Elle repose sur la décomposition des variables fonctionnelles sur une base construite par analyse en composantes principales ou par décomposition *Partial Least Squares* (PLS) simultanées, cette dernière méthode ayant l'avantage de prendre en compte le lien avec la covariable. La seconde étape permet ensuite d'estimer la densité de probabilité jointe des coefficients de la décomposition fonctionnelle. La densité jointe a été modélisée par un mélange de gaussiennes, comme cette modélisation semblait pertinente sur les

cas d'étude du PTS et de la rupture LiPoSo. Pour réduire le nombre de paramètres à estimer dans le mélange de gaussiennes et ainsi éviter des problèmes de surapprentissage, nous avons développé un algorithme d'estimation des paramètres du mélange de gaussiennes (noté sEM2). Cette méthode estime, par maximum de vraisemblance, des matrices de covariance creuses à l'aide d'une pénalisation Lasso sur les matrices de covariance. Dans cette méthode, il est possible de ne pas pénaliser de la même manière tous les éléments des matrices de covariance : les éléments diagonaux peuvent, par exemple, ne pas être pénalisés. Ainsi, trois pénalisations différentes ont été proposées et comparées. Plusieurs critères ont aussi été développés pour évaluer la qualité de la méthode globale de quantification des incertitudes et sélectionner de manière optimale les paramètres de la méthodologie tels que la taille de la base de décomposition ou encore le nombre de gaussiennes dans le mélange. Cette méthodologie de quantification des incertitudes a été appliquée à un exemple analytique, ainsi qu'aux cas du PTS et de la rupture LiPoSo. Les résultats obtenus ont permis notamment d'illustrer l'efficacité des décompositions simultanées et en particulier de la PLS simultanée pour approcher les variables fonctionnelles. Pour l'estimation de la densité, la méthode sEM2, sans pénalisation des éléments diagonaux des matrices de covariance, a donné de meilleurs résultats sur ces tests que les méthodes existantes.

La méthode de quantification proposée a aussi été utilisée pour adapter une technique de visualisation de données fonctionnelles, appelée *High Density Region* (HDR) boxplot. L'adaptation proposée permet de visualiser simultanément les incertitudes liées à des données issues de plusieurs variables fonctionnelles, tout en améliorant la qualité de la visualisation puisque cette nouvelle méthode utilise davantage de composantes de la décomposition fonctionnelle par rapport au HDR boxplot original.

Dans une deuxième partie, l'analyse de sensibilité globale des codes de calcul à entrées fonctionnelles dépendantes a été étudiée dans le but de déterminer l'influence des incertitudes des différents paramètres d'entrée sur la sortie et, en particulier, de quantifier la part de variabilité de la sortie expliquée par le groupe des variables aléatoires fonctionnelles (par rapport aux autres variables aléatoires scalaires). Cette analyse de sensibilité se base sur les indices obtenus par décomposition de la variance de la sortie, appelés indices de Sobol'. Tout d'abord, une revue des méthodes d'estimation de ces indices adaptées aux codes à entrées fonctionnelles et/ou dépendantes a été proposée. A l'issue de cette revue, la méthodologie suivante a été retenue. Les variables fonctionnelles sont modélisées à l'aide de la méthode de quantification des incertitudes développée dans la première partie. Pour tenir compte de la dépendance entre variables, celles-ci sont ensuite groupées selon la méthode développée par Jacques et al. (2006), puis les indices de Sobol' correspondant sont estimés par la méthode des plans répliqués. Cette méthodologie a pu être directement mise en œuvre sur le cas du PTS. Elle a permis de révéler la très grande influence d'une des variables scalaires, la hauteur du défaut, ainsi que celle non-négligeable des transitoires T-H.

Dans le cas de la rupture LiPoSo, où le code est plus coûteux en temps de calcul, une variante de cette méthodologie a été proposée. Celle-ci s'appuie sur la construction d'un métamodèle afin de réduire le nombre d'évaluations du code. En effet, son coût de calcul élevé rend impossible l'utilisation directe de méthodes d'analyse de sensibilité globale quantitative. Une solution consiste alors à approcher ce code par un modèle mathématique de substitution ayant un temps de calcul négligeable. Pour construire la base d'apprentissage du métamodèle, nous avons proposé une méthodologie d'échantillonnage uniforme pour des variables scalaires et fonctionnelles. Cette méthodologie consiste à échantillonner indépendamment les variables scalaires et les variables fonctionnelles selon des plans optimisés, puis à combiner les plans d'expériences ainsi obtenus. Pour cela, il est nécessaire que les domaines de variation des variables soient bornés. Pour les variables scalaires, si leur support n'est pas borné, une troncature ne conservant que les zones de plus haute densité est appliquée. Pour les variables fonctionnelles, le support de leur densité de probabilité estimée n'est pas borné (estimation par décomposition fonctionnelle et mélange de gaussiennes). Pour restreindre ce domaine, une troncature conservant les zones de plus haute densité a été proposée. Le support obtenu n'est pas un hypercube contrairement au support restreint des variables scalaires (indépendantes). L'échantillon uniforme de réalisations des variables scalaires est généré à partir d'un plan hypercube latin optimisé selon un critère de discrépance centrée. Celui des variables fonctionnelles est construit à l'aide d'un algorithme de recuit simulé développé par Auffray et al. (2012)

et optimisant le critère *maximin*. On a ensuite proposé de combiner les échantillons obtenus en optimisant une version régularisée du critère *maximin*. Enfin, un métamodèle processus gaussien est construit à l'aide de l'échantillon « mixte » (composé de variables scalaires et fonctionnelles) obtenu, et l'analyse de sensibilité est réalisée à partir de ce métamodèle. Cette méthodologie a été appliquée à un exemple analytique et au cas de la rupture LiPoSo avec succès. Dans ce second cas, elle a notamment permis de montrer l'influence prédominante des variables fonctionnelles sur les sorties du code.

Dans une dernière partie, d'autres approches pour la modélisation des codes de calculs à entrées fonctionnelles ont été explorées. En particulier, nous avons choisi de considérer que les paramètres d'entrée fonctionnels du code sont des paramètres incontrôlables dont les valeurs en entrée de chaque simulation du code ne sont pas directement connues. Par opposition, les autres paramètres d'entrée, scalaires, sont dits contrôlables et sont supposés connus pour chaque simulation. Le code étudié est alors vu comme un code stochastique, *i.e.* un code qui, pour un même jeu de paramètres d'entrée contrôlables, peut donner des sorties différentes, cette différence étant due aux paramètres incontrôlables. Pour un jeu de paramètres d'entrée contrôlables, sa sortie est alors une variable aléatoire caractérisée par sa densité de probabilité. Dans ce contexte, nous avons proposé d'approcher le code stochastique par un métamodèle prédisant à partir des entrées contrôlables la densité de probabilité de la sortie. L'objectif consiste donc à construire un métamodèle pour un code à entrées scalaires et dont la sortie est une fonction devant respecter certaines contraintes. Pour ce faire, nous avons développé plusieurs méthodes que l'on peut classer en deux familles. Dans la première, un métamodèle non-paramétrique de régression à noyau à sortie fonctionnelle est construit sous la contrainte que la prédiction du métamodèle soit une densité de probabilité. L'estimateur de la régression à noyau classique peut être vu comme une régression locale basée sur la minimisation de la distance quadratique. Nous avons proposé une adaptation de cette méthode consistant à remplacer la distance quadratique par la distance de Hellinger. Dans le deuxième type de méthode proposé, la densité de probabilité en sortie du code est d'abord décomposée sur une base fonctionnelle, construite sous contrainte, de manière à assurer que les fonctions approchées sur la base réduite soient des densités de probabilité. Ensuite, un métamodèle processus gaussien est construit pour prédire les coefficients de la décomposition en fonction des paramètres d'entrée du code. Deux nouvelles méthodes de décomposition fonctionnelle sous contrainte ont notamment été développées, à savoir les décompositions *Modified Magic Points* (MMP) et Minimisation Quadratique Alternée (MQA). Ces méthodes de métamodélisation ont été appliquées à deux cas analytiques et au cas du PTS. La régression à noyau utilisant la distance de Hellinger développée dans cette thèse a été plus performante sur les exemples étudiés que la régression classique. Parmi les métamodèles basés sur la décomposition fonctionnelle sous contrainte, ceux basés sur l'analyse en composantes principales contrainte ou sur la décomposition MQA donnent de bons résultats, alors que celui basé sur la méthode MMP approche beaucoup moins bien le code de calcul. Ces deux métamodèles basés sur ACPC et MQA ainsi que la régression à noyau utilisant la distance de Hellinger donnent généralement des résultats assez proches sur les applications étudiées.

