



Adaptive approximation of high-dimensional functions with tree tensor networks for Uncertainty Quantification

Cécile Haberstich

► To cite this version:

Cécile Haberstich. Adaptive approximation of high-dimensional functions with tree tensor networks for Uncertainty Quantification. Data Structures and Algorithms [cs.DS]. École centrale de Nantes, 2020. English. NNT : 2020ECDN0045 . tel-03185160

HAL Id: tel-03185160

<https://theses.hal.science/tel-03185160>

Submitted on 30 Mar 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE DE DOCTORAT DE

L'ÉCOLE CENTRALE DE NANTES

ÉCOLE DOCTORALE NO 601

*Mathématiques et Sciences et Technologies
de l'Information et de la Communication*

Spécialité : Mathématiques et leurs interactions

Par

Cécile HABERSTICH

**Adaptive approximation of high-dimensional functions with tree
tensor networks for Uncertainty Quantification.**

Thèse présentée et soutenue à Centrale Nantes, le 15/12/2020

Unité de Recherche : UMR 6629, Laboratoire de Mathématiques Jean Leray (LMJL)

Rapporteurs avant soutenance :

Lars Grasedyck Professeur, RWTH Aachen University (Allemagne)
Fabio Nobile Professeur, École Polytechnique Fédérale Lausanne (Suisse)

Composition du Jury :

Président :	Albert Cohen	Professeur des universités, Sorbonne Université, Paris
Examineurs :	Virginie Ehrlacher	Maître de Conférences HDR, École des Ponts Paristech, Marne la Vallée
Dir. de thèse :	Anthony Nouy	Professeur des universités, École Centrale de Nantes
Co-dir. de thèse :	Guillaume Perrin	Ingénieur de Recherche HDR, CEA Arpajon

ACKNOWLEDGEMENT

Je souhaite tout d'abord remercier les membres de mon jury. Merci à Fabio Nobile et Lars Grasedyck pour leur relecture approfondie du manuscrit ainsi que leurs questions lors de la soutenance qui témoignent de leur intérêt pour ces travaux de thèse. *First of all, I would like to thank the jury's members. Fabio Nobile and Lars Grasedyck, thank you, for your thorough reviewing of the manuscript as well as your questions during the defense, which show your interest in this work.* Merci à Albert Cohen pour avoir présidé ce jury et merci à Virginie Ehrlacher pour sa présence enthousiaste et son intérêt.

Anthony et Guillaume merci infiniment. Merci à tous les deux pour votre engagement et votre enthousiasme pour cette thèse, sans oublier votre bienveillance durant ces trois années.

TABLE OF CONTENTS

Introduction	8
1 Projection methods	29
1.1 Introduction	29
1.2 Interpolation	29
1.2.1 Interpolation in the one-dimensional case	31
1.2.2 Interpolation with any arbitrary approximation space	33
1.2.3 Interpolation with tensor product bases	34
1.3 Least-squares method	35
1.3.1 Weighted least-squares projection	36
1.3.2 Random sampling	38
1.3.3 Optimal sampling measure	40
2 Boosted Optimal Weighted Least-Squares	43
2.1 Introduction	43
2.2 Boosted optimal weighted least-squares method	44
2.2.1 Resampling and conditioning	45
2.2.2 Subsampling	49
2.3 The noisy case	51
2.4 Numerical experiments	54
2.4.1 Notations and objectives	54
2.4.2 Qualitative analysis of the boosted optimal weighted least-squares method . .	56
2.4.3 Quantitative analysis for polynomial approximations	59
2.4.4 A noisy example	68
2.4.5 Overall conclusion for all examples	69
2.5 Conclusion	70
3 Adaptive Boosted Optimal Weighted Least-Squares	75
3.1 Introduction	75
3.2 Notations	76
3.3 Optimal weighted least-squares with block-structured sampling	77
3.3.1 Approximation in a given space	77

TABLE OF CONTENTS

3.3.2	Adaptive approximation with a nested sequence of spaces	78
3.4	Boosted optimal weighted least-squares with block-structured sampling	79
3.4.1	Approximation in a given space	79
3.4.2	Adaptive approximation with a nested sequence of spaces	83
3.5	Numerical illustrations	85
3.5.1	Illustration of the stability of the adaptive boosted least-squares strategy . .	85
3.5.2	Illustration for polynomial approximation	89
3.6	Conclusion	92
4	Tree-based tensor formats	95
4.1	Introduction	95
4.2	Tensor spaces	95
4.3	Tensor ranks and tree-based tensor formats	96
4.3.1	Dimension partition tree	97
4.3.2	Tree tensor networks and their representation	98
4.4	Principal Component Analysis for multivariate functions	101
4.4.1	α -principal subspaces	101
4.4.2	Accuracy of the empirical α -principal subspaces	102
4.4.3	The case of functions with Sobolev regularity	103
4.5	Approximation power of tree tensor networks	105
4.5.1	Truncation in tree-based format	105
4.5.2	Approximation rates for Sobolev functions	106
5	Principal Component Analysis for Tree-based tensor formats	107
5.1	Approximation of α -principal subspaces	108
5.1.1	Choosing an oblique projection verifying a stability property	108
5.1.2	Choosing the boosted optimal weighted least-squares projection	110
5.2	Estimation of the α -principal subspaces	113
5.2.1	Accuracy of the empirical α -principal subspaces	113
5.2.2	Adaptive estimation of the α -principal component subspaces	116
5.3	Learning tree tensor networks using PCA	117
5.3.1	Description of the algorithm	117
5.3.2	Error analysis	118
5.3.3	Complexity analysis	121
5.3.4	Heuristics used in practice	122
5.4	Numerical examples	124
5.4.1	Notations and objectives	124
5.4.2	Adaptive determination of the approximation spaces in the leaves	125

5.4.3	Adaptive estimation of the α -principal components subspaces	126
5.5	Conclusions	130
6	Tree adaptation	137
6.1	Introduction	137
6.2	Estimation of α -ranks	139
6.2.1	Principle and algorithm	139
6.2.2	Numerical illustration	141
6.3	Leaves-to-root construction of the tree with local deterministic optimizations	143
6.3.1	Max-mean rank strategy	143
6.3.2	Ballani and Grasedyck's strategy	145
6.4	Leaves-to-root construction of the tree with local stochastic optimizations	146
6.4.1	Principle and algorithm	146
6.4.2	Illustration of the algorithm	147
6.4.3	Illustration of the choices of the parameters	147
6.5	Adaptation of the tree with global stochastic optimizations	150
6.5.1	Principle and algorithm	150
6.5.2	Illustration of the algorithm	150
6.5.3	Illustration of the choice of the parameters	151
6.6	Numerical examples	153
6.7	Conclusions	157
	Conclusion	159
A	Approximate fast greedy algorithm	172
A.1	Computational strategy for the approximate fast greedy algorithm	172
A.2	Complexity analysis	173
A.3	Illustration	174
B	Sampling of multivariate probability distribution in Tree-Based tensor formats	177
B.1	Probability distributions in tree-based format	177
B.1.1	Marginal distributions	178
B.1.2	Conditional distributions	179
B.2	Sampling from multivariate probability distributions in tree-based tensor formats . .	180

INTRODUCTION (EN FRANÇAIS)

Contexte

Grâce à des modèles de calcul, les chercheurs et les ingénieurs peuvent remplacer des expériences physiques particulièrement coûteuses et complexes à mettre en œuvre par des simulations numériques. La réponse d'un modèle dépendant de certains paramètres peut être représentée par une fonction $u(x_1, \dots, x_d)$ de plusieurs variables. Malgré l'amélioration continue des ressources de calcul, la plupart de ces simulations restent très coûteuses d'un point de vue informatique. En outre, la résolution des problèmes de quantification de l'incertitude (UQ), comprenant notamment la propagation des incertitudes, la calibration des modèles, les problèmes inverses ou l'analyse de sensibilité [87], nécessite un grand nombre d'évaluations des modèles. Une solution consiste alors à remplacer le modèle coûteux par un modèle approché, moins coûteux à évaluer, ce qui revient à remplacer la fonction qui représente la réponse du modèle par une approximation.

Le but de l'approximation est de remplacer une fonction u par une fonction plus simple (c'est-à-dire plus facile à évaluer) de telle sorte que la distance (mesurant la qualité de l'approximation) entre la fonction u et son approximation, notée u^* , soit aussi petite que possible. En général, l'approximation est recherchée dans un sous-espace de fonctions V_m décrit par un certain nombre de paramètres m . Une séquence de tels sous-espaces $(V_m)_{m \geq 1}$ est appelée outil d'approximation (ou classe de modèles), dont la complexité est mesurée par son nombre de paramètres m . L'erreur de meilleure approximation de u par des éléments de V_m est définie par la distance minimale entre u et tout élément de V_m . L'outil d'approximation doit être adapté à la classe de fonctions que nous voulons approximer, ce qui se traduit par une erreur de meilleure approximation qui converge vers zéro assez rapidement avec m .

Il existe différentes approches pour construire l'approximation d'une fonction, soit à partir d'informations directes sur la fonction (par exemple des évaluations ponctuelles ou des mesures linéaires...), soit à partir d'équations satisfaites par la fonction. Dans cette thèse, nous nous intéressons aux fonctions de type boîte noire (éventuellement bruitées), ce qui correspond à la première situation où l'information sur u est donnée par des échantillons. Par conséquent, pour construire l'approximation de la fonction u , nous utilisons les évaluations $y^i = u(x^i)$ pour un ensemble de points x^i ou $y^i = u(x^i) + e^i$ dans le cas de données bruitées. Cependant, dans le contexte d'un modèle coûteux, cette approximation devrait être construite en utilisant un nombre réduit

d'échantillons, idéalement proche du nombre de paramètres m . Les échantillons x^i peuvent être soit indépendants et identiquement distribués selon une mesure μ , soit sélectionnés de manière adaptative. Cette seconde approche est particulièrement pertinente lorsqu'aucune évaluation de la fonction n'a encore été faite, ce qui sera le cadre de cette thèse.

Des méthodes typiques pour construire une approximation à partir d'échantillons sont l'interpolation et les méthodes des moindres carrés [28]. Les méthodes d'interpolation classiques comprennent l'interpolation polynomiale et l'interpolation par spline (où les échantillons sont en général choisis par rapport à V_m , pour garantir de bonnes propriétés des opérateurs d'interpolation), ainsi que le krigeage (qui est assez efficace lorsque les échantillons sont donnés et peut être amélioré avec une sélection adaptative des échantillons) [82]. Pour un ensemble de m points, x^1, \dots, x^m l'interpolation u^* de u dans V_m est définie par

$$u(x^i) = u^*(x^i), \text{ pour tout } 1 \leq i \leq m.$$

L'interpolation est bien étudiée dans le cas unidimensionnel mais dans le cas multivarié ($d > 1$), trouver de bons points pour l'interpolation est un problème difficile, en particulier lorsque V_m est un espace d'approximation non classique. En pratique, [18, 19] proposent des stratégies pour construire de bonnes séquences de points pour l'interpolation parcimonieuse utilisant des polynômes ou splines tensorisés. Cependant, pour les espaces d'approximation non classiques, même si certains ensembles de points ont montré leur efficacité (voir [65, 15]), il n'y a pas de garantie pour l'erreur d'interpolation.

Pour n échantillons x^1, \dots, x^n indépendants et identiquement distribués selon une mesure μ , la méthode des moindres carrés classique définit l'approximation de u comme le minimiseur de

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n (v(x^i) - y^i)^2.$$

Un aspect intéressant des méthodes des moindres carrés est qu'elles sont capables, sous certaines conditions sur le nombre d'échantillons n , de garantir une approximation stable et une erreur proche de l'erreur de meilleure approximation mesurée avec la norme L_μ^2 . Toutefois, pour ce faire, elles peuvent exiger une taille d'échantillon n bien supérieure à m (voir [22]). Les méthodes des moindres carrés sont un cas particulier de minimisation de risque empirique en apprentissage supervisé [30]. D'autres fonctionnelles de risque peuvent être utilisées, mais elles nécessitent également de nombreuses évaluations afin d'assurer la stabilité.

Lorsque les échantillons x^i peuvent être sélectionnés selon une mesure que l'on choisit, l'erreur

d'estimation peut être améliorée en utilisant les moindres carrés pondérés [32, 73], qui définit l'approximation de u par

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n w^i (v(\tilde{x}^i) - y^i)^2,$$

où les poids sont adaptés à V_m et les points \tilde{x}^i sont des échantillons indépendants et identiquement distribués selon une mesure bien choisie. Cela peut permettre de réduire la taille de l'échantillon n pour atteindre la même erreur d'approximation, par rapport aux moindres carrés standards. Dans [24], les auteurs introduisent une mesure d'échantillonnage optimale dont la densité par rapport à la mesure de référence dépend de l'espace d'approximation V_m . Ils montrent que la projection des moindres carrés pondérés construite avec des échantillons de cette mesure est stable en espérance avec des valeurs de n plus faibles qu'avec les moindres carrés classiques. Une approche similaire est proposée dans [68], où la principale différence est que les échantillons aléatoires sont plus structurés tout en garantissant des résultats de stabilité similaires. Néanmoins, dans les deux cas, la condition nécessaire pour avoir une stabilité exige toujours un nombre élevé d'échantillons n , par rapport à une méthode d'interpolation. Les auteurs proposent également des méthodes de moindres carrés pondérés optimales dans le cas d'une approximation adaptative (lorsque V_m est choisi dans une séquence d'espaces d'approximation imbriqués), voir [67] et [4]. Dans cette thèse, nous proposons une nouvelle méthode de projection sur un espace d'approximation fixe V_m qui assure la stabilité de la projection des moindres carrés en espérance avec une taille d'échantillon proche de m . Nous proposons également une nouvelle méthode pour le cadre adaptatif qui réduit de manière significative le nombre d'échantillons et présente toujours des propriétés de stabilité.

Dans de nombreuses applications, les fonctions peuvent dépendre d'un nombre potentiellement élevé de variables $d \gg 1$. Lorsque la dimension d augmente, l'utilisation d'outils d'approximation standards adaptés aux fonctions régulières (par exemple les splines pour les fonctions de Sobolev) conduit à une complexité des méthodes d'approximation qui croît de manière exponentielle avec d . C'est ce qu'on appelle la malédiction de la dimension, expression introduite par [12]. Par conséquent, le nombre d'évaluations nécessaires pour approcher la fonction u avec des outils d'approximation naïfs peut exploser. Dans cette thèse, nous supposons que les fonctions présentent des structures de faible dimension, de sorte que nous pouvons nous attendre à une bonne approximation en utilisant un nombre limité d'évaluations de la fonction. L'exploitation de ces structures de la fonction nécessite généralement des outils d'approximation particuliers [29], qui peuvent dépendre de l'application. Certains outils d'approximation parmi les plus courants sont rappelés ci-après.

Une première approche consiste à utiliser la parcimonie de u en choisissant un ensemble particulier de fonctions $\{\varphi_j\}_{j \in \Lambda}$ parmi un ensemble de fonctions de base tensorisées, de sorte que

u puisse être écrit sous la forme $\sum_{j \in \Lambda} a_j \varphi_j(x)$. Des méthodes d'approximation parcimonieuses ont été envisagées pour résoudre les problèmes de quantification d'incertitude, voir par exemple [23, 6, 5, 18, 19] pour une approximation polynomiale parcimonieuse.

Des classes de modèles plus structurées pour traiter les problèmes de grande dimension sont les modèles additifs, introduits par [41], $\sum_{i=1}^d u_j(x_j)$ ou plus généralement $\sum_{\alpha \in T} u_\alpha(x_\alpha)$ où $T \subset 2^{\{1, \dots, d\}}$ est soit fixe (cela correspond à une approximation linéaire), soit un paramètre libre (cela correspond à une approximation non linéaire). Il existe également des modèles multiplicatifs $\prod_{j=1}^d u_j(x_j)$ ou plus généralement $\prod_{\alpha \in T} u_\alpha(x_\alpha)$, où T est également un sous-ensemble de $2^{\{1, \dots, d\}}$ et la même distinction entre approximation linéaire et non linéaire est faite, selon que T est fixe ou non. Ces types d'outils d'approximation sont standards en statistique et en modélisation probabiliste (modèles graphiques, réseaux bayésiens).

Des classes de modèles plus complexes basées sur la composition des fonctions ont également été proposées. Parmi elles, les modèles ridge peuvent être écrits sous la forme $v(Wx)$ avec $W \in \mathbb{R}^{k \times d}$, où v appartient à une classe de fonctions de k variables avec une paramétrisation de faible dimension. Un exemple typique est le perceptron $a\sigma(w^T x + b)$, où σ est une fonction d'activation donnée, introduit dans [83]. Ces modèles de composition comprennent également le modèle de projection poursuit $\sum_{j=1}^k v_j(w_j^T x)$, où pour $j = 1, \dots, k$, $w_j \in \mathbb{R}^d$, voir [41, 42]. Un cas particulier est celui des réseaux de neurones avec une couche cachée $\sum_{j=1}^d a_j \sigma(w_j^T x + b_j)$. On peut aussi considérer des compositions de fonctions plus générales, telles que $v_L \circ v_{L-1} \circ \dots \circ v_1$. Dans le cas des réseaux de neurones profonds, pour chaque $j = 1, \dots, L$, les fonctions v_j s'écrivent $v_j(t) = \sigma(A_j t + b_j)$, où σ est une fonction d'activation choisie, tandis que $b_j \in \mathbb{R}^k$ et $A_j \in \mathbb{R}^{k \times d}$ sont des paramètres à estimer [45]. Malgré son formalisme général et les nombreux succès qu'ils ont acquis en matière d'approximation, les réseaux de neurones profonds nécessitent généralement un nombre élevé d'évaluations pour apprendre les paramètres, ce qui n'est pas possible dans le contexte d'évaluations coûteuses. De plus, lorsque l'on considère les fonctions d'activation non linéaires générales, il est difficile de garantir une approximation stable dont l'erreur est proche de l'erreur de meilleure approximation, ce qui est un objectif central de cette thèse.

Dans ce travail, nous nous concentrons sur les modèles de fonctions de faible rang et en particulier sur l'ensemble des fonctions dans les formats de tenseurs basés sur des arbres ou les réseaux de tenseurs basés sur des arbres, qui peuvent être considérés comme une classe particulière de réseaux de neurones. Plus précisément, pour α un sous-ensemble de $D = \{1, \dots, d\}$ et $\alpha^c = D \setminus \alpha$ son complémentaire dans D , nous considérons les fonctions qui peuvent être écrites

$$v(x) = \sum_{i=1}^{r_\alpha} u_i^\alpha(x_\alpha) u_i^{\alpha^c}(x_{\alpha^c}), \text{ for all } \alpha \in T, \quad (1)$$

où T est un arbre de partition de dimension. Un arbre de partition de dimension est une collection particulière de sous-ensembles de D , quelques exemples d'arbres sont donnés Figure 1. L'entier minimal r_α tel que v peut être écrit sous la forme (1) est appelé le rang α de v , noté $\text{rank}_\alpha(v)$.

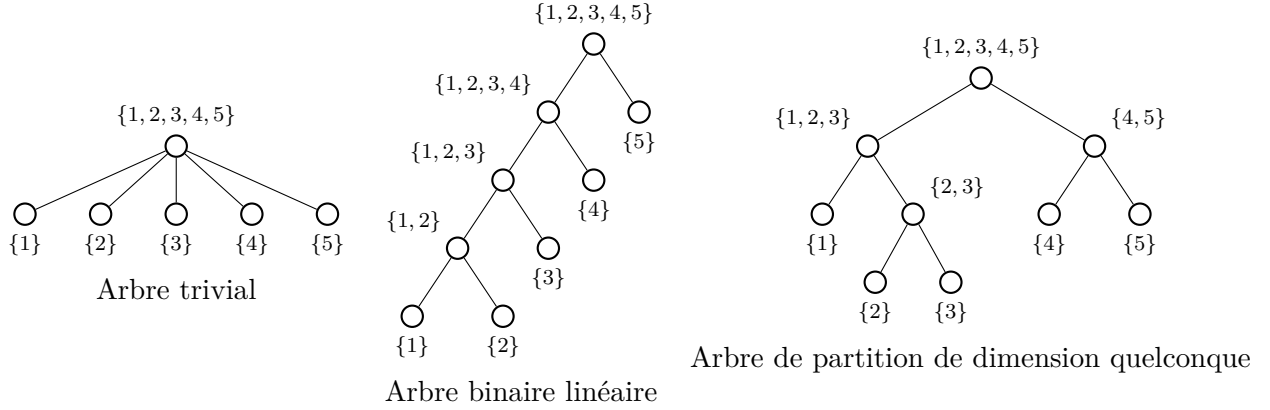


Figure 1 – Exemples d'arbres de partition de dimension

L'ensemble des fonctions dans un format de tenseurs basé sur des arbres est défini par $\mathcal{T}_r^T(\mathcal{H}) = \{v \in \mathcal{H} : \text{rank}_\alpha(v) \leq r_\alpha, \alpha \in T\}$, avec $r = (r_\alpha)_{\alpha \in T}$ un tuple d'entiers, et \mathcal{H} un espace d'approximation de fonctions multivariées. Il comprend le format de Tucker pour un arbre trivial, le format tensor-train (train de tenseurs) pour un arbre binaire linéaire [79] et le format hiérarchique plus général pour un arbre de partition de dimension quelconque [54]. Toute fonction dans $\mathcal{T}_r^T(\mathcal{H})$ admet une paramétrisation multilinéaire avec des paramètres formant un réseau de tenseurs de faible ordre, d'où le nom de réseaux de tenseurs basés sur des arbres, voir la Figure 2 pour une illustration pour $d = 5$. Considérer les arbres de dimension pour T donne de bonnes propriétés topologiques et géométriques à l'ensemble $\mathcal{T}_r^T(\mathcal{H})$ [35, 37]. Les réseaux de tenseurs basés sur des arbres sont particulièrement pertinents pour l'approximation en grande dimension, car la complexité de la paramétrisation d'une fonction est linéaire en la dimension d et polynomiale en les rangs [53]. Ils peuvent également être utilisés pour l'approximation de fonctions univariées [2, 3]. Toutefois, les rangs (et donc le nombre d'évaluations n) nécessaires pour calculer une approximation dans un format de tenseurs basé sur des arbres avec une tolérance donnée peuvent dépendre fortement de l'arbre choisi T . La classe de modèles $\mathcal{T}_r^T(\mathcal{H})$ peut également être interprétée comme une classe de fonctions qui sont des compositions de fonctions multilinéaires, la structure des compositions étant donnée par l'arbre. Elle a été identifiée avec une classe particulière de réseaux de neurones profonds (plus précisément les réseaux somme-produit ou les circuits arithmétiques) avec une connectivité parcimonieuse [26]. L'erreur d'approximation associée à cette classe de modèles a deux contributions : l'erreur de troncature (due aux rangs finis r) et l'erreur de discrétisation (due à

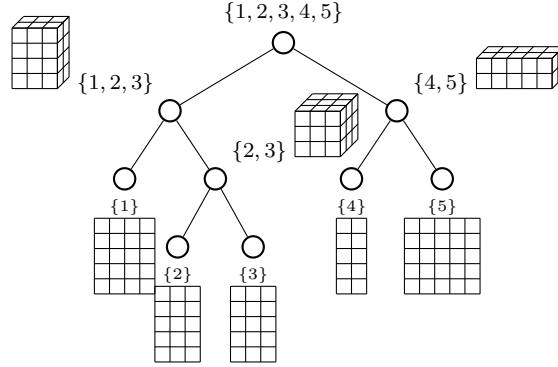


Figure 2 – Un réseau de tenseurs basé sur des arbres admet une paramétrisation multilinéaire avec des paramètres formant un réseau de tenseurs de faible ordre.

l'introduction de l'espace de dimension finie \mathcal{H}).

Les réseaux de tenseurs basés sur des arbres sont un outil d'approximation qui permettent d'obtenir de bonnes performances pour de nombreuses classes de fonctions. Dans [84] ou plus récemment dans [7, 2, 3], les auteurs montrent que pour les fonctions u avec une régularité Sobolev, ils atteignent (quel que soit l'arbre) un taux d'approximation optimal ou proche de l'optimal [61]. Ils prouvent également que pour les fonctions u avec une régularité Sobolev mixte, l'approximation dans un format de tenseur basé sur des arbres atteint presque la performance idéale obtenue par une approximation avec croix hyperbolique. Pour une revue de la littérature sur les méthodes d'approximation de faible rang, voir [47]. Nous renvoyons également le lecteur à la monographie [53] et à [8, 36, 59, 60, 77, 76].

Dans la littérature, des algorithmes construisant des approximations dans des formats de tenseurs basés sur des arbres en utilisant des évaluations ponctuelles de fonctions ont déjà été proposés. D'une part, il y a des approches d'apprentissage statistique qui utilisent des évaluations aléatoires et non structurées des fonctions [48, 55]. La robustesse et l'efficacité de ces algorithmes sont démontrées par des expériences numériques, mais ces algorithmes sont principalement basés sur des heuristiques et manquent d'une analyse approfondie. D'autre part, il y a les algorithmes qui utilisent des évaluations adaptatives et structurées des tenseurs, voir [64] (pour le format Tucker), ou [80] et [78] pour les formats de tenseurs basés sur des arbres. Parmi les approches adaptatives, la méthode de [78] présente un intérêt particulier. L'auteur propose une variante de la décomposition en valeurs singulières d'ordre supérieur (HOSVD) (des adaptations antérieures de la HOSVD ont été faites en [46] et [80]). Le principe consiste à construire une hiérarchie de sous-espaces optimaux qui aboutit à la construction d'un espace produit tensoriel dans lequel la fonction u

est projetée, pour définir l'approximation finale u^* . En se basant sur des hypothèses fortes sur l'erreur d'estimation faite dans la détermination des sous-espaces, l'auteur de [78] montre qu'avec un certain nombre d'évaluations de l'ordre de la complexité de stockage du format de tenseurs, l'approximation satisfait l'erreur souhaitée à des constantes près dépendant de certains opérateurs de projection, qui ne sont pas quantifiées.

Toutes ces conclusions montrent que les résultats théoriques sur la convergence des algorithmes existants sont limités. Dans cette thèse, nous proposons un algorithme qui construit une approximation u dans un format de tenseur basé sur des arbres, en utilisant un échantillonnage adaptatif et structuré avec quelques garanties théoriques, et nous proposons des stratégies heuristiques pour obtenir une approximation avec une précision souhaitée et une complexité quasi optimale.

Contributions

Plus précisément, les contributions de cette thèse peuvent être résumées à travers les objectifs suivants :

1. Proposer une méthode de projection dans un espace d'approximation linéaire, basée sur des techniques d'échantillonnage, qui soit stable et dont la construction nécessite un nombre d'évaluations proche de la dimension de l'espace d'approximation.
2. Proposer une stratégie pour construire des projections linéaires sur une séquence d'espaces d'approximation imbriqués, en utilisant un nombre réduit d'échantillons.
3. Proposer une méthode pour construire l'approximation d'une fonction dans un format de tenseur basé sur des arbres, avec des garanties de stabilité théoriques et proposer des heuristiques pour contrôler l'erreur.
4. Proposer une stratégie pour choisir l'arbre de dimension T afin de réduire les rangs de l'approximation à une précision donnée et donc sa complexité et le nombre d'évaluations requises.

Cette thèse est divisée en six chapitres. Les chapitres 1 et 4 font le point sur les méthodes de projection sur les espaces d'approximation linéaires et les formats de tenseur basés sur des arbres respectivement. Les chapitres 2, 3, 5 et 6 correspondent aux principales contributions de cette thèse en traitant les quatre objectifs ci-dessus. Dans l'annexe, on présente les aspects pratiques pour l'échantillonnage de distributions de probabilités multivariées dans des formats de tenseurs basés sur des arbres.

Dans le chapitre 2, nous proposons une nouvelle méthode de projection, appelée méthode des moindres carrés pondérés optimaux boostés, qui assure la stabilité de la projection des moindres carrés avec une taille d'échantillon proche de celui de l'interpolation ($n = m$). Elle consiste à échantillonner selon une mesure associée à l'optimisation d'un critère de stabilité sur une collection de n -échantillons indépendants, et à rééchantillonner selon cette mesure jusqu'à ce qu'une condition de stabilité soit remplie. Une méthode greedy est alors proposée pour enlever des points à l'échantillon obtenu. Des propriétés de quasi-optimalité en espérance sont obtenues pour la projection des moindres carrés pondérés, avec ou sans la procédure greedy. Si les observations sont polluées par un bruit, la propriété de quasi-optimalité est perdue, en raison d'un terme d'erreur supplémentaire dû au bruit. Ce dernier terme d'erreur peut cependant être réduit en augmentant n .

Afin de contrôler l'erreur d'approximation, on doit pouvoir choisir l'espace d'approximation de manière adaptative (jusqu'à une dimension suffisamment élevée pour atteindre une certaine précision). Pour l'approximation polynomiale et les applications en UQ, plusieurs stratégies pour construire de manière adaptative une séquence d'espaces d'approximation $(V_{m_l})_{l \geq 1}$ ont été proposées, voir [68, 18, 19]. Afin de limiter le nombre total d'évaluations de la fonction, il est important de réutiliser les échantillons qui ont servis pour construire l'approximation dans V_{m_l} afin de construire l'approximation dans $V_{m_{l+1}}$. Cependant, la projection de la méthode des moindres carrés pondérés optimaux boostés du chapitre 2 dépend de l'espace d'approximation, de sorte que la réutilisation des échantillons qui ont servis à construire une projection pour en construire une autre n'est pas simple. Dans [68], l'auteur propose une méthode adaptative optimale des moindres carrés pondérés qui recycle tous les échantillons d'une étape à l'autre. Dans le chapitre 3, nous proposons une méthode adaptative des moindres carrés pondérés optimaux boostés inspirée de [68], en utilisant à nouveau des procédures de rééchantillonnage, de conditionnement et greedy. Cette stratégie fonctionne pour une séquence générale d'espaces d'approximation imbriqués et nous l'appliquons pour l'approximation adaptative parcimonieuse.

Le chapitre 5 fournit un algorithme adapté de [78] pour construire l'approximation dans un format de tenseur basé sur des arbres $\mathcal{T}_r^T(\mathcal{H})$. En utilisant une approche allant des feuilles à la racine, l'algorithme construit grâce à l'analyse en composantes principales (ACP) séquentiellement le long de l'arbre de dimension T , des sous-espaces de faible dimension de fonctions de groupes de variables associés à chaque noeud de l'arbre. Une illustration du principe de l'algorithme est donnée dans Figure 3. Nous analysons l'algorithme dans le cas où la projection vérifie une propriété de stabilité et dans le cas particulier de la projection des moindres carrés optimaux boostés. En utilisant cette stratégie, l'erreur d'approximation a plusieurs contributions venant de la discrétisation (due à \mathcal{H}), de la troncature (due à l'approximation de faible rang) et de l'estimation (due au nombre limité d'échantillons). Nous proposons un algorithme adaptatif partiellement heuristique

qui contrôle simultanément les erreurs de discrétisation, de troncature et d'estimation.

L'algorithme proposé au chapitre 5 fonctionne pour toute fonction u et tout arbre de partition de dimension T . Cependant, les rangs et donc le nombre d'évaluations n nécessaires pour atteindre une précision donnée peuvent dépendre fortement de l'arbre choisi T . Le choix de l'arbre qui minimise le nombre d'évaluations n pour une précision donnée est un problème d'optimisation combinatoire qui ne peut pas être résolu en pratique. Dans [48], les auteurs proposent un algorithme stochastique qui explore un nombre raisonnable d'arbres de dimension avec la même arité (nombre maximal d'enfants que des noeuds d'un arbre). L'idée clé est de favoriser l'exploration des arbres conduisant à des faibles rangs. Dans [10], les auteurs proposent une stratégie déterministe qui construit un arbre de dimension des feuilles à la racine par concaténation successive des noeuds, comme le suggère la Figure 4. Les regroupements sont décidés de manière à minimiser une certaine fonctionnelle des α -rangs estimés. L'arbre sélectionné peut alors être utilisé pour calculer l'approximation de u . Au nombre d'évaluations nécessaires pour calculer l'approximation s'ajoute le nombre d'évaluations de la fonction utilisée pour estimer les α -rangs. Ces deux coûts doivent être pris en compte pour évaluer l'efficacité d'une stratégie. Dans le chapitre 6, nous proposons trois stratégies adaptatives différentes pour réaliser l'optimisation de l'arbre. Une stratégie stochastique globale inspirée de [48] qui explore plusieurs arbres de dimension et sélectionne celui qui minimise une fonctionnelle des α -rangs estimés. Les deux autres stratégies incluent l'optimisation des arbres à l'intérieur de l'algorithme de construction de l'approximation présentée au chapitre 5. Au fur et à mesure que l'algorithme va des feuilles à la racine, le nombre d'arbres possibles (et donc le nombre de α -rangs à évaluer) diminue fortement. C'est pourquoi nous proposons (en plus de la stratégie stochastique) une stratégie déterministe (qui permettra d'explorer un plus grand nombre d'arbres).

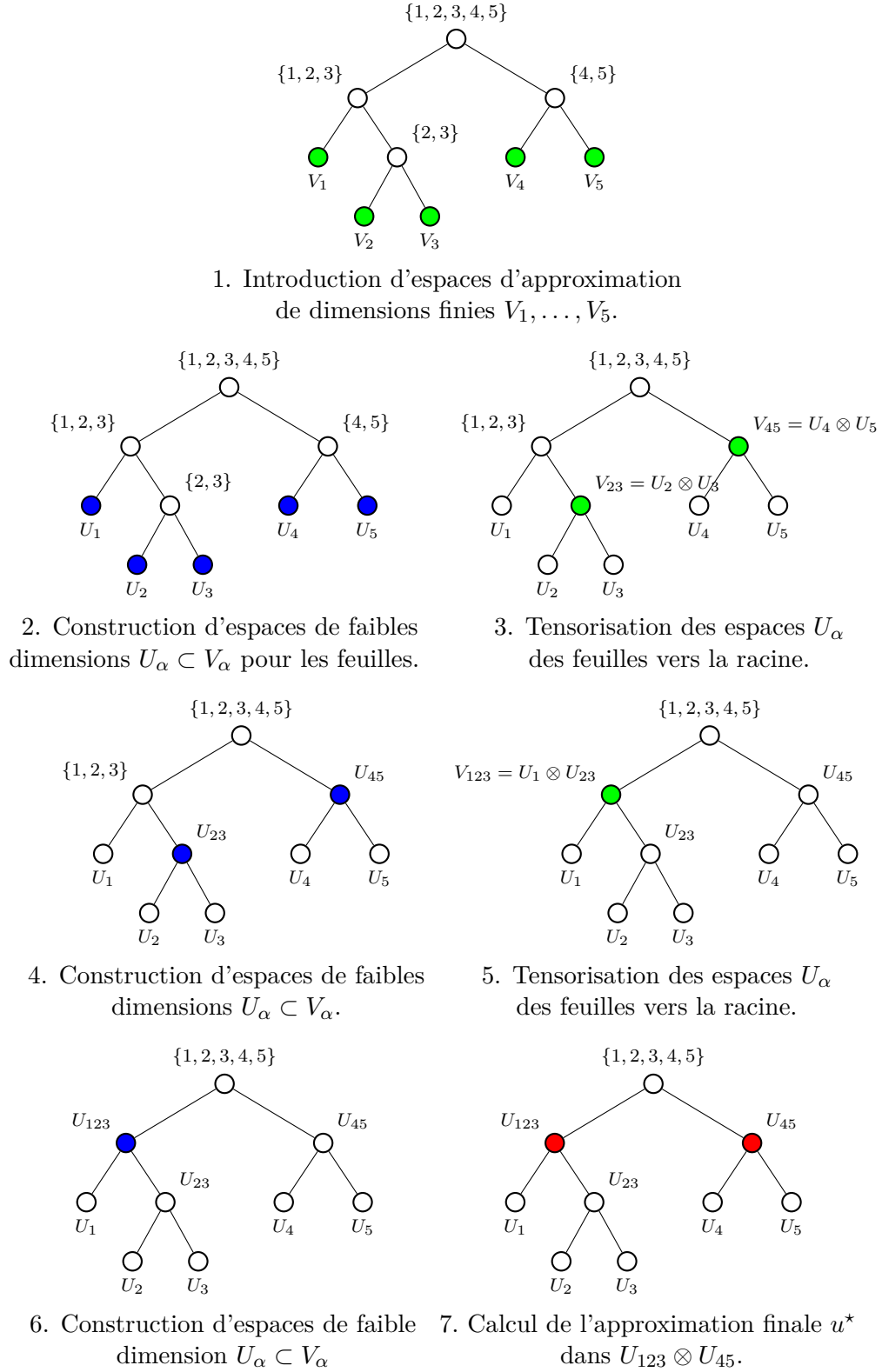
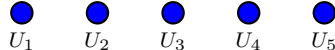
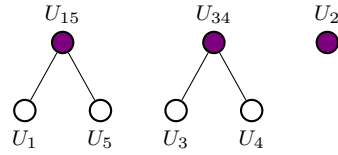
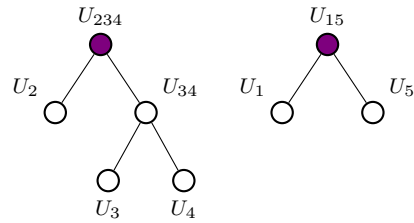


Figure 3 – Illustration de l'algorithme allant des feuilles à la racine du chapitre 5 pour la construction de l'approximation d'une fonction dans un format de tenseur basé sur des arbres.

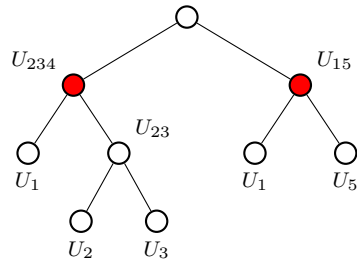
- 
 $U_1 \quad U_2 \quad U_3 \quad U_4 \quad U_5$
1. Construction d'espaces de faibles dimensions $U_\alpha \subset V_\alpha$



2. Test de plusieurs configurations de paires, et choix de celle qui donne des espaces U_α de faibles dimensions.



3. Recommencer pour le niveau suivant: test de plusieurs configurations de paires, et choix de celle qui donne des espaces de faibles dimensions U_α .



4. Calcul de l'approximation finale u^* dans $U_{234} \otimes U_{15}$.

Figure 4 – Construction de l'arbre des feuilles vers la racine avec optimisation locale des regroupements.

INTRODUCTION

Context

Thanks to computational models, researchers and engineers are able to replace expensive physical experiments by numerical simulations. The response of a model depending on some parameters can be represented by a function $u(x_1, \dots, x_d)$ of several variables. Despite the continuous improvement of hardware resources, most of these simulations remain very costly to compute. Furthermore, solving uncertainty quantification (UQ) problems, which include in particular forward uncertainty propagation, model calibration, inverse problems or sensitivity analysis [87, 34], require a high number of model's evaluations. One remedy is then to replace the costly model by a surrogate model, cheaper to evaluate, which amounts in approximating the function that represents the model's response.

The goal of approximation is to replace a function u by a simpler one (i.e. easier to evaluate) such that the distance (measuring the quality of the approximation) between the function u and its approximation, denoted u^* , is as small as possible. In general, the approximation is searched in a subspace of functions V_m described by a number of parameters m . A sequence of such subspaces $(V_m)_{m \geq 1}$ is called an approximation tool (or a model class), whose complexity is measured by its number of parameters m . The error of best approximation of u by an element of V_m is defined by the minimal distance between u and any element of V_m . The approximation tool should be adapted to the class of functions we want to approximate, meaning that the error of best approximation should converge to zero quickly enough with m .

There exist different approaches to construct a function's approximation either based on direct information on the function (e.g. point evaluations, linear measurements...) or equations satisfied by the function. In this thesis, we are interested in black-box (possibly noisy) functions (which corresponds to the first situation where the information on u is given by samples). Therefore, to construct the approximation of the function u , we rely on evaluations $y^i = u(x^i)$ for a set of points x^i or $y^i = u(x^i) + e^i$ in the noisy case. However, in the context of costly model, this approximation should be constructed using a reduced number of samples, ideally close to the number of parameters m . The samples x^i may either be independent and identically distributed from a measure μ or adaptively selected. This second situation is particularly relevant when no function's evaluation has already been made and this will be the framework for this thesis.

Typical methods to construct a sample-based approximation are interpolation and least-squares methods [28]. Classical interpolation methods include polynomial and spline interpolation (where the samples are in general chosen relatively to V_m , to ensure good properties of the interpolation operators), as well as kriging (which is quite efficient when the samples are given and may be improved with adaptive selection of the samples) [82]. For a given set of m points, x^1, \dots, x^m the interpolation u^* of u in V_m is defined by

$$u(x^i) = u^*(x^i), \text{ for all } 1 \leq i \leq m.$$

Interpolation is well-studied in the one-dimensional case, see for example [57], but in the multivariate case ($d > 1$), finding good points for the interpolation is a challenging problem, in particular when V_m is a non-classical approximation space. In practice, for sparse polynomial or tensor product spline interpolation, [18, 19, 16] propose strategies to construct good sequences of interpolation points. However, for non-classical approximation spaces, even if some sets of points have shown their efficiency (see [65, 15]), there is no associated guarantee on the resulting distance between u and its interpolation.

Given n samples x^1, \dots, x^n independent and identically distributed from a measure μ , standard least-squares methods define the approximation of u as the minimizer of

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n (v(x^i) - y^i)^2.$$

An interesting aspect of least-squares methods is that they are able, under conditions on the number of samples n , to guarantee a stable approximation and an error close to the best approximation error measured in L^2_μ norm. However, to do so, they may require a sample size n much higher than m (see [70, 22, 69, 93, 1]). Least-squares methods are a particular case of empirical risk minimization in supervised learning [30]. Other functionals can be used in this risk minimization, but they also require many evaluations in order to ensure the stability.

When the samples x^i can be selected, the estimation error can be improved using weighted least-squares [32, 73], by solving

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n w^i (v(\tilde{x}^i) - y^i)^2,$$

where the weights are adapted to V_m and the points \tilde{x}^i are independent and identically distributed samples from a well-chosen measure. It may allow to decrease the sample size n to reach the

same approximation error, compared to standard least-squares. In [24], the authors introduce an optimal sampling measure whose density with respect to the reference measure depends on the approximation space V_m . In [68], a similar optimal sampling approach is proposed whose major difference is that the random samples are more structured than in [24]. They both show that the weighted least-squares projection built with samples from this measure is stable in expectation under a milder condition on n compared to classical least-squares. Nevertheless, to ensure the stability of the weighted projection, these two optimal sampling measures still require a high number of samples n , compared to an interpolation method. The authors also propose optimal weighted least-squares methods in the case of adaptive approximation (when V_m is picked in a sequence of nested approximation spaces), see [68] and [4]. In this thesis, we propose a new projection method onto a fixed approximation space V_m which ensures the stability of the least-squares projection in expectation with a sample size close to m . We also propose a new method for the adaptive setting which significantly reduces the number of samples and still presents stability properties.

In many applications, functions may depend on a potentially high number of variables $d \gg 1$. When the dimension d increases, using standard approximation tools adapted to smooth functions (e.g. splines for Sobolev functions) leads to a complexity of the approximation methods which grows exponentially with d . This is the so-called curse of dimensionality, expression introduced in [12]. As a consequence, the number of evaluations necessary to approximate the function u with naive approximation tools may explode. In this thesis, we assume that the functions present some low-dimensional structures, so that we can expect a good approximation using a limited number of evaluations of the function. Exploiting these structures of the function usually requires particular approximation tools [29], which may be application dependent. Some of the most usual ones are detailed hereafter.

A first approach consists in using the sparsity of u by choosing a particular set of functions $\{\varphi_j\}_{j \in \Lambda}$ among a set of tensorized bases functions, such that $u(x)$ can be written as an expansion $\sum_{j \in \Lambda} a_j \varphi_j(x)$. Sparse approximation methods have been considered for solving uncertainty quantification problems, see e.g. [23, 6, 5, 18, 19, 74, 75] for sparse polynomial approximation.

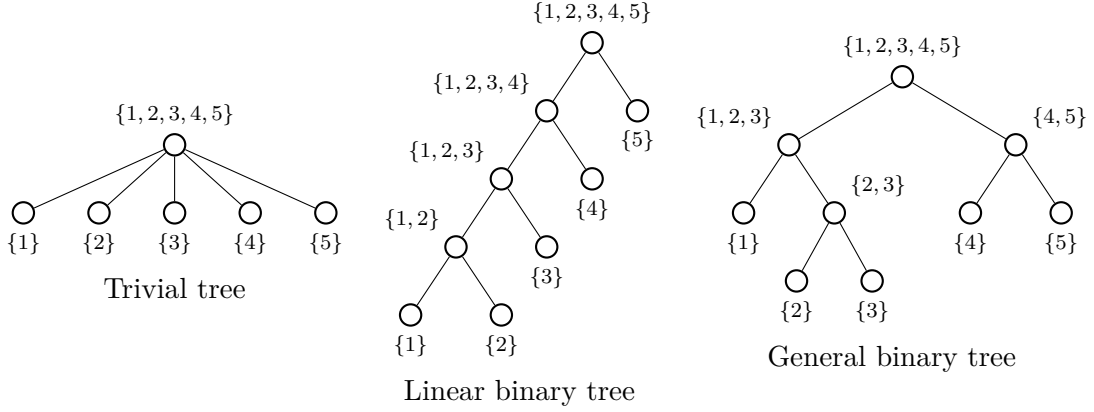
More structured model classes to deal with high-dimensional problems are additive models, introduced by [41], $\sum_{i=1}^d u_i(x_i)$ or more generally $\sum_{\alpha \in T} u_\alpha(x_\alpha)$ where $T \subset 2^{\{1, \dots, d\}}$ is either fixed (this corresponds to linear approximation) or a free parameter (this corresponds to non-linear approximation). There are also multiplicative models $\prod_{j=1}^d u_j(x_j)$ or more generally $\prod_{\alpha \in T} u_\alpha(x_\alpha)$, where T is also a subset of $2^{\{1, \dots, d\}}$ and the same distinction between linear and non-linear approximation is made, depending if T is fixed or not. These types of approximation tools are standard in statistics and probabilistic modelling (graphical models, bayesian networks).

Furthermore, more complex model classes based on compositions of functions have been proposed. Among them ridge models can be written under the form $v(Wx)$ with $W \in \mathbb{R}^{k \times d}$, where v belongs to a model class of functions of k variables with a low-dimensional parametrization. A typical example is the perceptron $a\sigma(w^T x + b)$, where σ is a given activation function, introduced in [83]. These compositional models also include projection pursuit models of the form $\sum_{j=1}^k v_j(w_j^T x)$, where for $j = 1, \dots, k$, $w_j \in \mathbb{R}^d$, [41, 42], a particular case being neural networks with one hidden layer $\sum_{j=1}^d a_j \sigma(w_j^T x + b_j)$. More general compositions of functions $v_L \circ v_{L-1} \circ \dots \circ v_1$ have been considered. They include the trendy deep neural networks, which are such that for each $j = 1, \dots, L$, the functions v_j write $v_j(t) = \sigma(A_j t + b_j)$, where σ is a chosen activation function, while $b_j \in \mathbb{R}^k$ and $A_j \in \mathbb{R}^{k \times d}$ are parameters to estimate [45]. In spite of its general formalism and the many successes that it has acquired in approximation, the use of deep neural networks generally requires a high number of evaluations to learn the parameters, which is not affordable in the context of costly evaluations. Furthermore, when considering general non-linear activation functions, it is difficult to guarantee a stable approximation whose error is close to the best approximation error, which is a central objective of this thesis.

In this work, we focus on the model class of rank-structured functions and in particular on the set of functions in tree-based tensor formats or tree tensor networks, which can be seen as a particular class of neural networks. More specifically, for α a subset of $D = \{1, \dots, d\}$ and $\alpha^c = D \setminus \alpha$ its complementary subset in D , we focus on functions that can be written as

$$v(x) = \sum_{i=1}^{r_\alpha} u_i^\alpha(x_\alpha) u_i^{\alpha^c}(x_{\alpha^c}), \text{ for all } \alpha \in T, \quad (1)$$

where T is a dimension partition tree, which is a particular collection of subsets of D . Some examples of dimension trees are given in Figure 1. The minimal integer r_α such that v can be written under the form (1) is called the α -rank of v , denoted $\text{rank}_\alpha(v)$. The set of functions in tree-based tensor format is defined by $\mathcal{T}_r^T(\mathcal{H}) = \{v \in \mathcal{H} : \text{rank}_\alpha(v) \leq r_\alpha, \alpha \in T\}$, with $r = (r_\alpha)_{\alpha \in T}$ a tuple of integers, and \mathcal{H} some approximation space of multivariate functions. It includes the Tucker format for a trivial tree, the tensor-train format for a linear binary tree [79] and the more general hierarchical format for a general dimension partition tree [54]. Any function in $\mathcal{T}_r^T(\mathcal{H})$ admits a multilinear parametrization with parameters forming a tree network of low-order tensors, hence the name tree tensor networks, see Figure 2 for an illustration in the case where $d = 5$. Considering dimension trees for T gives nice topological and geometrical properties to the model class $\mathcal{T}_r^T(\mathcal{H})$ [35, 37]. Tree tensor networks are particularly relevant for high-dimensional approximation, because the complexity of the parametrization of a function in tree-based tensor format is linear in the dimension d and polynomial in the ranks [53]. It can also be used for one-dimensional approximation [2, 3].

Figure 1 – Examples of dimension partition trees with $d = 5$.

However the ranks and therefore the number of evaluations n necessary to compute an approximation in tree-based tensor format with a given tolerance may strongly depend on the chosen tree T . The model class $\mathcal{T}_r^T(\mathcal{H})$ can also be interpreted as a class of functions that are compositions of multilinear functions, the structure of compositions being given by the tree. It has been identified with a particular class of deep neural networks (more precisely sum-product networks or arithmetic circuits) with a sparse connectivity and without parameter sharing [26]. The approximation error associated to this model class has two contributions : the truncation error (due to the finite ranks r) and the discretization error (due to the introduction of finite-dimensional spaces \mathcal{H}).

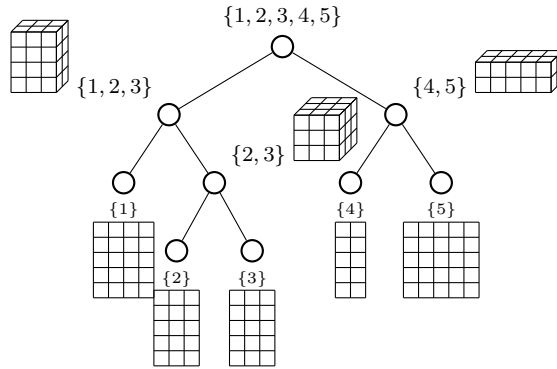


Figure 2 – A tree-based tensor network admits a multilinear parametrization with parameters forming a tree network of low-order tensors.

Tree tensor networks are an approximation tool that achieve good performances for many classes of functions. In [84] or more recently in [7, 2, 3], the authors show that for functions u with Sobolev

regularity, they achieve (whatever the tree is) an approximation rate which is optimal or close to optimal [61]. They also prove that for functions u with mixed Sobolev regularity, the approximation in tree-based tensor format achieves almost the ideal performance obtained by hyperbolic cross approximation. For a literature review on low-rank approximation methods, see [47]. Also, we refer the reader to monograph [53] and surveys [8, 36, 59, 60, 77, 76].

In the literature, some algorithms constructing approximations in tree-based tensor formats using points evaluations of functions have already been proposed. On the one hand, there are learning approaches that use random and unstructured evaluations of the functions [48, 55]. Robustness and effectiveness of such algorithms are observed on numerical experiments. However these algorithms are mainly based on heuristics and lack of theoretical analysis. On the other hand, there are algorithms that use adaptive and structured evaluations of tensors, see [64] (for the Tucker format), or [80] and [78] for tree-based tensor formats. Among adaptive approaches, the method from [78] is of particular interest, the author proposes a variant of higher-order singular value decomposition (HOSVD) (previous adaptations of HOSVD can be found in [46] and [80]). The principle is to construct a hierarchy of optimal subspaces that results in a final tensor product of subspaces in which the function u is projected, to define the final approximation u^* . Under strong assumptions on the estimation error made in the determination of subspaces, the author in [78] shows that with a number of evaluations scaling in the storage complexity of the tree-based tensor format, the approximation satisfies the desired error up to constants depending on some projection operators, which are not quantified.

All these conclusions show that theoretical results on the convergence of existing algorithms are limited. In this thesis we propose an algorithm that constructs an approximation u in tree-based tensor format, using adaptive and structured sampling with some theoretical guarantees, and we propose heuristic strategies for obtaining an approximation with a desired precision and near-optimal complexity.

Contributions

More precisely the contributions of this thesis can be summarized through the following objectives:

1. Propose a projection method in a linear space, based on sampling techniques, which is stable and whose construction requires a number of evaluations close to the dimension of the approximation space.
2. Propose a strategy for computing linear projections onto a nested sequence of approximation spaces, using a reduced number of samples.

3. Propose a method to construct the approximation of a function in tree-based tensor format, with theoretical stability guarantees and propose heuristics to control the error.
4. Propose a strategy to choose the tree structure T in order to reduce ranks of the approximation at given precision and therefore its complexity and the required number of samples.

This thesis is divided into six chapters. Chapters 1 and 4 provide a state-of-the-art on projection methods onto linear approximation spaces and tree-based tensor formats respectively. Chapters 2, 3, 5 and 6 correspond to the main contributions of this thesis addressing the four objectives listed above. In the appendix implementation aspects for sampling multivariate probability distributions in tree-based tensor formats are given.

In Chapter 2, we propose a new projection method, called boosted optimal weighted least-squares method, that ensures stability of the least-squares projection with a sample size close to the interpolation regime $n = m$. It consists in sampling according to a measure associated with the optimization of a stability criterion over a collection of independent n -samples, and resampling according to this measure until a stability condition is satisfied. A greedy method is then proposed to remove points from the obtained sample. Quasi-optimality properties in expectation are obtained for the weighted least-squares projection, with or without the greedy procedure. If the observations are polluted by a noise, quasi-optimality property is lost, because of an additional error term due to the noise. This latter error term can however be reduced by increasing n .

In order to control the approximation error, we should be able to choose the approximation space adaptively (with a dimension high enough to reach a certain accuracy). For polynomial approximation and applications in UQ, several strategies to construct adaptively a sequence of approximation spaces $(V_{m_l})_{l \geq 1}$ have been proposed in [43, 67, 18, 19]. To limit the total number of evaluations of the function, it is necessary to reuse the samples used to build the projection onto the space V_{m_l} in order to construct the one onto the space $V_{m_{l+1}}$. However, the optimal weighted boosted least-squares projection from Chapter 2 depends on the approximation space, such that reusing the samples used to construct a projection to construct another is not straightforward. In [68], the author proposes an adaptive optimal weighted least-squares method that recycles all samples from one step to another. In Chapter 3, we propose an adaptive boosted optimal weighted least-squares method inspired from [68], using again resampling, conditioning and greedy procedures. This strategy works for a general sequence of nested approximation spaces and we apply it for adaptive sparse polynomial approximation.

Chapter 5 provides an algorithm adapted from [78] to construct the approximation in a tree-based tensor format $\mathcal{T}_r^T(\mathcal{H})$. Using a leaves-to-root approach, the algorithm constructs, thanks to

a series of principal component analyses (PCA) along the tree structure T , low-dimensional subspaces of functions of groups of variables associated with each node of the tree. This algorithm is illustrated by the Figure 3. We analyse the algorithm in the case where the projection verifies a stability property and in the particular case where the projection is the boosted least-squares projection. Using this strategy the approximation error has several contributions due to discretization (due to \mathcal{H}), truncation (due to low-rank approximation) and estimation (due to the limited number of samples). We propose a partially heuristic adaptive algorithm that controls simultaneously the discretization, truncation and estimation errors.

The proposed algorithm from Chapter 5 works for any function u and any dimension partition tree T . However the ranks and therefore the number of evaluations n necessary to reach a given precision may strongly depend on the chosen tree T . Choosing the tree which minimizes the number of evaluations n for a given accuracy is a combinatorial optimization problem and it is not feasible in practice. In [48], the authors propose a stochastic algorithm that explores a reasonable number of dimension trees with the same arity (maximal number of children of any node of a tree). The key idea is to favour the exploration of trees yielding low ranks. In [10], the authors propose a deterministic strategy that constructs a dimension tree in a leaves-to-root approach by successive concatenations of nodes. The groupings are decided in order to minimize a certain functional based on estimated α -ranks. The selected tree can be used to compute the approximation of u . The number of function's evaluations used to estimate the α -ranks adds up to the number of evaluations necessary to compute the approximation. This total number of samples should be used to evaluate the efficiency of a strategy. In Chapter 6, we propose three different adaptive strategies to perform tree optimization. A global stochastic strategy inspired from [48] which explores several dimension trees and select the one minimizing a complexity functional depending on α -ranks estimations. The two other strategies include tree optimization inside the algorithm for the construction of the approximation presented in Chapter 5, as suggested in Figure 4. As the algorithm goes from the leaves to the root, the number of possible trees (and thus the number of α -ranks to be evaluated) decreases sharply. This is why we propose (in addition to the stochastic strategy) a deterministic strategy (which will explore a larger number of trees).

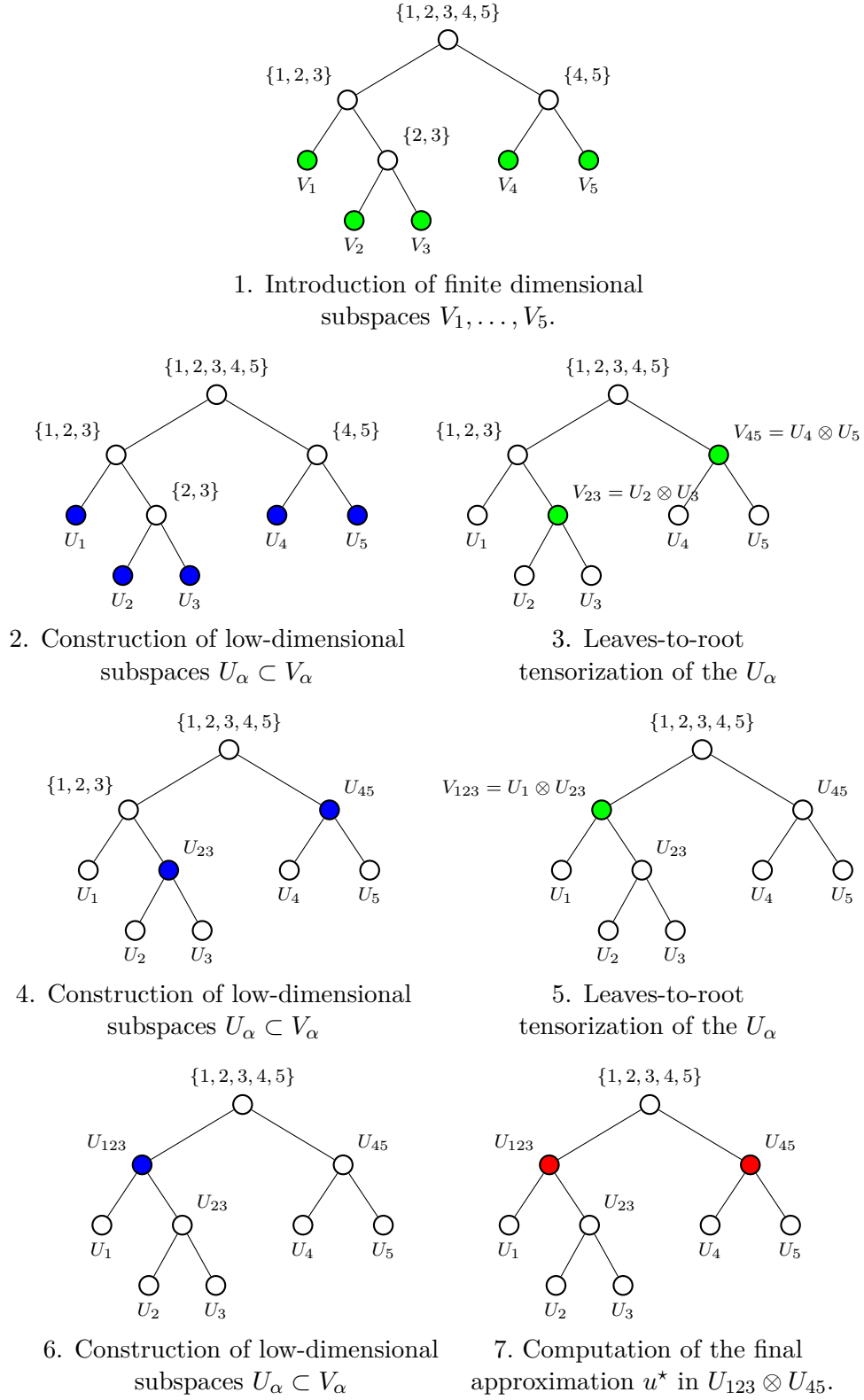
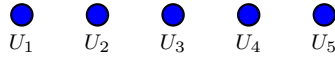
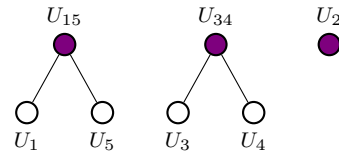


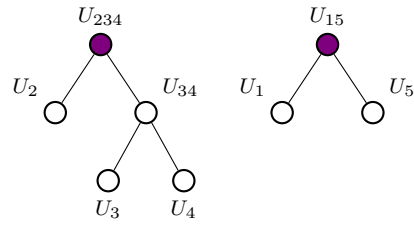
Figure 3 – Illustration of the leaves-to-root algorithm from Chapter 5 for the construction of the approximation of a function in tree-based tensor format.


 U_1 U_2 U_3 U_4 U_5

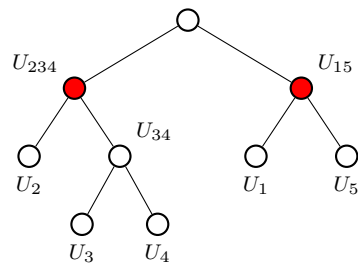
1. Construction of low-dimensional subspaces $U_\alpha \subset V_\alpha$



2. Try several pairings,
choose one which leads to
low-dimensional U_α .



3. Repeat for the next level:
try several pairings,
choose one which leads to
low-dimensional U_α .



4. Computation of the final
approximation u^* in $U_{234} \otimes U_{15}$.

Figure 4 – Leaves-to-root construction of the tree with local optimizations

PROJECTION METHODS

Contents

1.1	Introduction	29
1.2	Interpolation	29
1.2.1	Interpolation in the one-dimensional case	31
1.2.2	Interpolation with any arbitrary approximation space	33
1.2.3	Interpolation with tensor product bases	34
1.3	Least-squares method	35
1.3.1	Weighted least-squares projection	36
1.3.2	Random sampling	38
1.3.3	Optimal sampling measure	40

1.1 Introduction

In this chapter we present two classical families of methods to construct the approximation of a function onto a linear space V_m , using evaluations of the function at suitably chosen points: interpolation and least-squares regression [28].

The outline of this chapter is as follows. In Section 1.2, we present the approximation of multivariate functions by interpolation. In Section 1.3, we present weighted least-squares projections. Then, we recall the optimal sampling measure from [24], and outline its limitations.

1.2 Interpolation

In this section, we begin by recalling the definition of interpolation of a real-valued function u defined on a subset $\mathcal{X} \subset \mathbb{R}^d$ onto an approximation space V_m .

We consider a m -dimensional linear space V_m whose basis is denoted $\{\varphi_j\}_{j=1}^m$, where the φ_j can be polynomials, splines or wavelets for instance. We denote by

$$\boldsymbol{\varphi} = (\varphi_1, \dots, \varphi_m) : \mathcal{X} \rightarrow \mathbb{R}^m$$

the m -dimensional vector-valued function such that $\boldsymbol{\varphi}(x) = (\varphi_1(x), \dots, \varphi_m(x))^T$.

Let $\Gamma_m = \{x^i\}_{i=1}^m$ be a set of m points in \mathcal{X} . The interpolation of u is defined as the approximation $u^\star \in V_m$ such that

$$u^\star(x^i) = u(x^i) \text{ for all } x^i \in \Gamma_m.$$

u^\star can be written under the form

$$u^\star(x) = \sum_{j=1}^m a^j \varphi_j(x),$$

where the coefficients $(a^j)_{j=1}^m$ are solution of the following linear system,

$$\underbrace{\begin{bmatrix} \varphi_1(x^1) & \varphi_2(x^1) & \dots & \varphi_m(x^1) \\ \varphi_1(x^2) & \varphi_2(x^2) & \dots & \varphi_m(x^2) \\ \dots & & & \\ \varphi_1(x^m) & \varphi_2(x^m) & \dots & \varphi_m(x^m) \end{bmatrix}}_{\mathbf{V}_\varphi(x^1, \dots, x^m)} \begin{bmatrix} a^1 \\ a^2 \\ \vdots \\ a^m \end{bmatrix} = \begin{bmatrix} u(x^1) \\ u(x^2) \\ \vdots \\ u(x^m) \end{bmatrix}.$$

Here, the matrix $\mathbf{V}_\varphi(x^1, \dots, x^m)$ is called the Vandermonde like-matrix. We say that Γ_m is unisolvent for the space $V_m = \text{span}\{\varphi_j\}_{j=1}^m$ if $\mathbf{V}_\varphi(x^1, \dots, x^m)$ is invertible, which implies that u^\star is uniquely defined for any u . A set of points that maximizes the determinant of the Vandermonde-like matrix $\mathbf{V}_\varphi(x^1, \dots, x^m)$ is called a set of Fekete points [40]. If Γ_m is unisolvent, the interpolation operator $\mathcal{I}_m : \mathcal{X} \rightarrow V_m$ is defined by $\mathcal{I}_m u = u^\star$ and the associated Lebesgue constant L_m (in L^∞ norm) is defined by

$$L_m = \|\mathcal{I}_m\| = \sup_{v \neq 0} \frac{\|\mathcal{I}_m v\|_{L^\infty}}{\|v\|_{L^\infty}}.$$

It holds

$$\|u - u^\star\|_{L^\infty} \leq (1 + L_m) \inf_{v \in V_m} \|u - v\|_{L^\infty}.$$

The Lebesgue constant L_m can be expressed in terms of interpolation functions, but in general an expression of L_m can not be explicitly found. In dimension one, interpolation is well understood for various types of approximations (e.g. with polynomials, splines). In particular for polynomial spaces V_m there exists good sets of points Γ_m , such that the Lebesgue constant L_m is controlled. Some of them are detailed in Section 1.2.1. However in higher dimension, finding good points for polynomial approximation is a challenging problem. In Section 1.2.2, we consider the multivari-

ate setting and present several sets of interpolation points that have demonstrated their efficiency in practice. In Section 1.2.3, we consider the case where the approximation space is constructed from tensor product bases and present the construction of interpolation operator for approximation spaces associated to downward closed index sets. The purpose of these three sections is not to give an exhaustive list of already studied interpolation points but an insight on the most classical interpolation points and in particular the ones to which we will compare the boosted optimal weighted least-squares method, contribution of this thesis.

1.2.1 Interpolation in the one-dimensional case

In this subsection, we recall some well-known results for one-dimensional polynomial interpolation i.e. with V_m a space of degree $m - 1$ polynomials. A lower bound for the Lebesgue constant is given by the Erdos theorem [33], which shows that in the best case, L_m grows as the logarithm of the dimension m . More precisely for any set of points, there exists a constant $C > 0$ such that,

$$L_m \geq \frac{2}{\pi} \log(m + 1) - C.$$

Uniformly distributed points.

For uniformly distributed points over $[-1, 1]$, it holds

$$L_m \sim \frac{2^{m+1}}{em \log(m)} \text{ as } m \rightarrow \infty,$$

that shows that uniformly distributed points is clearly not a good option for interpolation.

Chebyshev points.

Another classical set of points is the Chebyshev set of points, which is defined as the solution of

$$\min_{x^1, \dots, x^m} \max_{x \in [-1, 1]} \prod_{j=1}^m |x - x^j|.$$

For a given m , these points are near optimal asymptotically according to the Erdos theorem, since

$$L_m \sim \frac{2}{\pi} \log(m) \text{ as } m \rightarrow \infty.$$

However, $\Gamma_m \not\subset \Gamma_{m+1}$ (i.e the set of the m Chebyshev points is not included in the set of $m + 1$ Chebyshev points), so that they are not really adapted for adaptive approximation.

Gauss-quadrature type points.

When the approximation space V_m is the span of the univariate Legendre polynomials $\{\varphi_j\}_{j=1}^m$, the Gauss-Legendre points are well-suited for interpolation. For $1 \leq i \leq m$, the i^{th} point of the Gauss-Legendre set of m points is given by the i^{th} root of φ_m , the Legendre polynomial of degree $m - 1$. The Lebesgue constant for the point set consisting of the Gauss-Legendre points is

$$L_m \leq C\sqrt{m}, \text{ see [88].}$$

Similarly, when the approximation space is the span of the Hermite polynomials, the Gauss-Hermite points can be defined as the roots of the Hermite polynomials.

Clenshaw-Curtis points.

Clenshaw-Curtis points are the extrema of Chebyshev polynomials. For $m \geq 1$, the sequence of m Clenshaw-Curtis points is given by $\Gamma_m = \{\cos(\frac{j\pi}{m-1}) : j = 0, \dots, m-1\}$ and the Lebesgue constant verifies

$$L_m \leq \frac{2}{\pi} \log(m-1) + 1.$$

The interest is that $\Gamma_{2^m} \subset \Gamma_{2^{m+1}}$, which is useful for adaptive approximation. But this requires to double the dimension of the approximation space at each iteration.

Leja points.

Another sequence of points which is suitable for adaptive approximation is the Leja sequence, which is defined recursively for $j \geq 2$ by

$$x^j \in \arg \max_{x \in [-1,1]} \prod_{l=1}^j |x - x^l|,$$

with x^1 taken equal to 0 or 1 in general. The definition is such that this sequence is not unique. An equivalent definition for the Leja sequence of points is given by

$$x^j \in \arg \max_{x \in [-1,1]} |\det \mathbf{V}_\varphi(x^1, \dots, x^{j-1}, x)| \text{ for } j \geq 2.$$

Therefore Leja sequences can be seen as a greedy D -optimal experimental design [38]. It does not exist any explicit expression for the Lebesgue constant associated to this set of points, but it has been proven that it grows subexponentially [39].

\mathcal{R} -Leja points are real part of the Leja points defined on the complex unit disc. In other terms, they are defined as the projection on the real line of complex Leja points. In dimension one, it has

been proven (see [21]) that

$$L_m \leq C(m+1)^2.$$

1.2.2 Interpolation with any arbitrary approximation space

Leja points.

The definition of the Leja points can be extended to the cases where $d > 1$, for any compact set E of \mathbb{R}^d [66], for $k \geq 2$, by

$$x^k \in \arg \max_{x \in E} |\det \mathbf{V}_\varphi(x^1, \dots, x^{k-1}, x)|.$$

When d is high, this sequence of points may be costly to compute, therefore it can be replaced by a so-called sequence of discrete Leja points which relies on a LU factorization with row-pivoting of the Vandermonde-like matrix [15]. To deal with the case of unbounded domains, weighted versions of discrete Leja points have been proposed [16].

Fekete points.

A set of points that maximizes the determinant of the Vandermonde-like matrix $\mathbf{V}_\varphi(x^1, \dots, x^m)$ is called a set of Fekete points [40]. These points are not limited to polynomial approximation but can be used in a very general setting. In general they are very expensive to compute because it requires a multivariate optimization. Therefore methods to compute approximate Fekete points based on QR factorizations with column pivoting of the Vandermonde-like matrix have been proposed in [86], [14] and [15]. To deal with the case of unbounded domains, weighted Fekete points have also been introduced [51]. There exist connections between Leja and Fekete points [66].

Magic points.

In [65], the authors propose a sequence of interpolation points called magic points which provides in practice good results in terms of the behaviour of the Lebesgue constant. An interest of this method is that it is not limited to polynomial approximation. Furthermore, these points can be relatively simply determined through a greedy algorithm. The procedure is as follows: we start from a candidate set of points Γ and determine a first point x^1 and an index i_1 such that

$$|\varphi_{i_1}(x^1)| = \max_{x \in \Gamma} \max_{1 \leq i \leq m} |\varphi_i(x)|.$$

Then for $k \geq 1$, we define

$$\tilde{\varphi}_i^k(x) = \varphi_i(x) - \sum_{l=1}^k \sum_{p=1}^k \varphi_{i_l}(x) a_{l,p}^k \varphi_i(x^p),$$

with the matrix $A = (a_{l,p}^k)_{1 \leq l, p \leq k}$ being the inverse of the matrix

$$\begin{bmatrix} \varphi_{i_1}(x^1) & \varphi_{i_2}(x^1) & \dots & \varphi_{i_k}(x^1) \\ \varphi_{i_1}(x^2) & \varphi_{i_2}(x^2) & \dots & \varphi_{i_k}(x^2) \\ \dots & \dots & \dots & \dots \\ \varphi_{i_1}(x^k) & \varphi_{i_2}(x^k) & \dots & \varphi_{i_k}(x^k) \end{bmatrix},$$

such that $\tilde{\varphi}_{i_l}^k(x) = 0$ for all $1 \leq l \leq k$ and $x \in \mathcal{X}$ and $\tilde{\varphi}_i^k(x^p) = 0$ for all $1 \leq p \leq k$ and $1 \leq i \leq m$. Then we determine the point $x^{k+1} \in \Gamma$ and an index i_{k+1} such that

$$|\tilde{\varphi}_{i_{k+1}}^k(x^{k+1})| = \max_{x \in \Gamma} \max_{1 \leq i \leq m} |\tilde{\varphi}_i^k(x)|.$$

The Lebesgue constant associated with the magic points behaves relatively well in practice [92].

1.2.3 Interpolation with tensor product bases

In this subsection, we consider the case where \mathcal{X} has a product structure $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_d$, then multivariate interpolation on tensor product bases can be constructed from univariate interpolation.

For each $k \in \{1, \dots, d\}$, we consider $\{\psi_{i_k}^k\}_{i_k \geq 1}$ a set of univariate functions defined on \mathcal{X}_k . We define for all $i = (i_1, \dots, i_d) \in \mathbb{N}^d$,

$$\varphi_i(x) = \prod_{k=1}^d \psi_{i_k}^k(x_k). \quad (1.1)$$

Let us introduce a set of multi-indices $\Lambda_m \subset \mathbb{N}^d$ such that $\#\Lambda_m = m$. The set $\Lambda_m \subset \mathbb{N}^d$ is said to be downward closed if for all $\alpha \in \Lambda_m$ and any β such that $\beta \leq \alpha$ (i.e. for all $1 \leq k \leq d$, $\beta_k \leq \alpha_k$), then $\beta \in \Lambda_m$. We introduce V_m the associated approximation space

$$V_m = \text{span}\{\varphi_i : i = (i_1, \dots, i_d) \in \Lambda_m\}.$$

Furthermore, for each $k \in \{1, \dots, d\}$, we consider $\{x_k^{i_k}\}_{i_k=1}^{m_k}$ a sequence of points in \mathcal{X}_k unisolvent for the basis $\{\psi_{i_k}^k\}_{i_k=1}^{m_k}$, therefore defining a unique interpolation operator denoted $\mathcal{I}_{m_k}^k : \mathcal{X}_k \mapsto V_{m_k}$.

When Λ_m is a product set, i.e. $\Lambda_m = \Lambda_{m_1}^1 \times \dots \times \Lambda_{m_d}^d$, the grid of interpolation points

$$\Gamma_m = \bigotimes_{k=1}^d \{x_k^1, \dots, x_k^{m_k}\}$$

is unisolvent for the tensorized basis $\{\varphi_i\}_{i \in \Lambda_m}$ from (1.1) and the associated interpolation operator is uniquely defined as

$$\mathcal{I}_m = \bigotimes_{k=1}^d \mathcal{I}_{m_k}^k.$$

When Λ_m is a general downward closed set, for each $k \in \{1, \dots, d\}$, the univariate interpolation operator $\mathcal{I}_{m_k}^k$ can be expressed as a sum of differences operators:

$$\mathcal{I}_{m_k}^k = \sum_{l=1}^{m_k} \Delta_l^k, \text{ where } \Delta_l^k = \mathcal{I}_l^k - \mathcal{I}_{l-1}^k \text{ and using the convention } \mathcal{I}_0^k = 0.$$

When $d > 1$, we consider nested sequences of points $\{x_k^{i_k}\}_{i_k=1}^{m_k}$ for all $k \in \{1, \dots, d\}$. Furthermore, for each $i = (i_1, \dots, i_d) \in \Lambda_m$ we introduce the tensorized differences operators $\Delta_i = \bigotimes_{k=1}^d \Delta_{i_k}^k$ where $i = (i_1, \dots, i_d)$ and then the interpolation operator \mathcal{I}_m on the space $V_m = \text{span}\{\varphi_i : i = (i_1, \dots, i_d) \in \Lambda_m\}$ can be expressed as

$$\mathcal{I}_m = \sum_{i \in \Lambda_m} \Delta_i.$$

It is associated with the sequence of interpolation points

$$\Gamma_m = \{x^i = (x_1^{i_1}, \dots, x_d^{i_d}) : i \in \Lambda_m\}$$

which is unisolvent for V_m [23].

Remark 1.1. Kriging interpolation.

Another widely used interpolation technique is kriging. Contrary to the interpolation techniques mentioned above, the approximation is not constructed in V_m a linear space with a fixed basis but in kernel-based space, which in general depends on the chosen points. This method has proved its importance but is not used in this thesis (possibly in future work). For more details the reader is referred to [82].

1.3 Least-squares method

Let \mathcal{X} be a subset of \mathbb{R}^d equipped with a probability measure μ , with $d \geq 1$. We consider a function u from $L_\mu^2(\mathcal{X})$, the Hilbert space of square-integrable real-valued functions defined on \mathcal{X} .

We let $\|\cdot\|_{L_\mu^2}$ be the natural norm in $L_\mu^2(\mathcal{X})$ defined by

$$\|v\|_{L_\mu^2}^2 = \int_{\mathcal{X}} v(x)^2 d\mu(x).$$

When there is no ambiguity, $L_\mu^2(\mathcal{X})$ will be simply denoted by L_μ^2 , and the norm $\|\cdot\|_{L_\mu^2}$ and associated inner product $(\cdot, \cdot)_{L_\mu^2}$ will be denoted by $\|\cdot\|$ and (\cdot, \cdot) respectively.

Let V_m be a m -dimensional subspace of L_μ^2 , with $m \geq 1$, and $\{\varphi_j\}_{j=1}^m$ be an orthonormal basis of V_m . The best approximation of u in V_m is given by its orthogonal projection defined by

$$P_{V_m} u := \arg \min_{v \in V_m} \|u - v\|.$$

Least-squares regression, which defines the approximation by solving

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n (u(x^i) - v(x^i))^2,$$

where the x^i are independent and identically distributed (i.i.d.) samples drawn from the measure μ . However, to guarantee a stable approximation and an error close to the best approximation error, least-squares regression may require a sample size n much higher than m (see [70, 22, 69, 93]). This issue can be mitigated by weighted least-squares projection, which is obtained by solving

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n w(x^i) (u(x^i) - v(x^i))^2,$$

where the x^i are points not necessarily drawn from μ and the $w(x^i)$ are corresponding weights. A suitable choice of weights and points may allow to decrease the sample size to reach the same approximation error, see e.g. [32, 73]. In [24], the authors introduce an optimal sampling measure ρ with a density $w(x)^{-1}$ with respect to the reference measure μ which depends on the approximation space. Choosing i.i.d. samples x^i from this optimal measure, one obtains with high probability $1 - \eta$ a stable approximation and an error of the order of the best approximation error using a sample size $n = \mathcal{O}(m \log(m\eta^{-1}))$. Nevertheless, the necessary condition for having stability requires a sample size n much higher than m , especially when a small probability η is desired.

1.3.1 Weighted least-squares projection

Letting $\mathbf{x}^n := \{x^i\}_{i=1}^n$ be a set of n points in \mathcal{X} , we consider the weighted least-squares projection defined by

$$Q_{V_m}^{\mathbf{x}^n} u := \arg \min_{v \in V_m} \|u - v\|_{\mathbf{x}^n},$$

where $\|\cdot\|_{\mathbf{x}^n}$ is a discrete semi-norm defined for v in L_μ^2 by

$$\|v\|_{\mathbf{x}^n}^2 := \frac{1}{n} \sum_{i=1}^n w(x^i) v(x^i)^2,$$

where w is a given non negative function defined on \mathcal{X} . We recall that $\boldsymbol{\varphi} = (\varphi_1, \dots, \varphi_m) : \mathcal{X} \rightarrow \mathbb{R}^m$ denotes the m -dimensional vector-valued function such that $\boldsymbol{\varphi}(x) = (\varphi_1(x), \dots, \varphi_m(x))^T$. Also we denote by $\mathbf{G}_{\mathbf{x}^n}$ the empirical Gram matrix defined by

$$\mathbf{G}_{\mathbf{x}^n} := \frac{1}{n} \sum_{i=1}^n w(x^i) \boldsymbol{\varphi}(x^i) \otimes \boldsymbol{\varphi}(x^i).$$

The stability of the weighted least-squares projection can be characterized by

$$Z_{\mathbf{x}^n} := \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2,$$

which measures a distance between the empirical Gram matrix and the identity matrix \mathbf{I} , with $\|\cdot\|_2$ the matrix spectral norm. For any v in V_m , we have

$$(1 - Z_{\mathbf{x}^n})\|v\|^2 \leq \|v\|_{\mathbf{x}^n}^2 \leq (1 + Z_{\mathbf{x}^n})\|v\|^2. \quad (1.2)$$

We have the following properties that will be useful in subsequent analyses.

Lemma 1.1. *Let \mathbf{x}^n be a set of n points in \mathcal{X} such that $Z_{\mathbf{x}^n} = \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2 \leq \delta$ for some $\delta \in [0, 1)$. Then, for any v in V_m , we have*

$$(1 - \delta)\|v\|^2 \leq \|v\|_{\mathbf{x}^n}^2 \leq (1 + \delta)\|v\|^2 \quad (1.3)$$

and the weighted least-squares projection $Q_{V_m}^{\mathbf{x}^n} u$ associated with \mathbf{x}^n satisfies

$$\|u - Q_{V_m}^{\mathbf{x}^n} u\|^2 \leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \|u - P_{V_m} u\|_{\mathbf{x}^n}^2. \quad (1.4)$$

Proof. The property (1.3) directly follows from (1.2) and the assumption that $Z_{\mathbf{x}^n} \leq \delta$. Using the property of the orthogonal projection $P_{V_m} u$ and (1.3), we have that

$$\begin{aligned} \|u - Q_{V_m}^{\mathbf{x}^n} u\|^2 &= \|u - P_{V_m} u\|^2 + \|P_{V_m} u - Q_{V_m}^{\mathbf{x}^n} u\|^2 \\ &\leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \|P_{V_m} u - Q_{V_m}^{\mathbf{x}^n} u\|_{\mathbf{x}^n}^2. \end{aligned}$$

Using the fact that $Q_{V_m}^{\mathbf{x}^n}$ is an orthogonal projection on V_m with respect to the semi-norm $\|\cdot\|_{\mathbf{x}^n}$, we have that for any v , $\|Q_{V_m}^{\mathbf{x}^n} v\|_{\mathbf{x}^n} \leq \|v\|_{\mathbf{x}^n}$. We deduce that

$$\|P_{V_m} u - Q_{V_m}^{\mathbf{x}^n} u\|_{\mathbf{x}^n} = \|Q_{V_m}^{\mathbf{x}^n} (P_{V_m} u - u)\|_{\mathbf{x}^n} \leq \|P_{V_m} u - u\|_{\mathbf{x}^n},$$

from which we deduce (1.4). \square

We now provide a result which bounds the L_μ^2 error by a best approximation error with respect to a weighted supremum norm.

Theorem 1.2. *Let \mathbf{x}^n be a set of n points in \mathcal{X} such that $Z_{\mathbf{x}^n} = \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2 \leq \delta$ for some $\delta \in [0, 1)$. Then,*

$$\|u - Q_{V_m}^{\mathbf{x}^n} u\| \leq (B + (1 - \delta)^{-1/2}) \inf_{v \in V_m} \|u - v\|_{\infty, w} \quad (1.5)$$

where $B^2 = \int_{\mathcal{X}} w(x)^{-1} d\mu(x)$ and $\|v\|_{\infty, w} = \sup_{x \in \mathcal{X}} w(x)^{1/2} |v(x)|$.

Proof. Using the triangular inequality and Lemma 1.1 we note that for any $v \in V_m$,

$$\|u - Q_{V_m}^{\mathbf{x}^n} u\| \leq \|u - v\| + (1 - \delta)^{-1/2} \|v - Q_{V_m}^{\mathbf{x}^n} u\|_{\mathbf{x}^n},$$

and $\|v - Q_{V_m}^{\mathbf{x}^n} u\|_{\mathbf{x}^n} = \|Q_{V_m}^{\mathbf{x}^n}(v - u)\|_{\mathbf{x}^n} \leq \|u - v\|_{\mathbf{x}^n}$.

We then conclude by using the inequalities $\|u - v\|_{\mathbf{x}^n} \leq \|u - v\|_{\infty, w}$ and

$$\|u - v\| \leq \left(\int_{\mathcal{X}} w(x)^{-1} d\mu(x) \right)^{1/2} \sup_{x \in \mathcal{X}} w(x)^{1/2} |u(x) - v(x)|. \quad \square$$

In the case where w^{-1} is the density of a probability measure with respect to μ (which will be the case in the rest of the paper), the constant B from Theorem 1.2 is equal to 1.

1.3.2 Random sampling

We consider the measure ρ on \mathcal{X} with density w^{-1} with respect to μ , i.e. $d\rho = w^{-1}d\mu$. If the x^1, \dots, x^n are i.i.d. random variables drawn from the measure ρ , or equivalently if $\mathbf{x}^n = (x^1, \dots, x^n)$ is drawn from the product measure $\rho^{\otimes n} := \boldsymbol{\rho}^n$ on \mathcal{X}^n , then for any function v in L_μ^2 (not only those in V_m), we have

$$\mathbb{E}(\|v\|_{\mathbf{x}^n}^2) = \|v\|^2. \quad (1.6)$$

The condition (1.6) restricted to all functions $v \in V_m$ implies that the empirical Gram matrix $\mathbf{G}_{\mathbf{x}^n}$ satisfies

$$\mathbb{E}(\mathbf{G}_{\mathbf{x}^n}) = \frac{1}{n} \sum_{i=1}^n \mathbb{E}(w(x^i) \boldsymbol{\varphi}(x^i) \otimes \boldsymbol{\varphi}(x^i)) = \mathbf{I}.$$

The random variable $Z_{\mathbf{x}^n} = \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2$ quantifies how much the random matrix $\mathbf{G}_{\mathbf{x}^n}$ deviates from its expectation. For any $\delta \in [0, 1)$, if

$$\mathbb{P}(Z_{\mathbf{x}^n} > \delta) \leq \eta, \quad (1.7)$$

then for all $v \in V_m$, Equation (1.3) holds with probability higher than $1 - \eta$. We directly conclude from Theorem 1.2 that the weighted least-squares projection $Q_{V_m}^{\mathbf{x}^n}$ satisfies Equation (1.5) with

probability higher than $1 - \eta$ (and $B = 1$).

Now, we present results in expectation which relate the L_μ^2 error with the best approximation in L_μ^2 . We have the following result from [24] for a conditional weighted least-squares projection, here stated in a slightly different form.

Theorem 1.3 ([24]). *Let \mathbf{x}^n be drawn from the measure $\boldsymbol{\rho}^n$ and let $Q_{V_m}^{\mathbf{x}^n} u$ be the associated weighted least-squares projection of u . For any $\delta \in [0, 1]$ and $\eta \in [0, 1]$ such that Equation (1.7) holds,*

$$\mathbb{E}(\|u - Q_{V_m}^{\mathbf{x}^n, C} u\|^2) \leq (1 + (1 - \delta)^{-1})\|u - P_{V_m} u\|^2 + \eta\|u\|^2, \quad (1.8)$$

where $Q_{V_m}^{\mathbf{x}^n, C} u = Q_{V_m}^{\mathbf{x}^n} u$ if $Z_{\mathbf{x}^n} \leq \delta$ and 0 otherwise.

Proof. We have

$$\mathbb{E}(\|u - Q_{V_m}^{\mathbf{x}^n, C} u\|^2) = \mathbb{E}(\|u - Q_{V_m}^{\mathbf{x}^n} u\|^2 \mathbb{1}_{Z_{\mathbf{x}^n} \leq \delta}) + \|u\|^2 \mathbb{E}(\mathbb{1}_{Z_{\mathbf{x}^n} > \delta}),$$

with $\mathbb{E}(\mathbb{1}_{Z_{\mathbf{x}^n} > \delta}) = \mathbb{P}(Z_{\mathbf{x}^n} > \delta)$ and therefore according to Equation (1.7) $\mathbb{E}(\mathbb{1}_{Z_{\mathbf{x}^n} > \delta}) \leq \eta$. Then using Lemma 1.1 and Equation (1.6), we have

$$\begin{aligned} \mathbb{E}(\|u - Q_{V_m}^{\mathbf{x}^n} u\|^2 \mathbb{1}_{Z_{\mathbf{x}^n} \leq \delta}) &\leq \mathbb{E}((\|u - P_{V_m} u\|^2 + (1 - \delta)^{-1}\|u - P_{V_m} u\|_{\mathbf{x}^n}^2) \mathbb{1}_{Z_{\mathbf{x}^n} \leq \delta}) \\ &\leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \mathbb{E}(\|u - P_{V_m} u\|_{\mathbf{x}^n}^2) \\ &= (1 + (1 - \delta)^{-1})\|u - P_{V_m} u\|^2, \end{aligned}$$

which concludes the proof. \square

Also, we have the following quasi-optimality property for the weighted least-squares projection associated with the distribution $\boldsymbol{\rho}^n$ conditioned to the event $\{Z_{\mathbf{x}^n} \leq \delta\}$.

Theorem 1.4. *Let \mathbf{x}^n be drawn from the measure $\boldsymbol{\rho}^n$ and let $Q_{V_m}^{\mathbf{x}^n} u$ be the associated weighted least-squares projection of u . For any $\delta \in [0, 1]$ and $\eta \in [0, 1]$ such that Equation (1.7) holds,*

$$\mathbb{E}(\|u - Q_{V_m}^{\mathbf{x}^n} u\|^2 | Z_{\mathbf{x}^n} \leq \delta) \leq (1 + (1 - \delta)^{-1}(1 - \eta)^{-1})\|u - P_{V_m} u\|^2. \quad (1.9)$$

Proof. From Lemma 1.1, we have that

$$\mathbb{E}(\|u - Q_{V_m}^{\mathbf{x}^n} u\|^2 | Z_{\mathbf{x}^n} \leq \delta) \leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \mathbb{E}(\|u - P_{V_m} u\|_{\mathbf{x}^n}^2 | Z_{\mathbf{x}^n} \leq \delta),$$

and

$$\mathbb{E}(\|u - P_{V_m} u\|_{\mathbf{x}^n}^2 | Z_{\mathbf{x}^n} \leq \delta) \leq \mathbb{E}(\|u - P_{V_m} u\|_{\mathbf{x}^n}^2) \mathbb{P}(Z_{\mathbf{x}^n} \leq \delta)^{-1},$$

and we conclude by using $\mathbb{P}(Z_{\mathbf{x}^n} \leq \delta) \geq 1 - \eta$ and the property (1.6). \square

1.3.3 Optimal sampling measure

An inequality of the form (1.7) can be obtained by concentration inequalities. A suitable sampling distribution can then be obtained by an optimization of the obtained upper bound. An optimal choice for w based on matrix Chernoff inequality is derived in [24] and given by

$$w(x)^{-1} = \frac{1}{m} \sum_{j=1}^m \varphi_j(x)^2 = \frac{1}{m} \|\boldsymbol{\varphi}(x)\|_2^2. \quad (1.10)$$

Using this distribution, we obtain the following result, for which we provide a sketch of proof following [24]. The result is here provided in a slightly more general form than in [24].

Theorem 1.5. *Let $\eta \in [0, 1)$ and $\delta \in [0, 1)$. Assume \mathbf{x}^n is drawn from the product measure $\rho^n = \rho^{\otimes n}$, with ρ having the density (1.10) with respect to μ . If the sample size n is such that¹*

$$n \geq n(\delta, \eta, m) := d_\delta^{-1} m \log(2m\eta^{-1}), \quad (1.11)$$

with $d_\delta := -\delta + (1 + \delta) \log(1 + \delta)$, then $Z_{\mathbf{x}^n} = \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2$ satisfies (1.7).

Proof. We have $\mathbf{G}_{\mathbf{x}^n} = \frac{1}{n} \sum_{i=1}^n \mathbf{A}_i$ where the $\mathbf{A}_i = w(x^i) \boldsymbol{\varphi}(x^i) \otimes \boldsymbol{\varphi}(x^i)$ are random matrices such that $\mathbb{E}(\mathbf{A}_i) = \mathbf{I}$ and $\|\mathbf{A}_i\|_2 = w(x^i) \|\boldsymbol{\varphi}(x^i)\|_2^2 = m$. The matrix Chernoff inequality from [91, Theorem 5.1] gives that the minimal and maximal eigenvalues of $\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}$ satisfy

$$\mathbb{P}(\lambda_{\min}(\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}) < -\delta) \vee \mathbb{P}(\lambda_{\max}(\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}) > \delta) \leq m \exp(-nd_\delta/m).$$

Under the condition (1.11), we have that $m \exp(-nd_\delta/m) \leq \eta/2$ and using a union bound, we deduce Equation (1.7). \square

Remark 1.2. Noticing that $d_\delta \leq \delta^2$, then a sufficient condition for satisfying the condition (1.11) is $n \geq \delta^{-2} m \log(2m\eta^{-1})$.