Les différentes méthodes développées au cours de cette thèse ont permis de construire une méthodologie de traitement des incertitudes d'un code de calcul pour les deux applications étudiées, permettant de modéliser ces incertitudes, de les visualiser, de les propager sur la sortie du code, et de réaliser une analyse de sensibilité de ce simulateur. Cette méthodologie a été illustrée sur plusieurs exemples analytiques ainsi que sur les deux cas d'étude de scénarios accidentels sur des réacteurs nucléaires. Les résultats obtenus ont permis d'évaluer les méthodes développées, de vérifier qu'elles remplissaient bien les objectifs fixés et de les comparer aux méthodes existantes. L'application des méthodes développées dans cette thèse ainsi que des considérations théoriques permettent de mettre en évidence plusieurs voies d'amélioration.

Dans la méthodologie de quantification des incertitudes de variables fonctionnelles proposée, la décomposition PLS est utilisée pour réduire la dimension des variables fonctionnelles tout en tenant compte de leur lien avec la sortie du code. Cependant, cette technique permet de ne rendre compte que de relations linéaires. Pour tenir compte de liens plus complexes, une solution pourrait être de s'intéresser aux

méthodes de régression PLS non-linéaire (Rosipal, 2010) et d'essayer de les adapter pour la décomposition de variables fonctionnelles.

Concernant l'analyse de sensibilité de codes de calcul coûteux et à entrées fonctionnelles, plusieurs améliorations peuvent être envisagées dans les étapes d'échantillonnage et de métamodélisation. Pour l'échantillonnage uniforme « mixte » (combinant les variables scalaires et fonctionnelles), les supports non bornés des variables scalaires ainsi que ceux des variables fonctionnelles sont tronqués. Il pourrait être intéressant d'assouplir la troncature en autorisant l'échantillonnage d'une certaine proportion de points en dehors du support tronqué. Par ailleurs, au lieu de combiner les échantillons réalisés séparément sur chacun des domaines de variation des entrées scalaires et du groupe des entrées fonctionnelles, on pourrait envisager d'adapter notre méthode d'échantillonnage pour échantillonner simultanément ces deux espaces. Cela nécessite de définir une distance sur l'ensemble de ces deux espaces. Une troisième voie d'amélioration de la méthodologie serait d'adapter l'algorithme d'échantillonnage, proposé par Auffray et al. (2012), utilisé pour générer les coefficients de la décomposition des variables fonctionnelle pour optimiser la discrépance plutôt que le critère *maximin*. En effet, dans le cas scalaire, ce critère donne de meilleurs résultats en pratique (Damblin et al., 2013).

Dans le cadre de la métamodélisation des codes par approche paramètres contrôlables/incontrôlables, une possibilité d'amélioration serait d'étudier les fonctions de répartition et non les densités de probabilité de la sortie du code stochastique. En effet, pour estimer les densités de probabilité à partir des réalisations de la sortie, la méthode d'estimation à noyau utilisée nécessite l'estimation de paramètres comme la largeur de bande, alors que l'estimation classique de la fonction de répartition empirique ne nécessite pas de paramètre et converge uniformément vers la fonction de répartition. La sortie du métamodèle devrait dans ce cas respecter d'autres contraintes, à savoir que la sortie approchée est bornée entre 0 et 1 et qu'elle est croissante. De plus, l'utilisation de l'estimation polynomiale locale de degré 1 ou supérieur (Fan et Gijbels, 1996) en remplaçant la distance L^2 par celle de Hellinger pourrait améliorer la qualité de la métamodélisation. Enfin, d'autres distances que la distance L^2 ou celle de Hellinger pourraient être utilisées dans les méthodes développées. En particulier, il est possible d'utiliser des distances donnant plus de poids aux queues de distribution puisque les tests réalisés ont montré une moins bonne prédiction des quantiles extrêmes.

Bibliographie

Affleck-Graves, J., Money, A. et Troskie, C. (1979). A principal component index subject to constraints. *Investment Analysts Journal*.

Anstett-Collin, F., Goffart, J., Mara, T. et Denis-Vidal, L. (2015). Sensitivity analysis of complex models : Coping with dynamic and static inputs. *Reliability Engineering & System Safety*, 134:268 – 275.

Antoniadis, A. et Fan, J. (2001). Regularization of wavelet approximations. *Journal of the American Statistical Association*, 96(455):939–967.

Askey, R. et Wilson, J. A. (1985). *Some basic hypergeometric orthogonal polynomials that generalize Jacobi polynomials*, volume 319. American Mathematical Soc.

Auder, B. et Fischer, A. (2012). Projection-based curve clustering. *Journal of Statistical Computation and Simulation*, 82(8):1145–1168.

Auffray, Y., Barbillon, P. et Marin, J.-M. (2012). Maximin design on non hypercube domains and kernel interpolation. *Statistics and Computing*, 22(3):703–712.

Babuška, I., Nobile, F. et Tempone, R. (2010). A stochastic collocation method for elliptic partial differential equations with random input data. *SIAM Review*, 52(2):317–355.

Bachoc, F. (2013). *Parametric estimation of covariance function in Gaussian-process based Kriging models. Application to uncertainty quantification for computer experiments*. Thèse de doctorat, Université Paris-Diderot-Paris VII.

Baringhaus, L. et Franz, C. (2004). On a new multivariate two-sample test. *Journal of Multivariate Analysis*, 88(1):190–206.

Baudin, M., Dutfoy, A., Iooss, B. et Popelin, A.-L. (2015). Open TURNS : An industrial software for uncertainty quantification in simulation. *arXiv preprint arXiv :1501.05242*.

Bayarri, M., Berger, J., Cafeo, J., Garcia-Donato, G., Liu, F., Palomo, J., Parthasarathy, R., Paulo, R., Sacks, J. et Walsh, D. (2007). Computer model validation with functional output. *The Annals of Statistics*, pages 1874–1906.

Bebendorf, M., Maday, Y. et Stamm, B. (2014). Comparison of some reduced representation approximations. *In Reduced Order Methods for Modeling and Computational Reduction*, pages 67–100. Springer.

Beck, A. et Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202.

Berveiller, M. (2005). *Eléments finis stochastiques : approches intrusive et non intrusive pour des analyses de fiabilité*. Thèse de doctorat, Université Blaise Pascal-Clermont-Ferrand II.

Bettonvil, B. (1995). Factor screening by sequential bifurcation. *Communications in Statistics-Simulation and Computation*, 24(1):165–185.

Bien, J. et Tibshirani, R. J. (2011). Sparse estimation of a covariance matrix. *Biometrika*, 98(4):807–820.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford University Press.

Blatman, G. et Sudret, B. (2010). An adaptive algorithm to build up sparse polynomial chaos expansions for stochastic finite element analysis. *Probabilistic Engineering Mechanics*, 25(2):183–197.

Byrd, R. H., Lu, P., Nocedal, J. et Zhu, C. (1995). A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208.

Cameron, R. H. et Martin, W. T. (1947). The orthogonal development of non-linear functionals in series of Fourier-Hermite functionals. *Annals of Mathematics*, 48(2):385–392.

Champion, M., Chastaing, G., Gadat, S. et Prieur, C. (2015). L2-Boosting for sensitivity analysis with dependent inputs. *Statistica Sinica*.

Chastaing, G., Gamboa, F. et Prieur, C. (2014). Generalized Sobol sensitivity indices for dependent variables : Numerical methods. *Journal of Statistical Computation and Simulation*, 85(7):1306–1333.

Chastaing, G., Gamboa, F., Prieur, C. et al. (2012). Generalized Hoeffding-Sobol decomposition for dependent variables-application to sensitivity analysis. *Electronic Journal of Statistics*, 6:2420–2448.