Remark 1.3. The quantile function of $Z_{\mathbf{x}^n}$ is defined for $t \in [0, 1]$ by $F_{Z_{\mathbf{x}^n}}^-(t) = \inf\{\delta : F_{Z_{\mathbf{x}^n}}(\delta) \geq t\}$, where $F_{Z_{\mathbf{x}^n}}$ is the cumulative density function of the random variable $Z_{\mathbf{x}^n}$. For given n and η , $F_{Z_{\mathbf{x}^n}}^-(1 - \eta)$ is the minimal δ such that Equation (1.7) is satisfied. Denoting by $\delta_c(\eta, n) = \min\{\delta : n \geq n(\delta, \eta, m)\}$, we clearly have $F_{Z_{\mathbf{x}^n}}^-(1 - \eta) \leq \delta_c(\eta, n)$. The closer δ_c is from $F_{Z_{\mathbf{x}^n}}^-(1 - \eta)$, the sharper the condition on the sample size n is for satisfying Equation (1.7).

Theorem 1.5 states that using the optimal sampling density from Equation (1.10), a stable projection of u is obtained with a sample size $n = \mathcal{O}(m \log(m\eta^{-1}))$ with high probability. Note that a small probability η , and therefore a large sample size n , may be required for controlling the term $\eta \|u\|^2$ in the error bound (1.8) for the conditional projection, or for obtaining a quasi-optimality

¹Note that the constant in the condition (1.11) differs from the one given in the reference [24] for $\delta = 1/2$, which was incorrect.

property (1.9) in conditional expectation with a quasi-optimality constant close to $1 + (1 - \delta)^{-1}$. This will be improved in the next chapter by proposing a new distribution (obtained by resampling, conditioning and subsampling) allowing to obtain stability of the empirical Gram matrix with a moderate sample size.

BOOSTED OPTIMAL WEIGHTED LEAST-SQUARES

Contents

2.1	Introduction	43
2.2	Boosted optimal weighted least-squares method	44
2.2.1	Resampling and conditioning	45
2.2.2	Subsampling	49
2.3	The noisy case	51
2.4	Numerical experiments	54
2.4.1	Notations and objectives	54
2.4.2	Qualitative analysis of the boosted optimal weighted least-squares method	56
2.4.3	Quantitative analysis for polynomial approximations	59
2.4.4	A noisy example	68
2.4.5	Overall conclusion for all examples	69
2.5	Conclusion	70

2.1 Introduction

This chapter is based on the article [52]. We begin by recalling some notions and notations already introduced in the previous chapter but necessary for a self-contained presentation of this chapter.

Here the goal is to construct the approximation of a function u onto a linear space V_m using evaluations of the function at suitably chosen points. We consider a weighted least-squares projection, obtained by solving

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n w(x^i) (u(x^i) - v(x^i))^2,$$

where the x^i are points sampled according to a measure constructed from the optimal measure ρ introduced by [24] and recalled in the subsection 1.3.3 from Chapter 1, and where $w(x^i)$ are the corresponding weights.

Here we introduce a boosted least-squares method which enables us to ensure almost surely some stability property of the weighted least-squares projection (i.e of the Gram matrix or of the $L_\mu^\infty \rightarrow L_\mu^2$ projection) with a sample size close to an interpolation method (i.e $n = m$). It consists in sampling according to a measure associated with the optimization of a stability criterion over a collection of independent n -samples, and resampling according to this measure until a stability condition is satisfied. A greedy method is then proposed to remove points from the obtained sample. Quasi-optimality properties in expectation are obtained for the weighted least-squares projection, with or without the greedy procedure.

If the observations are polluted by a noise, here modeled by a random variable e , then the weighted least-squares projection is defined as the solution of

$$\min_{v \in V_m} \frac{1}{n} \sum_{i=1}^n w(x^i) (y^i - v(x^i))^2,$$

where $y^i = u(x^i) + e^i$, with $\{e^i\}_{i=1}^n$ i.i.d realizations of the random variable e . Quasi-optimality property is lost in the case of noisy observations, because of an additional error term due to the noise. This latter error term can however be reduced by increasing n .

The outline of this chapter is as follows. In Section 2.2, we present the boosted least-squares method and analyze it in the noise-free case. The theoretical results are extended to the noisy case in Section 2.3. In Section 2.4, we present numerical examples.

2.2 Boosted optimal weighted least-squares method

Let \mathcal{X} be a subset of \mathbb{R}^d equipped with a probability measure μ , with $d \geq 1$. We consider a function u from $L_\mu^2(\mathcal{X})$, the Hilbert space of square-integrable real-valued functions defined on \mathcal{X} . Let V_m be a m -dimensional subspace of L_μ^2 , with $m \geq 1$, and $\{\varphi_j\}_{j=1}^m$ be an orthonormal basis of V_m . We also denote by

$$\boldsymbol{\varphi} = (\varphi_1, \dots, \varphi_m) : \mathcal{X} \rightarrow \mathbb{R}^m$$

the m -dimensional vector-valued function such that $\boldsymbol{\varphi}(x) = (\varphi_1(x), \dots, \varphi_m(x))^T$.

In this chapter, we propose an improved weighted least-squares method by proposing distributions over \mathcal{X}^n having better properties than $\boldsymbol{\rho}^n = \rho^{\otimes n}$, where ρ is the optimal weighted least-squares

measure from [24] (see Subsection 1.3.3 from Chapter 1). The function w defining the weighted least-squares projections will always be taken such that w^{-1} is the density of the optimal sampling measure ρ with respect to the reference measure μ , defined by

$$w(x)^{-1} = \frac{1}{m} \sum_{j=1}^m \varphi_j(x)^2 = \frac{1}{m} \|\varphi(x)\|_2^2. \quad (2.1)$$

2.2.1 Resampling and conditioning

The first improvement we propose consists in drawing M independent samples $\{\mathbf{x}^{n,i}\}_{i=1}^M$, with $\mathbf{x}^{n,i} = (x^{1,i}, \dots, x^{n,i})$, from the distribution ρ^n , and then in selecting a sample $\mathbf{x}^{n,\star}$ which satisfies

$$\|\mathbf{G}_{\mathbf{x}^{n,\star}} - \mathbf{I}\|_2 = \min_{1 \leq i \leq M} \|\mathbf{G}_{\mathbf{x}^{n,i}} - \mathbf{I}\|_2,$$

where $\mathbf{G}_{\mathbf{x}}$ denotes the empirical Gram matrix associated with a sample \mathbf{x} in \mathcal{X}^n . If several samples $\mathbf{x}^{n,i}$ are solutions of the minimization problem, $\mathbf{x}^{n,\star}$ is selected at random among the minimizers. We denote by $\rho^{n,\star}$ the probability measure of $\mathbf{x}^{n,\star}$. The probability that the stability condition $Z_{\mathbf{x}^{n,\star}} = \|\mathbf{G}_{\mathbf{x}^{n,\star}} - \mathbf{I}\|_2 \leq \delta$ is verified can be made arbitrarily high, playing on M , as it is shown in the following lemma (whose proof is trivial).

Lemma 2.1. *For any $\delta \in [0, 1)$ and $\eta \in (0, 1)$, if $n \geq n(\delta, \eta, m) := d_\delta^{-1} m \log(2m\eta^{-1})$, with $d_\delta := -\delta + (1 + \delta) \log(1 + \delta)$ (the condition from (1.11)), then*

$$\mathbb{P}(Z_{\mathbf{x}^{n,\star}} \leq \delta) \geq 1 - \eta^M.$$

Therefore, we can choose a probability η arbitrary close to 1, so that the condition on the number of samples from Lemma 2.1 requires a smaller sample size n , and still obtain the stability condition with a probability at least $1 - \eta^M$. This latter can be made arbitrarily close to 1 by choosing a sufficiently large M . Even if ρ^n has a product structure, for $M > 1$, the distribution $\rho^{n,\star}$ does not have a product structure, i.e. the components of $\mathbf{x}^{n,\star} = (x^{1,\star}, \dots, x^{n,\star})$ are not independent, and does not satisfy the assumptions of Theorems 1.3 and 1.4. In particular $\mathbb{E}(\mathbf{G}_{\mathbf{x}^{n,\star}})$ may not be equal to \mathbf{I} and in general, $\mathbb{E}(\|v\|_{\mathbf{x}^{n,\star}}^2) \neq \|v\|^2$ for an arbitrary function v when $M > 1$. Therefore, a new analysis of the resulting weighted least-squares projection is required.

Remark 2.1. Note that since the function $\mathbf{x} \mapsto \|\mathbf{G}_{\mathbf{x}} - \mathbf{I}\|_2$ defined on \mathcal{X}^n is invariant through permutations of the components of \mathbf{x} , we have that the components of $\mathbf{x}^{n,\star}$ have the same marginal distribution.

In order to ensure that the stability property is verified almost surely, we consider a sample $\tilde{\mathbf{x}}^n$ of $\mathbf{x}^{n,\star}$ conditioned on the event

$$A_\delta = \{\|\mathbf{G}_{\mathbf{x}^{n,\star}} - \mathbf{I}\|_2 \leq \delta\}, \quad (2.2)$$

which is such that for any function f , $\mathbb{E}(f(\tilde{\mathbf{x}}^n)) = \mathbb{E}(f(\mathbf{x}^{n,*})|A_\delta)$. Let $\tilde{\rho}^n$ be the distribution of $\tilde{\mathbf{x}}^n$. A sample $\tilde{\mathbf{x}}^n$ from the distribution $\tilde{\rho}^n$ is obtained by a simple rejection method, which consists in drawing samples $\mathbf{x}^{n,*}$ from the distribution $\rho^{n,*}$ until A_δ is satisfied. It follows that $\mathbb{P}(Z_{\tilde{\mathbf{x}}^n} \leq \delta) = 1$.

Remark 2.2. Let J be the number of trials necessary to get a sample $\mathbf{x}^{n,*}$ verifying the stability condition A_δ . This random variable J follows a geometric distribution with a probability of success $\mathbb{P}(A_\delta)$. Therefore J is almost surely finite and

$$\mathbb{P}(J \geq k) = (1 - \mathbb{P}(A_\delta))^k,$$

i.e. the probability to have J greater than k decreases exponentially with k . An other property of the geometric distribution is that $\mathbb{E}(J) = \frac{1}{\mathbb{P}(A_\delta)} \leq \frac{1}{1-\eta^M}$ (as $\mathbb{P}(A_\delta) \geq 1 - \eta^M$), such that the average number of trials increases when η tends to 1. In particular for $M = 1$, we have $\mathbb{E}(J) = 2$ for $\eta = 0.5$, and $\mathbb{E}(J) = 100$ for $\eta = 0.99$. $\mathbb{P}(A_\delta)$ increases with M and therefore $\mathbb{E}(J)$ decreases with M .

Now we provide a result on the distribution of $\tilde{\mathbf{x}}^n$ which will be later used for the analysis of the corresponding least-squares projection.

Lemma 2.2. *Let $\tilde{\mathbf{x}}^n$ be a sample following the distribution $\tilde{\rho}^n$, which is the distribution $\rho^{n,*}$ conditioned on the event A_δ defined by (2.2). Assume that $n \geq n(\delta, \eta, m)$ for some $\eta \in (0, 1)$ and $\delta \in (0, 1)$. Then for any function v in L_μ^2 and any $0 < \varepsilon \leq 1$,*

$$\mathbb{E}(\|v\|_{\tilde{\mathbf{x}}^n}^2) \leq C(\varepsilon, M)(1 - \eta^M)^{-1} \mathbb{E}(\|v\|_{\mathbf{x}^n}^{2/\varepsilon})^\varepsilon, \quad (2.3)$$

with

$$C(\varepsilon, M) = M \frac{(1 - \varepsilon)^{1-\varepsilon}}{(M - \varepsilon)^{1-\varepsilon}} \leq M.$$

In particular, for $\varepsilon = 1$,

$$\mathbb{E}(\|v\|_{\tilde{\mathbf{x}}^n}^2) \leq M(1 - \eta^M)^{-1} \|v\|^2. \quad (2.4)$$

Also, if $\|v\|_{\infty, w} = \sup_{x \in \mathcal{X}} w(x)^{1/2} |v(x)| < \infty$,

$$\mathbb{E}(\|v\|_{\tilde{\mathbf{x}}^n}^2) \leq C(\varepsilon, M)(1 - \eta^M)^{-1} \|v\|_{\infty, w}^{2-2\varepsilon} \|v\|^{2\varepsilon}. \quad (2.5)$$

Proof. See appendix. □

Corollary 2.3. *Let $\tilde{\mathbf{x}}^n$ be a sample following the distribution $\tilde{\rho}^n$ and assume that $n \geq n(\delta, \eta, m)$ for some $\eta \in (0, 1)$ and $\delta \in (0, 1)$. For any $v \in L_\mu^2$, the weighted least-squares projection $Q_{V_m}^{\tilde{\mathbf{x}}^n} v$ associated with the sample $\tilde{\mathbf{x}}^n$ satisfies*

$$\mathbb{E}(\|Q_{V_m}^{\tilde{\mathbf{x}}^n} v\|^2) \leq (1 - \delta)^{-1} M(1 - \eta^M)^{-1} \|v\|^2.$$

Proof. Since $Q_{V_m}^{\tilde{\mathbf{x}}^n} v \in V_m$, we have that

$$\|Q_{V_m}^{\tilde{\mathbf{x}}^n} v\|^2 \leq (1 - \delta)^{-1} \|Q_{V_m}^{\tilde{\mathbf{x}}^n} v\|_{\tilde{\mathbf{x}}^n}^2 \leq (1 - \delta)^{-1} \|v\|_{\tilde{\mathbf{x}}^n}^2,$$

where we have used the fact that $Q_{V_m}^{\tilde{\mathbf{x}}^n}$ is an orthogonal projection with respect to the semi-norm $\|\cdot\|_{\tilde{\mathbf{x}}^n}$. Taking the expectation and using (2.4), we obtain

$$\mathbb{E}(\|Q_{V_m}^{\tilde{\mathbf{x}}^n} v\|^2) \leq (1 - \delta)^{-1} M(1 - \eta^M)^{-1} \|v\|^2.$$

□

Theorem 2.4. *Let $\tilde{\mathbf{x}}^n$ be a sample following the distribution $\tilde{\rho}^n$ and assume that $n \geq n(\delta, \eta, m)$ for some $\eta \in (0, 1)$ and $\delta \in (0, 1)$. The weighted least-squares projection $Q_{V_m}^{\tilde{\mathbf{x}}^n} u$ associated with the sample $\tilde{\mathbf{x}}^n$ satisfies the quasi-optimality property*

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}^n} u\|^2) \leq (1 + (1 - \delta)^{-1}(1 - \eta^M)^{-1}M) \|u - P_{V_m} u\|^2. \quad (2.6)$$

Also, assuming $\|u\|_{\infty, w} \leq L$, we have

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}^n} u\|^2) \leq \left(\|u - P_{V_m} u\|^2 + (1 - \delta)^{-1}(1 - \eta^M)^{-1}D(M, L, m, \|u - P_{V_m} u\|^2) \right) \quad (2.7)$$

where for all $\alpha \geq 0$, $D(M, L, m, \alpha) = \inf_{0 < \varepsilon \leq 1} \tilde{D}(M, L, m, \alpha, \varepsilon)$, with

$$\tilde{D}(M, L, m, \alpha, \varepsilon) := C(\varepsilon, M)(L(1 + c_m))^{2-2\varepsilon} \alpha^\varepsilon.$$

Here, $C(\varepsilon, M)$ is the constant defined in Lemma 2.2 and c_m the supremum of $\|P_{V_m} v\|_{\infty, w}$ over functions v such that $\|v\|_{\infty, w} \leq 1$.

Proof. From Lemma 1.1, we have that

$$\|u - Q_{V_m}^{\tilde{\mathbf{x}}^n} u\|^2 \leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \|u - P_{V_m} u\|_{\tilde{\mathbf{x}}^n}^2$$

holds almost surely, and from Lemma 2.2, we have that

$$\mathbb{E}(\|u - P_{V_m} u\|_{\tilde{\mathbf{x}}^n}^2) \leq C(\varepsilon, M)(1 - \eta^M)^{-1} \mathbb{E}(\|u - P_{V_m} u\|_{\tilde{\mathbf{x}}^n}^{2/\varepsilon})^\varepsilon$$

for all $\varepsilon \in (0, 1]$. Combining the above inequalities and then taking the infimum over ε , we obtain

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}^n} u\|^2) \leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1}(1 - \eta^M)^{-1} \inf_{0 < \varepsilon \leq 1} C(\varepsilon, M) \mathbb{E} \left(\|u - P_{V_m} u\|_{\tilde{\mathbf{x}}^n}^{2/\varepsilon} \right)^\varepsilon. \quad (2.8)$$

The particular case $\varepsilon = 1$ yields Equation (2.6). The second property (2.7) is simply deduced from

Equation (2.8) by using the property (2.5) of Lemma 2.2 and by noting that $\|u - P_{V_m}u\|_{\infty,w} \leq (1 + c_m)\|u\|_{\infty,w}$. \square

Remark 2.3. The constant c_m in Theorem 2.4 is such that $c_m \leq m$. Indeed, $P_{V_m}v(x) = \sum_{i=1}^m a_i \varphi_i(x)$ with

$$a_i = (v, \varphi_i) = \int_{\mathcal{X}} v(x) \varphi_i(x) d\mu(x) = \int_{\mathcal{X}} v(x) \varphi_i(x) w(x) d\rho(x),$$

so that

$$|a_i| \leq \|v\|_{\infty,w} \int_{\mathcal{X}} |\varphi_i(x)| w(x)^{1/2} d\rho(x) \leq \|v\|_{\infty,w} \left(\int_{\mathcal{X}} \varphi_i(x)^2 w(x) d\rho(x) \right)^{1/2} = \|v\|_{\infty,w},$$

where we have used Cauchy-Schwarz inequality. Therefore,

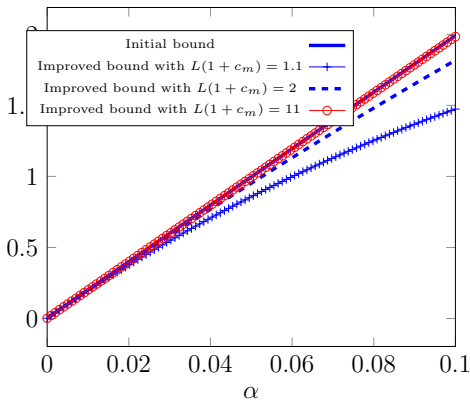
$$\begin{aligned} \|P_{V_m}v\|_{\infty,w} &\leq \|v\|_{\infty,w} \sup_{x \in \mathcal{X}} w(x)^{1/2} \sum_{i=1}^m |\varphi_i(x)| \\ &\leq \|v\|_{\infty,w} \sup_{x \in \mathcal{X}} w(x)^{1/2} m^{1/2} \left(\sum_{i=1}^m \varphi_i(x)^2 \right)^{1/2} = m \|v\|_{\infty,w}, \end{aligned}$$

thanks to the definition of w (see Equation (2.1)).

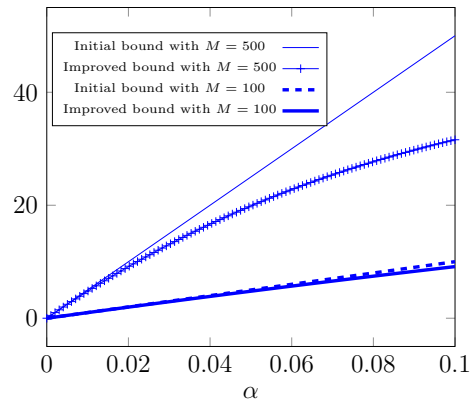
Remark 2.4. About the constant $D(L, M, m, \alpha)$.

The value of ε that minimizes $\tilde{D}(M, L, m, \alpha, \varepsilon)$ has no explicit expression, however numerical estimations can be calculated. Figure 2.1 illustrates the fact that $D(M, L, m, \|u - P_{V_m}u\|^2)$ may be lower than $M\|u - P_{V_m}u\|^2$ for some conditions on M, L, m meaning that (2.7) should provide a sharper result than (2.6). The legend "Initial bound" refers to the bound presented in Equation in (2.6), and the legend "Improved bound" refers to the bound presented in Equation (2.7).

On the two Figures 2.1a and 2.1b, the x -axis represents the best approximation error, so that for



(a) $M = 100, \delta = 0.5, \eta = 0.01^{1/M}$



(b) $L(1 + c_m) = 2, \delta = 0.5, \eta = 0.01^{1/M}$

Figure 2.1 – Improvement of the bound for different values of $L(c_m + 1)$ and M .

a given $L = \|u\|_{\infty, w}$ and a given $c_m = \max_{v \in V_m} \frac{\|P_{V_m} v\|_{\infty, w}}{\|v\|_{\infty, w}}$, the left part of the curve corresponds to functions u which can be well approximated in V_m whereas on the contrary, the right part of the curve corresponds to functions which are not well approximated in V_m (having a high best approximation error in V_m). We observe that the bound from (2.7) improves the bound from (2.6) when M is high ($M \geq 100$), see Figure 2.1b and when $L(1 + c_m)$ is small ($L(1 + c_m) \leq 1.1$), see Figure 2.1a. The details of the calculations to obtain these graphs are given in the Appendix of this chapter.

2.2.2 Subsampling

For a given δ , without resampling, the number of samples necessary to guarantee the stability with high probability η has to be greater than $d_\delta^{-1} m \log(2m\eta^{-1})$. When resampling M times, only $d_\delta^{-1} m \log(2m\eta^{-1/M})$ samples are necessary to guarantee the stability with the same probability η . Although the resampling enables us to choose δ and η such that n is smaller than with the initial strategy from [24], the value of n may still be high compared to an interpolation method. Therefore, to further decrease the sample size, for each generated sample $\tilde{\mathbf{x}}^n$, we propose to select a subsample which still verifies the stability condition.

We start with a sample $\tilde{\mathbf{x}}^n = (\tilde{x}^1, \dots, \tilde{x}^n)$ satisfying $\|\mathbf{G}_{\tilde{\mathbf{x}}^n} - \mathbf{I}\|_2 \leq \delta$ and then select a subsample $\tilde{\mathbf{x}}_K^n = (\tilde{x}^k)_{k \in K}$ with $K \subset \{1, \dots, n\}$ such that the empirical Gram matrix defined by $\mathbf{G}_{\tilde{\mathbf{x}}_K^n} = \frac{1}{\#K} \sum_{k \in K} w(\tilde{x}^k) \varphi(\tilde{x}^k) \otimes \varphi(\tilde{x}^k)$ still satisfies

$$\|\mathbf{G}_{\tilde{\mathbf{x}}_K^n} - \mathbf{I}\|_2 \leq \delta.$$

In practice, the set K is constructed by a greedy procedure. We start with $K = \{1, \dots, n\}$. Then at each step of the greedy procedure, we select k^* in K such that

$$\|\mathbf{G}_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n} - \mathbf{I}\|_2 = \min_{k \in K} \|\mathbf{G}_{\tilde{\mathbf{x}}_{K \setminus \{k\}}^n} - \mathbf{I}\|_2.$$

If $\|\mathbf{G}_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n} - \mathbf{I}\|_2 \leq \delta$ and $\#K > n_{\min}$ then k^* is removed from K . Otherwise, the algorithm is stopped. We denote by $\tilde{\rho}_g^n$ the distribution of the sample $\tilde{\mathbf{x}}_K^n$ produced by this greedy algorithm.

Theorem 2.5. Assume $n \geq n(\delta, \eta, m)$ for some $\eta \in (0, 1)$ and $\delta \in (0, 1)$, and let $\tilde{\mathbf{x}}_K^n$ be a sample produced by the greedy algorithm with $\#K \geq n_{\min}$. The weighted least-squares projection $Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u$ associated with the sample $\tilde{\mathbf{x}}_K^n$ satisfies the quasi-optimality property

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2) \leq (1 + \frac{n}{n_{\min}} (1 - \delta)^{-1} (1 - \eta^M)^{-1} M) \|u - P_{V_m} u\|^2. \quad (2.9)$$

Also, assuming $\|u\|_{\infty, w} \leq L$, we have

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2) \leq \|u - P_{V_m} u\|^2 + \frac{n}{n_{\min}} (1 - \delta)^{-1} (1 - \eta^M)^{-1} D(M, L, m, \|u - P_{V_m} u\|^2) \quad (2.10)$$

where $D(M, L, m, \|u - P_{V_m} u\|^2)$ is defined in Theorem 2.4.

Proof. Since $Z_{\tilde{\mathbf{x}}_K^n} \leq \delta$, from Lemma 1.1, we have that for any $v \in V_m$, the least-squares projection associated with $\tilde{\mathbf{x}}_K^n$ satisfies

$$\begin{aligned} \|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2 &\leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \|u - P_{V_m} u\|_{\tilde{\mathbf{x}}_K^n}^2 \\ &\leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \frac{n}{\#K} \|u - P_{V_m} u\|_{\tilde{\mathbf{x}}^n}^2, \end{aligned}$$

where the second inequality simply results from

$$\|v\|_{\tilde{\mathbf{x}}_K^n}^2 = \frac{1}{\#K} \sum_{k \in K} w(\tilde{x}^k) v(\tilde{x}^k)^2 \leq \frac{1}{\#K} \sum_{k=1}^n w(\tilde{x}^k) v(\tilde{x}^k)^2 = \frac{n}{\#K} \|v\|_{\tilde{\mathbf{x}}^n}^2.$$

Therefore, since $\#K \geq n_{\min}$, we obtain from Lemma 2.2 that

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2) \leq \|u - P_{V_m} u\|^2 + \frac{n}{n_{\min}} (1 - \delta)^{-1} (1 - \eta^M)^{-1} \inf_{0 < \varepsilon \leq 1} C(\varepsilon, M) \mathbb{E} \left(\|u - P_{V_m} u\|_{\tilde{\mathbf{x}}^n}^{\frac{2}{\varepsilon}} \right)^\varepsilon.$$

The particular case $\varepsilon = 1$ yields the first property. For the second property, the proof follows the one of the property (2.7) in Theorem 2.4. \square

Corollary 2.6. Assume $n \geq n(\delta, \eta, m)$ for some $\eta \in (0, 1)$ and $\delta \in (0, 1)$, and let $\tilde{\mathbf{x}}_K^n$ be a sample produced by the greedy algorithm with $\#K \geq n_{\min}$. The weighted least-squares projection $Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u$ associated with the sample $\tilde{\mathbf{x}}_K^n$ satisfies

$$\mathbb{E}(\|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} v\|^2) \leq (1 - \delta)^{-1} M (1 - \eta^M)^{-1} \frac{n}{n_{\min}} \|v\|^2.$$

Proof. Since $Q_{V_m}^{\tilde{\mathbf{x}}_K^n} v \in V_m$, we have that

$$\|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} v\|^2 \leq (1 - \delta)^{-1} \|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} v\|_{\tilde{\mathbf{x}}_K^n}^2 \leq (1 - \delta)^{-1} \|v\|_{\tilde{\mathbf{x}}_K^n}^2 \leq (1 - \delta)^{-1} \frac{n}{\#K} \|v\|_{\tilde{\mathbf{x}}^n}^2,$$

where we have used the fact that $Q_{V_m}^{\tilde{\mathbf{x}}_K^n}$ is an orthogonal projection with respect to the semi-norm $\|\cdot\|_{\tilde{\mathbf{x}}_K^n}$ and the fact that $\|v\|_{\tilde{\mathbf{x}}_K^n}^2 \leq \frac{n}{\#K} \|v\|_{\tilde{\mathbf{x}}^n}^2$. Taking the expectation, using Equation (2.4) and assuming that $\#K \leq n_{\min}$, we obtain

$$\mathbb{E}(\|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} v\|^2) \leq (1 - \delta)^{-1} M (1 - \eta^M)^{-1} \frac{n}{n_{\min}} \|v\|^2.$$

□

If we set $n_{\min} = m$, it may happen that the algorithm runs until $\#K = m$, the dimension of the approximation space. Choosing $n \geq n(\delta, \eta, m)$ then yields a quasi-optimality constant depending on $\log(m)$. It has to be compared with the optimal behaviour of the Lebesgue constant for polynomial interpolation in one dimension. If we choose $n_{\min} = n/\beta$ for some fixed $\beta \geq 1$ independent of m , then we have $\frac{n}{n_{\min}} \leq \beta$ and a quasi-optimality constant independent of m in Theorem 2.5, but the algorithm may stop before reaching the interpolation regime ($n = m$).

Remark 2.5. Concerning the greedy subsampling, a direct approach to remove a point is to calculate the norm of $\|\mathbf{G}_{\tilde{\mathbf{x}}_{K \setminus \{k\}}^n} - \mathbf{I}\|_2$ for each $k \in K$. However, it involves to calculate this norm for $\#K$ points and this each time a point is removed. In the appendix A of this thesis, we present a method which enables us to choose k^* by solving one eigenproblem and then performing simple matrix-vector multiplications. Indeed, knowing the eigenvalues of a symmetric matrix, there exists bounds on the eigenvalues of a rank-one update of this matrix (see [17] and [44] for more details, as well as the more recent results from [13] that we use in practice).

2.3 The noisy case

We here consider the case where the observations are polluted with a noise, which is modeled by a random variable e . More precisely the observed data take the form

$$y^i = u(\tilde{x}^i) + e^i,$$

where $\{e^i\}_{i \in K}$ are i.i.d realizations of the random variable e and $\{\tilde{x}^i\}_{i \in K} = \tilde{\mathbf{x}}_K^n$ are the points built with the boosted least-squares method. We assume the noise is independent of $\tilde{\mathbf{x}}^n$ and K and centered, i.e. $\mathbb{E}(e) = 0$, and with bounded variance $\sigma^2 = \mathbb{E}(|e|^2) < \infty$. More general cases could be considered as in [22] or [71].

The weighted discrete least-squares projection of u over V_m is defined by

$$u_{\tilde{\mathbf{x}}_K^n} := \arg \min_{v \in V_m} \frac{1}{\#K} \sum_{i \in K} w(\tilde{x}^i) (y^i - v(\tilde{x}^i))^2.$$

Theorem 2.7. *Assume $n \geq n(\delta, \eta, m)$ for some $\eta \in (0, 1)$ and $\delta \in (0, 1)$, and let $\tilde{\mathbf{x}}_K^n$ be a sample produced by the greedy algorithm with $\#K \geq n_{\min}$. The weighted least-squares projection $u_{\tilde{\mathbf{x}}_K^n}$ associated with the sample $\tilde{\mathbf{x}}_K^n$ and the data affected by the noise e , satisfies*

$$\begin{aligned} \mathbb{E}(\|u - u_{\tilde{\mathbf{x}}_K^n}\|^2) &\leq \left(1 + \frac{2n}{n_{\min}}(1 - \delta)^{-1}(1 - \eta^M)^{-1}M\right) \|u - P_{V_m}u\|^2 \\ &\quad + \frac{2\sigma^2 mn}{n_{\min}^2}(1 - \delta)^{-1}(1 - \eta^M)^{-1}M. \end{aligned} \tag{2.11}$$

Proof. Thanks to the Pythagorean equality, it holds

$$\begin{aligned}\|u - u^{\tilde{\mathbf{x}}_K^n}\|^2 &= \|u - P_{V_m} u\|^2 + \|P_{V_m} u - u^{\tilde{\mathbf{x}}_K^n}\|^2 \\ &= \|u - P_{V_m} u\|^2 + \|Q_{V_m}^{\tilde{\mathbf{x}}_K^n}(P_{V_m} u - u) + Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u - u^{\tilde{\mathbf{x}}_K^n}\|^2\end{aligned}$$

where $Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u$ is the boosted least-squares projection of the noiseless evaluations of u over V_m . Then using the triangular inequality,

$$\|u - u^{\tilde{\mathbf{x}}_K^n}\|^2 \leq \|u - P_{V_m} u\|^2 + 2\|Q_{V_m}^{\tilde{\mathbf{x}}_K^n}(P_{V_m} u - u)\|^2 + 2\|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u - u^{\tilde{\mathbf{x}}_K^n}\|^2.$$

Taking the expectation and using Corollary 2.6, it comes

$$\begin{aligned}\mathbb{E}(\|u - u^{\tilde{\mathbf{x}}_K^n}\|^2) &\leq (1 + 2\frac{n}{n_{min}}(1 - \delta)^{-1}M(1 - \eta^M)^{-1})\|u - P_{V_m} u\|^2 \\ &\quad + 2\mathbb{E}(\|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u - u^{\tilde{\mathbf{x}}_K^n}\|^2).\end{aligned}\tag{2.12}$$

Then, we note that

$$\|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u - u^{\tilde{\mathbf{x}}_K^n}\|^2 \leq \sum_{k=1}^m |b_k|^2$$

where $\mathbf{b} = (b_k)_{k=1}^m$ is solution to

$$\mathbf{G}_{\tilde{\mathbf{x}}_K^n} \mathbf{b} = \boldsymbol{\beta}, \quad \boldsymbol{\beta} := \left(\frac{1}{\#K} \sum_{i \in K} e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) \right)_{1 \leq k \leq m}.$$

Since $\|\mathbf{G}_{\tilde{\mathbf{x}}_K^n}^{-1}\|_2^2 \leq (1 - \delta)^{-1}$ it holds

$$\sum_{k=1}^m |b_k|^2 \leq (1 - \delta)^{-1} \sum_{k=1}^m |\beta_k|^2$$

and

$$\sum_{k=1}^m |\beta_k|^2 = \sum_{k=1}^m \frac{1}{(\#K)^2} \sum_{i \in K} \sum_{j \in K} e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) e^j w(\tilde{x}^j) \varphi_k(\tilde{x}^j).$$

We have,

$$\begin{aligned}&\mathbb{E} \left(\frac{1}{\#K^2} \sum_{i \in K} \sum_{j \in K} e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) e^j w(\tilde{x}^j) \varphi_k(\tilde{x}^j) \right) = \\ &\mathbb{E} \left(\frac{1}{\#K^2} \sum_{i \in K} \sum_{j \in K} e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) e^j w(\tilde{x}^j) \varphi_k(\tilde{x}^j) | K \right) = \\ &\mathbb{E} \left(\frac{1}{\#K^2} \sum_{i \in K} \sum_{j \in K} \mathbb{E}(e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) e^j w(\tilde{x}^j) \varphi_k(\tilde{x}^j) | K) \right).\end{aligned}$$

By construction K is independent from the noise,

$$\mathbb{E}(e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) e^j w(\tilde{x}^j) \varphi_k(\tilde{x}^j) | K) = \mathbb{E}(e^i e^j) \mathbb{E}(w(\tilde{x}^i) \varphi_k(\tilde{x}^i) w(\tilde{x}^j) \varphi_k(\tilde{x}^j) | K).$$

Therefore, for $i \neq j$

$$\mathbb{E}(e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) e^j w(\tilde{x}^j) \varphi_k(\tilde{x}^j) | K) = 0$$

and for $i = j$

$$\mathbb{E}(e^i w(\tilde{x}^i) \varphi_k(\tilde{x}^i) e^j w(\tilde{x}^j) \varphi_k(\tilde{x}^j) | K) = \sigma^2 \mathbb{E}(w(\tilde{x}^i)^2 \varphi_k(\tilde{x}^i)^2 | K).$$

Then using $\#K \geq n_{min}$,

$$\begin{aligned} \sum_{k=1}^m \mathbb{E}(|\beta_k|^2) &= \sigma^2 \mathbb{E} \left(\frac{1}{\#K^2} \sum_{i \in K} w(\tilde{x}^i)^2 \sum_{k=1}^m \varphi_k(\tilde{x}^i)^2 \right) \\ &= m \sigma^2 \mathbb{E} \left(\frac{1}{\#K^2} \sum_{i \in K} w(\tilde{x}^i)^2 \right) \text{ by definition of } w, \text{ see Equation (2.1)} \\ &\leq \frac{m}{n_{min}^2} \sigma^2 \mathbb{E} \left(\sum_{i=1}^n w(\tilde{x}^i)^2 \right). \end{aligned}$$

To bound the term $\mathbb{E}(\sum_{i=1}^n w(\tilde{x}^i)^2)$, we use Equation (2.4) with $v = 1$,

$$\mathbb{E} \left(\sum_{i=1}^n w(\tilde{x}^i)^2 \right) \leq n \mathbb{E} \left(\|1\|_{x^n}^2 \right) \leq n M (1 - \eta^M)^{-1} \|1\|^2 = n M (1 - \eta^M)^{-1}.$$

All in all,

$$\sum_{k=1}^m \mathbb{E}(|\beta_k|^2) \leq \sigma^2 \frac{mn}{n_{min}^2} (1 - \eta^M)^{-1} M.$$

□

When there is no subsampling ($n_{min} = n$) the bound from Equation (2.11) becomes

$$\begin{aligned} \mathbb{E}(\|u - u^{\tilde{x}_K^n}\|^2) &\leq (1 + (1 - \delta)^{-1} (1 - \eta^M)^{-1} M) \|u - P_{V_m} u\|^2 \\ &\quad + \frac{2\sigma^2 m}{n} (1 - \delta)^{-1} (1 - \eta^M)^{-1} M. \end{aligned} \tag{2.13}$$

When $n_{min} = m$ (allowing subsampling to reach $\#K = m$), the bound becomes

$$\begin{aligned} \mathbb{E}(\|u - u^{\tilde{x}_K^n}\|^2) &\leq (1 + \frac{2n}{m} (1 - \delta)^{-1} (1 - \eta^M)^{-1} M) \|u - P_{V_m} u\|^2 \\ &\quad + 2\sigma^2 \frac{n}{m} (1 - \delta)^{-1} (1 - \eta^M)^{-1} M. \end{aligned} \tag{2.14}$$

In this particular case the influence of the noise may be more important, as for an interpolation

method and there is no guarantee for convergence. Then in the noisy case, using $n_{min} = \frac{n}{\beta}$ for some fixed $\beta > 1$ allows to better control the noise term.

2.4 Numerical experiments

2.4.1 Notations and objectives

In this section, we focus on polynomial approximation spaces V_m . The aim is to compare the performance of the method we propose with the optimal weighted least-squares method and interpolation (see Chapter 1). We are not trying to be exhaustive in this comparison, but to cover a quite large panel of state-of-the art approximation methods.

First, we consider interpolation performed on deterministic set of points (Gauss-Hermite points for a Gaussian measure, abbreviated **I-GaussH** and Gauss-Legendre points for a uniform measure, abbreviated **I-GaussL**), magic points, abbreviated **I-Magic**, (see [65]), Leja points, abbreviated **I-Leja**, (see [9],[15] and [16] for their weighted version dealing with unbounded domains), and Fekete points, abbreviated **I-Fekete**, (see [86] and [51] for their weighted version dealing with unbounded domains). The three last sets of points are chosen among a large set of points in \mathcal{X} .

Then, we consider least-squares methods, more precisely standard least-squares methods, abbreviated **SLS**, optimal weighted least-squares projection (introduced in [24]), abbreviated **OWLS**, and also the boosted optimal weighted least-squares projections we propose, abbreviated **BLS**, **c-BLS** and **s-BLS** when we respectively use resampling, conditioning, and subsampling plus resampling and conditioning. Algorithms 2.1, 2.2 and 2.3 present respectively the algorithms to generate the sample for **BLS**, **c-BLS** and **s-BLS**.

Algorithm 2.1 Presentation of the **BLS** sampling

Inputs: δ, η, M, V_m .

Outputs: $\mathbf{x}^{n,*}$ for the **BLS** sampling.

for $i = 1, \dots, M$ **do**

 Sample $\mathbf{x}^{n,i} \sim \rho^n$

end for

 Select at random $I^* \in \arg \min_{1 \leq i \leq M} \|\mathbf{G}_{\mathbf{x}^{n,i}} - \mathbf{I}\|_2$.

 Set $\mathbf{x}^{n,*} = \mathbf{x}^{n,I^*}$

Remark 2.6. For a fixed approximation space V_m , it must be noticed that the methods **OWLS**, **BLS**, **I-GaussH**, **I-GaussL** or **I-Leja** points, do not depend on the choice of the orthonormal basis associated with V_m , as the quantity $Z_{\mathbf{x}^n}$ is independent of this choice. This is however not the case for the methods **I-Magic** [65] or **I-Fekete** [86], for which the particular choice of the basis will be mentioned in each example.

Remark 2.7. The difficulty of the optimal least-squares methods lies in the fact that we sample from a standard probability density function (PDF). We need a sampling technique that is both accurate

Algorithm 2.2 Presentation of the **c-BLS** sampling

Inputs: δ, η, M, V_m .**Outputs:** $\tilde{\mathbf{x}}^n$ for the **c-BLS** sampling.

```

 $z = \infty$ 
while  $z > \delta$  do
  for  $i = 1, \dots, M$  do
    Sample  $\mathbf{x}^{n,i} \sim \rho^n$ 
  end for
  Select at random  $I^* \in \arg \min_{1 \leq i \leq M} \|\mathbf{G}_{\mathbf{x}^{n,i}} - \mathbf{I}\|_2$ .
  Set  $\mathbf{x}^{n,*} = \mathbf{x}^{n,I^*}$  and set  $z = \|\mathbf{G}_{\mathbf{x}^{n,*}} - \mathbf{I}\|_2$ 
end while
Set  $\tilde{\mathbf{x}}^n = \mathbf{x}^{n,*}$ 

```

Algorithm 2.3 Presentation of the **s-BLS** sampling

Inputs: δ, η, M, V_m .**Outputs:** $\tilde{\mathbf{x}}_K^n$, for the **s-BLS** sampling.

```

 $z = \infty$ 
while  $z > \delta$  do
  for  $i = 1, \dots, M$  do
    Sample  $\mathbf{x}^{n,i} \sim \rho^n$ 
  end for
  Select at random  $I^* \in \arg \min_{1 \leq i \leq M} \|\mathbf{G}_{\mathbf{x}^{n,i}} - \mathbf{I}\|_2$ .
  Set  $\mathbf{x}^{n,*} = \mathbf{x}^{n,I^*}$  and set  $z = \|\mathbf{G}_{\mathbf{x}^{n,*}} - \mathbf{I}\|_2$ 
end while
Set  $\tilde{\mathbf{x}}^n = \mathbf{x}^{n,*}$ 

```

Set $K = \{1, \dots, n\}$ and $\tilde{\mathbf{x}}_K^n = \tilde{\mathbf{x}}^n$ Set $z = \|\mathbf{G}_{\tilde{\mathbf{x}}_K^n} - \mathbf{I}\|_2$.**while** $z \leq \delta$ **do**Select k^* such that

$$\|\mathbf{G}_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n} - \mathbf{I}\|_2 = \min_{k \in K} \|\mathbf{G}_{\tilde{\mathbf{x}}_{K \setminus \{k\}}^n} - \mathbf{I}\|_2.$$

Set $z = \|\mathbf{G}_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n} - \mathbf{I}\|_2$ If $z \leq \delta$, set $K \leftarrow K \setminus \{k^*\}$ **end while**

and as fast as possible. Sampling from univariate densities can be done with classical techniques such as rejection sampling [31], inverse transform sampling [31] and slice sampling [81]. This latter is used for all the numerical examples. To sample from multivariate densities we use a sequential sampling technique, which only requires samplings from univariate densities, see appendix B.

In the next section, two kinds of comparisons are performed. First, we compare qualitatively the distributions of the random variable $Z_{\mathbf{x}^n}$ and the distributions of the n -points sample \mathbf{x}^n . These analyses depend only on the choice of the approximation space V_m , and does not involve a function to approximate. Secondly, we compare quantitatively the efficiency of the different methods to approximate functions. We consider analytical functions on \mathbb{R}^d or $[-1, 1]^d$ equipped with Gaussian or uniform measures.

2.4.2 Qualitative analysis of the boosted optimal weighted least-squares method

Analysis of the stability

The objective of this paragraph is to compare the stability of the boosted optimal weighted least-squares method, using subsampling from Section 2.2.2 or not, respectively **s-BLS** and **c-BLS**, with two other state-of-the art methods, standard least-squares method, abbreviated **SLS** and **OWLS** method. As explained in Section 1.3.1, the stability of the least-squares projection can be characterized by the random variable $Z_{\mathbf{x}^n} = \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2$. The closer $Z_{\mathbf{x}^n}$ is to 0, the more stable the approximation is. In this paragraph, we compare the distribution of this random variable $Z_{\mathbf{x}^n}$ for the different sampling methods. For the **SLS** method, the sampling measure is the reference measure μ . In the **OWLS** method, the sampling measure ρ is the measure with density w^{-1} with respect to the reference measure μ , chosen as defined in (2.1).

In this paragraph, we consider $d = 1$ and we present results for polynomial approximation spaces $V_m = \mathbb{P}_5$, where 5 is the polynomial degree and with μ a Gaussian or uniform measure. Figures 2.2a and 2.2b show that using **OWLS** instead of **SLS** shifts the distribution of the random variable $Z_{\mathbf{x}^n}$ to the left. Without surprise, we see that conditioning $Z_{\mathbf{x}^n}$ by the event $A_\delta = \{Z_{\mathbf{x}^n} \leq \delta\}$ yields a distribution whose support is included in $[0, \delta]$. As expected, we also notice that very similar results are obtained for the **OWLS** and **c-BLS** methods when choosing $M = 1$. In the same manner, increasing the number of resampling M also shifts the PDF of $Z_{\mathbf{x}^n}$ to the low values, and decreases its variability. When interested in maximizing the probability of A_δ , **c-BLS** method is therefore an interesting alternative to **SLS** and **OWLS**. At last, looking at Figures 2.3b and 2.3a, we observe that the greedy selection moves the PDF of $Z_{\mathbf{x}^n}$ to the high values. This was expected: to switch from **c-BLS** to **s-BLS**, the size of the sample is reduced, as points are adaptively removed. However, as it is conditioned by A_δ , it remains better than **SLS** and **OWLS** methods.