Chevreuil, M., Lebrun, R., Nouy, A. et Rai, P. (2013). A least-squares method for sparse low rank approximation of multivariate functions. *arXiv preprint arXiv :1305.0030*.

Coifman, R. R. et Wickerhauser, M. V. (1992). Entropy-based algorithm for best basis selection. *IEEE Trans. Inf. Theory*, 38(2):713–718.

Conover, W. J. (1971). *Practical Nonparametric Statistics*. Wiley.

Crosetto, M. et Tarantola, S. (2001). Uncertainty and sensitivity analysis : tools for GIS-based model implementation. *International Journal of Geographical Information Science*, 15(5):415–437.

Csiszár, I. (1967). Information-type measures of difference of probability distributions and indirect observations. *Studia Sci. Math. Hungar.*, 2:299–318.

Cukier, R., Levine, H. et Shuler, K. (1978). Nonlinear sensitivity analysis of multiparameter model systems. *Journal of Computational Physics*, 26(1):1 – 42.

Cukier, R., Schaibly, J. et Shuler, K. E. (1975). Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. III. Analysis of the approximations. *The Journal of Chemical Physics*, 63(3):1140–1149.

Da Veiga, S. et Marrel, A. (2012). Gaussian process modeling with inequality constraints. *Annales de la Faculté des Sciences de Toulouse*, 21(3):529–555.

Da Veiga, S., Wahl, F. et Gamboa, F. (2009). Local polynomial estimation for sensitivity analysis on models with correlated inputs. *Technometrics*, 51(4):452–463.

Damblin, G., Couplet, M. et Iooss, B. (2013). Numerical studies of space-filling designs : optimization of Latin Hypercube Samples and subprojection properties. *Journal of Simulation*, 7(4):276–289.

De Boor, C. (2001). *A practical guide to splines*. Springer-Verlag.

De Rocquigny, E., Devictor, N. et Tarantola, S. (2008). *Uncertainty in industrial practice*. Wiley.

Delicado, P. (2011). Dimensionality reduction when data are density functions. *Computational Statistics & Data Analysis*, 55(1):401–420.

Dellino, G., Kleijnen, J. P. et Meloni, C. (2010). Robust optimization in simulation : Taguchi and Response Surface Methodology. *International Journal of Production Economics*, 125(1):52 – 59.

Dempster, A. P., Laird, N. M. et Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Donoho, D. L. et Johnstone, I. M. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrica*, 81(3):425–455.

Duong, T. et Hazelton, M. (2003). Plug-in bandwidth matrices for bivariate kernel density estimation. *Journal of Nonparametric Statistics*, 15(1):17–30.

Efron, B. et Stein, C. (1981). The jackknife estimate of variance. *The Annals of Statistics*, 9(3):586–596.

Epanechnikov, V. A. (1969). Non-parametric estimation of a multivariate probability density. *Theory of Probability & Its Applications*, 14(1):153–158.

Fan, J. et Gijbels, I. (1996). *Local polynomial modelling and its applications : Monographs on statistics and applied probability*, volume 66. CRC Press.

Fan, J. et Li, R. (2001). Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360.

Fang, K.-T., Li, R. et Sudjianto, A. (2006). *Design and modeling for computer experiments*. CRC Press.

Ferraty, F. et Vieu, P. (2006). *Nonparametric Functional Data Analysis*. Springer Series in Statistics. Springer.

Fort, J.-C., Klein, T., Lagnoux, A. et Laurent, B. (2013). Estimation of the Sobol indices in a linear functional multidimensional model. *Journal of Statistical Planning and Inference*, 143(9):1590–1605.

Franco, J., Vasseur, O., Corre, B. et Sergent, M. (2009). Minimum Spanning Tree : A new approach to assess the quality of the design of computer experiments. *Chemometrics and Intelligent Laboratory Systems*, 97(2):164 – 169.

Friedman, J., Hastie, T. et Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical Lasso. *Biostatistics*, 9(3):432–441.

Fromont, M., Laurent, B., Lerasle, M. et Reynaud-Bouret, P. (2012). Kernels based tests with non-asymptotic bootstrap approaches for two-sample problem. *In 25th Annual Conference on Learning Theory*, volume 23, pages 1–22.

Fruth, J., Roustant, O. et Kuhnt, S. (2015). Sequential designs for sensitivity analysis of functional inputs in computer experiments. *Reliability Engineering & System Safety*, 134:260–267.

Gannaz, I. (2007). *Estimation par ondelettes dans les modèles partiellement linéaires*. Thèse de doctorat, Université Joseph Fourier - Grenoble I.

Gelfand, I. M. et Fomin, S. V. (2000). *Calculus of variations*. Dover Publications.

Ghanem, R. G. et Spanos, P. D. (1991). *Stochastic finite elements : a spectral approach*. Springer.

Gibbs, A. L. et Su, F. E. (2002). On choosing and bounding probability metrics. *International statistical review*, 70(3):419–435.

Gilquin, L., Prieur, C. et Arnaud, E. (2015). Replication procedure for grouped Sobol' indices estimation in dependent uncertainty spaces. *Information and Inference : A Journal of the IMA*.

Ginsbourger, D., Rosspopoff, B., Pirot, G., Durrande, N. et Renard, P. (2013). Distance-based Kriging relying on proxy simulations for inverse conditioning. *Advances in Water Resources*, 52:275–291.

Goldfarb, D. et Idnani, A. (1983). A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical programming*, 27(1):1–33.

Gupta, M. R. et Chen, Y. (2011). *Theory and Use of the EM Algorithm*, volume 3. Now Pub.

Hall, J., Tarantola, S., Bates, P. et Horritt, M. (2005). Distributed sensitivity analysis of flood inundation model calibration. *Journal of Hydraulic Engineering*, 131(2):117–126.

Hardle, W., Marron, J. S. et al. (1985). Optimal bandwidth selection in nonparametric regression function estimation. *The Annals of Statistics*, 13(4):1465–1481.

Hastie, T. et Tibshirani, R. (1990). *Generalized Additive Models*. Chapman and Hall/CRC.

Hastie, T., Tibshirani, R. et Friedman, J. (2001). *The elements of statistical learning : data mining, inference, and prediction*. Springer New York.

Helton, J., Johnson, J., Sallaberry, C. et Storlie, C. (2006). Survey of sampling-based methods for uncertainty and sensitivity analysis. *Reliability Engineering & System Safety*, 91(10-11):1175 – 1209.

Helton, J. C., Johnson, J. D., Oberkampf, W. L. et Sallaberry, C. J. (2010). Representation of analysis results involving aleatory and epistemic uncertainty. *International Journal of General Systems*, 39(6): 605–646.

Heuvelink, G., Burgers, S., Tiktak, A. et Den Berg, F. V. (2010). Uncertainty and stochastic sensitivity analysis of the GeoPEARL pesticide leaching model. *Geoderma*, 155(3):186–192.

Hoeffding, W. (1948). A class of statistics with asymptotically normal distribution. *The annals of mathematical statistics*, 19(3):293–325.

Homma, T. et Saltelli, A. (1996). Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety*, 52(1):1–17.

Hooker, G. (2007). Generalized functional anova diagnostics for high-dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics*, 16(3).

Höskuldsson, A. (1988). PLS regression methods. *Journal of chemometrics*, 2(3):211–228.

Hyndman, R. J. et Shang, H. L. (2010). Rainbow plots, bagplots, and boxplots for functional data. *Journal of Computational and Graphical Statistics*, 19(1):29–45.

Iman, R. L. et Conover, W. J. (1982). A distribution-free approach to inducing rank correlation among input variables. *Communications in Statistics - Simulation and Computation*, 11(3):311–334.

Iooss, B. (2011). Revue sur l'analyse de sensibilité globale de modéles numériques. *Journal de la Société Française de Statistique*, 152(1):3–25.

Iooss, B. et Ribatet, M. (2009). Global sensitivity analysis of computer models with functional inputs. *Reliability Engineering & System Safety*, 94(7):1194 – 1204.

Jacques, J., Lavergne, C. et Devictor, N. (2006). Sensitivity analysis in presence of model uncertainty and correlated inputs. *Reliability Engineering & System Safety*, 91(10-11):1126 – 1134.

Janon, A., Klein, T., Lagnoux, A., Nodet, M. et Prieur, C. (2014). Asymptotic normality and efficiency of two Sobol index estimators. *ESAIM : Probability and Statistics*, 18:342–364.

Jin, R., Chen, W. et Sudjianto, A. (2005). An efficient algorithm for constructing optimal design of computer experiments. *Journal of Statistical Planning and Inference*, 134(1):268–287.

Johnson, M. E., Moore, L. M. et Ylvisaker, D. (1990). Minimax and maximin distance designs. *Journal of statistical planning and inference*, 26(2):131–148.

Jolliffe, I. (2005). *Principal component analysis*. Wiley Online Library.

Kleijnen, J. P. (1997). Sensitivity analysis and related analyses : a review of some statistical techniques. *Journal of Statistical Computation and Simulation*, 57(1-4):111–142.

Kleijnen, J. P. (2007). *Design and analysis of simulation experiments*, volume 111. Springer.

Kneip, A. et Utikal, K. J. (2001). Inference for density families using functional principal component analysis. *Journal of the American Statistical Association*, 96(454):519–542.

Krishnamurthy, A. (2012). High-dimensional clustering with sparse gaussian mixture models.

Lebrun, R. et Dutfoy, A. (2009). Do Rosenblatt and Nataf isoprobabilistic transformations really differ ? *Probabilistic Engineering Mechanics*, 24(4):577 – 584.

Li, G., Rabitz, H., Yelvington, P. E., Oluwole, O. O., Bacon, F., Kolb, C. E. et Schoendorf, J. (2010). Global sensitivity analysis for systems with independent and/or correlated inputs. *The Journal of Physical Chemistry A*, 114(19):6022–6032.

Lilburne, L. et Tarantola, S. (2009). Sensitivity analysis of spatial models. *International Journal of Geographical Information Science*, 23(2):151–168.

Loève, M. (1955). *Probability theory*. Springer.

López-Pintado, S. et Romo, J. (2009). On the concept of depth for functional data. *Journal of the American Statistical Association*, 104(486):718–734.

Ma, X. et Zabaras, N. (2011). Kernel principal component analysis for stochastic input model generation. *Journal of Computational Physics*, 230(19):7311–7331.

Maatouk, H. et Bay, X. (2014). Gaussian process emulators for computer experiments with inequality constraints.

Maday, Y., Nguyen, N. C., Patera, A. T. et Pau, G. S. (2007). A general, multipurpose interpolation procedure : the magic points. *In Proceedings of the 2nd International Conference on Scientific Computing and Partial Differential Equations*.

Mallat, S. (1989). A theory for multiresolution signal decomposition - The wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693.

Mallat, S. (2008). *A wavelet tour of signal processing*. Academic press.

Mara, T. A. (2009). Extension of the RBD-FAST method to the computation of global sensitivity indices. *Reliability Engineering & System Safety*, 94(8):1274 – 1281.

Mara, T. A. et Rakoto Joseph, O. (2008). Comparison of some efficient methods to evaluate the main effect of computer model factors. *Journal of Statistical Computation and Simulation*, 78(2):167–178.

Mara, T. A. et Tarantola, S. (2012). Variance-based sensitivity indices for models with dependent inputs. *Reliability Engineering & System Safety*, 107:115 – 121.

Marrel, A. (2008). *Mise en oeuvre et utilisation du métamodèle processus gaussien pour l'analyse de sensibilité de modèles numériques*. Thèse de doctorat, Institut national des sciences appliquées de Toulouse.

Marrel, A., Iooss, B., Da Veiga, S. et Ribatet, M. (2012). Global sensitivity analysis of stochastic computer models with joint metamodels. *Statistics and Computing*, 22(3):833–847.

Marrel, A., Iooss, B., Jullien, M., Laurent, B. et Volkova, E. (2011). Global sensitivity analysis for models with spatially dependent outputs. *Environmetrics*, 22(3):383–397.

Marrel, A., Iooss, B., Van Dorpe, F. et Volkova, E. (2008). An efficient methodology for modeling complex computer codes with Gaussian processes. *Computational Statistics & Data Analysis*, 52(10):4731–4744.

McCullagh, P., Nelder, J. A. et McCullagh, P. (1989). *Generalized linear models*, volume 2. Chapman and Hall London.

McKay, M. D., Beckman, R. J. et Conover, W. J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245.

Mclachlan, J. et Krishnan, T. (1997). *The EM algorithm and extension*. Wiley inter-science.

Meyer, Y. et Salinger, D. H. (1995). *Wavelets and operators*, volume 1. Cambridge university press.

Monod, H., Naud, C. et Makowski, D. (2006). Uncertainty and sensitivity analysis for crop models. *In Working with Dynamic Crop Models : Evaluation, Analysis, Parameterization, and Applications*. Elsevier.

Morris, M. D. (2012). Gaussian surrogates for computer models with time-varying inputs and outputs. *Technometrics*, 54(1):42–50.

Morris, M. D. (2014). Maximin distance optimal designs for computer experiments with time-varying inputs and outputs. *Journal of Statistical Planning and Inference*, 144:63 – 68. International Conference on Design of Experiments.

Morris, M. D. et Mitchell, T. J. (1995). Exploratory designs for computational experiments. *Journal of statistical planning and inference*, 43(3):381–402.

Muehlenstaedt, T., Fruth, J. et Roustant, O. (2014). Computer experiments with functional inputs and scalar outputs by a norm-based approach. *arXiv preprint arXiv :1410.0403*.

Muller, H.-G. et Stadtmuller, U. (1987). Variable bandwidth kernel estimators of regression curves. *The Annals of Statistics*, 15(1):182–201.

Myers, R. H., Montgomery, D. C. et Anderson-Cook, C. M. (2009). *Response surface methodology : process and product optimization using designed experiments*, volume 705. John Wiley & Sons.

Nadaraya, E. A. (1964). On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142.

Nelder, J. A. et Lee, Y. (1991). Generalized linear models for the analysis of Taguchi-type experiments. *Applied Stochastic Models and Data Analysis*, 7(1):107–120.

Oakley, J. et O'Hagan, A. (2002). Bayesian inference for the uncertainty distribution of computer model outputs. *Biometrika*, 89(4):769–784.

Ogura, H. (1972). Orthogonal functionals of the Poisson process. *Information Theory, IEEE Transactions on*, 18(4):473–481.

Oliveira-Brochado, A. et Martins, F. V. (2005). Assessing the number of components in mixture models : a review. Rapport technique, Universidade do Porto, Faculdade de Economia do Porto.

Parzen, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572.

Pebesma, E. J. et Heuvelink, G. B. (1999). Latin hypercube sampling of Gaussian random fields. *Technometrics*, 41(4):303–312.

Perrin, G., Soize, C., Duhamel, D. et Funfschilling, C. (2013). Karhunen–Loève expansion revisited for vector-valued random fields : Scaling, errors and optimal basis. *Journal of Computational Physics*, 242:607–622.

Popelin, A.-L. et Iooss, B. (2013). Visualization tools for uncertainty and sensitivity analyses on thermal-hydraulic transients. *In SNA + MC 2013 - Joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo*.

Pregibon, D. (1984). Review : P. McCullagh, J. A. Nelder, Generalized Linear Models. *Ann. Statist.*, 12(4):1589–1596.

Pronzato, L. et Müller, W. G. (2012). Design of computer experiments : space filling and beyond. *Statistics and Computing*, 22(3):681–701.

Ramsay, J. O. et Silverman, B. W. (2005). *Functional Data Analysis*. Springer Series in Statistics. Springer.

Rasmussen, C. E. et Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Reich, B. J., Kalendra, E., Storlie, C. B., Bondell, H. D. et Fuentes, M. (2012). Variable selection for high dimensional Bayesian density estimation : application to human exposure simulation. *Journal of the Royal Statistical Society : Series C (Applied Statistics)*, 61(1):47–66.

Rice, J. A. et Silverman, B. W. (1991). Estimating the Mean and Covariance Structure Nonparametrically When the Data are Curves. *Journal of the Royal Statistical Society. Series B (Methodological)*, 53(1): pp. 233–243.

Rigby, R. et Stasinopoulos, M. (1996). Mean and Dispersion Additive Models. *In* Härdle, W. et Schimek, M., éditeurs : *Statistical Theory and Computational Aspects of Smoothing*, Contributions to Statistics, pages 215–230. Physica-Verlag HD.

Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 27(3):832–837.

Rosipal, R. (2010). Nonlinear partial least squares : An overview. *In Chemoinformatics and Advanced Machine Learning Perspectives : Complex Computational Methods and Collaborative Techniques*, pages 169–189. IGI Global.

Rousseeuw, P. J., Ruts, I. et Tukey, J. W. (1999). The bagplot : A bivariate boxplot. *The American Statistician*, 53(4):382–387.

Ruffo, P., Bazzana, L., Consonni, A., Corradi, A., Saltelli, A. et Tarantola, S. (2006). Hydrocarbon exploration risk evaluation through uncertainty and sensitivity analyses techniques. *Reliability Engineering & System Safety*, 91(10-11):1155 – 1162.

Sacks, J., Welch, W. J., Mitchell, T. J. et Wynn, H. P. (1989). Design and Analysis of Computer Experiments. *Statistical Science*, 4(4):409–423.

Saltelli, A. (2002). Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280–297.

Saltelli, A., Chan, K. et Scott, E. (2000). *Sensitivity analysis*. Wiley series in probability and statistics. Wiley.

Saltelli, A., Tarantola, S. et Chan, K. P.-S. (1999). A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics*, 41(1):39–56.

Santner, T. J., Williams, B. et Notz, W. (2003). *The Design and Analysis of Computer Experiments*. Springer-Verlag.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464.

Scott, D. W. (1992). *Multivariate density estimation : theory, practice, and visualization*, volume 383. John Wiley & Sons.

Shannon, C. (1948). A Mathematical Theory of Communication. *Bell Sys. Tech. J.*, 27:379–423,623–656.

Sheather, S. J. et Jones, M. C. (1991). A reliable data-based bandwidth selection method for kernel density estimation. *Journal of the Royal Statistical Society. Series B (Methodological)*, 53(3):683–690.

Shi, J. Q., Wang, B., Murray-Smith, R. et Titterington, D. M. (2007). Gaussian process functional regression modeling for batch data. *Biometrics*, 63(3):714–723.

Sobol', I. M. (1993). Sensitivity estimates for non linear mathematical models. *Mathematical Modelling and Computational Experiments*, 4(1).

Sobol, I. M., Tarantola, S., Gatelli, D., Kucherenko, S. et Mauntz, W. (2007). Estimating the approximation error when fixing unessential factors in global sensitivity analysis. *Reliability Engineering & System Safety*, 92(7):957–960.

Stein, M. (1987). Large sample properties of simulations using Latin hypercube sampling. *Technometrics*, 29(2):143–151.

Sun, Y. et Genton, M. G. (2011). Functional boxplots. *Journal of Computational and Graphical Statistics*, 20(2):316–334.

Tarantola, S., Gatelli, D. et Mara, T. (2006a). Random balance designs for the estimation of first order global sensitivity indices. *Reliability Engineering & System Safety*, 91(6):717 – 727.

Tarantola, S., Nardo, M., Saisana, M. et Gatelli, D. (2006b). A new estimator for sensitivity analysis of model output : An application to the e-business readiness composite indicator. *Reliability Engineering & System Safety*, 91(10-11):1135 – 1141.

Terrell, G. R. et Scott, D. W. (1992). Variable kernel density estimation. *The Annals of Statistics*, 20(3):1236–1265.

Tissot, J.-Y. et Prieur, C. (2012a). Bias correction for the estimation of sensitivity indices based on random balance designs. *Reliability Engineering & System Safety*, 107:205–213.

Tissot, J.-Y. et Prieur, C. (2012b). Variance-based sensitivity analysis using harmonic analysis.

Tissot, J.-Y. et Prieur, C. (2015). A randomized orthogonal array-based procedure for the estimation of first- and second-order Sobol' indices. *Journal of Statistical Computation and Simulation*, 85(7):1358–1381.

Tukey, J. (1975). Mathematics and the picturing of data. *In Proceedings of the International Congress of Mathematicians*, volume 2, pages 523–521.

Tukey, J. W. (1977). *Exploratory data analysis*. Addison-Wesley Series in Behavioral Science.

Van Deun, K., Smilde, A., van der Werf, M., Kiers, H. et Van Mechelen, I. (2009). A structured overview of simultaneous component based data integration. *BMC Bioinformatics*, 10:246–261.

Vining, G. G. et Myers, R. H. (1990). Combining Taguchi and response surface philosophies : a dual response approach. *Journal of quality technology*, 22(1).

Volkova, E., Iooss, B. et Van Dorpe, F. (2008). Global sensitivity analysis for a numerical model of radionuclide migration from the RRC "Kurchatov Institute" radwaste disposal site. *Stochastic Environmental Research and Risk Assessment*, 22(1):17–31.

Wand, M. et Jones, M. (1995). Kernel smoothing. *Chapman&Hall, London*.

Wang, H. (2014). Coordinate descent algorithm for covariance graphical lasso. *Statistics and Computing*, 24(4):521–529.

Watson, G. S. (1964). Smooth regression analysis. *Sankhyā : The Indian Journal of Statistics, Series A*, 26(4):359–372.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J. et Morris, M. D. (1992). Screening, predicting, and computer experiments. *Technometrics*, 34(1):15–25.

Weyl, H. (1916). Über die gleichverteilung von zahlen mod. eins. *Mathematische Annalen*, 77(3):313–352.

Wiener, N. (1938). The homogeneous chaos. *American Journal of Mathematics*, 60(4):897–936.

Wold, H. (1966). *Estimation of Principal Components and Related Models by Iterative Least squares*, pages 391–420. Academic Press.

Wu, C. F. J. (1983). On the convergence properties of the EM algorithm. *The Annals of Statistics*, 11(1):95–103.

Xiu, D. (2009). Fast numerical methods for stochastic computations : a review. *Communications in computational physics*, 5(2-4):242–272.

Xiu, D. et Hesthaven, J. S. (2005). High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118–1139.

Xiu, D. et Karniadakis, G. E. (2002). The Wiener–Askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24(2):619–644.

Xu, C. et Gertner, G. Z. (2007). Extending a global sensitivity analysis technique to models with correlated parameters. *Computational Statistics & Data Analysis*, 51(12):5579 – 5590.

Xu, C. et Gertner, G. Z. (2008). Uncertainty and sensitivity analysis for models with correlated parameters. *Reliability Engineering & System Safety*, 93(10):1563 – 1573.

Zabalza, I., Dejean, J.-P. et Collombier, D. (1998). Prediction and density estimation of a horizontal well productivity index using generalized linear models. *In 6th European Conference on the Mathematics of Oil Recovery*.

Glossaire

ACP Analyse en Composantes Principales 28
ACPC Analyse en Composantes Principales Contrainte 127
ACPF Analyse en Composantes Principales Fonctionelle 29
ACPS Analyse en Composantes Principales Simultanée 29
ASG Analyse de Sensibilité Globale 81

BIC Bayesian Information Criterion 55

CS Critère de Sécurité 9
CSV CATHARE2-SSTH-VESTALE 10

EM Expectation-Maximization 51

HDR High Density Region 71

LHS Latin Hypercube Sampling 98
LiPoSo Liaison Pompe-Sommier 8

MMP Modified Magic Points 128
MQA Minimisation Quadratique Alternée 129

PLS Partial Least Squares 31
PTS Pressurized Thermal Shock (choc thermique pressurisé) 8

RelMSE Relative Mean Squared Error (erreur quadratique moyenne relative) 34

sEM sparse Expectation Maximization algorithm (algorithme pour l'estimation d'inverses de matrices de covariance creuses) 56
sEM2 Algorithme Expectation-Maximization pour l'estimation de matrices de covariance creuses 57
SPLS Simultaneous Partial Least Squares (décomposition Partial Least Squares simultanée) . . . 33

T-H Thermo-Hydraulique 9
T-M Thermo-Mécanique 9

Annexe A

Résolution du problème d'optimisation pénalisée pour l'estimation de matrices de covariance creuses

Dans cette annexe, on détaille deux méthodes permettant de résoudre le problème d'optimisation pénalisé de l'équation (2.25) utilisé dans le chapitre 2 pour estimer des matrices de covariance creuses.