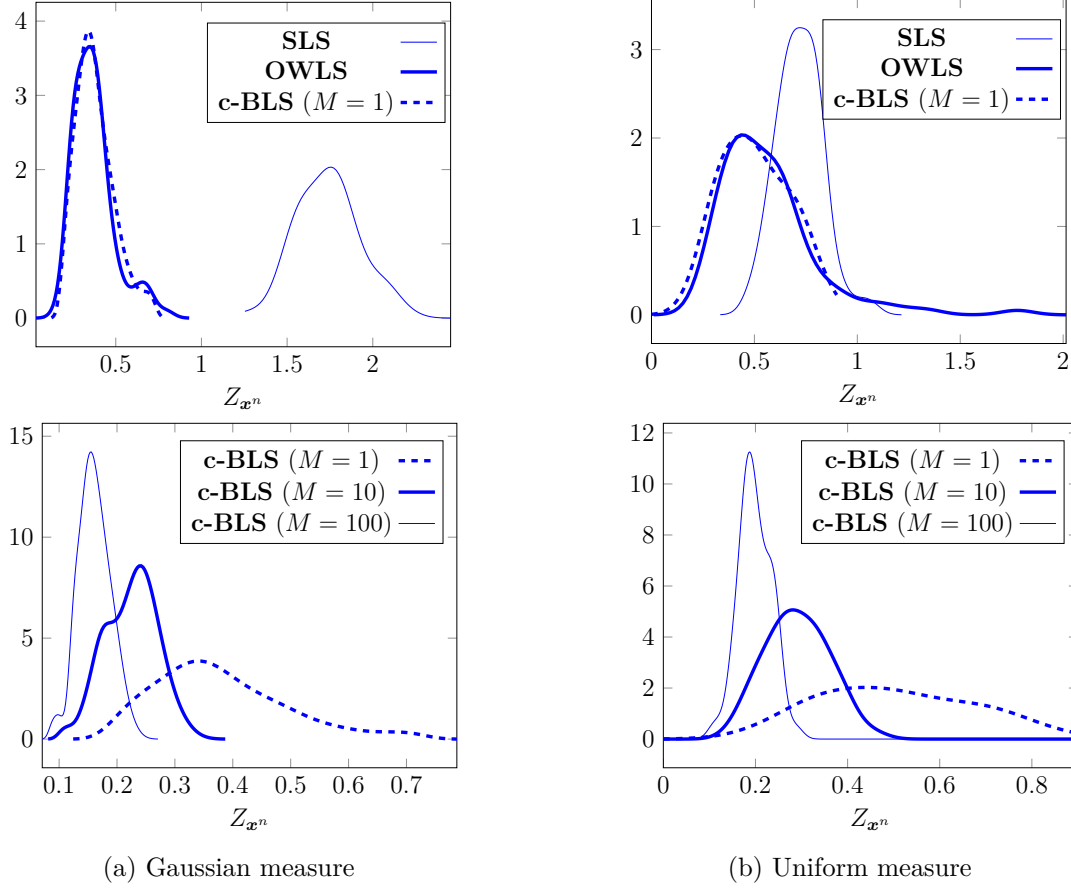


Figure 2.2 – Probability density function of $Z_{x^n} = \|\mathbf{G}_{x^n} - \mathbf{I}\|_2$ for $V_m = \mathbb{P}_5$, with $\delta = 0.9$ and $n = 100$.

Distribution of the sample points

In this paragraph, we consider $d = 1$, we are interested in the distributions of the points sampled with the **c-BLS** and **s-BLS** methods.

First, $n = 10$ points are sampled according to the **c-BLS** method for different values of M (from 1 to 50000). These points are then sorted in ascending order. After repeating this procedure $r = 1000$ times, the probability distributions of the sorted points are represented in Figures 2.4 and 2.5 (one color per point).

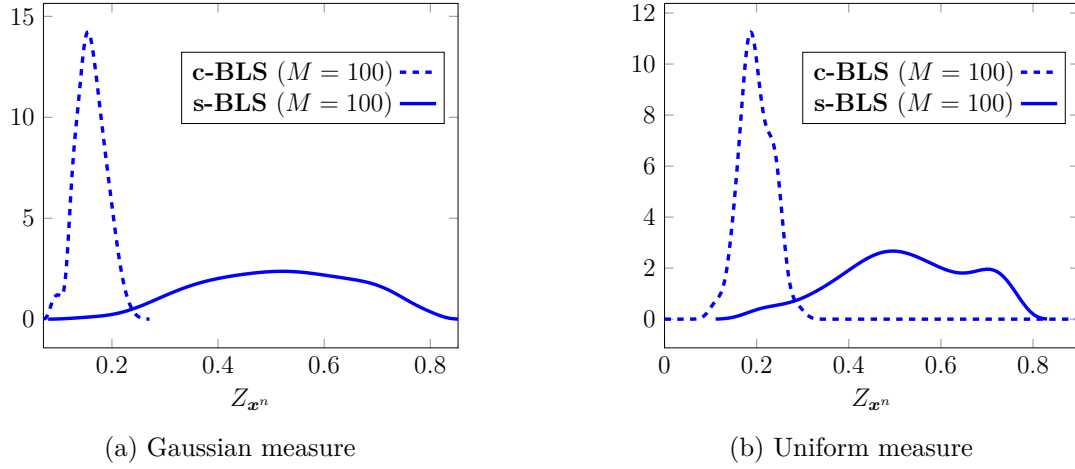


Figure 2.3 – Probability density function of $Z_{\mathbf{x}^n} = \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2$ for $V_m = \mathbb{P}_5$, with $\delta = 0.9$ and $n = 100$.

For μ the Gaussian or the uniform measure, when M is small, ($M = 1$ or $M = 10$), we notice a strong overlap between the support of the different distributions. This is no longer the case for the highest values of M ($M = 10000$ or $M = 50000$). Hence, the larger M , the further apart the points are from each other with high probability, and the more they concentrate around specific values. Secondly, $n = 6$ points are sampled according to the **s-BLS** method. To this end, a greedy procedure is applied to remove points from an initial sample of 10 points until we get the required number of points. In that case, as we fix the size of the sample, there is a priori no guarantee that the value of $Z_{\mathbf{x}^n}$ remains smaller than δ . The obtained 6-points sample is once again sorted in ascending order, and we repeat the procedure $r = 1000$ times. As previously, the distributions of the sorted points are represented in Figures 2.6 and 2.7 for different values of M . Only moderate values of M are considered, as we empirically observed that choosing M higher than 100 had very little influence on the results when considering subsampling.

Comparing the figures associated with the methods with or without greedy subsampling, we finally observe that **s-BLS** method provides results that are very close to **c-BLS** method with a very high value of M . This emphasizes the efficiency of the greedy selection to separate the support of the distributions of points.

In Figure 2.6a, the distributions of the sorted points associated to the **OWLS** method are represented in dashed lines. This shows that even if no resampling is carried out ($M = 1$), using the **s-BLS** method instead of **OWLS** improves the space filling properties of the obtained samples.

Remark 2.8. In Figures 2.4, 2.5, 2.6 and 2.7 black dots have been added to indicate the positions of the first n Gauss-Hermite points in the Gaussian case, and the n first Gauss-Legendre points in the uniform case. Interestingly, we observe that, in the Gaussian case, the distribution of points

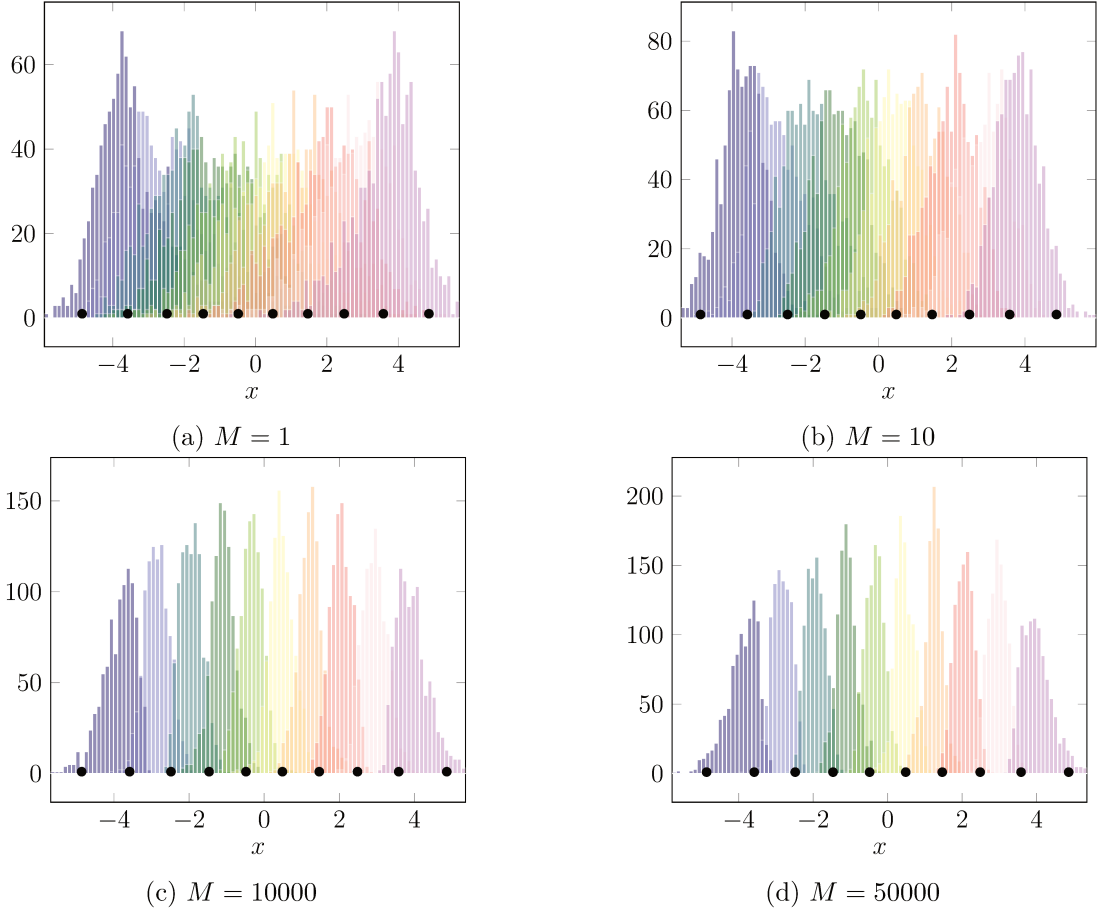


Figure 2.4 – Distributions of the $x^{(i)}, i = 1, \dots, 10$, with \mathbf{x}^{10} sampled from the **c-BLS** method for $V_m = \mathbb{P}_5$ and μ the Gaussian measure. On each graph, the black dots represent the 10 first Gauss-Hermite points.

spreads symmetrically around zero and in the uniform case, the distribution concentrates on the edges.

2.4.3 Quantitative analysis for polynomial approximations

In this paragraph, we want to compare the different methods introduced in Subsection 2.4.1 in terms of approximation errors. The quality of the approximation u^* of a function $u \in L^2_\mu(\mathcal{X})$ is assessed by estimating the error of approximation

$$\varepsilon(u^*)^2 = \frac{1}{n_{test}} \sum_{x \in \mathbf{x}_{test}} (u(x) - u^*(x))^2,$$

with \mathbf{x}_{test} a set of n_{test} test samples. In practice, we choose $n_{test} = 1000$. To study the robustness of the methods, we compute 10 times the approximations and draw 10 different test samples \mathbf{x}_{test}

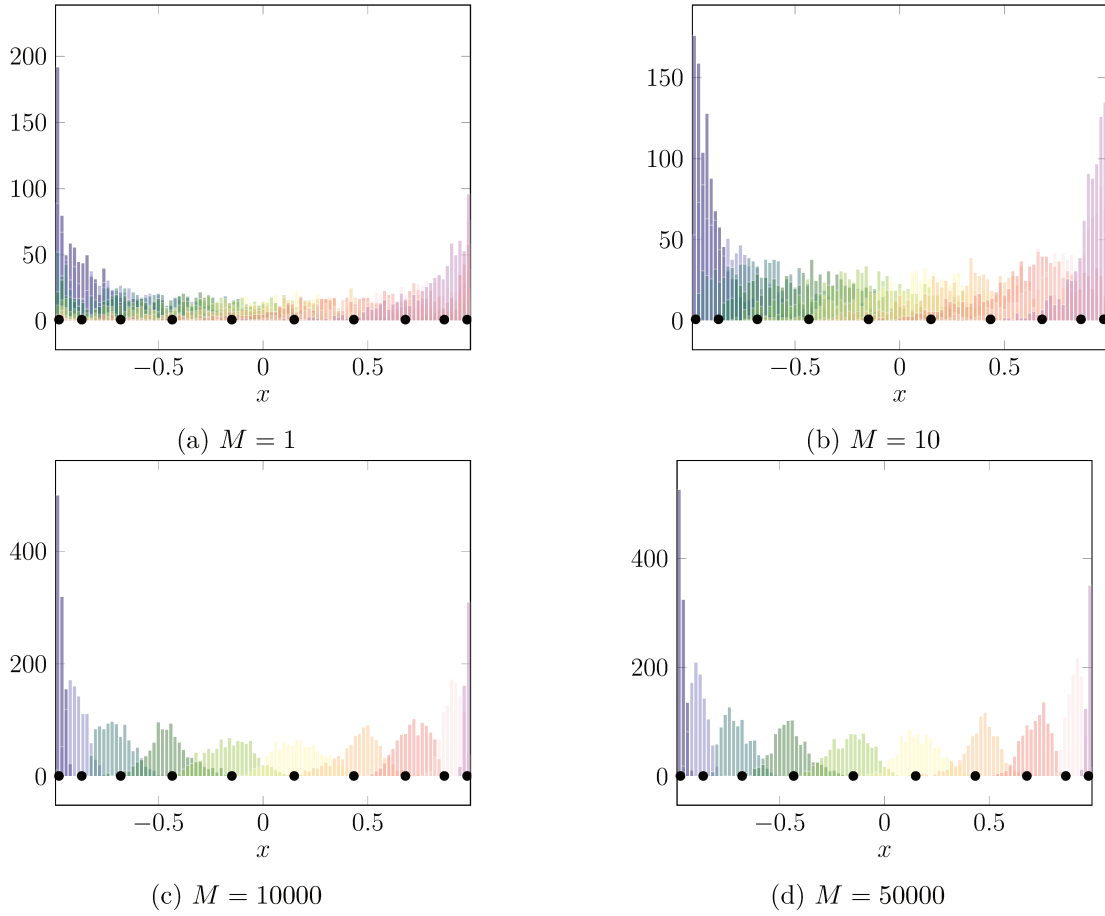


Figure 2.5 – Distributions of the $x^{(i)}, i = 1, \dots, 10$, with \mathbf{x}^{10} sampled from the **c-BLS** method for $V_m = \mathbb{P}_5$ and μ the uniform measure. On each graph, the black dots represent the 10 first Gauss-Legendre points.

and compute empirical confidence intervals of level 10% and 90% for the errors of approximation.

For each example, two kinds of comparisons are performed.

- Tables (a) present the results for the methods **OWLS**, **c-BLS**, and **s-BLS**. The number of samples is chosen to ensure the stability of the empirical Gram matrix with high probability or almost surely. For the **OWLS** method, the number of samples is equal to $n = \lceil d_\delta^{-1} m \log(2m\eta^{-1}) \rceil$, with $\delta = 0.9$ and $\eta = 0.01$ such that the stability of the empirical Gram matrix is ensured with probability greater than 0.99. For the **c-BLS** method, the initial number of samples n is also equal to $n = \lceil d_\delta^{-1} m \log(2m\eta^{-1}) \rceil$ but choosing $\delta = 0.9$ and $\eta = 0.01^{1/M}$ (M is specified in the example). Since the resulting sample $\tilde{\mathbf{x}}^n$ satisfies the event A_δ from Equation (2.2), the stability of the empirical Gram matrix is guaranteed. For the **s-BLS** method, the initial sample is the same than for the **c-BLS** method and a greedy

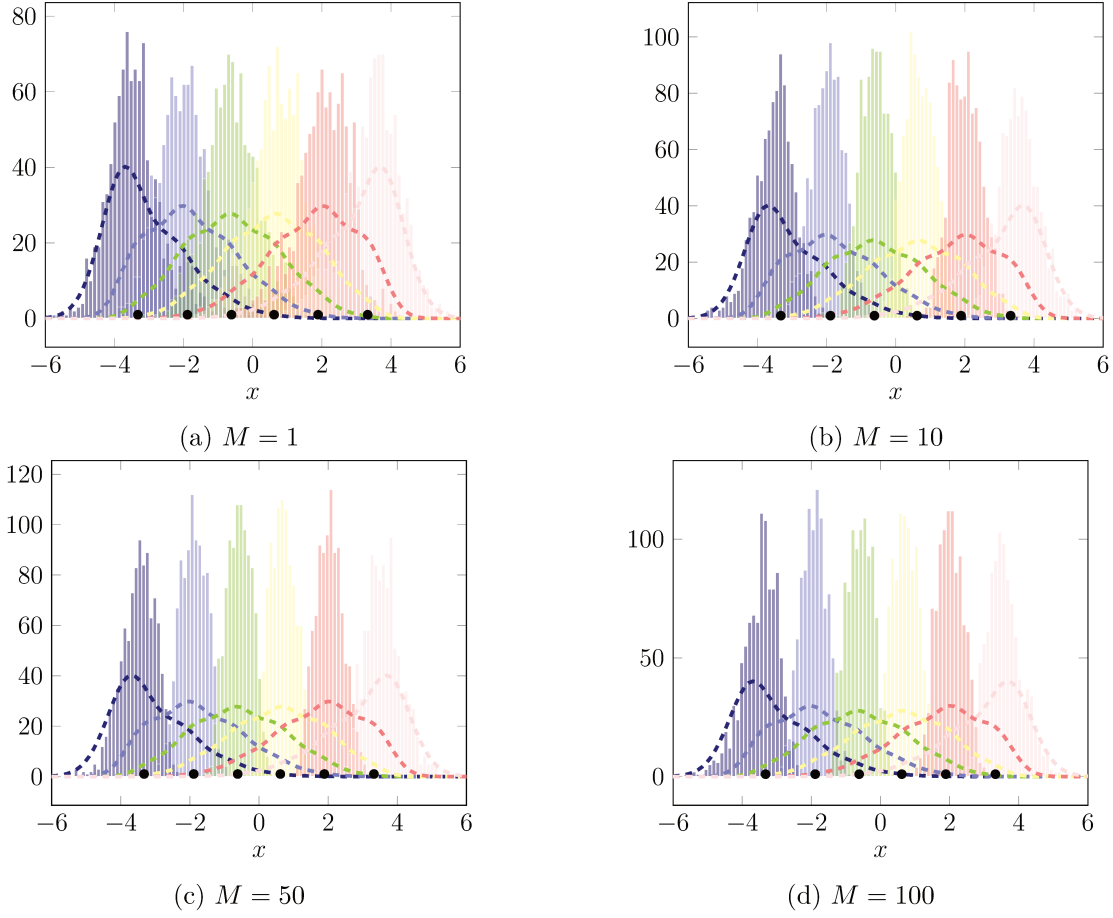


Figure 2.6 – Distributions of the $x^{(i)}, i = 1, \dots, 6$, with \mathbf{x}^6 sampled from the **s-BLS** method (colored histograms) and **OWLS** method (dashed lines) for $V_m = \mathbb{P}_5$ with μ the gaussian measure. On each graph, the black dots represent the 6 first Gauss-Hermite points.

removal of points is performed as long as the event A_δ is satisfied.

- Tables (b) compare all the \mathcal{I} — methods with **OWLS**, **BLS** and **s-BLS**. For all methods, except **s-BLS**, the number of samples n is taken equal to the dimension of the approximation space m . In this particular comparison, the **BLS** method only consists in a resampling strategy. For the **s-BLS** method, the initial number of samples is equal to $n = \lceil d_\delta^{-1} m \log(2m\eta^{-1}) \rceil$, with $\delta = 0.9$ and $\eta = 0.01^{1/M}$ (M is specified in the example) and since the resulting sample $\tilde{\mathbf{x}}^n$ satisfies A_δ , the stability of the empirical Gram matrix is guaranteed. Then the greedy selection of points is performed as long as $n \geq m$, implying that the resulting sample used to build the approximation may not lead to a stable empirical Gram matrix.

Remark 2.9. In the interpolation regime $n = m$, the stability condition from Equation (1.7) can not be reached for the **BLS** method. Indeed, choosing M arbitrary big enables us

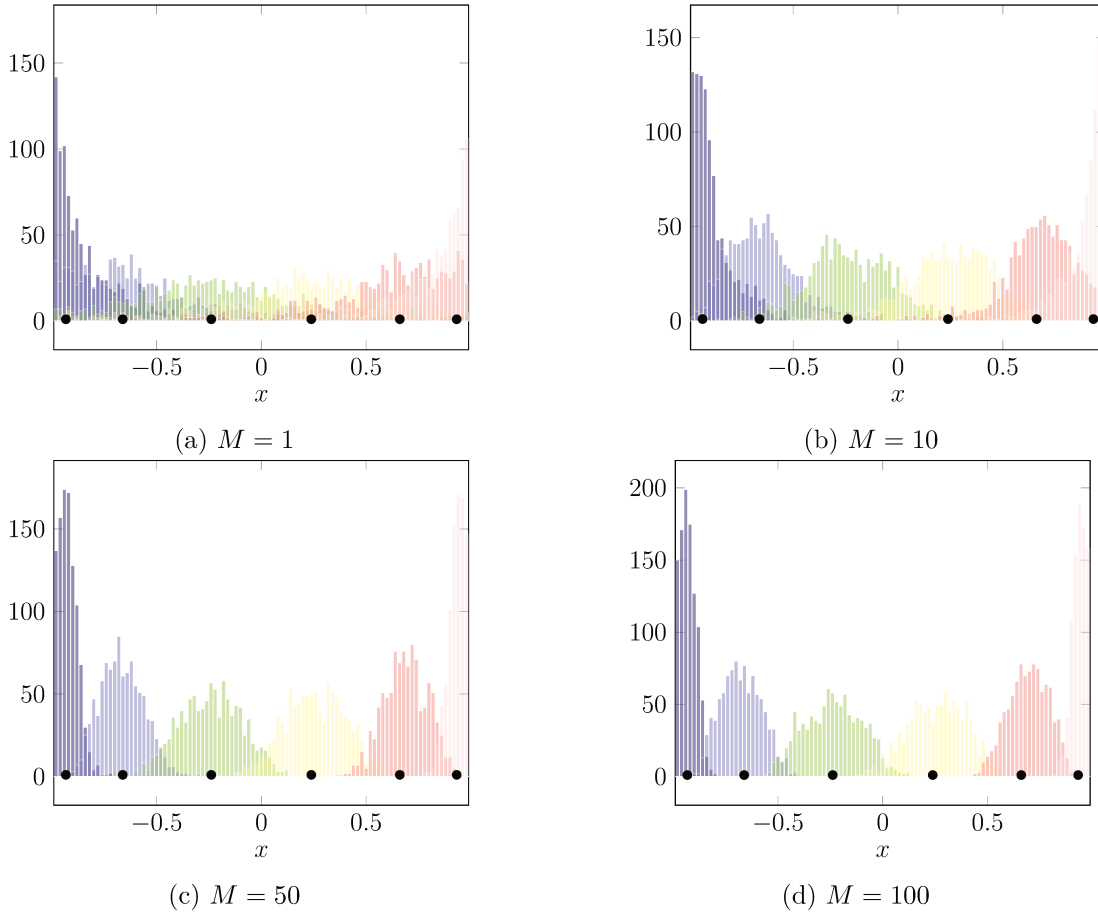


Figure 2.7 – Distributions of the $x^{(i)}, i = 1, \dots, 6$, with x^6 sampled from the **s-BLS** method for $V_m = \mathbb{P}_5$ and μ the uniform measure. On each graph, the black dots represent the 6 first Gauss-Legendre points.

to choose η close to 1, but it still holds $\eta < 1$. It implies that the number of samples $n(\delta, \eta, m)$ necessary to get the stability condition from Theorem 1.5 has to be greater than $d_\delta^{-1} m \log(2m)$, that is itself strictly greater than m . In the case of controlled cost, this explains why we choose to use the **BLS** method without conditioning.

A first function

We consider $\mathcal{X} = \mathbb{R}$, equipped with the standard Gaussian measure μ and the function

$$u_1(x) = \exp\left(-\frac{1}{4}(x-1)^2\right).$$

The approximation space is $V_m = \mathbb{P}_{m-1} = \text{span}\{\varphi_i : 1 \leq i \leq m\}$, where the basis $\{\varphi_i\}_{i=1}^m$ is chosen as the Hermite polynomials of degree less than $m-1$. This is referred to as example 1.

OWLS			c-BLS ($M = 1$)		c-BLS ($M = 100$)		s-BLS ($M = 100$)	
p	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n
5	[-2.1; -1.8]	134	[-2.2; -1.9]	134	[-2.2; -1.8]	48	[-2.1; -1.8]	[6; 6]
10	[-3.1; -3.1]	265	[-3.2; -3.0]	265	[-3.2; -3.1]	108	[-3.0; -2.6]	[11; 13]
15	[-4.5; -4.2]	404	[-4.5; -4.3]	404	[-4.5; -4.3]	176	[-4.6; -4]	[16; 17]
20	[-5.8; -5.7]	548	[-5.9; -5.7]	548	[-5.9; -5.7]	249	[-5.9; -5.3]	[21; 23]
25	[-6.9; -6.7]	697	[-7; -6.8]	697	[-7; -6.7]	326	[-6.9; -6.6]	[26; 28]
30	[-8.3; -8.1]	848	[-8.4; -8.2]	848	[-8.4; -8.2]	405	[-8.4; -8.0]	[31; 36]
35	[-9.4; -9.3]	1001	[-9.4; -9.3]	1001	[-9.4; -9.2]	488	[-9.4; -8.9]	[37; 39]
40	[-10.7; -10.5]	1157	[-10.7; -10.5]	1157	[-10.7; -10.6]	572	[-10.8; -10.3]	[42; 44]

(a) Guaranteed stability with probability greater than 0.99 for **OWLS** and almost surely for the other methods.

Interpolation					Least-Squares regression		
I-GaussH		I-Magic	I-Leja	I-Fekete	OWLS	BLS ($M = 100$)	s-BLS
p	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$
5	[-2.2; -1.5]	[-0.5; -0.5]	[-1.3; -1.2]	[-1.3; -1.2]	[-1.8; -0.3]	[-2.1; -1.6]	[-2.1; -1.8]
10	[-3.1; -3.0]	[-0.8; -0.8]	[-2.1; -2.0]	[-1.9; -1.9]	[-2.6; -1.0]	[-3.1; -1.9]	[-3.0; -2.5]
15	[-4.5; -4.2]	[-1.4; -1.4]	[-2.6; -2.5]	[-2.4; -2.3]	[-4.4; -1.6]	[-4.3; -2.2]	[-4.6; -3.9]
20	[-5.9; -5.8]	[-2.2; -2.2]	[-4.2; -4.2]	[-4.1; -4.1]	[-5.9; -3.2]	[-6.1; -4.3]	[-5.9; -5.1]
25	[-6.8; -6.7]	[-3.4; -3.4]	[-4.4; -4.3]	[-5.1; -5.0]	[-6.8; -3.9]	[-6.4; -4.7]	[-6.9; -6.0]
30	[-8.6; -8.4]	[-4.5; -4.4]	[-5.7; -5.6]	[-6.1; -6.1]	[-9.1; -4.4]	[-8.8; -5.2]	[-8.5; -7.2]
35	[-9.3; -9.2]	[-5.3; -5.3]	[-7.1; -7.0]	[-7.7; -7.6]	[-10.4; -4.9]	[-9.0; -6.4]	[-9.4; -8.5]
40	[-9.8; -9.7]	[-7.0; -7.0]	[-8.6; -8.5]	[-8.5; -8.5]	[-10.9; -6.7]	[-10.8; -7.1]	[-11.0; -9.7]

(b) Given cost: $n = m$ Table 2.1 – Approximation error ε in log-10 scale for the example 1. Abbreviations are defined in Subsection 2.4.1.

For this example, looking at Table 2.1a, we first observe that the approximation error decreases in a similar way for the four methods **OWLS**, **c-BLS** ($M = 1$), **c-BLS** ($M = 100$) and **s-BLS** ($M = 100$) when the size of the approximation space increases. However, the results for the **c-BLS** ($M = 100$) method are using less evaluations of the function. Indeed, by resampling, that is to say by increasing the value of M , the bound of the probability of getting a stable approximation is $1 - \eta^M$ instead of $1 - \eta$. Hence if η is chosen equal to 0.01 for $M = 1$, taking η equal to $0.01^{1/M}$ for higher values of M does not modify the bound of the probability of getting a stable approximation, but allows us to strongly reduce the number of samples needed to guarantee the same stability condition (the minimum number of samples is given in Lemma 2.1 by $n = \lceil d_\delta^{-1} m \log(2m\eta^{-1}) \rceil$). Regarding the number of evaluations of the function, the **s-BLS** ($M = 100$) method, which uses greedy subsampling is even better.

Looking at Table 2.1b, we also observe that for all the methods, the error of approximation decreases when the size of the approximation space increases. Nevertheless, it is interesting to

notice that among the interpolation methods, the **\mathcal{I} -Magic** method is less accurate than the others. In practice, for the **s-BLS** method, letting the greedy algorithm reach the interpolation regime ($m = n$) may provide a sample $\tilde{\mathbf{x}}^n$ which does not guarantee the stability. However, focusing on the upper bound of the errors, we also see that when $n = m$, only the **s-BLS** method seems to be able to provide results that are compatible to the ones of the **\mathcal{I} -GaussH** method.

Remark 2.10. The points for the **\mathcal{I} -Magic**, **\mathcal{I} -Fekete** and **\mathcal{I} -Leja** are chosen among a sufficiently large and dense discretization of \mathcal{X} . Here, in the case where $\mathcal{X} = \mathbb{R}$, we choose a uniform discretization of the bounded interval $[-10, 10]$ with 10000 points.

A second function

In this section, we consider $\mathcal{X} = [-1, 1]$ equipped with the uniform measure and the function

$$u_2(x) = \frac{1}{1 + 5x^2}.$$

The approximation space is $V_m = \mathbb{P}_{m-1} = \text{span}\{\varphi_i : 1 \leq i \leq m\}$, where the basis $\{\varphi_i\}_{i=1}^m$ is chosen as the Legendre polynomials of degree less than $m - 1$. This is referred to as example 2. For this example, the same observations than for the first function can be made:

- when resampling, it is possible to guarantee the stability of the approximation at a lower cost, without increasing the approximation error (see Table 2.2a),
- when resampling and also subsampling, it is possible to guarantee the stability of the approximation at a cost close to the interpolation regime, without increasing the approximation error (see Table 2.2a),
- the **s-BLS** method is comparable to interpolation in terms of the accuracy of the approximation (see Table 2.2b).

The only difference is that the **\mathcal{I} -Magic** method behaves almost as well as the other **\mathcal{I} -** method, which was not the case with the Gaussian measure.

A third function

We here consider the function

$$u_3(x) = \sum_{i=0}^p \exp\left(-\frac{i}{2}\right) \psi_i(x)$$

where $\mathcal{X} = \mathbb{R}$ is equipped with the Gaussian measure, $(\psi_1, \dots, \psi_m) = \mathbf{U}(\varphi_1, \dots, \varphi_m)$, with $\{\varphi_i\}_{i=1}^m$ the set of Hermite polynomials of degree less than p and \mathbf{U} an orthogonal matrix. In practice \mathbf{U} is taken as the matrix of the left singular vectors of a $m \times m$ matrix \mathbf{A} , whose elements are i.i.d. realizations of a standard Gaussian random variable $\mathcal{N}(0, 1)$.

OWLS			c-BLS ($M = 1$)		c-BLS ($M = 100$)		s-BLS ($M = 100$)	
p	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n
5	[-1.3; -1.2]	134	[-1.3; -1.2]	134	[-1.3; -1.3]	48	[-1.2; -0.9]	[6; 6]
10	[-2.4; -2.4]	265	[-2.4; -2.4]	265	[-2.4; -2.4]	108	[-2.3; -1.9]	[11; 11]
15	[-3.1; -3.1]	405	[-3.2; -3.2]	405	[-3.2; -3.2]	176	[-3.1; -2.8]	[16; 16]
20	[-4.3; -4.2]	548	[-4.3; -4.3]	548	[-4.3; -4.3]	249	[-4.2; -4.1]	[21; 23]
25	[-5.0; -4.8]	697	[-5.1; -5.0]	697	[-5.1; -5.0]	326	[-5.0; -4.7]	[26; 29]
30	[-6.2; -6.1]	848	[-6.2; -6.2]	848	[-6.2; -6.2]	405	[-6.1; -5.8]	[31; 35]
35	[-6.9; -6.9]	1001	[-6.9; -6.9]	1001	[-6.9; -6.9]	488	[-6.9; -6.6]	[36; 40]
40	[-8.0; -8.0]	1157	[-8.1; -8.1]	1157	[-8.1; -8.1]	572	[-8.0; -7.7]	[42; 46]

(a) Guaranteed stability with probability greater than 0.99 for **OWLS** and almost surely for the other methods.

Interpolation					Least-Squares regression		
I-GaussL		I-Magic	I-Leja	I-Fekete	OWLS	BLS ($M = 100$)	s-BLS
p	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$
5	[-1.3; -1.3]	[-0.9; -0.8]	[-1.1; -1.1]	[-1.1; -1.1]	[-0.6; 0.5]	[-1.1; -0.3]	[-1.2; -1.0]
10	[-2.3; -2.3]	[-2.2; -2.2]	[-2.2; -2.2]	[-2.2; -2.2]	[-0.9; 1.1]	[-1.7; 0.6]	[-2.3; -1.6]
15	[-3.2; -3.1]	[-2.9; -2.8]	[-3.0; -2.9]	[-3.0; -2.9]	[-2.0; 1.5]	[-2.5; 1.2]	[-2.9; -2.6]
20	[-4.2; -4.2]	[-4.0; -3.9]	[-4.1; -4.1]	[-4.1; -4.1]	[-1.9; 1.1]	[-2.8; 1.4]	[-4.1; -3.3]
25	[-5.1; -5.0]	[-4.9; -4.8]	[-4.9; -4.8]	[-4.8; -4.8]	[-2.4; 1.5]	[-3.5; 1.5]	[-4.8; -4.2]
30	[-6.1; -6.0]	[-5.7; -5.7]	[-6.0; -5.9]	[-6.0; -5.9]	[-3.2; 2.1]	[-4.0; 0.5]	[-5.8; -5.0]
35	[-6.9; -6.9]	[-6.6; -6.6]	[-6.7; -6.7]	[-6.7; -6.7]	[-4.3; 3.0]	[-4.0; 1.2]	[-6.7; -5.3]
40	[-7.9; -7.9]	[-7.7; -7.7]	[-7.8; -7.8]	[-7.8; -7.8]	[-5.6; 3.8]	[-4.7; 0.1]	[-7.7; -6.6]

(b) Given cost: $n = m$.

Table 2.2 – Approximation error ε in log-10 scale for the example 2. Abbreviations are defined in 2.4.1.

In this example, p is chosen equal to 40, the approximation space $V_m = \text{span}\{\psi_i : 1 \leq i \leq m\}$, and we consider different \mathbf{U} for each trial. The basis (ψ_1, \dots, ψ_m) is chosen as approximation basis. This is referred to as example 3 and the associated results are summarized in Table 2.3a and Table 2.3b. Hence, in the same manner than in Table 2.1 and Table 2.2, we notice that

- the error of approximation decreases when the size of the approximation space increases for all methods, we however notice that the **I-Magic** does not perform as well as the other methods,
- the errors associated with **s-BLS** ($M = 100$) method are almost the same than the ones associated with the **OWLS**, **c-BLS** ($M = 1$) and **c-BLS** ($M = 100$) methods while being based on a number of evaluations of the function tending to the interpolation regime,
- the **s-BLS** method provides better results than the **OWLS** and **BLS** ($M = 100$) methods when n is chosen equal to m (interpolation regime). It also provides really better results than all the **I**-methods, except the **I-GaussH** (Gauss-Hermite points are still used).

OWLS			c-BLS ($M = 1$)		c-BLS ($M = 100$)		s-BLS ($M = 100$)	
p	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n
5	[-1.7; -1.5]	134	[-1.7; -1.5]	134	[-1.7; -1.4]	48	[-1.6; -1.1]	[6; 7]
10	[-2.7; -2.6]	265	[-2.8; -2.6]	265	[-2.8; -2.6]	108	[-2.7; -2.3]	[11; 13]
15	[-3.9; -3.7]	405	[-3.8; -3.7]	405	[-3.9; -3.7]	176	[-3.7; -3.3]	[18; 23]
20	[-5.0; -4.8]	548	[-5.0; -4.7]	548	[-5.0; -4.6]	249	[-4.9; -4.4]	[24; 26]
25	[-6.1; -5.7]	697	[-6.0; -5.8]	697	[-6.0; -5.8]	326	[-6.0; -5.6]	[29; 34]
30	[-7.1; -6.9]	848	[-7.1; -6.9]	848	[-7.1; -6.9]	405	[-7.1; -6.7]	[34; 38]
35	[-8.2; -7.9]	1001	[-8.2; -8.0]	1001	[-8.2; -7.9]	488	[-8.3; -7.7]	[40; 45]
40	[-15.8; -15.4]	1157	[-15.8; -15.5]	1157	[-15.8; -15.2]	572	[-15.7; -15.4]	[41; 47]

(a) Guaranteed stability with probability greater than 0.99 for **OWLS** and almost surely for the other methods.

Interpolation					Least-Squares regression		
I-GaussH		I-Magic	I-Leja	I-Fekete	OWLS	BLS ($M = 100$)	s-BLS
p	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$
5	[-1.5; 0.2]	[-1.4; -0.7]	[-1.4; -0.8]	[-1.4; -1.0]	[-1.5; 0]	[-1.6; -1.0]	[-1.6; -1.1]
10	[-2.8; -1.2]	[-2.3; -1.6]	[-2.4; -1.8]	[-2.3; -1.5]	[-2.4; -1.3]	[-2.4; -1.4]	[-2.6; -1.7]
15	[-3.6; -2.5]	[-3.5; -2.6]	[-3.4; -2.4]	[-3.4; -2.7]	[-3.4; -0.9]	[-3.3; -1.6]	[-3.5; -1.0]
20	[-4.6; -3.6]	[-4.4; -3.6]	[-4.5; -3.9]	[-4.5; -4.0]	[-4.7; -3.8]	[-4.6; -3.4]	[-4.6; -3.8]
25	[-5.5; -4.2]	[-5.3; -4.7]	[-5.5; -4.8]	[-5.5; -4.9]	[-5.4; -3.3]	[-5.5; -4.1]	[-5.6; -5.1]
30	[-6.8; -5.6]	[-6.5; -5.8]	[-6.4; -5.9]	[-6.5; -5.8]	[-7.2; -5.5]	[-6.6; -4.8]	[-6.8; -5.8]
35	[-7.8; -6.5]	[-7.4; -6.5]	[-7.3; -6.9]	[-7.3; -6.9]	[-8.6; -6.8]	[-7.7; -6.4]	[-8.1; -7.2]
40	[-15.9; -14.6]	[-8.2; -6.3]	[-13.2; -11.4]	[-12.4; -11.1]	[-11.8; -4.2]	[-14.2; -7.7]	[-15.7; -15.1]

(b) Given cost: $n = m$.

Table 2.3 – Approximation error ε for the example 3. Abbreviations are defined in 2.4.1.

For this example, it is important to notice that the approximation space is not generated from a set of commonly-used polynomials, for which there exists adapted sequences of points for interpolation. This highlights the interest of the **s-BLS** method, which guarantees good sequences of points for the approximation, no matter what the approximation space is. The results obtained with the other **I**-methods show that it is difficult to choose a suitable set of initial points (in size and distribution).

Remark 2.11. As in example 2.4.3, the points for the **I-Magic**, **I-Fekete** and **I-Leja** are chosen among a sufficiently large and dense discretization of \mathcal{X} . Here, in the case where $\mathcal{X} = \mathbb{R}$, we choose a uniform discretization of the bounded interval $[-10, 10]$ with 10000 points.

Multi-dimensional example

Here, we consider $\mathcal{X} = [-1, 1]^d$, equipped with the uniform measure and the function

$$u(x) = \frac{1}{(1 - \frac{0.5}{2d} \sum_{i=1}^d x_i)^{d+1}}$$

We consider the hyperbolic cross polynomial approximation space defined by

$$\mathbb{P}_\Lambda = \text{span}\{\varphi_i(x) = \prod_{k=1}^d \psi_{i_k}^k(x) : i \in \Lambda\}, \text{ where } \Lambda = \{i = (i_1, \dots, i_d) : \prod_{k=1}^d (i_k + 1) \leq p + 1\}$$

where the $(\psi_{i_k}^k)_{i_k \geq 1}$ are sequences of univariate Legendre polynomials. For each $k \in \{1, \dots, d\}$, $\{\psi_{i_k}^k\}_{i_k \geq 1}$ is an orthonormal basis of $L_{\mu_k}^2(\mathcal{X}_k)$.

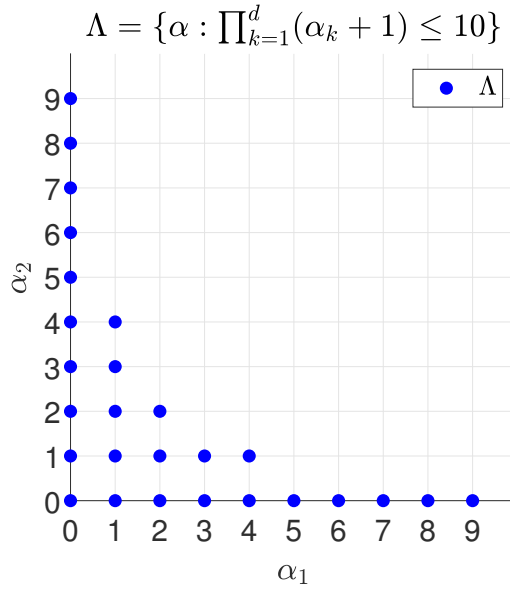


Figure 2.8 – Illustration of an hyperbolic cross index set for $d = 2$ and $p = 9$.

To define a set of interpolation points unisolvent for the tensorized basis $\{\varphi_i\}_{i \in \Lambda}$, we proceed as follows. For each $1 \leq k \leq d$, we introduce a set of points $(z_{i_k}^k)_{i_k=1}^{m_k} \in \mathcal{X}_k$ unisolvent for the space $\text{span}\{\psi_{i_k}^k\}_{i_k=1}^{m_k}$ (either Gauss, Leja, Fekete or Magic points). Then we introduce the multivariate sequence of points

$$\Gamma_\Lambda = \{z_i = (z_{i_1}^1, \dots, z_{i_d}^d) : i \in \Lambda, \# \Lambda = m\}.$$

OWLS				c-BLS ($M = 1$)		c-BLS ($M = 100$)		s-BLS ($M = 100$)	
p	m	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n
4	10	[-1.8; -1.7]	238	[-1.8; -1.8]	238	[-1.8; -1.8]	96	[-1.7; -1.5]	[10; 12]
9	27	[-3.3; -3.2]	727	[-3.3; -3.3]	727	[-3.3; -3.3]	341	[-3.2; -3.0]	[33; 38]
14	45	[-4.2; -4.1]	1282	[-4.2; -4.2]	1282	[-4.2; -4.2]	641	[-4.1; -3.9]	[58; 63]
19	66	[-5.6; -5.5]	1960	[-5.7; -5.5]	1960	[-5.7; -5.6]	1019	[-5.6; -5.5]	[92; 99]
24	87	[-6.5; -6.4]	2659	[-6.6; -6.4]	2659	[-6.5; -6.4]	1418	[-6.4; -6.3]	[130; 137]

(a) Guaranteed stability with probability greater than 0.99 for **OWLS** and almost surely for the other methods.

Interpolation						Least-Squares regression		
		\mathcal{I}-GaussL	\mathcal{I}-Magic	\mathcal{I}-Leja	\mathcal{I}-Fekete	OWLS	BLS ($M = 100$)	s-BLS
p	m	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$
4	10	[-1.0; -1.0]	[-1.0; -1.0]	[-1.6; -1.6]	[-1.0; -1.0]	[-1.2; 0.5]	[-1.5; 0.5]	[-1.6; -1.4]
9	27	[-1.7; -1.6]	[-2.2; -2.2]	[-3.1; -3.0]	[-1.7; -1.6]	[-2.7; -0.5]	[-2.7; -0.9]	[-2.9; -1.8]
14	45	[-2.2; -2.1]	[-3.1; -3.0]	[-3.7; -3.6]	[-2.2; -2.1]	[-3.4; -1.5]	[-3.2; -2.0]	[-3.6; -2.4]
19	66	[-2.7; -2.6]	[-4.0; -4.0]	[-5.4; -5.0]	[-2.7; -2.6]	[-4.8; 0.5]	[-4.5; -1.6]	[-5.1; -3.7]
24	87	[-3.1; -3.0]	[-4.1; -4.1]	[-6.1; -6.0]	[-3.0; -2.9]	[-5.3; -2.8]	[-5.2; -2.7]	[-5.6; -4.2]

(b) Given cost: $n = m$

Table 2.4 – Approximation error ε for the example 4 with $d = 2$. Abbreviations are defined in Subsection 2.4.1.

In Table 2.4, we observe that the best approximation errors are obtained with the least-squares regression methods, when the stability is guaranteed:

- in Table 2.4a, the **s-BLS** method strongly reduces the number of samples necessary to get this stability, it is about 1.5 times the interpolation regime.
- in Table 2.4b, the **\mathcal{I} -Leja** is the only interpolation method which performs better than the **s-BLS** method with a given cost. However, the **\mathcal{I} -Leja** is less accurate than the **s-BLS** method with guaranteed stability.

In Table 2.5, we observe that the conclusions are the same in dimension 4 than for the dimension 2. The only difference is that the number of samples necessary to get the stability is about 2 times the interpolation regime.

2.4.4 A noisy example

In this example, we consider the case where the evaluations are affected by a noise e , which fulfils the assumptions of Section 2.3 (the noise is independent from the sample $\tilde{\mathbf{x}}_K^n$, centred with a bounded conditional variance). Here we choose $e \sim \mathcal{N}(0, \sigma^2)$.

OWLS				c-BLS ($M = 1$)		c-BLS ($M = 100$)		s-BLS ($M = 100$)	
p	m	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n
4	23	[-1.5; -1.4]	608	[-1.5; -1.5]	608	[-1.5; -1.5]	279	[-1.5; -1.3]	[27; 33]
7	63	[-2.2; -2.0]	1862	[-2.2; -2.1]	1862	[-2.2; -2.0]	963	[-2.1; -1.9]	[99; 109]
10	93	[-2.2; -2.1]	2861	[-2.3; -2.1]	2861	[-2.3; -2.1]	1535	[-2.1; -2.0]	[164; 172]
13	153	[-2.4; -2.3]	4946	[-2.5; -2.4]	4946	[-2.5; -2.4]	2763	[-2.4; -2.3]	[291; 305]

(a) Guaranteed stability with probability greater than 0.99 for **OWLS** and almost surely for the other methods.

Interpolation						Least-Squares regression		
\mathcal{I}-GaussL		\mathcal{I}-Magic	\mathcal{I}-Leja	\mathcal{I}-Fekete	OWLS	BLS ($M = 100$)	s-BLS	
p	m	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$
4	23	[-0.6; -0.6]	[-1.0; -0.9]	[-1.3; -1.2]	[-0.6; -0.6]	[-1.1; 0.7]	[-1.2; 0.1]	[-1.0; 0.1]
7	63	[-0.9; -0.9]	[-1.4; -1.3]	[-1.2; -1.1]	[-0.9; -0.9]	[-1.2; -0.2]	[-1.7; 0.1]	[-1.6; -0.5]
10	93	[-1.0; -0.9]	[-1.3; -1.3]	[-0.5; -0.4]	[-1.0; -0.9]	[-1.2; 0]	[-1.1; 1.0]	[-1.4; 0.4]
13	153	[-1.2; -1.1]	[-1.6; -1.5]	[-2.2; -2.0]	[-1.2; -1.1]	[-1.7; -0.2]	[-1.5; -0.3]	[-1.6; -0.7]

(b) Given cost: $n = m$

Table 2.5 – Approximation error ε for the example 4 with $d = 4$. Abbreviations are defined in Subsection 2.4.1.