A.1 Algorithme de majorization-minimization (Bien et Tibshirani, 2011)

On introduit tout d'abord la définition d'une fonction majorée. Une fonction f est majorée par $g(.|x_0)$ si $\forall x, f(x) \leqslant g(x|x_0)$ et $f(x_0) = g(x_0|x_0)$. L'algorithme de majorization-minimization minimise à chaque itération une fonction majorant f. Il est initialisé en un point x_0 et à chaque étape, on définit le point suivant $x_n = \text{argmin}_x g(x|x_{n-1})$, où $g(.|x_{n-1})$ majore f. On peut montrer qu'à chaque itération n, $f(x_n) \leqslant f(x_{n-1})$. Les fonctions $g(.|x_n)$ majorant f doivent être plus faciles à minimiser que f pour que l'algorithme soit efficace.

Comme la fonction étudiée ici est la somme d'une fonction convexe et d'une fonction concave, on peut la majorer en remplaçant la fonction concave par sa tangente. On majore la fonction concave $\log\det\Sigma$ par $\log\det\Sigma_0 + \text{tr}(\Sigma_0^{-1}(\Sigma - \Sigma_0))$ et donc f est majorée par

$$g(\Sigma|\Sigma_0) = \log\det\Sigma_0 + \text{tr}(\Sigma_0^{-1}\Sigma) - p + \text{tr}(\Sigma^{-1}S) + \lambda \left\|P * \Sigma\right\|_1 .$$

Le problème de minimisation non convexe initial a donc été remplacé par la minimisation successive de problèmes convexes :

$$\hat{\Sigma}_n = \text{argmin}_\Sigma \left[\text{tr}\left(\Sigma_{n-1}^{-1}\Sigma\right) - p + \text{tr}(\Sigma^{-1}S) + \lambda \left\|P * \Sigma\right\|_1\right] . \tag{A.1}$$

Puisque l'équation (A.1) est convexe, on peut la résoudre avec un algorithme utilisant le gradient de la fonction. Bien et Tibshirani utilisent l'algorithme de descente de gradient généralisée, développé par Beck et Teboulle (2009). Celui-ci est une extension de l'algorithme de descente pour des fonctions non différentiables. Son taux de convergence est assez rapide puisqu'il faut $O(\varepsilon^{-1/2})$ itérations pour atteindre une valeur éloignée de l'optimum de ε. Le détail de cet algorithme d'optimisation n'est pas donné ici. La méthode de résolution de Bien et Tibshirani peut être résumée dans l'algorithme suivant :

Algorithme A.1 *Majorization-minimization* (Bien et Tibshirani, 2011)

1. Initialisation de la matrice Σ
2. Répéter jusqu'à convergence :
   (a) $\Sigma_0 \leftarrow \Sigma$
   (b) $\Sigma \leftarrow \text{argmin}_\Sigma \left[\text{tr}\left(\Sigma_0^{-1}\Sigma\right) - p + \text{tr}(\Sigma^{-1}S) + \lambda \left\|P * \Sigma\right\|_1\right]$

A.2 Résolution par l'algorithme de descente par coordonnée (Wang, 2014)

Wang (2014) a proposé un nouvel algorithme pour résoudre le problème d'optimisation (2.25). Celui-ci semble plus rapide et plus robuste que l'algorithme de Bien et Tibshirani d'après les tests présentés dans Wang (2014). L'algorithme proposé met à jour, dans la matrice recherchée Σ, une colonne et une ligne à chaque itération, tous les autres éléments de la matrice étant fixés. Sans perte de généralité, on ne présente que la mise à jour de la dernière ligne et de la dernière colonne de Σ. On réécrit les matrices S et Σ ainsi

$$\Sigma = \begin{pmatrix} \Sigma_{11} & \sigma_{12} \\ \sigma_{12}^T & \sigma_{22} \end{pmatrix} \text{ et } S = \begin{pmatrix} S_{11} & s_{12} \\ s_{12}^T & s_{22} \end{pmatrix}. \tag{A.2}$$

Σ_{11} et S_{11} sont des matrices de taille $(p-1)\times(p-1)$, σ_{12} et s_{12} sont des vecteurs de taille $p-1$, et σ_{22} et s_{22} sont respectivement la variance théorique et la variance empirique de la p-ième variable. On pose $\beta = \sigma_{12}$ et $\delta = \sigma_{22} - \sigma_{12}^T\Sigma_{11}^{-1}\sigma_{12}$. On écrit l'inverse de Σ en fonction de Σ_{11}, β et δ :

$$\Sigma^{-1} = \begin{pmatrix} \Sigma_{11}^{-1} + \Sigma_{11}^{-1}\beta\beta^T\Sigma_{11}^{-1}\delta^{-1} & -\Sigma_{11}^{-1}\beta\delta^{-1} \\ -\beta^T\Sigma_{11}^{-1}\delta^{-1} & \delta^{-1} \end{pmatrix}.$$

On peut ainsi réécrire les trois termes de la fonction à optimiser de l'équation (2.25) en fonction de Σ_{11}, β et δ :

$$\begin{aligned} \log\det\Sigma &= \log\delta + c_1, \\ \text{tr}(S\Sigma^{-1}) &= \beta^T\Sigma_{11}^{-1}S_{11}\Sigma_{11}^{-1}\beta\delta^{-1} - 2s_{12}^T\Sigma_{11}^{-1}\beta\delta^{-1} + s_{22}\delta^{-1} + c_2, \\ \rho\left\|\Sigma\right\|_1 &= 2\rho\left\|\beta\right\|_1 + \rho\left(\beta^T\Sigma_{11}^{-1}\beta + \delta\right) + c_3, \end{aligned}$$

où c_1, c_2 et c_3 sont des constantes qui ne font intervenir ni β ni δ. On peut donc réécrire le problème (2.25) selon β et δ de la manière suivante :

$$\text{argmin}_{\beta,\delta}\left[\log\delta + \beta^T\Sigma_{11}^{-1}S_{11}\Sigma_{11}^{-1}\beta\delta^{-1} - 2s_{12}^T\Sigma_{11}^{-1}\beta\delta^{-1} + s_{22}\delta^{-1} + 2\rho\left\|\beta\right\|_1 + \rho\beta^T\Sigma_{11}^{-1}\beta + \rho\delta\right].$$

Si l'on minimise selon δ seul, on obtient

$$\text{argmin}_\delta\left[\log\delta + a\delta^{-1} + \rho\delta\right],$$

où $a = \beta^T\Sigma_{11}^{-1}S_{11}\Sigma_{11}^{-1}\beta - 2s_{12}^T\Sigma_{11}^{-1}\beta + s_{22}$. On trouve donc

$$\hat{\delta} = \begin{cases} a & \text{si } \rho = 0 \\ (-1+\sqrt{1+4a\rho})/(2\rho) & \text{si } \rho \neq 0 \end{cases}. \tag{A.3}$$

De même, avec β, on obtient l'équation suivante

$$\text{argmin}_\beta\left[\beta^TV\beta - 2u^T\beta + 2\rho\left\|\beta\right\|_1\right], \tag{A.4}$$

où $V = \Sigma_{11}^{-1}S_{11}\Sigma_{11}^{-1}\delta^{-1} + \rho\Sigma_{11}^{-1}$ et $u = \Sigma_{11}^{-1}s_{12}\delta^{-1}$. On résout (A.4) en mettant à jour successivement les coordonnées de β jusqu'à convergence avec l'expression suivante :

$$\hat{\beta}_j = \mathcal{S}\left(u_j - \sum_{k\neq j}V_{kj}\hat{\beta}_k, \rho\right)/V_{jj}, \tag{A.5}$$

où $\mathcal{S}(x,t) = \text{sign}(x)(|x| - t)_+$. Cette résolution est appelée descente par coordonnées parce qu'à chaque itération, la fonction est minimisée sur une seule coordonnée. Les colonnes de Σ sont ensuite mises à jour par $\sigma_{12} = \beta$ et $\sigma_{22} = \delta + \beta^T \Sigma_1^{-1} \beta$. L'algorithme est résumé ci-dessous.