Table 2.6 presents the obtained results for two different approximation spaces' dimensions ($m = 10, 27$) and 3 different standard deviations of the noise ($\sigma = 0.1, 0.01, 0.001$). The noiseless case (corresponding to $\sigma = 0$ is also recalled). When the stability is guaranteed, see Table 2.6a, we observe that the influence of the noise is more important in the **s-BLS** method (which is in line with the expression from Equation (2.11), the higher n is the smaller the contribution due the noise is). Furthermore, when $m = 10$ ($p = 4$), the approximation error is determined by the dimension of the approximation space, whereas when $m = 27$ ($p = 9$), the approximation error decreases when σ decreases.

In Table 2.6b, we observe that the **\mathcal{I} -methods** are not robust to the noise (except **\mathcal{I} -Leja**), the approximation error is much higher than for the **s-BLS** method performed with $m = n$, for both dimensions of approximation spaces.

2.4.5 Overall conclusion for all examples

When the stability is guaranteed (see Table 2.1a, Table 2.2a, Table 2.3a, Table 2.4a), the **s-BLS** method, whose cost is close to m , behaves in the same way than the other least-squares regression methods and the best interpolation method. When $n = m$, the **s-BLS** method performs better than most of the interpolation methods. Depending on the choice of the approximation basis or the dimension of the problem d , the interpolation methods are more or less efficient. None

OWLS				c-BLS ($M = 1$)		c-BLS ($M = 100$)		s-BLS ($M = 100$)	
m	σ	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	n	$\log(\varepsilon(u^*))$	# K
10	0.1	[-1.8; -1.8]	238	[-1.8; -1.7]	238	[-1.8; -1.8]	96	[-1.6; -1.3]	[10; 11]
	0.01	[-1.8; -1.7]	238	[-1.8; -1.6]	238	[-1.8; -1.8]	96	[-1.6; -1.3]	[10; 11]
	0.001	[-1.8; -1.6]	238	[-1.8; -1.8]	238	[-1.8; -1.7]	96	[-1.7; -1.4]	[10; 12]
	0	[-1.8; -1.7]	238	[-1.8; -1.8]	238	[-1.8; -1.8]	96	[-1.7; -1.5]	[10; 12]
27	0.1	[-2.8; -2.6]	727	[-2.8; -2.6]	727	[-2.8; -2.6]	341	[-2.0; -1.8]	[31; 35]
	0.01	[-3.3; -3.2]	727	[-3.3; -3.3]	727	[-3.3; -3.3]	341	[-3.2; -3.0]	[30; 37]
	0.001	[-3.3; -3.3]	727	[-3.3; -3.3]	727	[-3.3; -3.3]	341	[-3.2; -2.9]	[31; 36]
	0	[-3.3; -3.2]	727	[-3.3; -3.3]	727	[-3.3; -3.3]	341	[-3.2; -3.0]	[33; 38]

(a) Guaranteed stability with probability greater than 0.99 for **OWLS** and almost surely for the other methods.

Interpolation					Least-Squares regression			
		\mathcal{I} -GaussL	\mathcal{I} -Magic	\mathcal{I} -Leja	\mathcal{I} -Fekete	OWLS	BLS ($M = 100$)	s-BLS
m	σ	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$	$\log(\varepsilon(u^*))$
10	0.1	[-1.1; -0.5]	[-1.1; -1.0]	[-1.5; -1.2]	[-1.2; -0.5]	[-0.8; 2]	[-1.4; 0.0]	[-1.6; -1.1]
	0.01	[-1.0; -0.9]	[-1.0; -1.0]	[-1.4; -1.2]	[-1.0; -0.9]	[-1.3; 1.2]	[-1.5; -0.1]	[-1.6; -1.2]
	0.001	[-1.0; -0.9]	[-1.0; -1.0]	[-1.4; -1.3]	[-1.0; -0.9]	[-1.5; 1.0]	[-1.6; -0.1]	[-1.6; -1.0]
	0	[-1.0; -1.0]	[-1.0; -1.0]	[-1.6; -1.6]	[-1.0; -1.0]	[-1.2; 0.5]	[-1.5; 0.5]	[-1.6; -1.4]
27	0.1	[1.5; 2.2]	[-0.8; 0.1]	[-1.5; -1.5]	[1.4; 2.2]	[0.8; 7.6]	[-0.7; 1.7]	[-1.8; -0.7]
	0.01	[-0.2; 0.3]	[-2.5; -1.5]	[-2.8; -2.7]	[-0.2; 0.3]	[-1.4; 6.8]	[-2.3; 0.4]	[-3.0; -2.3]
	0.001	[-2.0; -1.4]	[-2.2; -2.2]	[-2.8; -2.8]	[-2.0; -1.5]	[-2.2; -0.3]	[-2.8; -0.6]	[-2.9; -1.9]
	0	[-1.7; -1.6]	[-2.2; -2.2]	[-3.1; -3.0]	[-1.7; -1.6]	[-2.7; -0.5]	[-2.7; -0.9]	[-2.9; -1.8]

(b) Given cost: $n = m$ Table 2.6 – Approximation error ε for the example 4 noisy with $d = 2$. Abbreviations are defined in Subsection 2.4.1.

of them give the best results in all situations, whereas the **s-BLS** method with stability guaranteed is as efficient as the best interpolation method while having a number of samples tending to m .

The example where the data are polluted with noise shows a greater robustness of the **s-BLS** method (with the stability guaranteed) compared to the interpolation methods with a number of samples not too far from the dimension of the approximation space.

2.5 Conclusion

We have proposed a method to construct the projection of a function u in a given approximation space V_m with dimension m . In this method, the approximation is a weighted least-squares projection associated with random points sampled from a suitably chosen distribution. We obtained quasi-optimality properties (in expectation) for the weighted least-squares projection, with or without reducing the size of the sample by a greedy removal of points.

The error bound in the quasi-optimality property depends on the number of points selected by the greedy algorithm. The more points removed, the larger the bound will be. Therefore, if the goal is an accurate control of the error, as few points as possible should be removed. On the contrary, if the goal is to reduce the cost as much as possible but allows a larger error, the maximum number of points may be removed from the sample, which in some cases leads to $n = m$, the dimension of the approximation space.

As the convergence of this greedy algorithm to the interpolation regime is not systematic, it would be interesting to look for an optimal selection of the sub-sample with regard to the stability criterion. That is to say that we would select the k points to remove at the same time, however it is a combinatorial problem.

With this method, the points are sampled from a distribution which depends on the approximation space. In some cases, the approximation space is chosen adaptively among a sequence of spaces, then to have a reasonable number of function's evaluations, the samples used to build the projection onto a given approximation space should be reused to construct the projection onto a larger approximation space. In the next chapter, we propose an adaptive boosted optimal weighted least-squares method which constructs a sequence of boosted weighted least-squares projections, associated to a nested sequence of spaces.

Appendix of this Chapter

Proof of Lemma 2.2

We recall that for any sample \mathbf{x}^n , $Z_{\mathbf{x}^n} = \|\mathbf{G}_{\mathbf{x}^n} - \mathbf{I}\|_2$ and $\mathbb{P}(A_\delta) \geq 1 - \eta^M$ (Lemma 2.1). By definition of $\mathbf{x}^{n,*}$, we have $\mathbf{x}^{n,*} = \mathbf{x}^{n,I^*}$, where given the M samples $\mathbf{x}^{n,1}, \dots, \mathbf{x}^{n,M}$, the random variable I^* follows the uniform distribution on the set $\arg \min_{1 \leq i \leq M} Z_{\mathbf{x}^{n,i}}$ (possibly reduced to a singleton). The property (2.4) is a particular case of Equation (2.3) for $\varepsilon = 1$. However, let us first provide a simple proof of property (2.4). We have

$$\begin{aligned} \mathbb{E} \left(\|v\|_{\mathbf{x}^n}^2 \right) &= \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,*}}^2 | A_\delta \right) \leq \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,*}}^2 \right) \mathbb{P}(A_\delta)^{-1} \\ &\leq \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,I^*}}^2 \right) (1 - \eta^M)^{-1} \leq \sum_{j=1}^M \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,j}}^2 \right) (1 - \eta^M)^{-1} \\ &= \|v\|^2 M (1 - \eta^M)^{-1}. \end{aligned}$$

Now let us consider the proof of the other inequalities. We first note that $A_\delta = \{Z_{\mathbf{x}^{n,I^*}} \leq \delta\} = \{\min_{1 \leq i \leq M} Z_{\mathbf{x}^{n,i}} \leq \delta\}$. We consider the events $B_j = \{I^* = j\}$ which form a complete set of events. From the definition of I^* and A_δ and the fact that the samples $\mathbf{x}^{n,i}$ are i.i.d., it is clear that $\mathbb{P}(B_j) = \mathbb{P}(B_1) = M^{-1}$ and $\mathbb{P}(B_j \cap A_\delta) = \mathbb{P}(B_1 \cap A_\delta)$ for all j . Therefore,

$$\mathbb{P}(A_\delta \cap B_1) = \frac{1}{M} \sum_{j=1}^M \mathbb{P}(A_\delta \cap B_j) = \frac{1}{M} \mathbb{P}(A_\delta) \geq \frac{(1 - \eta^M)}{M}.$$

Then

$$\begin{aligned} \mathbb{E} \left(\|v\|_{\mathbf{x}^n}^2 \right) &= \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,*}}^2 | A_\delta \right) = \sum_{j=1}^M \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,j}}^2 | A_\delta \cap B_j \right) \mathbb{P}(B_j) = \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,1}}^2 | A_\delta \cap B_1 \right) \\ &= \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,1}}^2 \mathbf{1}_{A_\delta \cap B_1} \right) \mathbb{P}(A_\delta \cap B_1)^{-1} \\ &\leq \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,1}}^2 \mathbf{1}_{Z_{\mathbf{x}^{n,1}} \leq \delta} \mathbf{1}_{\min_{2 \leq i \leq M} Z_{\mathbf{x}^{n,i}} \geq Z_{\mathbf{x}^{n,1}}} \right) M (1 - \eta^M)^{-1} \\ &= \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,1}}^2 \mathbf{1}_{Z_{\mathbf{x}^{n,1}} \leq \delta} \mathbb{E} \left(\mathbf{1}_{\min_{2 \leq i \leq M} Z_{\mathbf{x}^{n,i}} \geq Z_{\mathbf{x}^{n,1}}} | \mathbf{x}^{n,1} \right) \right) M (1 - \eta^M)^{-1} \\ &= \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,1}}^2 \mathbf{1}_{Z_{\mathbf{x}^{n,1}} \leq \delta} \mathbb{E} \left(\mathbf{1}_{Z_{\mathbf{x}^{n,2}} > Z_{\mathbf{x}^{n,1}}} | \mathbf{x}^{n,1} \right)^{M-1} \right) M (1 - \eta^M)^{-1}. \end{aligned}$$

Using Hölder's inequality, we have that for any $0 < \varepsilon \leq 1$,

$$\begin{aligned} \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,*}}^2 | A_\delta \right) &\leq \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,1}}^{\frac{2}{\varepsilon}} \mathbf{1}_{Z_{\mathbf{x}^{n,1}} \leq \delta} \right)^\varepsilon \mathbb{E} \left(\mathbb{E} \left(\mathbf{1}_{Z_{\mathbf{x}^{n,2}} > Z_{\mathbf{x}^{n,1}}} | \mathbf{x}^{n,1} \right)^{\frac{M-1}{1-\varepsilon}} \right)^{1-\varepsilon} M (1 - \eta^M)^{-1} \\ &\leq \mathbb{E} \left(\|v\|_{\mathbf{x}^{n,1}}^{\frac{2}{\varepsilon}} \right)^\varepsilon \mathbb{E} \left(\mathbb{E} \left(\mathbf{1}_{Z_{\mathbf{x}^{n,2}} > Z_{\mathbf{x}^{n,1}}} | \mathbf{x}^{n,1} \right)^{\frac{M-1}{1-\varepsilon}} \right)^{1-\varepsilon} M (1 - \eta^M)^{-1}. \end{aligned}$$

For any measurable function f and any two i.i.d. random variables X and Y , we have that $\mathbb{E}(\mathbb{1}_{f(X) > f(Y)} | Y)$ is a uniform random variable on $(0, 1)$. Therefore $\mathbb{E}(\mathbb{1}_{Z_{\mathbf{x}^n, 2} > Z_{\mathbf{x}^n, 1}} | \mathbf{x}^{n, 1})$ is uniformly distributed on $(0, 1)$ and

$$\mathbb{E} \left(\mathbb{E} \left(\mathbb{1}_{Z_{\mathbf{x}^n, 2} > Z_{\mathbf{x}^n, 1}} | \mathbf{x}^{n, 1} \right)^{\frac{M-1}{1-\varepsilon}} \right) = \frac{1}{\frac{M-1}{1-\varepsilon} + 1} = \frac{1-\varepsilon}{M-\varepsilon}.$$

By combining the previous results, we obtain

$$\mathbb{E} \left(\|v\|_{\mathbf{x}^n, \star}^2 | A_\delta \right) \leq \mathbb{E}(\|v\|_{\mathbf{x}^n}^{\frac{2}{\varepsilon}})^\varepsilon M(1-\eta^M)^{-1} \frac{(1-\varepsilon)^{1-\varepsilon}}{(M-\varepsilon)^{1-\varepsilon}}.$$

For $\varepsilon = 1$, we recover the result (2.4). The last result simply follows from

$$\mathbb{E}(\|v\|_{\mathbf{x}^n}^{\frac{2}{\varepsilon}}) \leq \mathbb{E} \left(\|v\|_{\mathbf{x}^n}^2 \right) \|v\|_{\infty, w}^{2/\varepsilon - 2} = \|v\|^2 \|v\|_{\infty, w}^{2/\varepsilon - 2}.$$

Details for the remark 2.4

We recall that $\tilde{D}(M, L, m, \alpha, \varepsilon) = C(\varepsilon, M)(L(1+c_m))^{2-2\varepsilon}\alpha^\varepsilon$. Taking the logarithm of this expression it comes,

$$\log(\tilde{D}(M, L, m, \alpha, \varepsilon)) = \underbrace{\log(C(\varepsilon, M))}_I + \underbrace{\varepsilon \log(\alpha) + (2-2\varepsilon) \log(L(1+c_m))}_{II}$$

Considering I ,

$$\log(C(\varepsilon, M)) = \log \left(M \frac{(1-\varepsilon)^{1-\varepsilon}}{(M-\varepsilon)^{1-\varepsilon}} \right) = \log(M) + (1-\varepsilon) \log \left(\frac{1-\varepsilon}{M-\varepsilon} \right)$$

Taking the derivative with respect to ε ,

$$\begin{aligned} \frac{\partial I}{\partial \varepsilon} &= -\log(1-\varepsilon) - \frac{1-\varepsilon}{1-\varepsilon} + \frac{1-\varepsilon}{M-\varepsilon} + \log(M-\varepsilon) \\ &= \log \left(\frac{M-\varepsilon}{1-\varepsilon} \right) + \frac{1-\varepsilon}{M-\varepsilon} - 1. \end{aligned}$$

Looking at II ,

$$\frac{\partial II}{\partial \varepsilon} = -2 \log(L(1+c_m)) + \log(\alpha).$$

All in all $\frac{\partial I}{\partial \varepsilon} + \frac{\partial II}{\partial \varepsilon} = 0$ gives

$$-\log \left(\frac{1-\varepsilon}{M-\varepsilon} \right) + \frac{1-\varepsilon}{M-\varepsilon} = 1 + 2 \log(L(1+c_m)) - \log(\alpha).$$

If, we set $Y = \frac{1-\varepsilon}{M-\varepsilon}$, it comes

$$-\log(Y) + Y = 1 + 2 \log(L(1 + c_m)) - \log(\alpha)$$

also

$$\exp(Y) = \exp(1 + 2 \log(L(1 + c_m)) - \log(\alpha))Y.$$

This equation has no explicit solution, however numerically we can found the value ε . Some examples are shown in the remark 2.4 from this chapter.

ADAPTIVE BOOSTED OPTIMAL WEIGHTED LEAST-SQUARES

Contents

3.1	Introduction	75
3.2	Notations	76
3.3	Optimal weighted least-squares with block-structured sampling	77
3.3.1	Approximation in a given space	77
3.3.2	Adaptive approximation with a nested sequence of spaces	78
3.4	Boosted optimal weighted least-squares with block-structured sampling	79
3.4.1	Approximation in a given space	79
3.4.2	Adaptive approximation with a nested sequence of spaces	83
3.5	Numerical illustrations	85
3.5.1	Illustration of the stability of the adaptive boosted least-squares strategy	85
3.5.2	Illustration for polynomial approximation	89
3.6	Conclusion	92

3.1 Introduction

For adaptive approximation of functions, a classical approach is to consider approximations in a sequence of nested approximation spaces $(V_{m_l})_{l \geq 1}$ with increasing dimension m_l . In some cases, the sequence $(V_{m_l})_{l \geq 1}$ may be given a priori [20, 27] but this sequence may also be adaptively generated in the sense that the construction of $V_{m_{l+1}}$ depends on the projection onto the approximation space V_{m_l} [18, 19].

In this chapter, we present an adaptive version of the boosted optimal weighted least-squares method presented in Chapter 2 for computing projections associated with such nested sequences of approximation spaces $(V_{m_l})_{l \geq 1}$. The difficulty is that the optimal sampling measure (also used to construct the boosted least-squares projection) depends on the approximation spaces $(V_{m_l})_{l \geq 1}$.

However, to keep a reasonable number of function's evaluations, we should avoid to draw new samples to compute the weighted least-squares projection at each iteration of the adaptive strategy. Here, the proposed adaptive boosted optimal weighted least-squares method relies on the deterministic strategy from [68] (that recycles all the samples, going from the step l to the step $l + 1$). In practice, at each step of the adaptive strategy we observe that the number of samples used to build the projection is close to the one obtained with an interpolation method (i.e. $n = m_l$).

In Section 3.3, we recall the main results from [68], by presenting the weighted least-squares projection in a given approximation space using block-structured sampling and then its adaptation to the sequential setting which takes advantage of this particular structure of samples, and allows to reuse all the samples from one step to another. In Section 3.4.1, we present the boosted optimal weighted least-squares method with block-structured sampling and in Section 3.4.2 we present the adaptive boosted optimal weighted least-squares method.

3.2 Notations

We begin by recalling some notions and notations already introduced in the two first chapters of this thesis but necessary for a self-contained reading of this chapter. Let \mathcal{X} be a subset of \mathbb{R}^d equipped with a probability measure μ , with $d \geq 1$. We consider a function u from $L_\mu^2(\mathcal{X})$, the Hilbert space of square-integrable real-valued functions defined on \mathcal{X} . We let $\|\cdot\|_{L_\mu^2}$ be the natural norm in $L_\mu^2(\mathcal{X})$. When there is no ambiguity, $L_\mu^2(\mathcal{X})$ will be simply denoted L_μ^2 , the norm $\|\cdot\|_{L_\mu^2}^2$ and associated inner product $(\cdot, \cdot)_{L_\mu^2}$ will be denoted $\|\cdot\|$ and (\cdot, \cdot) respectively. Let V_m be a m -dimensional subspace of L_μ^2 , with $m \geq 1$. The weighted least-squares methods allow to define a particular projection of a function $u \in L_\mu^2(\mathcal{X})$ onto V_m , using n evaluations of this function at random points. Letting $\mathbf{x}^n := \{x^i\}_{i=1}^n$ be a set of n points in \mathcal{X} , we consider the weighted least-squares projection defined by

$$Q_{V_m}^{\mathbf{x}^n} u := \arg \min_{v \in V_m} \|u - v\|_{\mathbf{x}^n}, \quad (3.1)$$

where $\|\cdot\|_{\mathbf{x}^n}$ is a discrete semi-norm defined for v in L_μ^2 by

$$\|v\|_{\mathbf{x}^n}^2 := \frac{1}{n} \sum_{i=1}^n w(x^i) v(x^i)^2, \quad (3.2)$$

where w is a given non negative function defined on \mathcal{X} . We denote by $\{\varphi_j\}_{j=1}^m$ an orthonormal basis of V_m and we denote by $\boldsymbol{\varphi} = (\varphi_1, \dots, \varphi_m) : \mathcal{X} \rightarrow \mathbb{R}^m$ the m -dimensional vector-valued function such that $\boldsymbol{\varphi}(x) = (\varphi_1(x), \dots, \varphi_m(x))^T$. Also $P_{V_m} u$ denotes the orthogonal projection of u onto V_m .

In this chapter, the weight function w is chosen as

$$w(x)^{-1} = \frac{1}{m} \sum_{j=1}^m \varphi_j(x)^2 = \frac{1}{m} \|\boldsymbol{\varphi}(x)\|_2^2. \quad (3.3)$$

The optimal sampling measure from [24] will be referred to as optimal weighted least-squares measure and the one from [68], because of the particular structure of the samples, will be referred to as block-structured optimal weighted least-squares method.

3.3 Optimal weighted least-squares with block-structured sampling

3.3.1 Approximation in a given space

Let us consider for each $j = 1, \dots, m$, the measure $d\rho_j$ associated to the basis function φ_j defined by

$$\rho_j = \varphi_j(x)^2 d\mu.$$

For $\tau \geq 1$ an integer and $n = \tau m$, let $\mathbf{x}^n := \{x^i\}_{i=1}^n$ be a set of n points such that for each $j = 1, \dots, m$, the samples $x^{(j-1)\tau+1}, \dots, x^{j\tau}$ are drawn from the measure $d\rho_j$. In other words, for a given integer $\tau \geq 1$, the n independent samples (x^1, \dots, x^n) are drawn from the product measure

$$\boldsymbol{\gamma}^n = \left(\otimes_{j=1}^m \rho_j \right)^{\otimes \tau}. \quad (3.4)$$

It is important to notice that, contrary to the approach from [24], the samples $\{x^i\}_{i=1}^n$ are not identically distributed, they are only i.i.d by batch of τ samples. We denote by $\mathbf{G}_{\mathbf{x}^n}^m$ the empirical Gram matrix associated to the basis $\{\varphi_j\}_{j=1}^m$ of the approximation space V_m and the sample \mathbf{x}^n , which is defined by

$$\mathbf{G}_{\mathbf{x}^n}^m = \frac{1}{n} \sum_{i=1}^n w(x^i) \boldsymbol{\varphi}(x^i) \otimes \boldsymbol{\varphi}(x^i). \quad (3.5)$$

The random variable $Z_{\mathbf{x}^n}^m = \|\mathbf{G}_{\mathbf{x}^n}^m - \mathbf{I}\|_2$ quantifies the numerical stability of the weighted least-squares projection $Q_{V_m}^{\mathbf{x}^n}$.

The proof of the following lemma is inspired from [68], where they show that considering \mathbf{x}^n a n -sample sampled from $\boldsymbol{\gamma}^n$, with $n = \tau m$, it holds $\mathbb{E}(\mathbf{G}_{\mathbf{x}^n}^m) = \mathbf{I}$.

Lemma 3.1. *Let τ be an integer such that $\tau \geq 1$, and consider \mathbf{x}^n a n -sample from $\boldsymbol{\gamma}^n$, with*

$n = \tau m$. For all $v \in L_\mu^2(\mathcal{X})$ it holds

$$\mathbb{E}(\|v\|_{\mathbf{x}^n}^2) = \|v\|^2. \quad (3.6)$$

Proof. Using the definition of the discrete semi-norm recalled in Equation (3.2) and taking the expectation, we have

$$\begin{aligned} \mathbb{E}(\|v\|_{\mathbf{x}^n}^2) &= \frac{1}{n} \sum_{i=1}^n \mathbb{E}(w(x^i)v(x^i)^2) \\ &= \frac{1}{m} \frac{1}{\tau} \sum_{l=1}^{\tau} \sum_{j=1}^m \mathbb{E}(w(x^{(j-1)\tau+l})v(x^{(j-1)\tau+l})^2), \text{ where for all } 1 \leq l \leq \tau, x^{(j-1)\tau+l} \sim \rho_j \\ &= \frac{1}{m} \sum_{j=1}^m \mathbb{E}(w(x^j)v(x^j)^2), \text{ where for each } 1 \leq j \leq m, x^j \sim \rho_j \\ &= \frac{1}{m} \sum_{j=1}^m \int_{\mathcal{X}} w(x)v(x)^2 d\rho_j(x) \\ &= \int_{\mathcal{X}} v(x)^2 \frac{1}{m} \sum_{j=1}^m \varphi_j(x)^2 w(x) d\mu(x) = \|v\|^2, \end{aligned}$$

thanks to the definition (3.3) of w . □

We recall the following theorem, which states that the optimal weighted least-squares with block-structured sampling verifies a stability property.

Theorem 3.2. [68, Theorem 2]

For any $\eta, \delta \in (0, 1)$, if $n = \tau m$, with $\tau := \lceil d_\delta^{-1} \log(2m\eta^{-1}) \rceil$ and $d_\delta = -\delta + (1 + \delta) \log(1 + \delta)$, and \mathbf{x}^n is a n -sample sampled from γ^n , then the following properties are verified:

(i) the random variable $Z_{\mathbf{x}^n}^m$ satisfies the tail bound

$$\mathbb{P}(Z_{\mathbf{x}^n}^m > \delta) \leq \eta, \quad (3.7)$$

(ii) if $u \in L_\mu^2(\mathcal{X})$ then the conditioned estimator $Q_{V_m}^{\mathbf{x}^n, C} u$, defined by $Q_{V_m}^{\mathbf{x}^n, C} u = Q_{V_m}^{\mathbf{x}^n} u$ if $Z_{\mathbf{x}^n}^m \leq \delta$ and $Q_{V_m}^{\mathbf{x}^n, C} u = 0$ otherwise, satisfies

$$\mathbb{E}(\|u - Q_{V_m}^{\mathbf{x}^n, C} u\|^2) \leq \left(1 + (1 - \delta)^{-1}\right) \|u - P_{V_m} u\|^2 + \eta \|u\|^2. \quad (3.8)$$

3.3.2 Adaptive approximation with a nested sequence of spaces

Using the particular block-structure of the sample $\mathbf{x}^n \sim \gamma^n$, the author in [68] presents an adaptive strategy, which constructs weighted least-squares projections of u in a sequence of nested spaces

$V_{m_1} \subset V_{m_2} \subset \dots \subset V_{m_t} \subset L_\mu^2(\mathcal{X})$, of increasing dimension m_l , ensuring that the projections remain stable in expectation, simultaneously for all iterations from one to t .

Theorem 3.3. [68, Theorem 3] Let $\eta \in (0, 1)$, $s > 1$ be a real number and $t \geq 1$ be an integer. Given any sequence of nested spaces $(V_{m_l})_{1 \leq l \leq t}$ in $L_\mu^2(\mathcal{X})$ with dimensions $m_1 < \dots < m_t$, if

$$n_l = \tau_l m_l, \quad \tau_l := \lceil d_\delta^{-1} \ln \left(\zeta(s) m_l^{s+1} \eta^{-1} \right) \rceil \text{ for all } l = 1, \dots, t, \quad (3.9)$$

where $\zeta(s)$ denotes the Riemman-zeta function, then

$$\mathbb{P} \left(\bigcap_{l=1}^t \{Z_{\mathbf{x}^{n_l}}^{m_l} \leq \delta\} \right) \geq 1 - \eta \quad (3.10)$$

with $Z_{\mathbf{x}^{n_l}}^{m_l} = \|\mathbf{G}_{\mathbf{x}^{n_l}}^{m_l} - \mathbf{I}\|_2$, where $\mathbf{G}_{\mathbf{x}^{n_l}}^{m_l} \in \mathbb{R}^{m_l \times m_l}$ is the Gram matrix associated to an orthonormal basis of V_{m_l} defined by (3.5) and for any $l = 1, \dots, t$, the sample \mathbf{x}^{n_l} is generated according to the distribution γ^{n_l} (associated to the space V_{m_l}).

If $u \in L_\mu^2(\mathcal{X})$, then for any $l = 1, \dots, t$, the conditioned estimator $Q_{V_{m_l}}^{\mathbf{x}^{n_l}, C} u$ defined by $Q_{V_{m_l}}^{\mathbf{x}^{n_l}, C} u = Q_{V_{m_l}}^{\mathbf{x}^{n_l}} u$ if $Z_{\mathbf{x}^{n_l}}^{m_l} \leq \delta$ and $Q_{V_{m_l}}^{\mathbf{x}^{n_l}, C} u = 0$ otherwise, verifies

$$\mathbb{E}(\|u - Q_{V_{m_l}}^{\mathbf{x}^{n_l}, C} u\|^2) \leq \left(1 + (1 - \delta)^{-1}\right) \|u - P_{V_{m_l}} u\|^2 + \eta \|u\|^2. \quad (3.11)$$

3.4 Boosted optimal weighted least-squares with block-structured sampling

3.4.1 Approximation in a given space

In this section, we propose a boosted least-squares method whose principle is exactly the same than in Chapter 2 (or see also [52]) but exploiting the measure γ^n . For the sake of completeness, we remind in this section the main steps that consist in resampling according to γ^n until a stability criterion $Z_{\mathbf{x}^n}^m \leq \delta$ is satisfied and then removing points from the samples using a greedy strategy. Also we recall a theoretical result, which states that this method ensures the stability of the weighted least-squares projection in expectation.

The first step consists in drawing M independent samples $\{\mathbf{x}^{n,i}\}_{i=1}^M$, with $\mathbf{x}^{n,i} = (x^{1,i}, \dots, x^{n,i})$, from the distribution γ^n , and then selecting a sample $\mathbf{x}^{n,*}$ which satisfies

$$Z_{\mathbf{x}^{n,*}}^m = \min_{1 \leq i \leq M} Z_{\mathbf{x}^{n,i}}^m, \quad (3.12)$$

where $Z_{\mathbf{x}}^m = \|\mathbf{G}_{\mathbf{x}} - \mathbf{I}\|$ is the distance between the empirical Gram matrix associated to \mathbf{x} and the

identity matrix, which quantifies the numerical stability for a sample \mathbf{x} in \mathcal{X}^n . In the case where several samples $\mathbf{x}^{n,i}$ are solutions of the minimization problem, $\mathbf{x}^{n,*}$ is selected at random among the minimizers. We denote by $\gamma^{n,*}$ the probability measure of $\mathbf{x}^{n,*}$. If n fulfills the condition from Theorem 3.2, i.e $n \geq m \lceil d_\delta^{-1} \log(2m\eta^{-1}) \rceil$ the probability that the stability condition $Z_{\mathbf{x}^{n,*}}^m \leq \delta$ is verified is greater than $1 - \eta^M$ and can thus be made arbitrarily high, playing on M .

In order to ensure that the stability property is verified almost surely we consider the sample $\mathbf{x}^{n,*}$ conditioned on the event

$$A_\delta = \{Z_{\mathbf{x}^{n,*}}^m \leq \delta\}. \quad (3.13)$$

This conditioned sample is denoted $\tilde{\mathbf{x}}^n$ and the associated distribution $\tilde{\gamma}^n$. Such a sample is obtained by a simple rejection method, which consists in drawing samples $\mathbf{x}^{n,*}$ from the distribution $\gamma^{n,*}$ until A_δ is satisfied. It follows that $\mathbb{P}(Z_{\tilde{\mathbf{x}}^n}^m \leq \delta) = 1$ and that for any function f , $\mathbb{E}(f(\tilde{\mathbf{x}}^n)) = \mathbb{E}(f(\mathbf{x}^{n,*})|A_\delta)$.

Although the resampling enables us to choose δ and η such that n is smaller than with the initial strategy from [68], the value of n may still be high compared to an interpolation method. Therefore, to further decrease the sample size, for each generated sample $\tilde{\mathbf{x}}^n$, we propose to select a subsample which still verifies the stability condition.

We start with a sample $\tilde{\mathbf{x}}^n = (\tilde{x}^1, \dots, \tilde{x}^n)$ satisfying $Z_{\tilde{\mathbf{x}}^n}^m \leq \delta$ and then select a subsample $\tilde{\mathbf{x}}_K^n = (\tilde{x}^k)_{k \in K}$ with $K \subset \{1, \dots, n\}$ such that the random variable $Z_{\tilde{\mathbf{x}}_K^n}^m$ still satisfies

$$Z_{\tilde{\mathbf{x}}_K^n}^m \leq \delta.$$

In practice, the set K is constructed by a backward greedy procedure. Starting with $K = \{1, \dots, n\}$, at each step of the greedy procedure, we select k^* in K such that

$$Z_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n}^m = \min_{k \in K} Z_{\tilde{\mathbf{x}}_{K \setminus \{k\}}^n}^m. \quad (3.14)$$

If $Z_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n}^m \leq \delta$ and $\#K > n_{\min}$ then k^* is removed from K . Otherwise, the algorithm is stopped. We denote by $\tilde{\gamma}_g^n$ the distribution of the sample $\tilde{\mathbf{x}}_K^n$ produced by this greedy algorithm, the entire strategy is given by Algorithm 3.1.

Algorithm 3.1 Boosted optimal sampling from $\tilde{\gamma}_g^n$

Inputs: $\mu, \{\varphi_i\}_{i=1}^m, M, \delta, \eta$
Outputs: Sample $\tilde{\mathbf{x}}_K^n$ from $\tilde{\gamma}_g^n$

 Set $\tau = \lceil d_\delta^{-1} \log(2m\eta^{-1}) \rceil$ with $d_\delta = -\delta + (1 + \delta) \log(1 + \delta)$

 Set $z = \infty$
Resampling and Conditionning
while $z > \delta$ **do**
for $i = 1, \dots, M$ **do**

 Sample $\mathbf{x}^{n,i}$ according to γ^n
end for

 Select at random $I^* \in \arg \min_{1 \leq i \leq M} Z_{\mathbf{x}^{n,i}}^m$

 Set $\mathbf{x}^{n,*} = \mathbf{x}^{n,I^*}$ and $z = Z_{\mathbf{x}^{n,*}}^m$
end while

 Set $\tilde{\mathbf{x}}^n = \mathbf{x}^{n,*}$
Greedy subsampling

 Set $K = \{1, \dots, n\}$ and set $z = Z_{\tilde{\mathbf{x}}^n}^m$
while $z \leq \delta$ **do**

 Select $k^* \in K$ such that

$$Z_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n}^m = \min_{k \in K} Z_{\tilde{\mathbf{x}}_{K \setminus \{k\}}^n}^m$$

 Set $z = Z_{\tilde{\mathbf{x}}_{K \setminus \{k^*\}}^n}^m$

 If $z \leq \delta$, set $K \leftarrow K \setminus \{k^*\}$
end while

Theorem 3.4. Assume n satisfies the condition from Theorem 3.2 for any η and $\delta \in (0, 1)$. Let $\tilde{\mathbf{x}}_K^n$ be a sample produced by the greedy algorithm with $\#K \geq n_{\min}$, then the two following properties are verified:

(i) the random variable $Z_{\tilde{\mathbf{x}}_K^n}^m$ satisfies

$$\mathbb{P}(Z_{\tilde{\mathbf{x}}_K^n}^m \leq \delta) = 1. \quad (3.15)$$

(ii) The weighted least-squares projection $Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u$ satisfies the quasi-optimality property

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2) \leq \left(1 + \frac{n}{n_{\min}}(1 - \delta)^{-1}(1 - \eta^M)^{-1}M\right) \|u - P_{V_m} u\|^2. \quad (3.16)$$

Remark 3.1. Without subsampling, i.e. $n_{\min} = n$, the quasi-optimality property (3.16) is

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2) \leq \left(1 + (1 - \delta)^{-1}(1 - \eta^M)^{-1}M\right) \|u - P_{V_m} u\|^2. \quad (3.17)$$

When $n_{min} = \beta n$, the quasi-optimality property (3.16) holds too.

Proof. The first property is deduced from the definition of $\tilde{\mathbf{x}}_K^n$.

As the property $\mathbb{E}(\|v\|_{\mathbf{x}^n}^2) = \|v\|^2$ is still verified when $\mathbf{x}^n \sim \gamma^n$, see Lemma 3.1, the results are obtained in a similar manner than in Chapter 2 (or see [52]).

For any function $v \in L_\mu^2(\mathcal{X})$,

$$\mathbb{E}(\|v\|_{\mathbf{x}^n}^2) = \mathbb{E}(\|v\|_{\mathbf{x}^{n,*}}^2 | A_\delta) \leq \mathbb{E}(\|v\|_{\mathbf{x}^{n,*}}^2) \mathbb{P}(A_\delta)^{-1} \leq \sum_{k=1}^M \mathbb{E}(\|v\|_{\mathbf{x}^{n,j}}^2) (1 - \eta^M)^{-1} = \|v\|^2 M (1 - \eta^M)^{-1}.$$

Thanks to the Pythagorean equality, we have

$$\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2 = \|u - P_{V_m} u\|^2 + \|P_{V_m} u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2.$$

Using also that $(P_{V_m} - Q_{V_m}^{\tilde{\mathbf{x}}_K^n})u \in V_m$, and that $Z_{\tilde{\mathbf{x}}_K^n}^m \leq \delta$ almost surely, it comes

$$\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2 \leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \|P_{V_m} u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|_{\tilde{\mathbf{x}}_K^n}^2.$$

Using the fact that $Q_{V_m}^{\tilde{\mathbf{x}}_K^n}$ is an orthogonal projection with respect to the discrete semi-norm $\|\cdot\|_{\tilde{\mathbf{x}}_K^n}$, the Pythagorean equality for the discrete norm $\|\cdot\|_{\tilde{\mathbf{x}}_K^n}$ yields

$$\|P_{V_m} u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|_{\tilde{\mathbf{x}}_K^n}^2 + \|Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u - u\|_{\tilde{\mathbf{x}}_K^n}^2 = \|P_{V_m} u - u\|_{\tilde{\mathbf{x}}_K^n}^2,$$

which implies $\|P_{V_m} u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|_{\tilde{\mathbf{x}}_K^n}^2 \leq \|u - P_{V_m} u\|_{\tilde{\mathbf{x}}_K^n}^2$. Therefore,

$$\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2 \leq \|u - P_{V_m} u\|^2 + (1 - \delta)^{-1} \|u - P_{V_m} u\|_{\tilde{\mathbf{x}}_K^n}^2.$$

By definition of the discrete semi-norm and since $\#K \geq n_{min}$, $\|v\|_{\tilde{\mathbf{x}}_K^n}^2$ can be bounded by $\frac{n}{n_{min}} \|v\|_{\mathbf{x}^n}^2$ for all $v \in L_\mu^2$ and thus,

$$\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2 \leq \|u - P_{V_m} u\|^2 + \frac{n}{n_{min}} (1 - \delta)^{-1} \|P_{V_m} u - u\|_{\tilde{\mathbf{x}}_K^n}^2.$$

Taking the expectation, it comes

$$\mathbb{E}(\|u - Q_{V_m}^{\tilde{\mathbf{x}}_K^n} u\|^2) \leq \|u - P_{V_m} u\|^2 + \frac{n}{n_{min}} (1 - \delta)^{-1} M (1 - \eta^M)^{-1} \|P_{V_m} u - u\|^2,$$

which ends the proof. \square

3.4.2 Adaptive approximation with a nested sequence of spaces

In this section, we present the adaptive boosted optimal weighted least-squares strategy, which constructs a sequence of projections of u onto a sequence of nested spaces $(V_{m_l})_{l \geq 1}$, of increasing dimension m_l . We let $\{\varphi_i\}_{i \geq 1}$ be a set of orthonormal functions, such that for each $l \geq 1$, $\{\varphi_i\}_{i=1}^{m_l}$ is a basis of V_{m_l} .

In practice, we build two increasing sequences of sets of points $(\tilde{\mathbf{x}}^{n_l})_{l \geq 1}$ and $(\tilde{\mathbf{x}}_{K_l}^{n_l})_{l \geq 1}$, such that for all $l \geq 1$, $\tilde{\mathbf{x}}^{n_l} \subset \tilde{\mathbf{x}}^{n_{l+1}}$ and $\tilde{\mathbf{x}}_{K_l}^{n_l} \subset \tilde{\mathbf{x}}_{K_{l+1}}^{n_{l+1}}$, with $n_l = \tau_l m_l$ and τ_l is an integer. By construction $\tilde{\mathbf{x}}_{K_l}^{n_l}$ is also a subsample from $\tilde{\mathbf{x}}^{n_l}$, and the discrete projection $Q_{V_m}^{\tilde{\mathbf{x}}_{K_l}^{n_l}} u$ is built with $\tilde{\mathbf{x}}_{K_l}^{n_l}$. In the following paragraphs, we describe the method to construct these samples.

Initialization step

Let τ_1 be defined by $\tau_1 = \lceil d_\delta^{-1} \log(\zeta(s) m_1^{s+1} \eta^{-1}) \rceil$. For the initial space V_{m_1} , a n_1 -sample $\tilde{\mathbf{x}}^{n_1}$, with $n_1 = \tau_1 m_1$ is drawn from $\tilde{\gamma}^{n_1}$ and a backward greedy algorithm is used to obtain $\tilde{\mathbf{x}}_{K_1}^{n_1}$. The sample $\tilde{\mathbf{x}}_{K_1}^{n_1}$ whose distribution is $\tilde{\gamma}_g^{n_1}$ is obtained from Algorithm 3.1, using the parameters $\mu, \{\varphi_i\}_{i=1}^{m_1}, M, \delta, \eta$.

Transmission step

For each step l , the overall number of samples is $n_l = \tau_l m_l$ with τ_l a constant, which is defined by $\tau_l = \lceil d_\delta^{-1} \log(\zeta(s) m_l^{s+1} \eta^{-1}) \rceil$. At the previous steps already n_{l-1} samples have been drawn (with $n_{l-1} = \tau_{l-1} m_{l-1}$), such that if we add $p_l + q_l$ samples to n_{l-1} with $p_l = (m_l - (m_{l-1}))\tau_l$ and $q_l = m_{l-1}(\tau_l - \tau_{l-1})$ we have $n_l = n_{l-1} + p_l + q_l = \tau_{l-1} m_{l-1} + (m_l - (m_{l-1}))\tau_l + m_{l-1}(\tau_l - \tau_{l-1}) = m_l \tau_l$ samples.

Remark 3.2. The particular choices of the values of τ_l, p_l, q_l , that determine the number of samples n_l are based on the strategy from [68]. With these choices the author has proven that the optimal weighted least-squares projections are stable in expectation (see Section 3.3). Even if we do not have theoretical guarantees yet for our sampling strategy, the promising numerical results of the following part justify these heuristic choices in our strategy.

At the step l , the sample $\tilde{\mathbf{x}}^{n_l}$ is constructed by reusing the sample $\tilde{\mathbf{x}}^{n_{l-1}}$ and adding two blocks of samples from two different measures. We resample M times a p_l -sample from the measure $(\otimes_{j=1+m_{l-1}}^{m_l} d\rho_j)^{\otimes \tau_l}$, they are denoted $\{\mathbf{x}_{add}^{p_l, i}\}_{i=1}^M$ and a q_l -sample from the measure $(\otimes_{j=1}^{m_{l-1}} d\rho_j)^{\otimes (\tau_l - \tau_{l-1})}$, they are denoted $\{\mathbf{x}_{old}^{q_l, i}\}_{i=1}^M$.

For each i in $\{1, \dots, M\}$, we denote by $\bar{\mathbf{x}}^{n_l, i}$ the concatenation of the three samples $\tilde{\mathbf{x}}^{n_l-1}$, $\mathbf{x}_{add}^{p_l, i}$ and $\mathbf{x}_{old}^{q_l, i}$, such that $\bar{\mathbf{x}}^{n_l, i} = [\tilde{\mathbf{x}}^{n_l-1}, \mathbf{x}_{add}^{p_l, i}, \mathbf{x}_{old}^{q_l, i}]$. The samples $\bar{\mathbf{x}}^{n_l, i}$ has a total size equals to $n_{l-1} + p_l + q_l = \tau_{l-1}m_{l-1} + (m_l - (m_{l-1}))\tau_l + m_{l-1}(\tau_l - \tau_{l-1}) = m_l\tau_l$, which is n_l as desired. We select the sample $\bar{\mathbf{x}}^{n_l, \star}$ which satisfies,

$$Z_{\bar{\mathbf{x}}^{n_l, \star}}^{m_l} = \min_{1 \leq i \leq M_l} Z_{\bar{\mathbf{x}}^{n_l, i}}^{m_l}.$$

If several samples $\bar{\mathbf{x}}^{n_l, i}$ are solutions of the minimization problem, $\bar{\mathbf{x}}^{n_l, \star}$ is selected at random among the minimizers. The probability that the stability condition $Z_{\bar{\mathbf{x}}^{n_l, \star}}^{m_l} \leq \delta$ is verified can be made higher, playing on M . Furthermore, in order to ensure that this stability property is verified almost surely we consider the sample $\bar{\mathbf{x}}^{n_l, \star}$ conditioned by the event

$$A_\delta = \{Z_{\bar{\mathbf{x}}^{n_l, \star}}^{m_l} \leq \delta\}.$$

The resulting sample is denoted $\tilde{\mathbf{x}}^{n_l}$. In practice, the sample $\tilde{\mathbf{x}}^{n_l}$ is then obtained by a simple rejection method, drawing samples until $A_\delta = \{Z_{\bar{\mathbf{x}}^{n_l, \star}}^{m_l} \leq \delta\}$ is satisfied.

Remark 3.3. To condition the sample $\bar{\mathbf{x}}^{n_l, \star}$ by the event $A_\delta = \{Z_{\bar{\mathbf{x}}^{n_l, \star}}^{m_l-1} \leq \delta\}$, we assume that at each step l , for any \mathbf{x} verifying $Z_{\mathbf{x}}^{m_{l-1}} \leq \delta$,

$$\mathbb{P}(Z_{\bar{\mathbf{x}}^{n_l, \star}}^{m_l} \leq \delta \mid \tilde{\mathbf{x}}^{n_{l-1}} = \mathbf{x}) > 0,$$

where the first part $\tilde{\mathbf{x}}^{n_{l-1}}$ of the $\bar{\mathbf{x}}^{n_l, \star}$ is assumed to be fixed equal to \mathbf{x} . (This assumption remains to be shown). Then, since $\tilde{\mathbf{x}}^{n_{l-1}}$ satisfies $Z_{\tilde{\mathbf{x}}^{n_{l-1}}}^{m_{l-1}} \leq \delta$, the event A_δ has a non-zero probability of happening and our algorithm is well-defined.

To obtain $\tilde{\mathbf{x}}_{K_l}^{n_l}$, we reuse $\tilde{\mathbf{x}}_{K_{l-1}}^{n_{l-1}}$ and select points in $\tilde{\mathbf{x}}^{n_l}$ with a forward greedy algorithm. More precisely, starting from $K_l = K_{l-1}$ we select a k^\star in $\{1, \dots, n_l\} \setminus K_l$ such that the new sample $\tilde{\mathbf{x}}_{K_l \cup \{k^\star\}}^{n_{l-1}}$ verifies

$$Z_{\tilde{\mathbf{x}}_{K_l \cup \{k^\star\}}^{n_l}}^{m_l} = \min_{k \in \{1, \dots, n_l\} \setminus K_l} Z_{\tilde{\mathbf{x}}_{K_l \cup \{k\}}^{n_l}}^{m_l}.$$

We add k^\star to K_l and while $\tilde{\mathbf{x}}_{K_l \cup \{k^\star\}}^{n_l}$ does not satisfy the stability criterion, i.e. $Z_{\tilde{\mathbf{x}}_{K_l \cup \{k^\star\}}^{n_l}}^{m_l} > \delta$, we repeat the procedure. At the end of each step l , we keep both samples $\tilde{\mathbf{x}}^{n_l}$ and $\tilde{\mathbf{x}}_{K_l}^{n_l}$. The distribution of $\tilde{\mathbf{x}}_{K_l}^{n_l}$ is denoted $\tilde{\gamma}_g^{n_l}$. The overall strategy is described in Algorithm 3.2.

Algorithm 3.2 Adaptive Boosted Least-Squares**Inputs:** $t, s, \delta, \eta, M, (\varphi_j)_{j \geq 1}$ **Outputs:** samples $\tilde{\mathbf{x}}^{n_1}, \dots, \tilde{\mathbf{x}}^{n_t}$ and $\tilde{\mathbf{x}}_{K_1}^{n_1}, \dots, \tilde{\mathbf{x}}_{K_t}^{n_t}$ Sample $\tilde{\mathbf{x}}^{n_1}$ and $\tilde{\mathbf{x}}_{K_1}^{n_1}$ using Algorithm 3.1 with inputs $\mu, \{\varphi_i\}_{i=1}^{m_1}, M, \delta, \eta$.**for** $l = 2, \dots, t$ **do**Set $\tau_l = \lceil d_\delta^{-1} \log(\zeta(s)m_l^{s+1}\eta^{-1}) \rceil$, $p_l = (m_l - m_{l-1})\tau_l$, $q_l = m_{l-1}(\tau_l - \tau_{l-1})$, and $\tilde{\mathbf{x}}^{n_l} = \tilde{\mathbf{x}}^{n_{l-1}}$.
Set $z = \infty$.**Resampling and conditionning****while** $z > \delta$ **do****for** $i = 1, \dots, M$ **do**Sample $\mathbf{x}_{add}^{p_l, i} \sim (\otimes_{j=1+m_{l-1}}^{m_l} \rho_j)^{\otimes \tau_l}$.Sample $\mathbf{x}_{old}^{q_l, i} \sim (\otimes_{j=1}^{m_{l-1}} \rho_j)^{\otimes (\tau_l - \tau_{l-1})}$.Set $\bar{\mathbf{x}}^{n_l, i} = [\tilde{\mathbf{x}}^{n_{l-1}}, \mathbf{x}_{add}^{p_l, i}, \mathbf{x}_{old}^{q_l, i}]$ **end for**Select $I^* \in \arg \min_{1 \leq i \leq M} Z_{\bar{\mathbf{x}}^{n_l, i}}^{m_l}$ and set $\bar{\mathbf{x}}^{n_l, \star} = \bar{\mathbf{x}}^{n_l, I^*}$ $z = Z_{\bar{\mathbf{x}}^{n_l, \star}}^{m_l}$.**end while**Set $\tilde{\mathbf{x}}^{n_l} = \bar{\mathbf{x}}^{n_l, \star}$ **Greedy enrichment**Set $K_l = K_{l-1}$ and $\tilde{\mathbf{x}}_{K_l}^{n_l} = \tilde{\mathbf{x}}_{K_{l-1}}^{n_{l-1}}$.**while** $Z_{\tilde{\mathbf{x}}_{K_l}^{n_l}}^{m_l} > \delta$ **do**Select k^* in $\{1, \dots, n_l\} \setminus K_l$ such that

$$Z_{\tilde{\mathbf{x}}_{K_l \cup \{k^*\}}^{n_l}}^{m_l} = \min_{k \in \{1, \dots, n_l\} \setminus K_l} Z_{\tilde{\mathbf{x}}_{K_l \cup \{k\}}^{n_l}}^{m_l}. \quad (3.18)$$

Set $K_l = K_l \cup \{k^*\}$.**end while****end for**

3.5 Numerical illustrations

3.5.1 Illustration of the stability of the adaptive boosted least-squares strategy

Comparison with the adaptive optimal weighted least-squares with block-structured sampling

In this section, we want to illustrate the efficiency of the adaptive boosted least-squares method (abbreviated **a-BLS**) described by Algorithm 3.2, through $Z_{\tilde{\mathbf{x}}_{K_l}^{n_l}}^{m_l}$ and $\#K_l$. It is compared to the adaptive optimal weighted least-squares method from [68] (abbreviated **a-OWLS**), whose efficiency is measured through $Z_{\mathbf{x}_{n_l}}^{m_l}$ and n_l . Each algorithm is carried out 10 times and we provide 80% em-

pirical confidence intervals for the variables $Z_{\tilde{x}_{K_l}}^{m_l}$, $\#K_l$ and $Z_{x^{n_l}}^{m_l}$. We study the methods for the uniform measure on $\mathcal{X} = [-1, 1]^d$ and the standard Gaussian measure on $\mathcal{X} = \mathbb{R}^d$.

For $d = 1$, we consider the nested sequence of polynomial approximation spaces $(V_{m_l})_{l=1}^t$, where for each $l \geq 1$, $V_{m_l} = \mathbb{P}_{l-1}(\mathcal{X})$.

Gaussian measure					Uniform measure			
a-BLS		a-OWLS			a-BLS		a-OWLS	
m_l	$Z_{\tilde{x}_{K_l}}^{m_l}$	$\#K_l$	$Z_{x^{n_l}}^{m_l}$	n_l	$Z_{\tilde{x}_{K_l}}^{m_l}$	$\#K_l$	$Z_{x^{n_l}}^{m_l}$	n_l
1	[0; 0]	[1; 1]	[0; 0]	16	[0; 0]	[1; 1]	[0; 0]	16
2	[0.01; 0.07]	[2; 2]	[0.05; 0.18]	28	[0; 0.21]	[2; 2]	[0.02; 0.17]	28
3	[0.1; 0.58]	[3; 3]	[0.07; 0.2]	60	[0.34; 0.6]	[3; 3]	[0.06; 0.2]	60
4	[0.23; 0.69]	[4; 4]	[0.1; 0.18]	95	[0.25; 0.57]	[4; 4]	[0.1; 0.21]	95
5	[0.46; 0.66]	[5; 5]	[0.09; 0.22]	130	[0.53; 0.75]	[5; 5]	[0.09; 0.23]	130
6	[0.62; 0.78]	[6; 6]	[0.12; 0.21]	168	[0.37; 0.78]	[6; 6]	[0.1; 0.22]	168
7	[0.42; 0.78]	[7; 7]	[0.13; 0.22]	202	[0.67; 0.84]	[7; 7]	[0.12; 0.23]	202
8	[0.46; 0.74]	[8; 8]	[0.14; 0.2]	237	[0.55; 0.8]	[8; 8]	[0.14; 0.24]	237
9	[0.56; 0.87]	[9; 9]	[0.14; 0.19]	282	[0.6; 0.84]	[9; 9]	[0.13; 0.24]	282
10	[0.54; 0.8]	[10; 10]	[0.15; 0.2]	320	[0.6; 0.8]	[10; 10]	[0.15; 0.25]	320

Table 3.1 – Comparison between the **a-BLS** method and **a-OWLS** method in dimension 1, for the uniform and the Gaussian measure using $s = 2$, $\delta = 0.9$, $\eta = 0.01$, $M = 10$ and $t = 10$.

Table 3.1 shows that with the **a-BLS** method, we ensure the stability of the empirical Gram matrix with a number of samples $\#K_l = m_l$ and this for both the uniform and gaussian measures. Compared to the **a-OWLS** method using $n_l = \lceil d_\delta^{-1} \ln(\zeta(s)m_l^{s+1}\eta^{-1}) \rceil m_l$ for all $l = 1, \dots, t$, which is the condition given by Theorem 3.3, the number of evaluations is divided by at least 20 for $l \geq 3$.

In dimension $d > 1$, we consider the nested sequence of hyperbolic cross polynomial approximation spaces, $(V_{m_l})_{l=1}^t$, where for each $l \geq 1$,

$$V_{m_l} = \mathbb{P}_{\Lambda_l} = \text{span}\{\varphi_i(x) = \prod_{k=1}^d x_k^{i_k}, i \in \Lambda_l\}, \text{ where } \Lambda_l = \{i = (i_1, \dots, i_d) : \prod_{k=1}^d (i_k + 1) \leq l\}. \quad (3.19)$$

For an illustration of an hyperbolic cross indices sets for $d = 2$, see Figure 2.8 from Chapter 2.

Both for the uniform and the gaussian measures, Tables 3.2, 3.3 and 3.4 show respectively for $d = 2, 5$ and 8 that the number of samples necessary to ensure the stability condition is slightly larger than the dimension of the approximation space m_l . However, the gain compared to n_l is important, more precisely for $d = 2$ the number of evaluations is divided by at least 20 between the

Gaussian measure					Uniform measure			
a-BLS			a-OWLS		a-BLS		a-OWLS	
m_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\mathbf{x}^{n_l}}^{m_l}$	n_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\mathbf{x}^{n_l}}^{m_l}$	n_l
1	[0; 0]	[2; 2]	[0; 0]	16	[0; 0]	[2; 2]	[0; 0]	16
3	[0.23; 0.4]	[3; 3]	[0.09; 0.28]	62	[0.18; 0.42]	[3; 3]	[0.15; 0.27]	62
5	[0.56; 0.83]	[5; 5]	[0.19; 0.27]	136	[0.47; 0.76]	[5; 6]	[0.18; 0.32]	136
8	[0.64; 0.84]	[8; 8]	[0.25; 0.3]	256	[0.69; 0.82]	[8; 10]	[0.27; 0.38]	256
10	[0.77; 0.89]	[10; 11]	[0.22; 0.32]	348	[0.78; 0.89]	[10; 12]	[0.29; 0.38]	348
14	[0.77; 0.89]	[14; 16]	[0.27; 0.34]	532	[0.81; 0.88]	[15; 17]	[0.31; 0.41]	532
16	[0.78; 0.89]	[16; 19]	[0.26; 0.35]	616	[0.82; 0.87]	[18; 22]	[0.31; 0.39]	616
20	[0.84; 0.9]	[21; 23]	[0.28; 0.37]	808	[0.82; 0.89]	[24; 27]	[0.32; 0.37]	808
23	[0.86; 0.89]	[24; 28]	[0.29; 0.38]	943	[0.83; 0.9]	[28; 34]	[0.33; 0.39]	943
27	[0.83; 0.9]	[30; 35]	[0.29; 0.37]	1154	[0.85; 0.89]	[34; 38]	[0.33; 0.37]	1154

Table 3.2 – Comparison between the **a-BLS** method and **a-OWLS** method in dimension 2, for the uniform and the Gaussian measures using $s = 2$, $\delta = 0.9$, $\eta = 0.01$, $M = 10$ and $t = 10$.

Gaussian measure					Uniform measure			
a-BLS			a-OWLS		a-BLS		a-OWLS	
m_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\mathbf{x}^{n_l}}^{m_l}$	n_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\mathbf{x}^{n_l}}^{m_l}$	n_l
1	[0; 0]	[5; 5]	[0; 0]	16	[0; 0]	[5; 5]	[0; 0]	16
6	[0.63; 0.81]	[6; 6]	[0.36; 0.46]	182	[0.57; 0.7]	[6; 6]	[0.41; 0.61]	182
11	[0.84; 0.89]	[11; 13]	[0.44; 0.54]	396	[0.79; 0.89]	[11; 13]	[0.45; 0.57]	396
26	[0.87; 0.9]	[30; 32]	[0.46; 0.56]	1189	[0.87; 0.9]	[30; 37]	[0.56; 0.66]	1189
31	[0.86; 0.9]	[36; 41]	[0.48; 0.62]	1429	[0.87; 0.89]	[42; 49]	[0.51; 0.64]	1429
56	[0.88; 0.9]	[72; 77]	[0.52; 0.61]	2934	[0.88; 0.9]	[79; 91]	[0.56; 0.63]	2934
61	[0.88; 0.9]	[80; 86]	[0.51; 0.62]	3209	[0.89; 0.9]	[98; 111]	[0.53; 0.6]	3209
96	[0.89; 0.9]	[129; 138]	[0.53; 0.62]	5457	[0.89; 0.9]	[156; 174]	[0.58; 0.62]	5457
111	[0.89; 0.9]	[156; 169]	[0.52; 0.61]	6357	[0.89; 0.9]	[200; 218]	[0.55; 0.6]	6357
136	[0.89; 0.9]	[199; 211]	[0.55; 0.62]	8018	[0.9; 0.9]	[259; 287]	[0.52; 0.59]	8018

Table 3.3 – Comparison between the **a-BLS** method and **a-OWLS** method in dimension 5, for the uniform and the Gaussian measures using $s = 2$, $\delta = 0.9$, $\eta = 0.01$, $M = 10$ and $t = 10$.

Gaussian measure					Uniform measure			
a-BLS			a-OWLS		a-BLS		a-OWLS	
m_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\mathbf{x}^{n_l}}^{m_l}$	n_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\mathbf{x}^{n_l}}^{m_l}$	n_l
1	[0; 0]	[8; 8]	[0; 0]	16	[0; 0]	[8; 8]	[0; 0]	16
9	[0.76; 0.89]	[9; 10]	[0.48; 0.64]	317	[0.72; 0.85]	[9; 10]	[0.62; 0.81]	317
17	[0.83; 0.89]	[19; 22]	[0.53; 0.67]	706	[0.87; 0.9]	[20; 25]	[0.54; 0.68]	706
53	[0.89; 0.9]	[69; 77]	[0.6; 0.67]	2767	[0.88; 0.9]	[75; 83]	[0.72; 0.87]	2767
61	[0.89; 0.9]	[86; 94]	[0.61; 0.64]	3260	[0.89; 0.9]	[98; 113]	[0.67; 0.77]	3260

Table 3.4 – Comparison between the **a-BLS** method and **a-OWLS** method in dimension 8, for the uniform and the Gaussian measures using $s = 2$, $\delta = 0.9$, $\eta = 0.01$, $M = 10$ and $t = 5$.

a-OWLS method and the **a-BLS** method (for $l \geq 2$) and even 30 for $l = 10$, in dimension $d = 5$, it is divided by at least 30 (for $l \geq 3$) and 40 for $l = 10$. In dimension $d = 8$ the ratio $\frac{n_l}{\#K_l}$ is also greater than 30 for $l \geq 3$.

Comparison with the non-adaptive optimal boosted least-squares strategy

In this subsection, we compare the efficiency of the **a-BLS** strategy compared to the boosted least-squares method, as defined in Section 3.4 and abbreviated **s-BLS** (without using the fast algorithm described in Appendix A). For a given approximation space V_{m_l} ($l > 1$) of dimension m_l , the **a-BLS** strategy reuses samples from the previous step whereas the **s-BLS** uses only new samples. The efficiencies of the **a-BLS** strategy and the **s-BLS** strategy are respectively measured through $Z_{\tilde{x}_{K_l}}^{m_l}, \#K_l$ and $Z_{\tilde{x}_K}^{m_l}, \#K$. We recall that \tilde{x}_K^n is the sample associated to the distribution $\tilde{\rho}_g^{n_l}$ from Chapter 2 which is the boosted optimal weighted least-squares measure with subsampling and $\#K$ the number of points that are in \tilde{x}_K^n after the greedy subsampling.

Gaussian Measure					Uniform Measure			
a-BLS			s-BLS		a-BLS		s-BLS	
m_l	$Z_{\tilde{x}_{K_l}}^{m_l}$	$\#K_l$	$Z_{\tilde{x}_K}^{m_l}$	$\#K$	$Z_{\tilde{x}_{K_l}}^{m_l}$	$\#K_l$	$Z_{\tilde{x}_K}^{m_l}$	$\#K$
1	[0; 0]	[1; 1]	[0; 0]	[1; 1]	[0; 0]	[1; 1]	[0; 0]	[1; 1]
2	[0; 0.18]	[2; 2]	[0.3; 0.43]	[2; 2]	[0; 0.3]	[2; 2]	[0.33; 0.39]	[2; 2]
3	[0.13; 0.48]	[3; 4]	[0.17; 0.47]	[3; 4]	[0.17; 0.46]	[3; 4]	[0.16; 0.38]	[3; 3]
4	[0.25; 0.44]	[4; 6]	[0.38; 0.5]	[4; 6]	[0.24; 0.49]	[4; 5]	[0.31; 0.42]	[4; 6]
5	[0.22; 0.48]	[5; 7]	[0.31; 0.47]	[5; 7]	[0.3; 0.48]	[5; 6]	[0.33; 0.48]	[5; 6]
6	[0.34; 0.47]	[6; 9]	[0.36; 0.49]	[6; 8]	[0.19; 0.49]	[7; 8]	[0.29; 0.46]	[6; 9]
7	[0.32; 0.49]	[8; 10]	[0.38; 0.47]	[7; 10]	[0.27; 0.47]	[7; 9]	[0.38; 0.49]	[7; 10]
8	[0.37; 0.5]	[8; 12]	[0.34; 0.48]	[9; 11]	[0.34; 0.48]	[8; 10]	[0.42; 0.48]	[9; 12]
9	[0.45; 0.5]	[10; 14]	[0.4; 0.47]	[10; 13]	[0.37; 0.49]	[11; 13]	[0.37; 0.49]	[10; 12]
10	[0.39; 0.48]	[12; 15]	[0.36; 0.49]	[10; 13]	[0.37; 0.49]	[11; 14]	[0.43; 0.48]	[11; 14]

Table 3.5 – Comparison between the **s-BLS** method and the **a-BLS** method in dimension 1, for the uniform and the Gaussian measures using $s = 2$, $\delta = 0.5$, $\eta = 0.01$, $M = 10$ and $t = 10$.

In Table 3.5, when $d = 1$ for a given m_l , we observe that using the **a-BLS** strategy requires almost the same number of evaluations as for the **s-BLS** strategy and this both for the Gaussian and the uniform measures. The advantage is that the adaptive strategy recycles the samples from one step to another.

When $d = 2$, we observe in Table 3.7 that for $m_l \geq 16$, the boosted optimal weighted least-squares method **s-BLS** behaves a bit better than the adaptive boosted optimal weighted least-squares method **a-BLS**, the difference is stronger with the Gaussian measure.

Gaussian Measure					Uniform Measure			
a-BLS			s-BLS		a-BLS		s-BLS	
m_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\tilde{\mathbf{x}}_K^n}^{m_l}$	$\#K$	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\tilde{\mathbf{x}}_K^n}^{m_l}$	$\#K$
1	[0; 0]	[2; 2]	[0; 0]	[2; 2]	[0; 0]	[2; 2]	[0; 0]	[2; 2]
3	[0.16; 0.31]	[3; 3]	[0.36; 0.5]	[3; 4]	[0.1; 0.35]	[3; 3]	[0.15; 0.45]	[3; 4]
5	[0.35; 0.49]	[5; 7]	[0.36; 0.49]	[6; 7]	[0.37; 0.48]	[5; 7]	[0.29; 0.49]	[6; 7]
8	[0.41; 0.49]	[10; 15]	[0.39; 0.49]	[9; 12]	[0.42; 0.49]	[10; 14]	[0.41; 0.5]	[9; 12]
10	[0.42; 0.5]	[13; 21]	[0.42; 0.49]	[13; 15]	[0.42; 0.49]	[15; 20]	[0.47; 0.49]	[14; 16]
14	[0.45; 0.49]	[21; 30]	[0.44; 0.49]	[20; 22]	[0.44; 0.49]	[24; 31]	[0.46; 0.49]	[21; 23]
16	[0.44; 0.49]	[24; 39]	[0.47; 0.5]	[22; 25]	[0.47; 0.5]	[29; 38]	[0.47; 0.5]	[26; 29]
20	[0.46; 0.5]	[31; 55]	[0.47; 0.5]	[28; 33]	[0.46; 0.5]	[39; 68]	[0.48; 0.5]	[33; 39]
23	[0.46; 0.5]	[44; 62]	[0.47; 0.5]	[34; 42]	[0.47; 0.5]	[46; 73]	[0.48; 0.5]	[39; 47]
27	[0.47; 0.5]	[54; 71]	[0.48; 0.5]	[40; 45]	[0.48; 0.5]	[63; 87]	[0.48; 0.5]	[51; 59]

Table 3.6 – Comparison between the **s-BLS** method and the **a-BLS** method in dimension 2, for the uniform and the Gaussian measures using $s = 2$, $\delta = 0.5$, $\eta = 0.01$, $M = 10$ and $t = 10$.

Gaussian Measure					Uniform Measure			
a-BLS			s-BLS		a-BLS		s-BLS	
m_l	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\tilde{\mathbf{x}}_K^n}^{m_l}$	$\#K$	$Z_{\tilde{\mathbf{x}}_{K_l}^{m_l}}^{m_l}$	$\#K_l$	$Z_{\tilde{\mathbf{x}}_K^n}^{m_l}$	$\#K$
1	[0; 0]	[5; 5]	[0; 0]	[5; 5]	[0; 0]	[0; 0]	[0; 0]	[0; 0]
6	[0.32; 0.47]	[6; 8]	[0.31; 0.49]	[8; 11]	[0.34; 0.5]	[6; 10]	[0.38; 0.48]	[7; 11]
11	[0.46; 0.49]	[18; 25]	[0.42; 0.5]	[20; 28]	[0.46; 0.49]	[18; 24]	[0.44; 0.49]	[20; 27]
26	[0.48; 0.5]	[61; 79]	[0.46; 0.49]	[66; 75]	[0.49; 0.5]	[66; 84]	[0.48; 0.5]	[71; 92]
31	[0.49; 0.5]	[89; 115]	[0.46; 0.5]	[80; 104]	[0.49; 0.5]	[100; 126]	[0.48; 0.5]	[98; 118]
56	[0.49; 0.5]	[192; 253]	[0.48; 0.5]	[171; 203]	[0.49; 0.5]	[224; 258]	[0.49; 0.5]	[203; 256]

Table 3.7 – Comparison between the **s-BLS** method and the **a-BLS** method in dimension 5, for the uniform and the Gaussian measures using $s = 2$, $\delta = 0.5$, $\eta = 0.01$, $M = 10$ and $t = 5$.

When $d = 5$, we observe in Table 3.7 that for $m_l \geq 11$, the boosted optimal weighted least-squares method **s-BLS** behaves a bit better than the adaptive boosted optimal weighted least-squares method **a-BLS**, the difference is stronger with the Gaussian measure.

3.5.2 Illustration for polynomial approximation

Basis adaptation

Here we describe adaptive strategies for constructing a sequence of polynomial approximation spaces. This sequence will then be used to construct the boosted least-squares projection and perform adaptive approximation. When $d = 1$, we simply successively increase the degree of the polynomial space. In higher dimension, we adopt the following strategy originally proposed in [43]

and adapted for interpolation in [18] with bulk chasing procedure with θ a given parameter.

Let $\Lambda_l \subset \mathbb{N}^d$, such that $\#\Lambda_l = m_l$ and we let $V_{m_l} = \text{span}\{\varphi_i : i \in \Lambda_l\}$. Λ_l is a downward closed set (or lower set) if

$$\forall i \in \Lambda_l, i^* \leq i \Rightarrow i^* \in \Lambda_l.$$

We recall that for $i^* = (i_1^*, \dots, i_d^*)$ and $i = (i_1, \dots, i_d)$, the inequality $i^* \leq i$ means that $i_k^* \leq i_k$ for all $1 \leq k \leq d$. For Λ_l a downward closed set, we define its reduced margin,

$$\mathcal{M}_r(\Lambda_l) = \{i \in \mathbb{N}^d \setminus \Lambda_l : \forall k \text{ such that } i_k > 1, i - e_k \in \Lambda_l\}. \quad (3.20)$$

where $e_k \in \mathbb{N}^d$ is such that $(e_k)_j = \delta_{jk}$, for $k, j \in \mathbb{N}$.

At each step l , the Algorithm 3.3 selects a subset N_l of multi-indices in the reduced margin of the downward closed set Λ_l . The boosted least-squares projection associated to the set $\Lambda_l \cup N_l$ is computed and a corresponding error is calculated, the procedure is repeated until a certain tolerance is reached for this error.

In the next example, we want to compare the different methods for adaptive approximation:

Algorithm 3.3 Adaptive approximation for boosted least squares projection

Inputs: δ, η, M, θ , desired tolerance \mathcal{E} , function u .

Outputs: $\tilde{\mathbf{x}}_{K_l}^{n_l}, Q_{V_{m_l}}^{\tilde{\mathbf{x}}_{K_l}^{n_l}} u, \varepsilon_l$.

Initialization

Set $l = 1, \Lambda_l = \{1, \dots, 1\}, V_{m_l} = \text{span}\{\varphi_i : i \in \Lambda_l\}$.

Sample $\tilde{\mathbf{x}}_{K_l}^{n_l}$ from $\tilde{\gamma}_g^{n_l}$ associated to the space V_{m_l} and set $\varepsilon_l = 1$.

Adaptation

while $\varepsilon_l > \mathcal{E}$ and $l < t$ **do**

 Compute $\mathcal{M}_r(\Lambda_l)$ the reduced margin of the index set Λ_l .

 Set $\Lambda_l^* = \Lambda_l \cup \mathcal{M}_r(\Lambda_l)$ and $V_{m_l}^* = \text{span}\{\varphi_i : i \in \Lambda_l^*\}$.

 Sample $\tilde{\mathbf{x}}_{K_l}^{n_l}$ from $\tilde{\gamma}_g^{n_l}$ associated to the space $V_{m_l}^*$.

 Compute $Q_{V_{m_l}^*}^{\tilde{\mathbf{x}}_{K_l}^{n_l}} u = \sum_{i \in \Lambda_l^*} a_i \varphi_i$.

 Select $N_l \subset \mathcal{M}_r(\Lambda_l)$ the smallest set such that $\sum_{i \in N_l} a_i^2 \geq \theta \sum_{i \in \mathcal{M}_r(\Lambda_l)} a_i^2$.

 Update $\Lambda_{l+1} = \Lambda_l \cup N_l$ and $V_{m_{l+1}} = \text{span}\{\varphi_i : i \in \Lambda_{l+1}\}$.

 Sample $\tilde{\mathbf{x}}_{K_{l+1}}^{n_{l+1}}$ from $\tilde{\gamma}_g^{n_{l+1}}$ associated to the space $V_{m_{l+1}}$.

 Compute $Q_{V_{m_{l+1}}}^{\tilde{\mathbf{x}}_{K_{l+1}}^{n_{l+1}}} u = \sum_{i \in \Lambda_{l+1}} a_i \varphi_i$.

 Compute $\varepsilon_l^2 = \frac{\sum_{i \in N_l} a_i^2}{\sum_{i \in \Lambda_l^*} a_i^2}$.

 Update $l = l + 1$.

end while

adaptive interpolation with magic points (abbreviated **a- \mathcal{I} -Magic**), adaptive optimal weighted

least-squares (abbreviated **a-OWLS**) and the method presented in this chapter (abbreviated **a-BLS**), in terms of approximation w.r.t complexity. The quality of the approximation u^\star of a function $u \in L_\mu^2(\mathcal{X})$ is assessed by estimating the error of approximation with

$$\varepsilon(u^\star) = \left(\frac{1}{n_{test}} \sum_{x \in \mathbf{x}_{test}} (u(x) - u^\star(x))^2 \right)^{1/2}.$$

In practice, we choose $n_{test} = 1000$. To study the robustness of the methods, we compute 10 times the approximations and draw 10 different test samples \mathbf{x}_{test} and compute empirical confidence intervals of level 10% and 90% for the errors of approximation.

A first example with a bivariate function

Here, we consider $\mathcal{X} = [-1, 1]^d$, $d = 2$ equipped with the uniform measure and the function

$$u(x) = \frac{1}{(1 - \frac{0.5}{2d} \sum_{i=1}^d x_i)^{d+1}} \quad (3.21)$$

Figure 3.1 shows that the number of samples necessary to reach a certain precision with the **a-OWLS**-methods is much greater than the one used for the **a- \mathcal{I} -Magic** method or the **a-BLS**. The **a-BLS** reaches the desired tolerance $\mathcal{E} = 10^{-5}$ whereas the **a- \mathcal{I} -Magic** method does not, furthermore, the number of samples n_l necessary to reach a certain tolerance is greater with the **a- \mathcal{I} -Magic** method than with the **a-BLS** method.

Now, we consider that we only have access to noisy evaluations, $y^i = u(x^i) + e^i$ where e^i are i.i.d realizations from a Gaussian variable $e \sim \mathcal{N}(0, \sigma)$.

Figure 3.2 shows that the same conclusions can be drawn in the noisy case than in the previous one (see figure 3.1). However due to the noise, none of these methods reach the required precision in this case. We notice that both the **a-BLS** method and the **a-OWLS** are more stable than the **a- \mathcal{I} -Magic** method.

Figure 3.3 shows that, decreasing the noise level increases the reached precision. As in the previous case (with high noise-level), none of these methods reach the required precision in this case. We notice that both the **a-BLS** method and the **a-OWLS** are more stable than the **a- \mathcal{I} -Magic** method. Furthermore the **a-BLS** method requires fewer evaluations than the **a- \mathcal{I} -Magic** method to reach a certain precision.

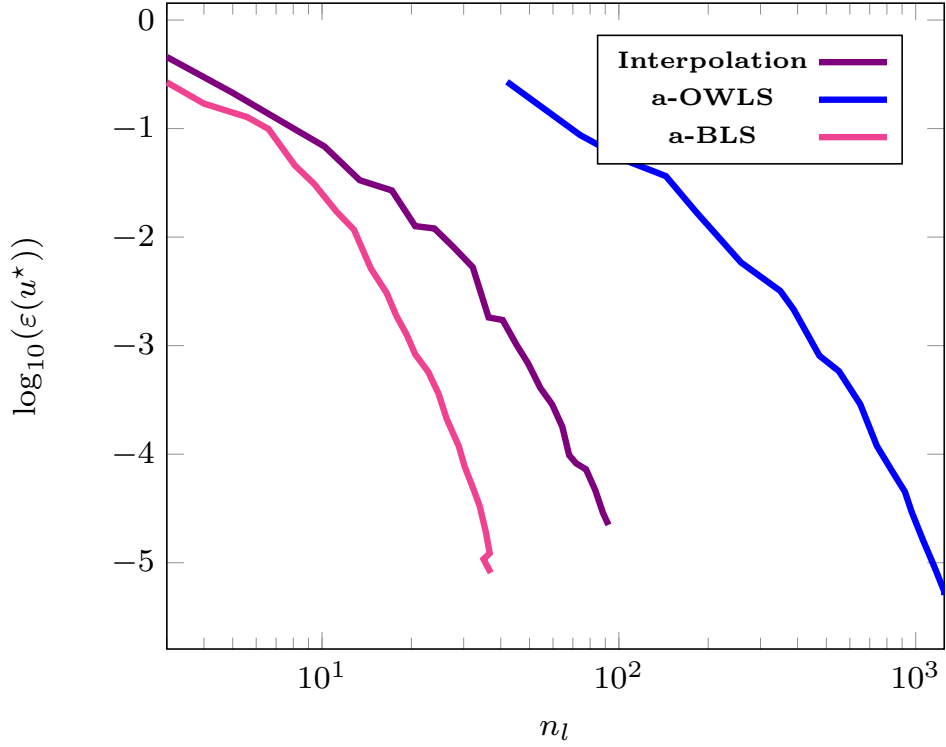


Figure 3.1 – Illustration of the Algorithm 3.3 on the first example (3.21) with required precision $\mathcal{E} = 10^{-5}$, $\delta = 0.9$ and $\eta = 0.01$.

3.6 Conclusion

We have proposed a method to construct a sequence of boosted least-squares estimators associated to a nested sequence of approximation spaces. Even if the numerical results are promising, we did not manage to show that for each $l > 1$, the constructed least-squares estimators are stable in expectation. The proposed strategy has an overall cost n_l that is in practice of the order of m_l . It remains to understand the conditions for observing this nice property.

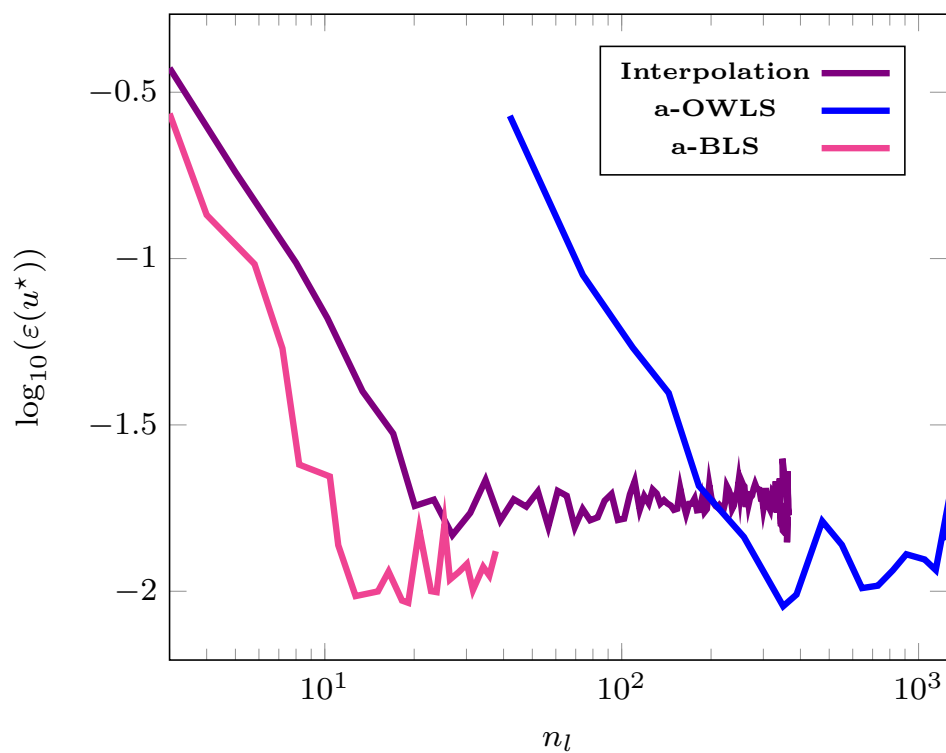


Figure 3.2 – Illustration of the Algorithm 3.3 for the first example (3.21) with noise with required precision $\mathcal{E} = 10^{-5}$, $\delta = 0.9$, $\eta = 0.01$ and $\sigma = 0.01$.

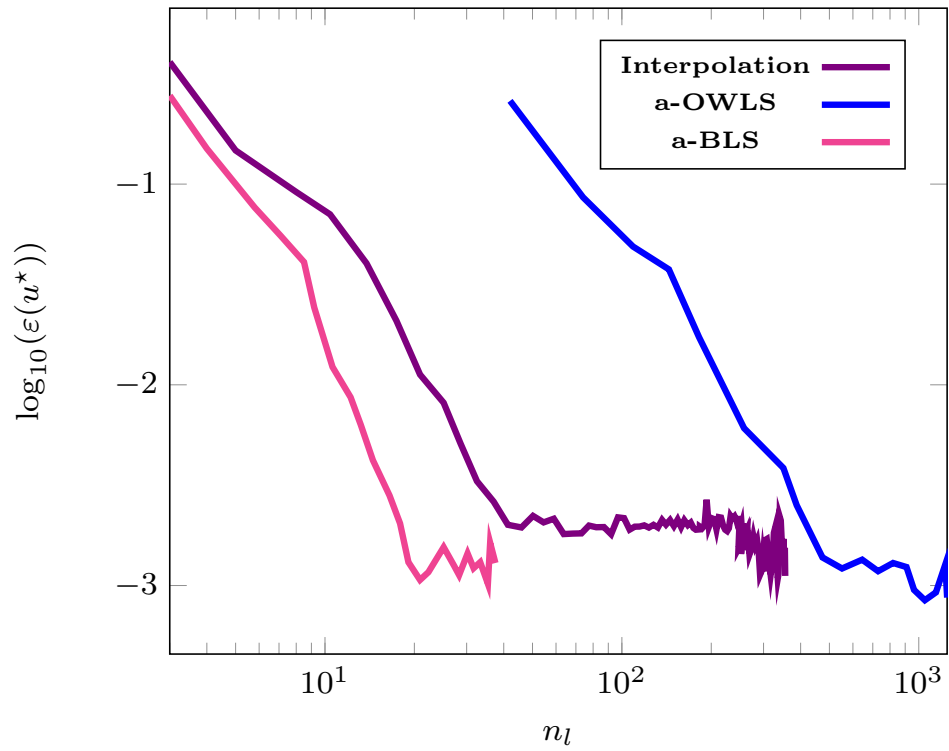


Figure 3.3 – Illustration of the Algorithm 3.3 for the first example (3.21) with noise with required precision $\mathcal{E} = 10^{-5}$, $\delta = 0.9$, $\eta = 0.01$ and $\sigma = 0.001$.

TREE-BASED TENSOR FORMATS

Contents

4.1	Introduction	95
4.2	Tensor spaces	95
4.3	Tensor ranks and tree-based tensor formats	96
4.3.1	Dimension partition tree	97
4.3.2	Tree tensor networks and their representation	98
4.4	Principal Component Analysis for multivariate functions	101
4.4.1	α -principal subspaces	101
4.4.2	Accuracy of the empirical α -principal subspaces	102
4.4.3	The case of functions with Sobolev regularity	103
4.5	Approximation power of tree tensor networks	105
4.5.1	Truncation in tree-based format	105
4.5.2	Approximation rates for Sobolev functions	106

4.1 Introduction

This chapter is devoted to the presentation of model classes of rank-structured functions with a focus on tree-based tensor formats. In Section 4.2, we introduce the tensor spaces of multivariate functions. We present in Section 4.3 the tree-based tensor formats. In Section 4.4, we present the extension of principal component analysis (PCA) for multivariate functions. In Section 4.5, we also recall results from [84] and [7] that provide conditions on the complexity of the tree-based tensor necessary to approximate a function with Sobolev regularity.

4.2 Tensor spaces

Let \mathcal{X} be a subset of \mathbb{R}^d with a product structure $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_d$ and $\mu = \mu_1 \otimes \dots \otimes \mu_d$ be a product measure on \mathcal{X} . For each $1 \leq \nu \leq d$, let us consider \mathcal{H}_ν a Hilbert space of univariate functions defined on \mathcal{X}_ν , equipped with the inner product $(\cdot, \cdot)_{\mathcal{H}_\nu}$ and the associated norm $\|\cdot\|_{\mathcal{H}_\nu}$.

The elementary tensor product $v^1 \otimes \dots \otimes v^d$ of d univariate functions $v^\nu \in \mathcal{H}_\nu$ is defined as the multivariate function v such that for $x = (x_1, \dots, x_d) \in \mathcal{X}$, $v(x) = (v^1 \otimes \dots \otimes v^d)(x_1, \dots, x_d) = v^1(x_1) \dots v^d(x_d)$. The span of these elementary tensor products is the algebraic tensor space $\mathcal{H}_1 \otimes_a \dots \otimes_a \mathcal{H}_d$, which is the set of functions v that can be written as a finite linear combination of elementary tensors [50]. A canonical inner product over this algebraic tensor space is defined by $(v^1 \otimes \dots \otimes v^d, u^1 \otimes \dots \otimes u^d) = (v^1, u^1)_{\mathcal{H}_1} \dots (v^d, u^d)_{\mathcal{H}_d}$. The associated canonical norm, denoted $\|\cdot\|$, verifies $\|v^1 \otimes \dots \otimes v^d\| = \|v^1\|_{\mathcal{H}_1} \dots \|v^d\|_{\mathcal{H}_d}$. A tensor Hilbert space \mathcal{H} is defined as the completion of this algebraic tensor space $\mathcal{H} = \mathcal{H}_1 \otimes \dots \otimes \mathcal{H}_d = \overline{\mathcal{H}_1 \otimes_a \dots \otimes_a \mathcal{H}_d}^{\|\cdot\|_{\mathcal{H}}}$. We will use the notation $\mathcal{H} = \mathcal{H}_1 \otimes \dots \otimes \mathcal{H}_d = \bigotimes_{\nu=1}^d \mathcal{H}_\nu$.

For each $1 \leq \nu \leq d$, we consider $L_{\mu_\nu}^2(\mathcal{X}_\nu)$, the Hilbert space of square-integrable, real-valued univariate functions defined on \mathcal{X}_ν , equipped with the probability measure μ_ν . The natural norm in $L_{\mu_\nu}^2(\mathcal{X}_\nu)$, denoted $\|\cdot\|_{L_{\mu_\nu}^2(\mathcal{X}_\nu)}$, is defined for $v^\nu \in L_{\mu_\nu}^2(\mathcal{X}_\nu)$ by

$$\|v^\nu\|_{L_{\mu_\nu}^2}^2 = \int_{\mathcal{X}_\nu} v^\nu(x_\nu)^2 d\mu_\nu(x_\nu).$$

The space $L_\mu^2(\mathcal{X})$ of square-integrable, real-valued functions defined on \mathcal{X} equipped with the canonical norm $\|v\|_{L_\mu^2}^2 = \int_{\mathcal{X}} v(x)^2 d\mu(x)$ can be identified with the completion of the algebraic tensor space $L_{\mu_1}^2(\mathcal{X}_1) \otimes \dots \otimes L_{\mu_d}^2(\mathcal{X}_d)$.

From now, for the clarity of the notations and when there is no ambiguity $L_\mu^2(\mathcal{X})$ will be simply denoted L_μ^2 , the norm $\|\cdot\|_{L_\mu^2}$ and associated inner product $(\cdot, \cdot)_{L_\mu^2}$ will be denoted $\|\cdot\|$ and (\cdot, \cdot) respectively.

4.3 Tensor ranks and tree-based tensor formats

The canonical rank r of a tensor $u \in \mathcal{H}$ is the minimal integer such that the multivariate function u can be written under the form

$$u(x_1, \dots, x_d) = \sum_{k=1}^r v_k^1(x_1) \dots v_k^d(x_d), \quad (4.1)$$

for v_k^ν functions in \mathcal{H}_ν . An approximation in the form (4.1) is called an approximation in canonical tensor format. This format has several drawbacks when $d > 2$ (see [85] and [56]) and is therefore not very convenient for computational purposes. This is the reason why other notions of ranks have been introduced. Of particular interest here is the notion of α -rank.

Let α denote a non-empty subset of $D = \{1, \dots, d\}$. We introduce $x_\alpha = (x_\nu)_{\nu \in \alpha}$, $\mu_\alpha = \bigotimes_{\nu \in \alpha} \mu_\nu$,

$\mathcal{X}_\alpha = \times_{\nu \in \alpha} \mathcal{X}_\nu$, and for its complementary set $\alpha^c = D \setminus \alpha$, $x_{\alpha^c} = (x_\nu)_{\nu \in \alpha^c}$, $\mu_{\alpha^c} = \otimes_{\nu \in \alpha^c} \mu_\nu$ and $\mathcal{X}_{\alpha^c} = \times_{\nu \in \alpha^c} \mathcal{X}_\nu$. $\mathcal{H}_\alpha = \otimes_{\nu \in \alpha} \mathcal{H}_\nu$ and $\mathcal{H}_{\alpha^c} = \otimes_{\nu \in \alpha^c} \mathcal{H}_\nu$ denote respectively the corresponding Hilbert spaces of functions of groups of variables x_α and x_{α^c} .

A multivariate function $u(x_1, \dots, x_d)$ in \mathcal{H} can be identified with a bivariate function $u(x_\alpha, x_{\alpha^c})$, implicitly considering the reordering of the variables, which is an order-two tensor denoted by $\mathcal{M}_\alpha(u) \in \mathcal{H}_\alpha \otimes \mathcal{H}_{\alpha^c}$ and called the α -matricization of the tensor u . In the literature the terms unfoldings and flattenings are also employed. The α -rank of the tensor u is the canonical rank of the order-two tensor $\mathcal{M}_\alpha(u)$, denoted $\text{rank}_\alpha(u) = \text{rank}(\mathcal{M}_\alpha(u))$, which is by definition the minimal integer such that there exist functions $v_k^\alpha \in \mathcal{H}_\alpha$ and $v_k^{\alpha^c} \in \mathcal{H}_{\alpha^c}$ verifying

$$u(x_1, \dots, x_d) = \sum_{k=1}^{\text{rank}_\alpha(u)} v_k^\alpha(x_\alpha) v_k^{\alpha^c}(x_{\alpha^c}). \quad (4.2)$$

We also consider cases where α -ranks are infinite.

For a function u that admits the representation (4.2), the space $\overline{\text{span}\{v_k^\alpha\}_{k=1}^{\text{rank}_\alpha(u)} \cdot \|\cdot\|_{\mathcal{H}_\alpha}}$ is called the α -minimal subspace of u , denoted $U_\alpha^{\min}(u)$ and $\text{rank}_\alpha(u) = \dim(U_\alpha^{\min}(u)) = \dim(U_{\alpha^c}^{\min}(u))$ (see [35]).

4.3.1 Dimension partition tree

Considering $T \subset 2^{\{1, \dots, d\}}$, T is a dimension partition tree over $D = \{1, \dots, d\}$, if it has the following properties:

- D is the root of the tree T ,
- a node $\alpha \in T$ is a non empty subset of D , whose cardinality is denoted by $\#\alpha$,
- for each node $\alpha \in T$, the set of sons $S(\alpha)$ of α is either empty, if $\#\alpha = 1$, or forms a partition of α , if $\#\alpha > 1$.

The nodes α such that $\#\alpha = 1$ are the leaves of the tree T and the set of leaves is denoted $\mathcal{L}(T)$. By construction for each $\alpha \in \mathcal{L}(T)$, $S(\alpha) = \emptyset$. As an illustration, Figure 4.1 shows a particular dimension partition tree, with $d = 6$,

$$T = \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{2, 3\}, \{1, 2, 3\}, \{4, 5, 6\}, \{1, 2, 3, 4, 5, 6\}\}.$$

For a node α , $l(\alpha)$ denotes the level of the node α in the tree T . It is defined from the root to the leaves, such that $l(D) = 0$ and if $\beta \in S(\alpha)$, $l(\beta) = l(\alpha) + 1$. The maximum level of the nodes

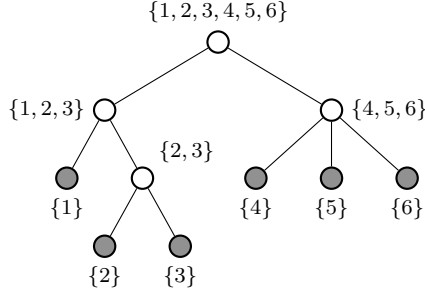


Figure 4.1 – Dimension partition tree with its leaves represented in gray

in T is the depth of the tree $\text{depth}(T) = \max_{\alpha \in T} l(\alpha)$.