Algorithme A.2 Descente par coordonnées (Wang, 2014)

---

1. $\Sigma^{(0)} = S$ ou $\text{diag}(S_{11}, \ldots, S_{nn})$
2. Répéter jusqu'à convergence
   (a) $\Sigma^{(k+1)} = \Sigma^{(k)}$
   (b) Pour $i = 1, \ldots, p$
      i. Écrire $\Sigma^{(k+1)}$ et S comme dans (A.2)
      ii. Calculer δ avec (A.3)
      iii. Résoudre le problème (A.4) en répétant (A.5) sur les coordonnées j jusqu'à convergence
      iv. Calculer $\sigma_{12}^{(k+1)} = \beta$, $\sigma_{21}^{(k+1)} = \beta^T$ et $\sigma_{22}^{(k+1)} = \delta + \beta^T \Sigma_{11}^{-1} \beta$
   (c) $k = k + 1$

Annexe B

Test d'adéquation à noyau

On présente, dans cette annexe, le test d'adéquation multivarié développé par Fromont et al. (2012) et utilisé comme critère pour valider la quantification des incertitudes proposée dans le chapitre 2. Soit $Z^1 = (Z_1^1, \ldots, Z_{n_1}^1)$ et $Z^2 = (Z_1^2, \ldots, Z_{n_2}^2)$ deux échantillons de variables aléatoires indépendantes et identiquement distribuées sur un espace mesurable $\mathcal{Z}$ et de densités respectives s_1 et s_2 par rapport à la mesure σ-finie ν de $\mathcal{Z}$. On fait l'hypothèse que $s_1, s_2 \in L^2(\mathcal{Z}, \nu)$. On note l'ensemble $Z = Z^1 \cup Z^2$ de taille $n = n_1 + n_2$. On définit enfin pour h mesurable, la norme $\|h\|_k = \left(\int_{\mathcal{Z}} |h|^k d\nu\right)^{1/k}$, et on note $\langle ., . \rangle$ le produit scalaire associé.

L'objectif est de tester, à partir des échantillons Z^1 et Z^2 , l'hypothèse H_0 : « $s_1 = s_2$ » contre l'hypothèse H_1 : « $s_1 \neq s_2$ ». Pour cela, on introduit un noyau symétrique $K : \mathcal{Z} \times \mathcal{Z} \mapsto \mathbb{R}$, satisfaisant

$$\int_{\mathcal{Z}^2} K^2(z, z')(s_1 + s_2)(z)(s_1 + s_2)(z') d\nu_z d\nu_{z'} < +\infty. \tag{B.1}$$

Fromont et al. (2012) donnent trois exemples de noyaux vérifiant cette propriété et qui peuvent être utilisés pour le test :

– Soit une famille finie $\{\phi_\lambda, \lambda \in \Lambda\}$ orthonormale pour $\langle ., . \rangle$. Le premier exemple de noyau s'écrit

$$K(z, z') = \sum_{\lambda \in \Lambda} \phi_\lambda(z)\phi_\lambda(z').$$

– Si $\mathcal{Z} = \mathbb{R}^d$ et si ν est la mesure de Lebesgue, on peut définir K pour $z = (z_1, \ldots, z_d)$ et $z' = (z'_1, \ldots, z'_d)$:

$$K(z, z') = \frac{1}{\prod_{i=1}^d h_i} k\left(\frac{z_1 - z'_1}{h_1}, \ldots, \frac{z_d - z'_d}{h_d}\right),$$

où $h = (h_1, \ldots, h_d)$ est un vecteur de largeurs de bande positives et $k \in L^2(\mathbb{R}^d)$.

– Le troisième exemple donné par Fromont et al. (2012) est la famille de noyaux de Mercer (ou semi-définis positifs). Un tel noyau s'écrit

$$K(z, z') = \langle \theta(z), \theta(z') \rangle_{\mathcal{H}_K},$$

où $\mathcal{H}_K$ un espace de Hilbert à noyau reproduisant associé à K, θ est une fonction de $\mathcal{Z}$ dans $\mathcal{H}_K$ et $\langle ., . \rangle_{\mathcal{H}_K}$ est le produit scalaire de $\mathcal{H}_K$.

Pour simplifier les calculs, on définit les trois variables suivantes

$$\begin{aligned} a_{n_1,n_2} &= \left(\frac{1}{n_1(n_1-1)} - c_{n_1,n_2}\right)^{1/2}, \\ b_{n_1,n_2} &= -a_{n_2,n_1} = -\left(\frac{1}{n_2(n_2-1)} - c_{n_1,n_2}\right)^{1/2}, \\ c_{n_1,n_2} &= \frac{1}{n_1 n_2(n_1+n_2-2)}. \end{aligned}$$

La statistique de test T_K est alors définie par

$$T_K = \sum_{i,j\in\{1,\ldots,n\},i\neq j} K(Z_i,Z_j)(\varepsilon_i^0\varepsilon_j^0 + c_{n_1,n_2}),$$

où $\varepsilon_i^0 = \begin{cases} a_{n_1,n_2} & \text{si } Z_i \in Z^1 \\ b_{n_1,n_2} & \text{si } Z_i \in Z^2 \end{cases}$. L'hypothèse H_0 est rejetée si T_K est supérieur à une valeur critique. Comme la distribution de T_K n'est pas connue, on définit la variable suivante pour choisir cette valeur critique :

$$T_K^\varepsilon = \sum_{i,j\in\{1,\ldots,n\},i\neq j} K(Z_i,Z_j)(\varepsilon_i\varepsilon_j + c_{n_1,n_2}),$$

où $\varepsilon_i = \begin{cases} a_{n_1,n_2} & \text{si } i \in \{R_1,\ldots,R_{n_1}\} \\ b_{n_1,n_2} & \text{sinon} \end{cases}$, avec $R = (R_1,\ldots,R_n)$ un vecteur aléatoire uniformément distribué sur les permutations de $\{1,\ldots,n\}$. On définit $q_{K,1-\alpha}^Z$ le quantile $1-\alpha$ de T_K^ε conditionnellement à Z et on obtient la règle de décision suivante : H_0 est rejeté si $T_k > q_{K,1-\alpha}^Z$. Sous l'hypothèse H_0, T_K et T_K^ε ont la même distribution conditionnellement à Z. Par conséquent, le test est exactement de niveau α.

Annexe C

Complément à l'étude des cas PTS et de la rutpure LiPoSo

On complète dans cette section l'étude de la quantification des incertitudes des variables fonctionnelles du cas du choc thermique pressurisé (PTS) et de la rupture LiPoSo, faites respectivement dans les sections 2.2.7.3 et 2.2.7.2. On présente ici une comparaison des erreurs quadratiques moyennes relatives (RelMSE) des décompositions réalisées séparément pour chaque variable fonctionnelle (ACP ou PLS) à celles obtenues avec les décompositions simultanées de toutes les variables (ACPS ou SPLS). L'objectif est de comparer les erreurs obtenues à nombre de composantes total égal.

C.1 Cas PTS

La Figure C.1 représente en fonction du nombre de composantes de l'ACPS (respectivement SPLS) les nombres de composantes (d_1, d_2, d_3) à garder dans les ACP (respectivement PLS) des trois transitoires pour obtenir une erreur d'approximation équivalente. Plus précisément, les courbes vertes, rouges et noires représentent le nombre maximal de composantes des décompositions de la température, de la pression et du coefficient d'échange tel que les erreurs quadratiques de ces décompositions sont supérieures ou égales à celle de la décomposition simultanée. La courbe bleue est la somme des trois autres courbes ($d_1+d_2+d_3$). Au dessus de 5 composantes pour l'ACP et 2 composantes pour la PLS, l'ACPS et la SPLS approchent au moins aussi bien les transitoires que les ACP ou la PLS sur chaque transitoire. En effet, pour un nombre de composantes fixé dans la décomposition simultanée, la somme des courbes vertes, rouge et noir pour les trois transitoires est supérieure à ce nombre de composantes. Ainsi, pour une approximation de même qualité, les décompositions réalisées indépendamment sur les trois transitoires nécessitent des tailles de base plus importantes. **Les décompositions simultanées donnent une meilleure approximation des fonctions de l'échantillon si la taille de la base choisie est supérieure ou égale à 2 pour la SPLS et 5 pour l'ACPS.**

C.2 Cas de la rutpure LiPoSo

La Figure C.2 représente en fonction du nombre de composantes de l'ACPS le nombre de composantes à garder dans les ACP des trois variables fonctionnelles pour obtenir une erreur d'approximation équivalente. Au dessus de 6 composantes, l'ACPS approche au moins aussi bien les variables fonctionnelles que les ACP sur chaque variable. En effet, pour un nombre d de composantes fixé dans la décomposition simultanée, la somme des courbes vertes, rouge et noir pour les trois transitoires est supérieure à d. Ainsi, la somme des tailles des bases des décompositions ACP de chaque variable sont plus élevées pour approximation de même qualité. **La décomposition ACP simultanée donne une meilleure**

approximation des fonctions de l'échantillon si la taille de la base est supérieure ou égale à 6.

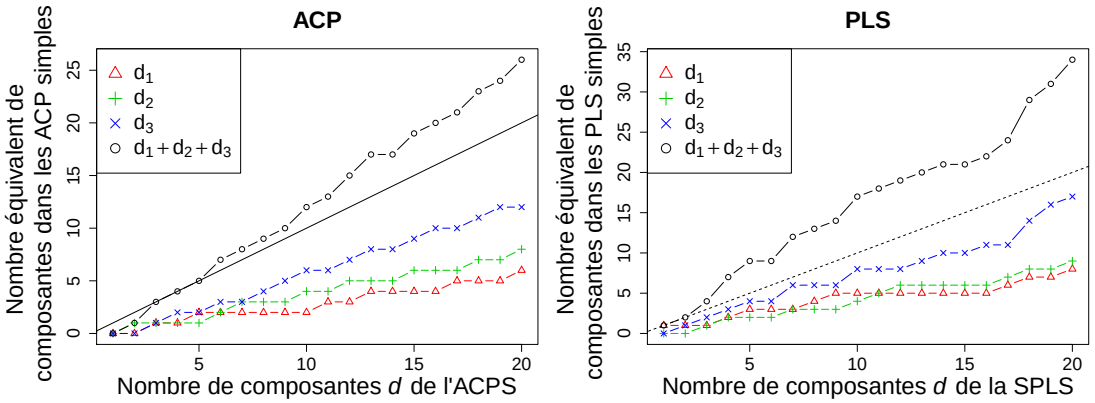

FIGURE C.1 – Cas PTS : nombre maximal de composantes des décompositions sur chaque transitoire tel que les RelMSE des décompositions simples soient supérieures ou égales à celle de la décomposition simultanée pour l'ACP (à gauche) et la PLS (à droite).

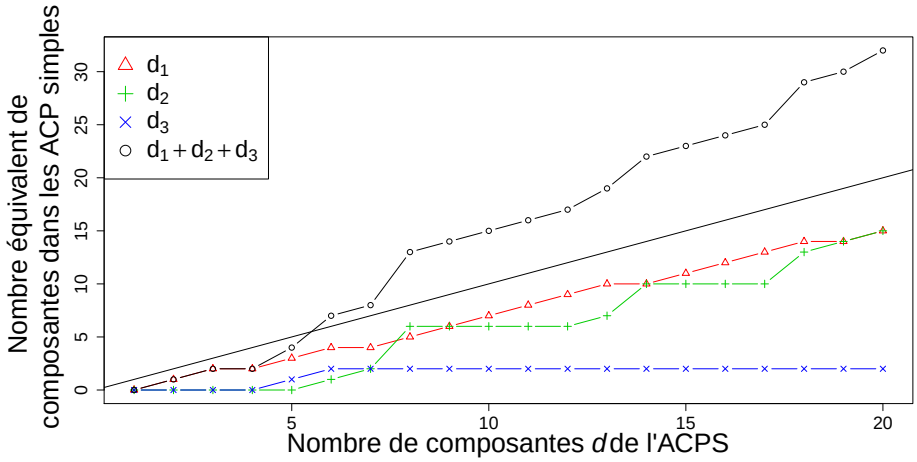

FIGURE C.2 – Cas LiPoSo : nombre maximal de composantes des décompositions ACP sur chaque transitoire tel que leurs RelMSE soient supérieures ou égales à celle de la décomposition ACPS.

Résumé

Cette thèse s'inscrit dans le cadre du traitement des incertitudes dans les simulateurs numériques, et porte plus particulièrement sur l'étude de deux cas d'application liés aux études de sûreté pour les réacteurs nucléaires. Ces deux applications présentent plusieurs caractéristiques communes. La première est que les entrées du code étudié sont fonctionnelles et scalaires, les entrées fonctionnelles étant dépendantes entre elles. La deuxième caractéristique est que la distribution de probabilité des entrées fonctionnelles n'est connue qu'à travers un échantillon de ces variables. La troisième caractéristique, présente uniquement dans un des deux cas d'étude, est le coût de calcul élevé du code étudié qui limite le nombre de simulations possibles. L'objectif principal de ces travaux de thèse était de proposer une méthodologie complète de traitement des incertitudes de simulateurs numériques pour les deux cas étudiés. Dans un premier temps, nous avons proposé une méthodologie pour quantifier les incertitudes de variables aléatoires fonctionnelles dépendantes à partir d'un échantillon de leurs réalisations. Cette méthodologie permet à la fois de modéliser la dépendance entre les variables fonctionnelles et de prendre en compte le lien entre ces variables et une autre variable, appelée covariable, qui peut être, par exemple, la sortie du code étudié. Associée à cette méthodologie, nous avons développé une adaptation d'un outil de visualisation de données fonctionnelles, permettant de visualiser simultanément les incertitudes et les caractéristiques de plusieurs variables fonctionnelles dépendantes. Dans un second temps, une méthodologie pour réaliser l'analyse de sensibilité globale des simulateurs des deux cas d'étude a été proposée. Dans le cas d'un code coûteux en temps de calcul, l'application directe des méthodes d'analyse de sensibilité globale quantitative est impossible. Pour pallier ce problème, la solution retenue consiste à construire un modèle de substitution ou métamodèle, approchant le code de calcul et ayant un temps de calcul très court. Une méthode d'échantillonnage uniforme optimisé pour des variables scalaires et fonctionnelles a été développée pour construire la base d'apprentissage du métamodèle. Enfin, une nouvelle approche d'approximation de codes coûteux et à entrées fonctionnelles a été explorée. Dans cette approche, le code est vu comme un code stochastique dont l'aléa est dû aux variables fonctionnelles supposées incontrôlables. Sous ces hypothèses, plusieurs métamodèles ont été développés et comparés. L'ensemble des méthodes proposées dans ces travaux a été appliqué aux deux cas d'application étudiés.

Mots-clés : quantification des incertitudes, variables fonctionnelles, métamodèle, analyse de sensibilité.

Abstract

This work relates to the framework of uncertainty quantification for numerical simulators, and more precisely studies two industrial applications linked to the safety studies of nuclear plants. These two applications have several common features. The first one is that the computer code inputs are functional and scalar variables, functional ones being dependent. The second feature is that the probability distribution of functional variables is known only through a sample of their realizations. The third feature, relative to only one of the two applications, is the high computational cost of the code, which limits the number of possible simulations. The main objective of this work was to propose a complete methodology for the uncertainty analysis of numerical simulators for the two considered cases. First, we have proposed a methodology to quantify the uncertainties of dependent functional random variables from a sample of their realizations. This methodology enables to both model the dependency between variables and their link to another variable, called covariate, which could be, for instance, the output of the considered code. Then, we have developed an adaptation of a visualization tool for functional data, which enables to simultaneously visualize the uncertainties and features of dependent functional variables. Second, a method to perform the global sensitivity analysis of the codes used in the two studied cases has been proposed. In the case of a computationally demanding code, the direct use of quantitative global sensitivity analysis methods is intractable. To overcome this issue, the retained solution consists in building a surrogate model or metamodel, a fast-running model approximating the computationally expensive code. An optimized uniform sampling strategy for scalar and functional variables has been developed to build a learning basis for the metamodel. Finally, a new approximation approach for expensive codes with functional outputs has been explored. In this approach, the code is seen as a stochastic code, whose randomness is due to the functional variables, assumed uncontrollable. In this framework, several metamodels have been developed and compared. All the methods proposed in this work have been applied to the two nuclear safety applications.

Keywords: uncertainty quantification, functional variables, metamodel, sensitivity analysis.