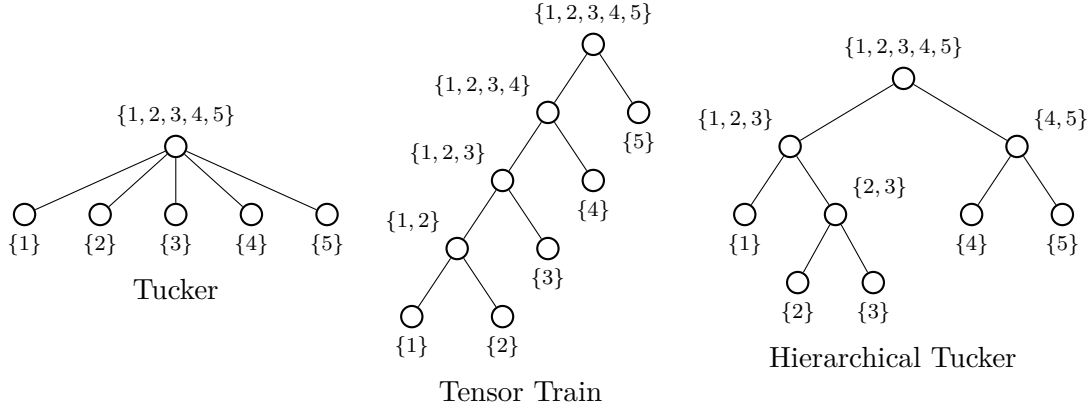


Figure 4.2 – Examples of dimension partition trees

4.3.2 Tree tensor networks and their representation

Considering a function u which admits the representation (4.2) for each node α of a dimension partition tree T , we define the T -rank of the function u , as the tuple $\text{rank}_T(u) = \{\text{rank}_\alpha(u)\}_{\alpha \in T}$, with the convention $\text{rank}_D(u) = 1$. With this notion of T -rank, we define an approximation format, called tree-based tensor format, which is the set of functions with T -rank bounded by $r = (r_\alpha)_{\alpha \in T}$,

$$\mathcal{T}_r^T(V) = \{u \in V : \text{rank}_T(u) \leq r\} = \bigcap_{\alpha \in T} \{u \in V : \text{rank}_\alpha(u) \leq r_\alpha\}, \quad (4.3)$$

where V is a certain finite-dimensional tensor space of multivariate functions.

A tensor format is an intersection of subsets of tensors with bounded α -rank, $\alpha \in T$, these for-

mats inherit most of the nice properties of low-rank approximation formats for order-two tensors. In particular, best approximation problems in the set $\mathcal{T}_r^T(V)$ are well posed [35].

Representation of tree tensor networks We consider a function u in tree-based tensor format $\mathcal{T}_r^T(V)$, whose T -rank is the tuple $(r_\alpha)_{\alpha \in T}$ and $V = \bigotimes_{\nu=1}^d V_\nu$ a finite-dimensional tensor product space. For each node α , the function u can be written under the form (4.2).

If $\alpha = \{\nu\} \in \mathcal{L}(T)$, we denote by $\{v_{k_\nu}^\nu\}_{k_\nu=1}^{r_\nu}$ a basis of the minimal subspace $U_\nu^{\min}(u)$. If $\alpha \notin \mathcal{L}(T)$, we denote its set of sons by $S(\alpha)$ and we have the relation of nestedness between the minimal subspace $U_\alpha^{\min}(u)$ associated to α , and the minimal subspace $U_\beta^{\min}(u)$ associated to $\beta \in S(\alpha)$:

$$U_\alpha^{\min}(u) \subset \bigotimes_{\beta \in S(\alpha)} U_\beta^{\min}(u). \quad (4.4)$$

Hence, if for all $\beta \in S(\alpha)$, $\{v_{k_\beta}^\beta\}_{k_\beta=1}^{r_\beta}$ is a basis of $U_\beta^{\min}(u)$, then

$$U_\alpha^{\min}(u) \subset \text{span}\left\{ \prod_{\beta \in S(\alpha)} v_{k_\beta}^\beta(x_\beta) : 1 \leq k_\beta \leq r_\beta, \beta \in S(\alpha) \right\} \quad (4.5)$$

contains a basis of $U_\alpha^{\min}(u)$.

For each $\alpha \in T \setminus \mathcal{L}(T)$, we let $\Lambda^\alpha = \times_{\beta \in S(\alpha)} \{1, \dots, r_\beta\}$ such that for each $k_\alpha \in \Lambda^\alpha$, $k_\alpha = (k_\beta)_{\beta \in S(\alpha)}$. Thanks to (4.5), each element of a basis $\{v_{k_\alpha}^\alpha\}_{k_\alpha=1}^{r_\alpha}$ of $U_\alpha^{\min}(u)$ can be expressed as

$$v_{k_\alpha}^\alpha(x_\alpha) = \sum_{\substack{1 \leq k_\beta \leq r_\beta \\ \beta \in S(\alpha)}} C_{k_\alpha, (k_\beta)_{\beta \in S(\alpha)}}^\alpha \prod_{\beta \in S(\alpha)} v_{k_\beta}^\beta(x_\beta)$$

where C^α is a tensor of coefficients in $\mathbb{R}^{r_\alpha \times (\times_{\beta \in S(\alpha)} r_\beta)}$.

For example, if $\alpha = \{\beta_1, \beta_2\}$,

$$v_{k_\alpha}^\alpha(x_\alpha) = \sum_{1 \leq k_{\beta_1} \leq r_{\beta_1}} \sum_{1 \leq k_{\beta_2} \leq r_{\beta_2}} C_{k_\alpha, k_{\beta_1}, k_{\beta_2}}^\alpha v_{k_{\beta_1}}^{\beta_1}(x_{\beta_1}) v_{k_{\beta_2}}^{\beta_2}(x_{\beta_2}).$$

Then, the function u can be expressed as

$$u(x) = \sum_{1 \leq k_D \leq n_D} C_{1, k_D}^D v_{k_D}^D(x) = \sum_{\substack{1 \leq k_\beta \leq r_\beta \\ \beta \in S(D)}} C_{1, (k_\beta)_{\beta \in S(D)}}^D \prod_{\beta \in S(D)} v_{k_\beta}^\beta(x_\beta),$$

with C^D a tensor of order $\#S(D)$, $C^D \in \mathbb{R}^{\times_{\beta \in S(D)} r_\beta \times r_D}$. We use the convention $r_D = 1$, $U_D^{\min}(u) = \text{span}\{u\}$.

If for each $\nu \in \{1, \dots, d\}$, $\{\varphi_{k_\nu}^\nu\}_{k_\nu=1}^{m_\nu}$ denotes an orthonormal basis of V_ν , this yields

$$u(x) = \sum_{\substack{1 \leq k_\alpha \leq r_\alpha \\ \alpha \in T}} \prod_{\alpha \in T \setminus \mathcal{L}(T)} C_{k_\alpha, (k_\beta)_{\beta \in S(\alpha)}}^\alpha \prod_{\nu \in \mathcal{L}(T)} \sum_{1 \leq k_\nu \leq m_\nu} \varphi_{k_\nu}^\nu(x_\nu). \quad (4.6)$$

Representation complexity of the tree tensor networks

The representation complexity of a tree-based tensor u is defined by the number of parameters necessary for its representation. The parameters correspond to the tensors of coefficients C^α for $\alpha \in T \setminus \mathcal{L}(T)$ and the coefficients of the basis functions coefficients $v_{k_\nu}^\nu$ for $\nu \in \mathcal{L}(T)$. The representation complexity is thus given by

$$\mathcal{S}(T, r, m) = \sum_{\nu \in \mathcal{L}(T)} m_\nu r_\nu + \sum_{\alpha \in T \setminus \mathcal{L}(T)} r_\alpha \prod_{\beta \in S(\alpha)} r_\beta, \quad (4.7)$$

with $m_\nu = \dim(V_\nu)$ for $\nu \in \mathcal{L}(T)$. Denoting s the arity of the tree, that is to say the maximal number of sons a node can have, $R = \max_{\alpha \in T} r_\alpha$ the maximum α -rank and $M = \max_{\nu \in \mathcal{L}(T)} m_\nu$ the maximum dimension of the bases of the spaces V_ν , as the number of interior nodes in the tree $\#T$ is bounded by $d - 1$, the storage complexity is bounded by

$$\mathcal{S}(T, r, m) \leq dMR + (d - 2)R^{s+1} + R^s. \quad (4.8)$$

Tree-based tensor formats as deep neural networks with multilinear units

For a leaf node $\alpha = \{\nu\} \in \mathcal{L}(T)$, let denote $\varphi^\nu(x_\nu) = (\varphi_{k_\nu}^\nu(x_\nu))_{k_\nu=1}^{m_\nu} \in \mathbb{R}^{m_\nu}$ and C^ν the tensor corresponding to the coefficients of the basis $(v_{k_\nu}^\nu(x_\nu))_{k_\nu=1}^{r_\nu}$ into the basis $(\varphi_{k_\nu}^\nu(x_\nu))_{k_\nu=1}^{m_\nu}$. The tensor C^ν can be identified with a linear function $f^\nu : \mathbb{R}^{n_\nu} \rightarrow \mathbb{R}^{r_\nu}$ and

$$(v_{k_\nu}^\nu(x_\nu))_{k_\nu=1}^{r_\nu} = f^\nu(\varphi^\nu(x_\nu)).$$

For each node $\alpha \in T \setminus \mathcal{L}(T)$, a tensor $C^\alpha \in \mathbb{R}^{r_\alpha \times (\times_{\beta \in S(\alpha)} r_\beta)}$ can be identified with a \mathbb{R}^{r_α} -valued multilinear function $f^\alpha : \times_{\beta \in S(\alpha)} \mathbb{R}^{r_\beta} \rightarrow \mathbb{R}^{r_\alpha}$ and

$$(v_{k_\alpha}^\alpha(x_\alpha))_{k_\alpha=1}^{r_\alpha} = f^\alpha((v_{k_\beta}^\beta(x_\beta))_{\beta \in S(\alpha)}).$$

Finally the function u has the representation

$$u(x) = f^D((v_{k_D}^D(x_\alpha))_{\alpha \in S(D)}),$$

so that u is a composition of multilinear maps, which is a particular class of deep neural networks, known as sum-product networks (or arithmetic circuits).

4.4 Principal Component Analysis for multivariate functions

We here consider functions in L^2_μ . For a subspace $V_\alpha \subset L^2_{\mu_\alpha}(\mathcal{X}_\alpha)$, we denote by P_{V_α} the orthogonal projection from $L^2_{\mu_\alpha}(\mathcal{X}_\alpha)$ to V_α , and by \mathcal{P}_{V_α} the orthogonal projection from $L^2_\mu(\mathcal{X})$ to $V_\alpha \otimes L^2_{\mu_{\alpha^c}}(\mathcal{X}_{\alpha^c})$, such that for all x_{α^c} in \mathcal{X}_{α^c} , $(\mathcal{P}_{V_\alpha} u)(\cdot, x_{\alpha^c}) = P_{V_\alpha} u(\cdot, x_{\alpha^c})$.

4.4.1 α -principal subspaces

Let u be a function of L^2_μ . For each $\alpha \subset D$, the function u admits a singular value decomposition

$$u(x) = \sum_{k=1}^{\text{rank}_\alpha(u)} \sigma_\alpha^k v_k^\alpha(x_\alpha) v_k^{\alpha^c}(x_{\alpha^c}), \text{ with } \|v_k^\alpha\|_{L^2_{\mu_\alpha}(\mathcal{X}_\alpha)} = \|v_k^{\alpha^c}\|_{L^2_{\mu_{\alpha^c}}(\mathcal{X}_{\alpha^c})} = 1. \quad (4.9)$$

Here, $\sigma_\alpha^1 \geq \sigma_\alpha^2 \geq \dots$ are the singular values, which are assumed to be sorted in decreasing order, and $v_k^\alpha \in L^2_{\mu_\alpha}(\mathcal{X}_\alpha)$ and $v_k^{\alpha^c} \in L^2_{\mu_{\alpha^c}}(\mathcal{X}_{\alpha^c})$ are respectively the left and right singular functions. For $r_\alpha \leq \text{rank}_\alpha(u)$, the truncated singular value decomposition of u up to the rank r_α is then given by

$$u_{r_\alpha}(x) = \sum_{k=1}^{r_\alpha} \sigma_\alpha^k v_k^\alpha(x_\alpha) v_k^{\alpha^c}(x_{\alpha^c}). \quad (4.10)$$

The r_α dominant left singular functions, $\{v_k^\alpha\}_{k=1}^{r_\alpha}$, are called the α -principal components of u , while the linear span of these r_α functions, denoted by U_α , is called the α -principal subspace of u . The function $u_{r_\alpha} = \mathcal{P}_{U_\alpha} u$ is optimal for the approximation of u , in the sense that it is the best approximation of u with α -rank r_α bounded by r_α , i.e

$$\|u - \mathcal{P}_{U_\alpha} u\|_{L^2_\mu(\mathcal{X})} = \min_{\substack{v \in L^2_\mu(\mathcal{X}) \\ \text{rank}_\alpha(v) \leq r_\alpha}} \|u - v\|_{L^2_\mu(\mathcal{X})} = \min_{\substack{W_\alpha \subset L^2_{\mu_\alpha}(\mathcal{X}_\alpha) \\ \dim(W_\alpha) \leq r_\alpha}} \|u - \mathcal{P}_{W_\alpha} u\|_{L^2_\mu(\mathcal{X})}. \quad (4.11)$$

We denote by $e_{r_\alpha}^\alpha(u)$ the best approximation error associated to $\mathcal{P}_{U_\alpha} u$, which verifies

$$e_{r_\alpha}^\alpha(u)^2 := \|u - \mathcal{P}_{U_\alpha} u\|_{L^2_\mu(\mathcal{X})}^2 = \sum_{k=r_\alpha+1}^{\text{rank}_\alpha(u)} (\sigma_\alpha^k)^2.$$

Noticing that $\|u - \mathcal{P}_{W_\alpha} u\|_{L^2_\mu(\mathcal{X})}^2 = \mathbb{E} \left(\|u(\cdot, X_{\alpha^c}) - \mathcal{P}_{W_\alpha} u(\cdot, X_{\alpha^c})\|_{L^2_{\mu_\alpha}(\mathcal{X}_\alpha)}^2 \right)$, with X_{α^c} a random variable whose probability distribution is μ_{α^c} , we introduce the empirical α -principal subspace of u , written \hat{U}_α , as the solution of

$$\min_{\substack{\dim(W_\alpha)=r_\alpha \\ W_\alpha \subset L^2_{\mu_\alpha}(\mathcal{X}_\alpha)}} \frac{1}{z_{\alpha^c}} \sum_{k=1}^{z_{\alpha^c}} \|u(\cdot, x_{\alpha^c}^k) - \mathcal{P}_{W_\alpha} u(\cdot, x_{\alpha^c}^k)\|_{L^2_{\mu_\alpha}(\mathcal{X}_\alpha)}^2, \quad (4.12)$$

where the $\{x_{\alpha^c}^k\}_{k=1}^{z_{\alpha^c}}$ are z_{α^c} i.i.d. realizations of X_{α^c} .

$\|u - \mathcal{P}_{\hat{U}_\alpha} u\|$ is the reconstruction error associated to the empirical α -principal subspace. It is a random variable and this is the reason why we are interested in quantifying $\mathbb{E}(\|u - \mathcal{P}_{\hat{U}_\alpha} u\|^2)$. In the next paragraph, we present some recent results obtained in [72] and [25].

4.4.2 Accuracy of the empirical α -principal subspaces

Replacing U_α by \hat{U}_α implies an estimation error. We would like to know how well can the empirical α -principal subspace \hat{U}_α approximate the α -principal subspace U_α . In this paragraph, we detail two existing results which connect the empirical reconstruction error of a function u with the minimal reconstruction error $e_{r_\alpha}^\alpha(u)$.

In a first time, we recall the result from [72] which derives high-probability bounds for this reconstruction error of the empirical α -principal subspace and look into the particular cases where the singular values satisfy polynomial or exponential upper bounds. We recall the theorem from [72] using our notations.

Theorem 4.1. [72, Theorem 2] *Let $u \in L_\mu^2$, and $\{\sigma_\alpha^k\}_{k \geq 1}$ and $\{v_k^\alpha\}_{k \geq 1}$ be respectively the α -singular values and the α -left singular functions of u . Assume that the random variables $\{\Gamma_\alpha^k = \frac{1}{\sigma_\alpha^k}(u(\cdot, X_{\alpha^c}), v_k^\alpha)\}_{k \geq 1}$ are independent and that there is a constant $R > 0$ such that*

$$\sup_{k \geq 1} \sup_{q \geq 1} q^{-1/2} (\mathbb{E} |\Gamma_\alpha^k|^q)^{1/q} \leq R. \quad (4.13)$$

Consider $1 \leq r'_\alpha \leq r_\alpha$ such that $\sigma_\alpha^{r'_\alpha} \geq \sqrt{2} \sigma_\alpha^{r_\alpha+1}$. If the number of samples z_{α^c} verifies

$$z_{\alpha^c} \geq c_1(R) \max \left(r'_\alpha, \sum_{k > r_\alpha} \frac{\sigma_\alpha^k}{\sigma_\alpha^{r'_\alpha}} \right)$$

with $c_1(R)$ a constant depending on R , then it holds

$$\|u - \mathcal{P}_{\hat{U}_\alpha} u\|^2 \leq c(R) e_{r_\alpha}^\alpha(u)^2 \text{ with probability at least } 1 - \exp(-c_2(R) z_{\alpha^c}), \quad (4.14)$$

where $e_{r_\alpha}^\alpha(u)$ and $\|u - \mathcal{P}_{\hat{U}_\alpha} u\|$ are respectively the minimal reconstruction error of u and the reconstruction error of the empirical α -principal subspace of u , and $c_2(R)$ and $c(R)$ two other constants depending on R .

This theorem is applied in two particular cases of interest, when the singular values satisfy either a polynomial or exponential upper bound, as summarized in the following corollaries.

Corollary 4.2. [72, Corollary 3] Assume the Equation (4.13) of Theorem 4.1 is verified and that for some $\beta > 1$ and $K > 0$, it holds $(\sigma_\alpha^k)^2 \leq Kk^{-\beta}$ for all $k \geq 1$.

If $z_{\alpha^c} \geq r_\alpha c_1(R, K, \beta)$, then

$$\|u - \mathcal{P}_{\hat{U}_\alpha} u\|^2 \leq c(R, K, \beta) e_{r_\alpha}^\alpha(u)^2 \text{ with probability at least } 1 - \exp(-c_2(R, K, \beta) z_{\alpha^c}),$$

with $c(R, K, \beta)$, $c_1(R, K, \beta)$ and $c_2(R, K, \beta)$ constants depending on β, K and R .

Corollary 4.3. [72, Corollary 4] Assume the Equation (4.13) of Theorem 4.1 is verified and that the singular values of u satisfy an exponential upper bound, i.e. for some $\gamma \in (0, 1]$ and $\beta, K > 0$ it holds $K^{-1} \exp(-\beta k^\gamma) \leq (\sigma_\alpha^k)^2 \leq K \exp(-\beta k^\gamma)$ for all $k \geq 1$.

If $z_{\alpha^c} \geq r_\alpha c_1(R, K, \beta, \gamma)$, then

$$\|u - \mathcal{P}_{\hat{U}_\alpha} u\|^2 \leq c(R, K, \beta, \gamma) e_{r_\alpha}^\alpha(u)^2 \text{ with probability at least } 1 - \exp(-c_2(R, K, \beta, \gamma) z_{\alpha^c}),$$

with $c(R, K, \beta, \gamma)$, $c_1(R, K, \beta, \gamma)$ and $c_2(R, K, \beta, \gamma)$ constants depending on β, K, γ and R .

In practice, the assumption on the independence of the Γ_α^k but also the assumption (4.13) is restrictive and we would like to work with functions that do not necessary verify it.

Secondly, in [25], the authors show that in the case where the minimal reconstruction error $e_{r_\alpha}^\alpha(u)$ has a certain algebraic decay, the same rate of convergence can be obtained for $\|u - \mathcal{P}_{\hat{U}_\alpha} u\|$ the reconstruction error of the empirical α -principal subspace of u if the number of samples z_{α^c} is chosen sufficiently high.

Theorem 4.4. [25] Assume that $C_4 := \int_{\mathcal{X}_{\alpha^c}} \|u(\cdot, X_{\alpha^c})\|_{L_{\mu_\alpha}^2}^4 d\mu_{\alpha^c} < \infty$. For all $r_\alpha, z_{\alpha^c} \geq 0$, one has

$$0 \leq \mathbb{E}(\|u - \mathcal{P}_{\hat{U}_\alpha} u\|^2) - e_{r_\alpha}^\alpha(u)^2 \leq C_4^{1/2} \left(\frac{r_\alpha}{z_{\alpha^c}} \right)^{1/2}.$$

In particular, when $e_{r_\alpha}^\alpha(u)^2$ has an algebraic decay with a certain rate r_α^{-s} for some $s > 1$, the same rate can be obtained for $\|u - \mathcal{P}_{\hat{U}_\alpha} u\|^2$ if the number of samples z_{α^c} is greater than r_α^{1+2s} .

Corollary 4.5. [25] Assume that $C_4 := \int_{\mathcal{X}_{\alpha^c}} \|u(\cdot, X_{\alpha^c})\|_{L_{\mu_\alpha}^2}^4 d\mu_{\alpha^c} < \infty$. If $e_{r_\alpha}^\alpha(u)^2 \leq C r_\alpha^{-s}$ and $z_{\alpha^c} \geq r_\alpha^{1+2s}$, we have that

$$\mathbb{E}(\|u - \mathcal{P}_{\hat{U}_\alpha} u\|^2) \leq (C + C_4^{1/2}) r_\alpha^{-s}.$$

4.4.3 The case of functions with Sobolev regularity

The decay of singular values can be estimated for some classical regularity classes. Here we assume \mathcal{X} is compact and μ is the uniform measure.

Functions with Sobolev regularity

Sobolev classes. The Sobolev space of functions u denoted $W_\mu^{s,2}$ is the space of functions u with bounded norms

$$\|u\|_{W_\mu^{s,2}}^2 = \sum_{|\mathbf{k}| \leq s} \|D^{\mathbf{k}}u\|_{L_\mu^2}^2$$

where for $\mathbf{k} \in \mathbb{N}^d$, $|\mathbf{k}| = k_1 + \dots + k_d$ and $D^{\mathbf{k}}u(x_1, \dots, x_d) = \frac{\partial^{|\mathbf{k}|}u}{\partial x_1^{k_1} \dots \partial x_d^{k_d}}(x_1, \dots, x_d)$.

We recall that the Kolmogorov r -width of a subset of functions $\mathcal{U} \subset \mathcal{H}$ represents the best approximation rate that can be achieved by an approximation g in a linear space V_r of dimension r , see [61]. It is defined by

$$d_r(\mathcal{U})_{\mathcal{H}} = \inf_{\dim(V_r)=r} \sup_{f \in \mathcal{U}} \inf_{g \in V_r} \|f - g\|.$$

The behaviour of the Kolmogorov r -width of Sobolev balls has been studied in [63] and [90]. The Kolmogorov r -width of the unit ball of $W_\mu^{s,2}$ is $d_r(W_\mu^{s,2})_{L_\mu^2} \sim r^{-s/d}$. This implies that to approximate a function u with tolerance ε , r has to be such that $r \sim \varepsilon^{-d/s}$.

From the above result and the relation between the Kolmogorov r -width and the minimal reconstruction error, the authors in [7] showed the following result.

Proposition 4.6. [7] *If $u \in W_\mu^{s,2}(\mathcal{X})$, then*

$$e_{r_\alpha}^\alpha(u) \leq C(s, d_\alpha) r_\alpha^{-s/d_\alpha} \|u\|_{W_\mu^{s,2}}, \quad d_\alpha = \min\{\#\alpha, d - \#\alpha\}$$

with $C(s, d_\alpha)$ a constant depending on s and d_α .

The proposition 4.6 implies that to reach a certain accuracy ε for the minimal reconstruction error, the rank r_α has to be of the order of $\varepsilon^{-d_\alpha/s}$.

Mixed Sobolev classes. The mixed Sobolev space denoted $W_{\mu, \text{mix}}^{s,2}$ is the space of functions u with bounded norms

$$\|u\|_{W_{\mu, \text{mix}}^{s,2}}^2 = \sum_{\mathbf{k} \leq s} \|D^{\mathbf{k}}u\|_{L_\mu^2}^2.$$

The behaviour of the Kolmogorov r -width of balls of mixed Sobolev spaces has been studied in [89]. The Kolmogorov r -width of the unit ball of $u \in W_{\mu, \text{mix}}^{s,2}$, is such that $d_r(W_{\mu, \text{mix}}^{s,2})_{L_\mu^2} \sim r^{-s} \log(r)^{s(d-1)}$. Using [84, Lemma 1], this implies that to approximate a function u with tolerance ε , r has to be of the order of $r \sim s^{-(d-1)} \varepsilon^{-1/s} |\log(\varepsilon)|^{(d-1)}$.

Again from the above result and the relation between the Kolmogorov r -width and the minimal reconstruction error, [7] showed the following result.

Proposition 4.7. [7] If $u \in W_{\mu, \text{mix}}^{s,2}(\mathcal{X})$, then

$$e_{r_\alpha}^\alpha(u) \leq (C(s, d_\alpha) r_\alpha^{-s} \log(r_\alpha)^{s(d_\alpha-1)}) \|u\|_{W_{\mu, \text{mix}}^{s,2}}, \quad d_\alpha = \min\{\#\alpha, d - \#\alpha\}$$

with $C(s, d_\alpha)$ a constant depending on s and d_α .

4.5 Approximation power of tree tensor networks

4.5.1 Truncation in tree-based format

Let T be a dimension tree, $V = \bigotimes_{\nu=1}^d V_\nu$ a finite-dimensional tensor product space and $u \in L_\mu^2(\mathcal{X})$. Consider a collection of α -principal subspaces U_α of dimension r_α with $\alpha \in T \setminus \{D\}$, defined as the solutions of Equation (4.11). The approximation u^* obtained by successive suitably ordered orthogonal projections,

$$u^* = \prod_{\alpha \in T \setminus \{D\}} \mathcal{P}_{U_\alpha} u \quad (4.15)$$

satisfies $u^* \in \mathcal{T}_r^T(V)$ with $V = V_1 \otimes \dots \otimes V_d$, $r = (r_\alpha)_{\alpha \in T}$ and

$$\|u - u^*\|^2 \leq \sum_{\alpha \in T \setminus \{D\}} e_{r_\alpha}^\alpha(u)^2. \quad (4.16)$$

For a proof see e.g [53] or [46].

If for all $\alpha \in T$, the α -rank is chosen such that

$$e_{r_\alpha}^\alpha(u)^2 \leq \frac{\varepsilon^2}{\#T - 1} \|u\|^2,$$

then u^* provides an approximation of u with relative precision ε , i.e.,

$$\|u - u^*\| \leq \varepsilon \|u\|.$$

The approximation u^* obtained by the truncated higher-order singular value decomposition (HOSVD) is a quasi-best approximation of u in $\mathcal{T}_r^T(V)$ satisfying the following Theorem proved in [46]. We state this result for $u \in L_\mu^2(\mathcal{X})$.

Theorem 4.8. [46, Theorem 17]. Let $u \in L_\mu^2(\mathcal{X})$. Consider T a dimension partition tree and the collection of α -principal component subspaces U_α of u with dimension r_α , determined by Equation (4.11). Let u^* be the approximation obtained by successive orthogonal projections from Equation (4.15). It holds

$$\|u - u^*\| \leq \sqrt{\#T - 1} \inf_{v \in \mathcal{T}_r^T(V)} \|u - v\|.$$

To get more specific results, further assumptions have to be made on the function's class. The particular cases of Sobolev and mixed-Sobolev regularities have been studied in [84, 7].

4.5.2 Approximation rates for Sobolev functions

In [84], the authors provide asymptotic upper bounds for the storage complexity of the tree-based tensor necessary to approximate with precision ε a function having Sobolev or mixed-Sobolev regularity. More recent results from [7] propose some refinements of these results.

Sobolev spaces

Theorem 4.9. [7] *Let $u \in W_{\mu}^{s,2}(\mathcal{X})$ and $0 < \varepsilon < 1$ and let $S(u, \varepsilon, d)$ be the minimal complexity $S(T, r, m)$ such that*

$$\inf_{v \in \mathcal{T}_r^T(V)} \|u - v\| \leq \varepsilon \|u\|_{W_{\mu}^{s,2}},$$

where $V = V_1 \otimes \dots \otimes V_d$ with V_{ν} a spline space of dimension m_{ν} . For any dimension partition tree T , it holds

$$S(u, \varepsilon, d) \leq C(d) \varepsilon^{-d/s}, \text{ with } C(d) \text{ a constant depending on } d.$$

Mixed Sobolev spaces

Theorem 4.10. [7] *For $u \in W_{\mu, \text{mix}}^{s,2}(\mathcal{X})$ and $0 < \varepsilon < 1$. We denote by $S(u, \varepsilon, d)$ the minimal complexity $S(T, r, m)$ such that*

$$\inf_{v \in \mathcal{T}_r^T(V)} \|u - v\| \leq \varepsilon \|u\|_{W_{\mu, \text{mix}}^{s,2}},$$

where $V = V_1 \otimes \dots \otimes V_d$ with V_{ν} a spline space of dimension m_{ν} . For any dimension partition tree T , it holds

- if T is a trivial tree (with depth one)

$$S(u, \varepsilon, d) \leq C(d) \varepsilon^{-d/(2s)} \log(\varepsilon^{-1})^{d(d-2)}.$$

- if T is a binary tree

$$S(u, \varepsilon, d) \leq C(d) \varepsilon^{-3/(2s)} \log(\varepsilon^{-1})^{3(d-2)}.$$

with $C(d)$ a constant depending exponentially on d .

PRINCIPAL COMPONENT ANALYSIS FOR TREE-BASED TENSOR FORMATS

Contents

5.1	Approximation of α-principal subspaces	108
5.1.1	Choosing an oblique projection verifying a stability property	108
5.1.2	Choosing the boosted optimal weighted least-squares projection	110
5.2	Estimation of the α-principal subspaces	113
5.2.1	Accuracy of the empirical α -principal subspaces	113
5.2.2	Adaptive estimation of the α -principal component subspaces	116
5.3	Learning tree tensor networks using PCA	117
5.3.1	Description of the algorithm	117
5.3.2	Error analysis	118
5.3.3	Complexity analysis	121
5.3.4	Heuristics used in practice	122
5.4	Numerical examples	124
5.4.1	Notations and objectives	124
5.4.2	Adaptive determination of the approximation spaces in the leaves	125
5.4.3	Adaptive estimation of the α -principal components subspaces	126
5.5	Conclusions	130

In this Chapter, we present an algorithm adapted from [78] that constructs the approximation in tree-based tensor format. We will use the same notations as in Chapter 4. Using a leaves-to-root approach, the algorithm constructs thanks to principal component analysis α -principal subspaces of u . In practice, we replace the α -principal subspaces of u by approximations, as explained in Section 5.1. In Section 5.2, we detail how these α -principal subspaces are estimated in practice. Section 5.3 describes the whole algorithm. Section 5.4 shows the efficiency of the algorithm on numerical examples.

5.1 Approximation of α -principal subspaces

In practice, we do not directly determine the α -principal subspaces of u , but an approximation of U_α is searched in a low-dimensional approximation subspace of $L_{\mu_\alpha}^2$, denoted by V_α , with $m_\alpha = \dim(V_\alpha) \geq r_\alpha$, by solving

$$\min_{\substack{\dim(W_\alpha)=r_\alpha \\ W_\alpha \subset V_\alpha}} \|u - \mathcal{P}_{W_\alpha} u\|_{L_\mu^2}^2 = \min_{\substack{\dim(W_\alpha)=r_\alpha \\ W_\alpha \subset V_\alpha}} \|u - \mathcal{P}_{V_\alpha} u\|_{L_\mu^2}^2 + \|\mathcal{P}_{V_\alpha} u - \mathcal{P}_{W_\alpha} u\|_{L_\mu^2}^2. \quad (5.1)$$

For a subspace $V_\alpha \subset L_{\mu_\alpha}^2(\mathcal{X}_\alpha)$, we recall that \mathcal{P}_{V_α} is the orthogonal projection from $L_\mu^2(\mathcal{X})$ to $V_\alpha \otimes L_{\mu_{\alpha^c}}^2(\mathcal{X}_{\alpha^c})$, such that for all x_{α^c} in \mathcal{X}_{α^c} , $(\mathcal{P}_{V_\alpha} u)(\cdot, x_{\alpha^c}) = P_{V_\alpha} u(\cdot, x_{\alpha^c})$ where P_{V_α} is the orthogonal projection from $L_{\mu_\alpha}^2(\mathcal{X}_\alpha)$ to V_α .

Since $\mathcal{P}_{W_\alpha} = \mathcal{P}_{W_\alpha} \mathcal{P}_{V_\alpha}$, solving (5.1) is equivalent solving

$$\min_{\substack{\dim(W_\alpha)=r_\alpha \\ W_\alpha \subset L_{\mu_\alpha}^2}} \|\mathcal{P}_{V_\alpha} u - \mathcal{P}_{W_\alpha} \mathcal{P}_{V_\alpha} u\|_{L_\mu^2}^2,$$

whose solution is the α -principal subspace of $\mathcal{P}_{V_\alpha} u$.

Since the orthogonal projection is usually not computable, the orthogonal projection P_{V_α} is replaced by an oblique projection \mathcal{Q}_{V_α} from $L_{\mu_\alpha}^2$ onto V_α . An approximate α -principal subspace U_α^\star is then obtained by solving

$$\min_{\substack{\dim(W_\alpha)=r_\alpha \\ W_\alpha \subset L_{\mu_\alpha}^2}} \|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{W_\alpha} \mathcal{Q}_{V_\alpha} u\|_{L_\mu^2}^2, \quad (5.2)$$

whose solution is the α -principal subspace of $\mathcal{Q}_{V_\alpha} u$, where \mathcal{Q}_{V_α} is the oblique projection from L_μ^2 to $V_\alpha \otimes L_{\mu_{\alpha^c}}^2$ defined by $(\mathcal{Q}_{V_\alpha} u)(\cdot, x_{\alpha^c}) = \mathcal{Q}_{V_\alpha} u(\cdot, x_{\alpha^c})$. \mathcal{Q}_{V_α} may be a sample-based projection. In the case where the samples are random, the quantity $\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|_{L_\mu^2}^2$ is thus a random variable. The objective of the next two sections is to define an oblique projection allowing us to control $\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|_{L_\mu^2}^2$ the minimal reconstruction error associated to the α -principal subspace of $\mathcal{Q}_{V_\alpha} u$ defined by the Equation (5.2).

5.1.1 Choosing an oblique projection verifying a stability property

In this section, we consider the general case where \mathcal{Q}_{V_α} is an oblique projection from $L_{\mu_\alpha}^2$ to V_α verifying a stability property, that is to say, for any function $f^\alpha \in L_{\mu_\alpha}^2$, it holds

$$\|f^\alpha - \mathcal{Q}_{V_\alpha} f^\alpha\| \leq C \|f^\alpha - P_{V_\alpha} f^\alpha\|, \quad (5.3)$$

where P_{V_α} is the orthogonal projection from $L_{\mu_\alpha}^2$ to V_α and C a constant independent of the approximation space V_α .

Lemma 5.1. Assume Q_{V_α} is a projection from $L_{\mu_\alpha}^2$ to V_α verifying the stability property from Equation (5.3) for all $f^\alpha \in L_{\mu_\alpha}^2$ and let \mathcal{Q}_{V_α} be the associated oblique projection from L_μ^2 to $V_\alpha \otimes L_{\mu_\alpha^c}^2$ such that $(\mathcal{Q}_{V_\alpha} u)(\cdot, x_{\alpha^c}) = Q_{V_\alpha} u(\cdot, x_{\alpha^c})$. Then it holds for all $u \in L_\mu^2$,

$$\|\mathcal{Q}_{V_\alpha} u\|^2 \leq C_1 \|u\|^2 \text{ with } C_1 = C^2.$$

where C the constant from Equation (5.3).

Proof. In a first time, let us show that the assumption (5.3) implies that for all $u \in L_\mu^2$, $\|u - \mathcal{Q}_{V_\alpha} u\| \leq C \|u - \mathcal{P}_{V_\alpha} u\|$. The function u has a representation

$$u(x) = \sum_{k=1}^{\text{rank}_\alpha(u)} u_k^\alpha(x_\alpha) u_k^{\alpha^c}(x_{\alpha^c})$$

with $\{u_k^\alpha\}_{k=1}^{\text{rank}_\alpha(u)}$ an orthogonal family of functions in $L_{\mu_\alpha}^2$.

Then

$$\begin{aligned} \|u - \mathcal{Q}_{V_\alpha} u\|^2 &= \left\| \sum_{k=1}^{\text{rank}_\alpha(u)} u_k^\alpha \otimes u_k^{\alpha^c} - \mathcal{Q}_{V_\alpha} \left(\sum_{k=1}^{\text{rank}_\alpha(u)} u_k^\alpha \otimes u_k^{\alpha^c} \right) \right\|^2 \\ &= \sum_{k=1}^{\text{rank}_\alpha(u)} \|(u_k^\alpha - Q_{V_\alpha} u_k^\alpha) \otimes u_k^{\alpha^c}\|^2 \\ &= \sum_{k=1}^{\text{rank}_\alpha(u)} \|u_k^\alpha - Q_{V_\alpha} u_k^\alpha\|_{L_{\mu_\alpha}^2}^2 \|u_k^{\alpha^c}\|_{L_{\mu_{\alpha^c}}^2}^2. \end{aligned}$$

Using the assumption (5.3), we obtain

$$\begin{aligned} \|u - \mathcal{Q}_{V_\alpha} u\|^2 &\leq \sum_{k=1}^{\text{rank}_\alpha(u)} C^2 \|u_k^\alpha - P_{V_\alpha} u_k^\alpha\|^2 \|u_k^{\alpha^c}\|^2 \\ &= \sum_{k=1}^{\text{rank}_\alpha(u)} C^2 \|u_k^\alpha \otimes u_k^{\alpha^c} - (P_{V_\alpha} u_k^\alpha) \otimes u_k^{\alpha^c}\|^2 \\ &= C^2 \|u - \mathcal{P}_{V_\alpha} u\|^2. \end{aligned}$$

For all $u \in L_\mu^2$, thanks to the Pythagorean equality $\|u - \mathcal{Q}_{V_\alpha} u\|^2 = \|u - \mathcal{P}_{V_\alpha} u\|^2 + \|\mathcal{P}_{V_\alpha} u - \mathcal{Q}_{V_\alpha} u\|^2$, so that

$$\begin{aligned} \|u - \mathcal{P}_{V_\alpha} u\|^2 + \|\mathcal{P}_{V_\alpha} u - \mathcal{Q}_{V_\alpha} u\|^2 &\leq C^2 \|u - \mathcal{P}_{V_\alpha} u\|^2, \text{ which implies} \\ \|\mathcal{P}_{V_\alpha} u - \mathcal{Q}_{V_\alpha} u\|^2 &\leq (C^2 - 1) \|u - \mathcal{P}_{V_\alpha} u\|^2. \end{aligned}$$

Using the triangular inequality $\|\mathcal{Q}_{V_\alpha} u\| - \|\mathcal{P}_{V_\alpha} u\| \leq \|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{V_\alpha} u\|$ and the Cauchy-Schwarz inequality $\sqrt{C^2 - 1} \|u - \mathcal{P}_{V_\alpha} u\| + \|\mathcal{P}_{V_\alpha} u\| \leq (1 + \sqrt{C^2 - 1})^{1/2} (\|u - \mathcal{P}_{V_\alpha} u\|^2 + \|\mathcal{P}_{V_\alpha} u\|^2)^{1/2}$, we get

$$\|\mathcal{Q}_{V_\alpha} u\| \leq C (\|u - \mathcal{P}_{V_\alpha} u\|^2 + \|\mathcal{P}_{V_\alpha} u\|^2)^{1/2} \leq C \|u\|,$$

which ends the proof. \square

Theorem 5.2. *Assume Q_{V_α} is a projection from $L_{\mu_\alpha}^2$ to V_α verifying the stability property (5.3) for all $f^\alpha \in L_{\mu_\alpha}^2$.*

Then, it holds

$$\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|^2 \leq C_1 e_{r_\alpha}^\alpha(u)^2, \quad (5.4)$$

with C_1 the constant from Lemma (5.1) and where $e_{r_\alpha}^\alpha(u)$ and $\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|$ are respectively the minimal reconstruction errors of u and $Q_{V_\alpha} u$ associated to the α -principal subspaces U_α and U_α^\star defined in Equations (4.11) and (5.2).

Proof. By definition, for all $v \in L_\mu^2$ with $\text{rank}_\alpha(v) \leq r_\alpha$,

$$\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\| = \min_{\text{rank}_\alpha(v) \leq r_\alpha} \|Q_{V_\alpha} u - v\|.$$

If we choose in particular $v = Q_{V_\alpha} \mathcal{P}_{U_\alpha} u$, where U_α is the α -principal subspace of u , defined in Equation (4.11), it comes

$$\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\| \leq \|Q_{V_\alpha} u - Q_{V_\alpha} \mathcal{P}_{U_\alpha} u\| = \|Q_{V_\alpha} (u - \mathcal{P}_{U_\alpha} u)\|.$$

Using Lemma 5.1 it comes,

$$\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|^2 \leq \|Q_{V_\alpha} (u - \mathcal{P}_{U_\alpha} u)\|^2 \leq C^2 e_{r_\alpha}^\alpha(u)^2.$$

\square

5.1.2 Choosing the boosted optimal weighted least-squares projection

In this paragraph, we propose to use the projection presented in Chapter 2 (or [52]), which verifies the stability property from Equation (5.3) in expectation.

Let $\{\varphi_i^\alpha\}_{i=1}^{m_\alpha}$ be an orthonormal basis of the approximation space $V_\alpha \subset L_{\mu_\alpha}^2(\mathcal{X}_\alpha)$, and ρ_α be the measure defined by

$$d\rho_\alpha(x_\alpha) = w^\alpha(x_\alpha)^{-1} d\mu_\alpha(x_\alpha), \quad w^\alpha(x_\alpha)^{-1} = \frac{1}{m_\alpha} \sum_{i=1}^{m_\alpha} \varphi_i^\alpha(x_\alpha)^2. \quad (5.5)$$

By construction, $w^\alpha(x_\alpha)^{-1}$ is the density of ρ_α with respect to the reference measure μ_α . As it is invariant by rotation of $\{\varphi_i^\alpha\}_{i=1}^{m_\alpha}$, ρ_α does not depend on the chosen basis but only on V_α . We let Q_{V_α} be the boosted optimal weighted least-squares projection introduced in Chapter 2, such that

for all $f^\alpha \in L_{\mu_\alpha}^2(\mathcal{X}_\alpha)$,

$$Q_{V_\alpha} f^\alpha = \arg \min_{g^\alpha \in V_\alpha} \|f^\alpha - g^\alpha\|_{\mathbf{x}_\alpha^{z_\alpha}},$$

with $\mathbf{x}_\alpha^{z_\alpha} := \{x_\alpha^i\}_{i=1}^{z_\alpha}$ a set of z_α points in \mathcal{X}_α and $\|\cdot\|_{\mathbf{x}_\alpha^{z_\alpha}}$ a discrete semi-norm defined for $f^\alpha \in L_{\mu_\alpha}^2$ by

$$\|f^\alpha\|_{\mathbf{x}_\alpha^{z_\alpha}}^2 = \frac{1}{z_\alpha} \sum_{i=1}^{z_\alpha} w^\alpha(x_\alpha^i) f^\alpha(x_\alpha^i)^2.$$

Here, $x_\alpha^1, \dots, x_\alpha^{z_\alpha}$ are dependent random variables drawn from a measure related to the measure ρ_α from Equation (5.5). Without going into too much details (a precise description of the sampling procedure can be found in Chapter 2), to select these z_α points in \mathcal{X}_α , we draw M times a n_α -sample according to the product measure $\rho_\alpha^{\otimes n_\alpha}$ and select in this collection of M samples the one minimizing a stability criterion (based on the empirical Gram matrix). We resample in this way, until a stability condition is verified. In a second time, we remove from this selected sample as many points as possible while maintaining the stability condition and guaranteeing a resulting number of samples z_α higher than a chosen constant $n_{\alpha, \min} = p_r n_\alpha$, with p_r a constant independent of m_α . As proved in Chapter 2, this sampling procedure allows us to ensure in expectation the stability of the projection. Indeed, Theorem 2.5 from Chapter 2 states that for any $f^\alpha \in L_{\mu_\alpha}^2$,

$$\mathbb{E}(\|f^\alpha - Q_{V_\alpha} f^\alpha\|^2) \leq (1 + \gamma) \|f^\alpha - P_{V_\alpha} f^\alpha\|^2, \text{ where } \gamma = p_r(1 - \delta)^{-1}(1 - \eta^M)^{-1}M. \quad (5.6)$$

In the case where V_α is a random linear space, the equation (5.6) becomes

$$\mathbb{E}(\|f^\alpha - Q_{V_\alpha} f^\alpha\|^2 | V_\alpha) \leq (1 + \gamma) \mathbb{E}(\|f^\alpha - P_{V_\alpha} f^\alpha\|^2 | V_\alpha).$$

Taking the expectation it comes

$$\mathbb{E}(\|f^\alpha - Q_{V_\alpha} f^\alpha\|^2) \leq (1 + \gamma) \mathbb{E}(\|f^\alpha - P_{V_\alpha} f^\alpha\|^2). \quad (5.7)$$

By extension, the oblique projection \mathcal{Q}_{V_α} from L_μ^2 to $V_\alpha \otimes L_{\mu_{\alpha^c}}^2$ such that $(\mathcal{Q}_{V_\alpha} u)(\cdot, x_{\alpha^c}) = Q_{V_\alpha} u(\cdot, x_{\alpha^c})$ is called boosted weighted least-squares projection.

Lemma 5.3. *Let Q_{V_α} be the boosted least-squares projection verifying the property (5.7) for all $f^\alpha \in L_{\mu_\alpha}^2$. Let \mathcal{Q}_{V_α} be the oblique projection from L_μ^2 to $V_\alpha \otimes L_{\mu_{\alpha^c}}^2$, such that $(\mathcal{Q}_{V_\alpha} u)(\cdot, x_{\alpha^c}) = Q_{V_\alpha} u(\cdot, x_{\alpha^c})$ and assume $n_\alpha \geq d_\delta^{-1} m_\alpha \log(2m_\alpha \eta^{-1})$, with $d_\delta^{-1} = -\delta + (1 + \delta) \log(1 + \delta)$ for some $\eta \in (0, 1)$ and some $\delta \in (0, 1)$. Then for all $u \in L_\mu^2$, it holds*

$$\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u\|^2) \leq C_1 \|u\|^2 \text{ with } C_1 = 2(1 + \gamma),$$

where $\gamma = p_r(1 - \delta)^{-1}(1 - \eta^M)^{-1}M$ is the constant associated to the boosted weighted least-squares projection with M the number of repetitions and p_r the constant defining the maximal proportion

of samples to be removed.

Proof. The reasoning is almost the same as for Lemma 5.1, but the stability is verified in expectation, which leads to some differences.

In a first time, let us show that the assumption (5.7) implies that for all $u \in L_{\mu}^2$, $\mathbb{E}(\|u - \mathcal{Q}_{V_{\alpha}} u\|^2) \leq (1 + \gamma) \mathbb{E}(\|u - \mathcal{P}_{V_{\alpha}} u\|^2)$, with $\gamma = p_r(1 - \delta)^{-1}(1 - \eta^M)^{-1}M$.

The function u has a representation

$$u(x) = \sum_{k=1}^{\text{rank}_{\alpha}(u)} u_k^{\alpha}(x_{\alpha}) u_k^{\alpha^c}(x_{\alpha^c}) \text{ with } \{u_k^{\alpha}\} \text{ an orthogonal family of functions.}$$

From Lemma 5.1, it holds

$$\|u - \mathcal{Q}_{V_{\alpha}} u\|^2 = \sum_{k=1}^{\text{rank}_{\alpha}(u)} \|u_k^{\alpha} - \mathcal{Q}_{V_{\alpha}} u_k^{\alpha}\|_{L_{\mu_{\alpha}}^2}^2 \|u_k^{\alpha^c}\|_{L_{\mu_{\alpha^c}}^2}^2.$$

By hypothesis on projection $\mathcal{Q}_{V_{\alpha}}$ we have $\mathbb{E}(\|u_k^{\alpha} - \mathcal{Q}_{V_{\alpha}} u_k^{\alpha}\|^2) \leq (1 + \gamma) \mathbb{E}(\|u_k^{\alpha} - \mathcal{P}_{V_{\alpha}} u_k^{\alpha}\|^2)$. Then

$$\begin{aligned} \mathbb{E}(\|u - \mathcal{Q}_{V_{\alpha}} u\|^2) &\leq \sum_{k=1}^{\text{rank}_{\alpha}(u)} (1 + \gamma) \mathbb{E}(\|u_k^{\alpha} - \mathcal{P}_{V_{\alpha}} u_k^{\alpha}\|^2) \|u_k^{\alpha^c}\|^2 \\ &= \sum_{k=1}^{\text{rank}_{\alpha}(u)} (1 + \gamma) \mathbb{E}(\|u_k^{\alpha} \otimes u_k^{\alpha^c} - (\mathcal{P}_{V_{\alpha}} u_k^{\alpha}) \otimes u_k^{\alpha^c}\|^2) \\ &= (1 + \gamma) \mathbb{E}(\|u - \mathcal{P}_{V_{\alpha}} u\|^2). \end{aligned}$$

Now, thanks to the Pythagorean equality, we have $\mathbb{E}(\|u - \mathcal{Q}_{V_{\alpha}} u\|^2) = \mathbb{E}(\|u - \mathcal{P}_{V_{\alpha}} u\|^2) + \mathbb{E}(\|\mathcal{Q}_{V_{\alpha}} u - \mathcal{P}_{V_{\alpha}} u\|^2)$, and then

$$\begin{aligned} \mathbb{E}(\|u - \mathcal{P}_{V_{\alpha}} u\|^2) + \mathbb{E}(\|\mathcal{Q}_{V_{\alpha}} u - \mathcal{P}_{V_{\alpha}} u\|^2) &\leq (1 + \gamma) \mathbb{E}(\|u - \mathcal{P}_{V_{\alpha}} u\|^2), \text{ which implies} \\ \mathbb{E}(\|\mathcal{Q}_{V_{\alpha}} u - \mathcal{P}_{V_{\alpha}} u\|^2) &\leq \gamma \mathbb{E}(\|u - \mathcal{P}_{V_{\alpha}} u\|^2). \end{aligned}$$

Using the triangular inequality $\|\mathcal{Q}_{V_{\alpha}} u\|^2 \leq 2\|\mathcal{Q}_{V_{\alpha}} u - \mathcal{P}_{V_{\alpha}} u\|^2 + 2\|\mathcal{P}_{V_{\alpha}} u\|^2$, we get

$$\mathbb{E}(\|\mathcal{Q}_{V_{\alpha}} u\|^2) \leq 2\gamma \mathbb{E}(\|u - \mathcal{P}_{V_{\alpha}} u\|^2) + 2\mathbb{E}(\|\mathcal{P}_{V_{\alpha}} u\|^2) \leq 2(\gamma + 1)\|u\|^2,$$

which ends the proof. \square

Theorem 5.4. Let $\mathcal{Q}_{V_{\alpha}}$ be the boosted least-squares projection verifying the property (5.6) for all $f^{\alpha} \in L_{\mu_{\alpha}}^2$. Let $\mathcal{Q}_{V_{\alpha}}$ be the oblique projection from L_{μ}^2 to $V_{\alpha} \otimes L_{\mu_{\alpha^c}}^2$, such that $(\mathcal{Q}_{V_{\alpha}} u)(\cdot, x_{\alpha^c}) = \mathcal{Q}_{V_{\alpha}} u(\cdot, x_{\alpha^c})$. Also assume $n_{\alpha} \geq d_{\delta}^{-1} m_{\alpha} \log(2m_{\alpha}\eta^{-1})$, with $d_{\delta}^{-1} = -\delta + (1 + \delta) \log(1 + \delta)$ for some

$\eta \in (0, 1)$ and some $\delta \in (0, 1)$, then it holds for all $u \in L_\mu^2$,

$$\mathbb{E}(\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|^2) \leq C_1 e_{r_\alpha}^\alpha(u)^2, \text{ with } C_1 = 2(1 + \gamma), \quad (5.8)$$

where $e_{r_\alpha}^\alpha(u)$ and $\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|$ are respectively the minimal reconstruction errors of u and $Q_{V_\alpha} u$ associated to the α -principal subspaces U_α and U_α^\star defined in Equations (4.11) and (5.2).

Proof. By definition, for all $v \in L_\mu^2$ with $\text{rank}_\alpha(v) \leq r_\alpha$,

$$\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\| = \min_{\text{rank}_\alpha(v) \leq r_\alpha} \|Q_{V_\alpha} u - v\|.$$

If we choose in particular $v = Q_{V_\alpha} \mathcal{P}_{U_\alpha} u$, where U_α is the α -principal subspace of u , defined in Equation (4.11), it comes,

$$\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\| \leq \|Q_{V_\alpha} u - Q_{V_\alpha} \mathcal{P}_{U_\alpha} u\| = \|Q_{V_\alpha}(u - \mathcal{P}_{U_\alpha} u)\|.$$

Taking the expectation and using Lemma 5.3 it comes,

$$\mathbb{E}(\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|^2) \leq \mathbb{E}(\|Q_{V_\alpha}(u - \mathcal{P}_{U_\alpha} u)\|^2) \leq C_1 e_{r_\alpha}^\alpha(u)^2.$$

□

Remark 5.1. When Q_{V_α} is a projection verifying a stability condition as in Equation (5.3), the term $\mathbb{E}(\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|^2)$ is bounded by $C^2 e_{r_\alpha}^\alpha(u)^2$. When Q_{V_α} is the boosted least-squares projection (verifying property (5.8)), the constant $2(1 + \gamma)$. We observe that having a quasi-optimality property verified in expectation leads to a wider bound (multiplied by a factor 2).

5.2 Estimation of the α -principal subspaces

5.2.1 Accuracy of the empirical α -principal subspaces

The approximation U_α^\star of the α -principal subspace is solution of Equation (5.2), which is equivalent to

$$\min_{\dim(U_\alpha^\star) = r_\alpha} \mathbb{E} \left(\|Q_{V_\alpha} u(\cdot, X_{\alpha^c}) - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u(\cdot, X_{\alpha^c})\|_{L_{\mu_\alpha}^2}^2 \right)$$

where $Q_{V_\alpha} u(\cdot, X_{\alpha^c})$ is a function-valued random variable. In practice, an estimation of U_α^\star , denoted \hat{U}_α^\star , can be obtained using z_{α^c} independent and identically distributed (i.i.d) samples $\{Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k)\}_{k=1}^{z_{\alpha^c}}$ of this random variable, where the $\{x_{\alpha^c}^k\}_{k=1}^{z_{\alpha^c}}$ are i.i.d. samples of X_{α^c} , and by solving

$$\min_{\dim(\hat{U}_\alpha^\star) = r_\alpha} \frac{1}{z_{\alpha^c}} \sum_{k=1}^{z_{\alpha^c}} \|Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k) - \mathcal{P}_{\hat{U}_\alpha^\star} Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k)\|_{L_{\mu_\alpha(X_{\alpha^c})}^2}^2. \quad (5.9)$$

Let $(\varphi_i^\alpha)_{i=1}^{m_\alpha}$ be an orthonormal basis of V_α . Then $Q_{V_\alpha} u(\cdot, x_\alpha^k)$ can be written

$$Q_{V_\alpha} u(\cdot, x_\alpha^k) = \sum_{i=1}^{m_\alpha} a_\alpha^{ik} \varphi_i^\alpha(\cdot), \quad (5.10)$$

where the coefficients a_α^{ik} depend on the samples $\{x_\alpha^l\}_{l=1}^{z_\alpha}$ in \mathcal{X}_α used to define the projection Q_{V_α} . Therefore, solving the Equation (5.9) requires evaluating the function u on a product grid $\{(x_\alpha^l, x_{\alpha^c}^k) : 1 \leq l \leq z_\alpha, 1 \leq k \leq z_{\alpha^c}\}$, where the samples $(x_\alpha^l, x_{\alpha^c}^k)$ are not i.i.d.. We denote by $\mathbf{A}^\alpha \in \mathbb{R}^{m_\alpha \times z_{\alpha^c}}$ the matrix formed with the coefficients (a_α^{ik}) . The truncated singular value decomposition of \mathbf{A}^α is

$$\mathbf{A}_{r_\alpha}^\alpha = \sum_{i=1}^{r_\alpha} \sigma_\alpha^i \mathbf{v}_\alpha^i (\mathbf{v}_{\alpha^c}^i)^T$$

where $\mathbf{v}_\alpha^i = (v_\alpha^{ik})_{1 \leq k \leq m_\alpha} \in \mathbb{R}^{m_\alpha}$ and $\mathbf{v}_{\alpha^c}^i = (v_{\alpha^c}^{ik})_{1 \leq k \leq z_{\alpha^c}} \in \mathbb{R}^{z_{\alpha^c}}$, and the $\sigma_\alpha^1 \geq \sigma_\alpha^2 \geq \dots \sigma_\alpha^{r_\alpha}$ are the singular values, which are assumed to be sorted in decreasing order.

The solution of Equation (5.9) is the subspace spanned by the functions

$$v_i^\alpha(\cdot) = \sum_{k=1}^{m_\alpha} v_\alpha^{ik} \varphi_k^\alpha(\cdot), \text{ for } 1 \leq i \leq r_\alpha.$$

Letting $\mathbf{V}_{r_\alpha}^\alpha = (\mathbf{v}_\alpha^1, \dots, \mathbf{v}_\alpha^{r_\alpha})$, we have

$$\sum_{k=1}^{z_{\alpha^c}} \|Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k) - P_{\hat{U}_\alpha^\star} Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k)\|_{L_{\mu_\alpha}^2}^2 = \|\mathbf{A}_{r_\alpha}^\alpha - \mathbf{V}_{r_\alpha}^\alpha (\mathbf{V}_{r_\alpha}^\alpha)^T \mathbf{A}_{r_\alpha}^\alpha\|_F^2 = \sum_{k > r_\alpha} (\sigma_\alpha^k)^2.$$

The rank r_α can be chosen such that $\sum_{k > r_\alpha} (\sigma_\alpha^k)^2 \leq \varepsilon^2 \sum_{k \geq 1} (\sigma_\alpha^k)^2$ implying that

$$\frac{1}{z_{\alpha^c}} \sum_{k=1}^{z_{\alpha^c}} \|Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k) - P_{\hat{U}_\alpha^\star} Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k)\|_{L_{\mu_\alpha}^2}^2 \leq \frac{\varepsilon^2}{z_{\alpha^c}} \sum_{k=1}^{z_{\alpha^c}} \|Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k)\|_{L_{\mu_\alpha}^2}^2.$$

Remark 5.2. The determination of \hat{U}_α^\star depends both on the samples $\{x_{\alpha^c}^k\}_{k=1}^{z_{\alpha^c}}$ and on the projection Q_{V_α} and thus on the samples $\{x_\alpha^l\}_{l=1}^{z_\alpha}$.

As underlined in Section 4.4.2, an interesting question is the behaviour of the reconstruction error $\|Q_{V_\alpha} u - P_{\hat{U}_\alpha^\star} Q_{V_\alpha} u\|$ associated with the empirical subspace \hat{U}_α^\star compared to the minimal reconstruction error $\|Q_{V_\alpha} u - P_{U_\alpha^\star} Q_{V_\alpha} u\|$ associated with U_α^\star . But a major difficulty comes from the fact that we have to deal here with the α -principal subspaces of $Q_{V_\alpha} u$. Choosing a sample-based projection Q_{V_α} where the samples are not deterministic but randomly drawn from a certain measure implies that Q_{V_α} is random and depends on samples of the function u , which makes tricky the interpretability of the hypotheses made on u .

Theorem 5.5. [72, Theorem 2] *Let u be in L_μ^2 and Q_{V_α} an oblique projection, which is fixed. For*

$\alpha \in T \setminus D$, let $\{\sigma_{\alpha,\star}^i\}_{i \geq 1}$ and $\{v_i^{\alpha,\star}\}_{i \geq 1}$ be respectively the α -singular values and the α -left singular vectors of $\mathcal{Q}_{V_\alpha} u$.

Also assume that the random variables $\{\Gamma_\alpha^i = \frac{1}{\sigma_{\alpha,\star}^i} (Q_{V_\alpha} u(\cdot, X_{\alpha^c}), v_i^{\alpha,\star})\}_{i \geq 1}$ are independent and that there is a constant $R > 0$ such that

$$\sup_{i \geq 1} \sup_{q \geq 1} q^{-1/2} \left(\mathbb{E} |\Gamma_\alpha^i|^q \right)^{1/q} \leq R. \quad (5.11)$$

Consider $1 \leq r'_\alpha \leq r_\alpha$ such that $\sigma_{\alpha,\star}^{r'_\alpha} \geq \sqrt{2} \sigma_{\alpha,\star}^{r_\alpha+1}$. If the number of samples z_{α^c} verifies

$$z_{\alpha^c} \geq c_1(R) \max(r'_\alpha, \sum_{k > r_\alpha} \frac{\sigma_\alpha^k}{\sigma_\alpha^{r'_\alpha}}) \text{ with } c_1(R) \text{ a constant depending on } R.$$

then it holds

$$\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2 \leq c(R) \mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2), \quad (5.12)$$

with probability at least $1 - \exp(-c_2(R) z_{\alpha^c})$, where $c_2(R)$ and $c(R)$ are two constants depending on R .

$\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2$ and $\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2$ are the reconstruction errors of $\mathcal{Q}_{V_\alpha} u$ respectively associated to the α -principal subspaces \hat{U}_α^\star and U_α^\star .

As already pointed out, the results from [72] are obtained under strong assumptions of the function u . In [25], the framework is wider.

Proposition 5.6. [25] Let u be in L_μ^2 and \mathcal{Q}_{V_α} be an oblique projection, which is fixed and assume that $C_\alpha = \left(\int_{\mathcal{X}_{\alpha^c}} \|\mathcal{Q}_{V_\alpha} u(\cdot, X_{\alpha^c})\|_{L_{\mu_\alpha}^2}^4 d\mu_{\alpha^c} \right)^{1/2} < \infty$. It holds

$$\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2) \leq \mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2) + C_\alpha \left(\frac{r_\alpha}{z_{\alpha^c}} \right)^{1/2}. \quad (5.13)$$

Under some assumptions on the function u , we could bound the term $\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2)$. Then, the number of samples z_{α^c} could be chosen to balance the second term $C_\alpha \left(\frac{r_\alpha}{z_{\alpha^c}} \right)^{1/2}$ with the first term, but in practice this number seems overestimated. Furthermore, due to the fact that \mathcal{Q}_{V_α} is a random projection, the hypotheses that have to be made in practice on u to ensure $C_\alpha < \infty$ are difficult to validate on concrete examples.

For these reasons, in the next section, we propose an adaptive strategy to estimate the empirical α -principal subspaces \hat{U}_α^\star with a given tolerance in order to choose a near-minimal number of samples z_{α^c} .

5.2.2 Adaptive estimation of the α -principal component subspaces

For a given number of samples z_{α^c} , the reconstruction error of the empirical α -principal subspace \hat{U}_{α}^* is estimated by leave-one-out cross validation. While this error is greater than the desired tolerance ε , we increase the dimension of \hat{U}_{α}^* . If for $\dim(\hat{U}_{\alpha}^*) = z_{\alpha^c}$, the tolerance is not reached, we increase the number of samples z_{α^c} and again estimate the leave-one-out error. We start from $z_{\alpha^c} = 1$ and have as upper bound, $z_{\alpha^c} \leq k_{PCA} m_{\alpha}$, where $k_{PCA} \in \mathbb{N}^*$ is a sampling factor. This procedure, presented in Algorithm 5.1, provides in many experiments a near-minimal number of samples z_{α^c} to get the desired accuracy.

Algorithm 5.1 Adaptive algorithm for the estimation of the α -principal components of a function-valued random variable $X_{\alpha^c} \mapsto Q_{V_{\alpha}} u(\cdot, X_{\alpha^c})$ with prescribed tolerance ε

Inputs: desired tolerance ε , random variable $u(\cdot, X_{\alpha^c})$, approximation space V_{α} , $(\varphi_i)_{i=1}^{m_{\alpha}}$ an orthonormal basis of V_{α} , oblique projection $Q_{V_{\alpha}}$ and sampling factor k_{PCA} .

Outputs: V_r^{α} matrix of singular vectors of A^{α}

Set $z_{\alpha^c} = 1$

Compute the vector A^{α} corresponding to the coefficients of one realization $Q_{V_{\alpha}} u(\cdot, X_{\alpha^c})$ in the orthonormal basis of V_{α} .

Set $\mathcal{E} = \infty$

while $\mathcal{E} > \varepsilon$ and $z_{\alpha^c} \leq k_{PCA} \dim(V_{\alpha})$ **do**

 Update $z_{\alpha^c} \leftarrow z_{\alpha^c} + 1$

 Update $A^{\alpha} = [A^{\alpha}, a^{\alpha}]$, with a^{α} the vector corresponding to the coefficients of one realization of $Q_{V_{\alpha}} u(\cdot, X_{\alpha^c})$ in the orthonormal basis of V_{α} .

 Set $r = 0$

while $\mathcal{E} > \varepsilon$ or $r \leq z_{\alpha^c}$ **do**

 Update $r \leftarrow r + 1$

 Estimate \mathcal{E} the leave-one-out cross validation error,

for $i = 1, \dots, z_{\alpha^c}$ **do**

 Determine the matrix $V_{\setminus i, r}^{\alpha}$ of r main left singular vectors of $A_{\setminus i}^{\alpha}$, which is A^{α} without its i^{th} column.

end for

 Set

$$\mathcal{E} = \frac{\sum_{i=1}^{z_{\alpha^c}} \|A_i^{\alpha} - V_{\setminus i, r}^{\alpha} (V_{\setminus i, r}^{\alpha})^T A_i^{\alpha}\|_2^2}{\sum_{i=1}^{z_{\alpha^c}} \|A_i^{\alpha}\|_2^2}. \quad (5.14)$$

end while

end while

Determine the matrix V_r^{α} of r left singular vectors of A^{α} .

Remark 5.3. The approximation space V_{α} could also be adaptively selected. Instead of choosing $V_{\alpha} = \bigotimes_{\beta \in S(\alpha)} U_{\beta}$, we could construct adaptively, relying on error estimates, $V_{\alpha} \subset \bigotimes_{\beta \in S(\alpha)} U_{\beta}$.

5.3 Learning tree tensor networks using PCA

Relying on the results from the two previous sections, we here present an algorithm that constructs an approximation u^* of the function $u \in L_\mu^2$ in a tree-based tensor format (tree tensor networks). This algorithm relies on the estimation of the α -principal subspaces of boosted optimal weighted least-squares projections of u .

5.3.1 Description of the algorithm

For a given dimension tree T , the algorithm runs through all nodes α starting from the leaves and going to the root of the tree, to determine the parameters of the tree-based tensor format representation (4.6) of u^* . For each node of the tree $\alpha \in T \setminus \{D\}$, we construct a subspace $\hat{U}_\alpha^* \subset V_\alpha$ that is an approximation of the α -principal subspace of u , as explained in subsection 5.2.2.

When $\alpha \in \mathcal{L}(T)$, V_α is a given finite dimensional approximation space of $L_{\mu_\alpha}^2(\mathcal{X}_\alpha)$ (e.g. splines, wavelets, polynomials, ...).

When $\alpha \notin \mathcal{L}(T)$, V_α is chosen as the tensor product space $\otimes_{\beta \in S(\alpha)} \hat{U}_\beta^*$ for $\beta \in S(\alpha)$. Each space \hat{U}_β^* is a statistical estimation of the α -principal subspace associated to the node β and therefore V_α is a random space, depending on the previously generated samples of u .

The final approximation u^* is defined as the projection of the function u on the tensor product space formed by the α -principal subspaces of $S(D)$, that is to say

$$u^* = \mathcal{Q}_{V_D} u \text{ where } V_D = \bigotimes_{\alpha \in S(D)} \hat{U}_\alpha^*, \quad (5.15)$$

with \mathcal{Q}_{V_D} a boosted optimal least-squares projection. The final approximation is in $\mathcal{T}_r^T(V)$ with $r = (r_\alpha)_{\alpha \in T}$ and $r_\alpha = \dim(\hat{U}_\alpha)$, $\alpha \in T \setminus \{D\}$ and $V = \bigotimes_{\nu=1}^d V_\nu$. The procedure is summarized in Algorithm 5.2.

Also, we give an illustration of the algorithm in Figure 5.1 with $d = 5$. The step 1 corresponds to the introduction of a finite-dimensional tensor product space $V = V_1 \otimes \dots \otimes V_5$ and a given dimension partition tree T . Then going from the leaves to the root, the algorithm constructs low-dimensional subspaces $U_\alpha \subset V_\alpha$ (corresponding to the trees with the blue nodes) and the final approximation is defined as the projection of u onto $\hat{U}_{123}^* \otimes \hat{U}_{45}^*$.

Remark 5.4. In practice, the boosted optimal weighted least-squares projection requires sampling from the optimal measure from Equation (5.5). To this end, we now make explicit the expression of this optimal measure, in the case where α is a leaf of the tree or an interior node.

- When $\alpha \in \mathcal{L}(T)$ is a leaf node, V_α is a given approximation space of univariate functions, such that sampling only implies one-dimensional distributions. One can then rely on standard

Algorithm 5.2 Construction of the approximation of a function in tree-based tensor format

Inputs: dimension tree T , function to approximate u , measure μ , finite-dimensional approximation spaces V_ν for $\nu \in \mathcal{L}(T)$, desired tolerance ε .

Outputs: approximation u^* in tree-based tensor format $\mathcal{T}_r^T(V)$

for $\alpha \in T$ going by decreasing level **do**

if $\alpha \notin \mathcal{L}(T)$ **then**

 Set $V_\alpha = \bigotimes_{\beta \in S(\alpha)} \hat{U}_\beta^*$

end if

 Compute \hat{U}_α^* the estimation of the α -principal subspaces with relative reconstruction error ε , thanks to Algorithm 5.1.

end for

Set $V_D = \bigotimes_{\alpha \in S(D)} \hat{U}_\alpha^*$.

Compute $u^* = \mathcal{Q}_{V_D} u$.

simulation methods such as rejection sampling, inverse transform sampling or slice sampling techniques, see [31].

- When $\alpha \notin \mathcal{L}(T)$, $V_\alpha = \bigotimes_{\beta \in S(\alpha)} \hat{U}_\beta^*$. For each β , we let $\{\psi_{k_\beta}^\beta\}_{k_\beta=1}^{r_\beta}$ be a basis of \hat{U}_β^* . The product basis of V_α is denoted by $\{\varphi_{i_\alpha}^\alpha\}_{i_\alpha=1}^{m_\alpha}$, where $m_\alpha = \prod_{\beta \in S(\alpha)} r_\beta$ and for $i_\alpha = 1, \dots, m_\alpha$,

$$\varphi_{i_\alpha}^\alpha(x_\alpha) = \prod_{\beta \in S(\alpha)} \psi_{k_\beta}^\beta(x_\beta), \text{ for } i_\alpha \equiv (k_\beta)_{\beta \in S(\alpha)}.$$

The sampling measure given by Equation (5.5) is such that

$$w^\alpha(x_\alpha)^{-1} = \frac{1}{m_\alpha} \sum_{i_\alpha=1}^{m_\alpha} \varphi_{i_\alpha}^\alpha(x_\alpha)^2 = \prod_{\beta \in S(\alpha)} \frac{1}{r_\beta} \sum_{1 \leq k_\beta \leq r_\beta} \psi_{k_\beta}^\beta(x_\beta)^2$$

and using the product structure of μ_α , we have

$$d\rho_\alpha(x_\alpha) = \prod_{\beta \in S(\alpha)} d\rho_\beta(x_\beta) \text{ with } d\rho_\beta(x_\beta) = \frac{1}{r_\beta} \sum_{k_\beta=1}^{r_\beta} \psi_{k_\beta}^\beta(x_\beta)^2 d\mu_\beta(x_\beta).$$

As for each $\beta \in S(\alpha)$, $d\rho_\beta(x_\beta)$ can be written in a tree-based tensor format, and its marginal distributions can be efficiently computed. Then sampling from ρ_β can be efficiently done through sequential sampling, some implementation details are given in Appendix B.

5.3.2 Error analysis

Lemma 5.7. For α an interior node of a tree T such that $V_\alpha = \bigotimes_{\beta \in S(\alpha)} \hat{U}_\beta^*$, it holds almost surely

$$\|u - \mathcal{P}_{V_\alpha} u\|^2 \leq \sum_{\beta \in S(\alpha)} \|u - \mathcal{P}_{\hat{U}_\beta^*} u\|^2.$$

Proof. Let γ be an element of $S(\alpha)$, we have:

$$\begin{aligned} \|u - \mathcal{P}_{V_\alpha} u\|^2 &= \|u - \prod_{\beta \in S(\alpha)} \mathcal{P}_{\hat{U}_\beta^*} u\|^2 \\ &= \|u - \mathcal{P}_{\hat{U}_\gamma^*} u\|^2 + \|\mathcal{P}_{\hat{U}_\gamma^*} u - \mathcal{P}_{\hat{U}_\gamma^*} \prod_{\beta \in S(\alpha) \setminus \gamma} \mathcal{P}_{\hat{U}_\beta^*} u\|^2 \\ &\leq \|u - \mathcal{P}_{\hat{U}_\gamma^*} u\|^2 + \|u - \prod_{\beta \in S(\alpha) \setminus \gamma} \mathcal{P}_{\hat{U}_\beta^*} u\|^2. \end{aligned}$$

Proceeding recursively, we obtain the desired result. \square

Lemma 5.8. *Let u^* be the approximation of the function constructed with Algorithm 5.2. Assume that for all $\alpha \in T$, \mathcal{Q}_{V_α} is a projection verifying either the assumptions from Theorem 5.2 or the boosted optimal weighted least-squares projection verifying the assumptions from Theorem 5.4. Assume that for all $\alpha \in T \setminus D$, the empirical α -principal subspaces of $\mathcal{Q}_{V_\alpha} u$ solutions of Equation (5.9), denoted \hat{U}_α^* , are adaptively estimated with Algorithm 5.1.*

The error of approximation is bounded in expectation as follows,

$$\mathbb{E}(\|u - u^*\|^2) \leq \sum_{\alpha \in T \setminus D} (2C_1)^{l(\alpha)} \mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^*} \mathcal{Q}_{V_\alpha} u\|^2) + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} e_{m_\alpha}^{\alpha, dis}(u)^2,$$

where C_1 is defined in Theorem 5.2 or 5.4 depending on the choice of the projection \mathcal{Q}_{V_α} , $l(\alpha)$ is the level of the node α in the tree T , $\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^*} \mathcal{Q}_{V_\alpha} u\|^2$ is the reconstruction error associated to \hat{U}_α^* , $\mathcal{L}(T)$ is the set of leaves of the tree T , and $e_{m_\alpha}^{\alpha, dis}(u) = \|u - \mathcal{P}_{V_\alpha} u\|_{L_\mu^2}$ the error of discretization made by introducing a finite-dimensional approximation space V_α for the leaf α .

Proof. The final approximation u^* is defined by $u^* = \mathcal{Q}_{V_D} u$.

If $\alpha \in \mathcal{L}(T)$, then V_α is a given deterministic approximation space. Therefore, $\mathbb{E}(\|u - \mathcal{Q}_{V_\alpha} u\|^2) \leq C_1 \|u - \mathcal{P}_{V_\alpha} u\|^2$ where C_1 is the constant from Theorem 5.2 or 5.4 depending on the choice of the projection.

For each $\alpha \in T \setminus \mathcal{L}(T)$, $V_\alpha = \bigotimes_{\beta \in S(\alpha)} \hat{U}_\beta^*$ and from Lemma 5.7,

$$\mathbb{E}(\|u - \mathcal{P}_{V_\alpha} u\|^2) \leq \sum_{\beta \in S(\alpha)} \mathbb{E}(\|u - \mathcal{P}_{\hat{U}_\beta^*} u\|^2) \quad (*).$$

Using the triangular inequality, we can write:

$$\begin{aligned} \|u - \mathcal{P}_{\hat{U}_\beta^*} u\| &= \|u - \mathcal{Q}_{V_\beta} u + \mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^*} \mathcal{Q}_{V_\beta} u + \mathcal{P}_{\hat{U}_\beta^*} \mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^*} u\| \\ &\leq \|u - \mathcal{Q}_{V_\beta} u + \mathcal{P}_{\hat{U}_\beta^*} \mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^*} u\| + \|\mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^*} \mathcal{Q}_{V_\beta} u\| \\ &\leq \|(id - \mathcal{P}_{\hat{U}_\beta^*})(u - \mathcal{Q}_{V_\beta} u)\| + \|\mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^*} \mathcal{Q}_{V_\beta} u\| \end{aligned}$$

so that

$$\|u - \mathcal{P}_{\hat{U}_\beta^\star} u\|^2 \leq 2\|u - \mathcal{Q}_{V_\beta} u\|^2 + 2\|\mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^\star} \mathcal{Q}_{V_\beta} u\|^2.$$

Using the inequality (\star) and taking the expectation, it comes

$$\mathbb{E}(\|u - \mathcal{P}_{V_\alpha} u\|^2) \leq \sum_{\beta \in S(\alpha)} 2\mathbb{E}(\|u - \mathcal{Q}_{V_\beta} u\|^2) + 2\mathbb{E}(\|\mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^\star} \mathcal{Q}_{V_\beta} u\|^2).$$

The term $\mathbb{E}(\|\mathcal{Q}_{V_\beta} u - \mathcal{P}_{\hat{U}_\beta^\star} \mathcal{Q}_{V_\beta} u\|^2)$ is the error due to principal component analysis. To deal with the term $\mathbb{E}(\|u - \mathcal{Q}_{V_\beta} u\|^2)$, we distinguish the case where β is a leaf or not. If β is a leaf, we use again Theorems 5.2 or 5.4. If β is not a leaf, we proceed recursively. Going through all nodes, we obtain

$$\mathbb{E}(\|u - u^\star\|^2) \leq \sum_{\alpha \in T \setminus D} (2C_1)^{l(\alpha)} \mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2) + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} \|u - \mathcal{P}_{V_\alpha} u\|^2.$$

□

Theorem 5.9. *Assume that for all $\alpha \in T$, \mathcal{Q}_{V_α} is a projection verifying either the assumptions from Theorem 5.2 or the boosted optimal weighted least-squares projection verifying the assumptions from Theorem 5.4.*

Assume that for all $\alpha \in T \setminus D$, the empirical α -principal subspaces of $\mathcal{Q}_{V_\alpha} u$ solutions of Equation (5.9), denoted \hat{U}_α^\star , are such that the reconstruction errors verify

$$\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2 | \mathcal{Q}_{V_\alpha} u) \leq C_2 \mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2 | \mathcal{Q}_{V_\alpha} u), \quad (5.16)$$

where $\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2$ is the reconstruction error associated with the α -principal subspace of U_α^\star solution of Equation (5.2). Then the error of approximation is bounded in expectation as follows:

$$\mathbb{E}(\|u - u^\star\|^2) \leq C_1 C_2 \sum_{\alpha \in T \setminus D} (2C_1)^{l(\alpha)} e_{r_\alpha}^\alpha(u)^2 + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} e_{m_\alpha}^{\alpha, dis}(u)^2, \quad (5.17)$$

where C_1 is defined in Theorem 5.2 or 5.4 depending on the choice of the projection \mathcal{Q}_{V_α} , $l(\alpha)$ is the level of the node α in the tree T , $\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2$ is the reconstruction error associated to \hat{U}_α^\star , $\mathcal{L}(T)$ is the set of leaves of the tree T , and $e_{m_\alpha}^{\alpha, dis}(u) = \|u - \mathcal{P}_{V_\alpha} u\|_{L_\mu^2}$ the error of discretization made by introducing a finite-dimensional approximation space V_α for the leaf α .

Proof. Taking the expectation in (5.16), we have for all $\alpha \in T \setminus \{D\}$.

$$\mathbb{E}(\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2 | \mathcal{Q}_{V_\alpha} u)) \leq C_2 \mathbb{E}(\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2 | \mathcal{Q}_{V_\alpha} u)),$$

which yields

$$\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2) \leq C_2 \mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2).$$

Besides, the term $\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2)$ can be bounded thanks to Theorem 5.4, such that

$$\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2) \leq C_2 C_1 e_{r_\alpha}^\alpha(u)^2.$$

Using this bound and theorem 5.8, it comes

$$\mathbb{E}(\|u - u^\star\|^2) \leq C_1 C_2 \sum_{\alpha \in T \setminus D} (2C_1)^{l(\alpha)} e_{r_\alpha}^\alpha(u)^2 + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} e_{m_\alpha}^{\alpha, dis}(u)^2,$$

which ends the proof. \square

The first term in Equation (5.23) $C_1 C_2 \sum_{\alpha \in T \setminus D} (2C_1)^{l(\alpha)} e_{r_\alpha}^\alpha(u)^2$ is the error due to the estimation of the principal components. The second term $\sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} e_{m_\alpha}^{\alpha, dis}(u)^2$ is due to the error of approximation made in the leaves.

Remark 5.5. In practice, the hypothesis (5.16) is related to the discussion made in the Subsection 5.2.

5.3.3 Complexity analysis

An important question is how many samples n are required by the algorithm to compute an approximation with precision ε ?

Estimation of the number of samples. The total number of evaluations necessary to build the approximation in tree-based tensor format using Algorithm 5.2, depends on the number of samples z_α used to build the projection and the number of samples z_{α^c} used to estimate the α -principal subspaces. Under the assumptions of Theorem 5.9, that is to say when choosing the boosted least-squares projection, z_α scales in $\mathcal{O}(m_\alpha \log(m_\alpha))$. Thus, for each node $\alpha \in T \setminus \{D\}$, the minimal number of samples is $N_\alpha = \mathcal{O}(m_\alpha \log(m_\alpha) z_{\alpha^c})$. Summing over all nodes it comes

$$n = \sum_{\alpha \in T} N_\alpha = \mathcal{O} \left(\sum_{\alpha \in T} m_\alpha \log(m_\alpha) z_{\alpha^c} \right).$$

When $\alpha \in \mathcal{L}(T)$, $m_\alpha = \dim(V_\alpha)$ and $r_\alpha = \dim(\hat{U}_\alpha^\star)$. When $\alpha \notin \mathcal{L}(T)$, $m_\alpha = \prod_{\beta \in S(\alpha)} r_\beta$. Then

$$n = \mathcal{O} \left(\sum_{\alpha \in \mathcal{L}(T)} m_\alpha z_{\alpha^c} + \sum_{\alpha \notin \mathcal{L}(T)} z_{\alpha^c} \prod_{\beta \in S(\alpha)} r_\beta \right) \text{ up to log factors.}$$

The choice of z_{α^c} is tricky as mentioned in the discussions made above. Therefore, we rely on the adaptive strategy from Algorithm 5.1. Assuming (as in [72]), that the number of samples z_{α^c} necessary to control the reconstruction error of the empirical α -principal subspace is proportional to r_α (that is observed in many practical applications), the total number of samples n is

$$\begin{aligned}
 n &= \sum_{\alpha \in T} N_\alpha = \sum_{\alpha \in T} \mathcal{O}(m_\alpha \log(m_\alpha) r_\alpha) \\
 &= \sum_{\alpha \in \mathcal{L}(T)} \mathcal{O}(m_\alpha \log(m_\alpha) r_\alpha) + \sum_{\alpha \notin \mathcal{L}(T)} \mathcal{O}(r_\alpha \prod_{\beta \in S(\alpha)} r_\beta \sum_{\beta \in S(\alpha)} \log(r_\beta)) \\
 &= \mathcal{O}(\sum_{\alpha \in \mathcal{L}(T)} m_\alpha r_\alpha + \sum_{\alpha \notin \mathcal{L}(T)} r_\alpha \prod_{\beta \in S(\alpha)} r_\beta) \text{ up to log factors} \\
 &= \mathcal{O}(\mathcal{S}(T, r, m)),
 \end{aligned}$$

where $\mathcal{S}(T, r, m)$, $r = \{r_\alpha\}_{\alpha \in T}$, $m = \{m_\alpha\}_{\alpha \in \mathcal{L}(T)}$ is the storage complexity of the tree-based tensor format $\mathcal{T}_r^T(V)$, defined by

$$\mathcal{S}(T, r, m) = \sum_{\alpha \in \mathcal{L}(T)} m_\alpha r_\alpha + \sum_{\alpha \notin \mathcal{L}(T)} r_\alpha \prod_{\beta \in S(\alpha)} r_\beta. \quad (5.18)$$

5.3.4 Heuristics used in practice

According to Lemma 5.8, the error of approximation is bounded in expectation by

$$\mathbb{E}(\|u - u^\star\|^2) \leq \sum_{\alpha \in T \setminus \{D\}} (2C_1)^{l(\alpha)} \mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2) + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} e_{m_\alpha}^{\alpha, dis}(u)^2.$$

The term $\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2)$ includes the error due to the truncation and estimation of the α -principal subspaces. The term $e_{m_\alpha}^{\alpha, dis}(u)$ is the discretization error made in the leaves, which comes from the introduction of finite-dimensional subspaces. These two contributions are amplified by constants depending on the boosted least-squares projection and the chosen tree. In the proposed adaptive strategies, to obtain a certain precision ε , we have to take the constants into account. Assuming that we want to reach a final error having the precision ε^2 , according to Lemma 5.8 the following assumptions are now considered:

- For all $\alpha \in T \setminus D$, the term $\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2)$ is controlled with Algorithm 5.1 with prescribed tolerance ε_{pca}^2 i.e.

$$\mathbb{E}(\|\mathcal{Q}_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} \mathcal{Q}_{V_\alpha} u\|^2) \leq \varepsilon_{pca}^2 := \frac{\varepsilon^2}{(2C_1)^{l(\alpha)} (\#T - 1)},$$

where $\#T$ is the number of nodes in the tree.

- For $\alpha \in \{1, \dots, d\}$, the discretization errors $e_{m_\alpha}^{\alpha, dis}(u)$ can be controlled by adapting the spaces

V_α , using the adaptive boosted optimal least-squares strategy described in Chapter 3, for the construction of a sequence of boosted least-squares projections adapted to the sequence of approximation spaces. For polynomial approximation, the sequence of nested approximation spaces is simply constructed by increasing the polynomial degree one by one. For wavelet approximation, it can be defined by increasing the resolution. The difficulty is that we have to perform this strategy for each sample k of the function-valued random variable $Q_{V_\alpha} u(\cdot, x_{\alpha^c}^k)$. To make things clearer the strategy is described in Algorithm 5.3.

Algorithm 5.3 Algorithm for adaptive approximation of $u(\cdot, x_{\alpha^c}^k)$ in a leaf.

Inputs: desired tolerance ε_{dis} , a sequence of nested approximation spaces $(V_\alpha^j)_{j \geq 1}$.

Outputs: \mathbf{a}^α the vector corresponding to the coefficients of the k^{th} realization of $Q_{V_\alpha^j} u(\cdot, x_{\alpha^c}^k)$ in the orthonormal basis of V_α^j , with j depending on the desired tolerance ε .

Set $\mathcal{E} = \infty$

while $\mathcal{E} > \varepsilon_{dis}$ **do**

 Compute the coefficients \mathbf{a}^α of the boosted least-squares projection $Q_{V_\alpha^j} u(\cdot, x_{\alpha^c}^k)$ in the orthonormal basis of V_α^j

 Set $\mathcal{E} = \frac{a_p^\alpha}{\sqrt{\sum_{i=1}^p (a_i^\alpha)^2}}$

 Set $j = j + 1$

end while

For all $\alpha \in \mathcal{L}(T)$, the term $e_{m_\alpha}^{\alpha, dis}(u)$ is controlled with Algorithm 5.3, using for all $\alpha \in \mathcal{L}(T)$

$$e_{m_\alpha}^{\alpha, dis}(u)^2 \leq \varepsilon_{dis}^2 := \frac{\varepsilon^2}{\frac{1}{2}(2C_1)^{l(\alpha)+1}d}.$$

Thus, under all these assumptions, we should get the desired accuracy for $\mathbb{E}(\|u - u^*\|^2)$. In practice, for small values of ε , ε_{pca} and ε_{dis} may be very small. Indeed when choosing the boosted least-squares projection the constant C_1 defined in Theorem 5.4 may be very high, particularly if the number of repetitions M is high or the proportion p_r of removed points is important or when $l(\alpha)$ is high (the case when using deep trees or when $d > 1$, $l(\alpha) \sim d$ for linear trees). As a consequence the ranks r_α and the number of samples z_{α^c} necessary to determine the α -principal subspaces may be high, and the dimension of the approximation spaces in the leaves may be high too.

In practice, we make the following heuristic choices:

- We replace the constant $C_1 = 2(1+p_r(1-\delta)^{-1}(1-\eta^M)^{-1}M)$ by $C_1 = 2(1+(1-\delta)^{-1}(1-\eta)^{-1})$, which corresponds to the boosted optimal weighted least-squares projection from Theorem 5.4, with no repetition ($M = 1$) and no subsampling ($p_r = 1$). In Chapter 2 (see [52]), we observed on all the examples (without noise) that these two choices give comparable accuracy

for the error of approximation. This leads us to take the value $C_1 = (1 - \delta^{-1})(1 - \eta^{-1})$ even when there are repetitions and subsampling.

- When $l(\alpha) \geq 3$, we replace $l(\alpha) = 3$ in the expressions of ε_{dis} and ε_{pca} . (In all the examples of this chapter the depth of the tree is lower or equal to 3 so that this heuristic does not apply but we have observed on some examples that taking $l(\alpha) = 1$ is not enough to control the precision).

5.4 Numerical examples

5.4.1 Notations and objectives

This numerical section aims at showing the efficiency of the following three contributions:

- Replacing a non-controlled projection (for example empirical interpolation as in [65]) by the boosted least-squares projection presented in Chapter 2, for which we obtained a bound in expectation of the approximation error. We choose the same parameters for this projection in all the numerical examples: $M = 100$, $\delta = 0.9$ and $\eta = 0.01$. The maximal proportion of samples to be removed p_r equals $\frac{m_\alpha}{n_\alpha}$, implying that points are removed while the stability condition is verified.
- Using the adaptive strategy for the determination of the approximation spaces in the leaves (described in Algorithm 5.3), thanks to the adaptive boosted least-squares strategy from Chapter 3.
- Using the adaptive strategy for the estimation of the α -principal components presented in Algorithm 5.1. In this whole numerical part, the sampling factor k_{PCA} is always taken equal to 3, it is an arbitrary choice. When the principal components are not adaptively chosen, we simply take $z_{\alpha^c} = m_\alpha$ (using notations from Algorithm 5.1).

To illustrate the efficiency of the strategies, we assess the quality of the approximation u^\star of a function $u \in L_\mu^2(\mathcal{X})$ by estimating the error of approximation by

$$\varepsilon(u^\star) = \left(\frac{1}{n_{test}} \sum_{x \in \mathbf{x}_{test}} (u(x) - u^\star(x))^2 \right)^{1/2}.$$

In practice, we choose $n_{test} = \#\mathbf{x}_{test} = 1000$. To study the robustness of the methods, we compute 10 times the approximations and draw 10 different test samples \mathbf{x}_{test} and compute empirical confidence intervals of level 10% and 90% for the errors of approximation.

5.4.2 Adaptive determination of the approximation spaces in the leaves

Henon-Heiles potential

The discretization error made in the leaves depends on the approximation spaces we choose. In this section we focus on polynomial approximation spaces and we use the adaptive strategy presented in Algorithm 5.3 to select the polynomial degree p necessary to reach the desired discretization error ε_{dis} .

In this section, we consider the Henon-Heiles potential (see [62]) defined on $\mathcal{X} = \mathbb{R}^d$ equipped with the standard gaussian measure μ and $d = 8$.

$$u(x_1, \dots, x_d) = \frac{1}{2} \sum_{i=1}^d x_i^2 + \sigma^* \sum_{i=1}^{d-1} (x_i x_{i+1}^2 - x_i^3) + \frac{\sigma^*}{16} \sum_{i=1}^{d-1} (x_i^2 + x_{i+1}^2)^2,$$

with $\sigma^* = 0.2$. We consider polynomial approximation spaces $V_\nu = \mathbb{P}_p(\mathcal{X}_\nu)$, $\nu \in D$. The Henon-Heiles function is such that there is no discretization error for $p \geq 4$.

	Without basis adaptation				With basis adaptation	
	$p = 15$		$p = 4$			
	\mathcal{S}	n	\mathcal{S}	n	\mathcal{S}	n
Interpolation	[761; 761]	[1097; 1097]	[431; 431]	[591; 591]	[461; 461]	[717; 717]
Boosted Least-squares	[761; 761]	[1108; 1109]	[431; 431]	[591; 591]	[461; 461]	[719; 720]

Table 5.1 – Comparison of the number of samples n without and with basis adaptation necessary to get an approximation error $\varepsilon = 10^{-14}$, using respectively interpolation with magic points and boosted least-squares as projections. The α -principal components are estimated with $z_{\alpha^c} = m_\alpha$.

Table 5.1 compares the storage complexity \mathcal{S} , the number of evaluations n in three cases,. In the first two cases, there is no basis adaptation and we use respectively $p = 15$ and $p = 4$ such that in both cases there is no discretization error. In the third case, there is an adaptation of the basis (thanks to Algorithm 5.3) with maximal polynomial degree $p = 15$. We observe that this adaptive basis strategy select a polynomial degree $p = 5$ for each leaf. This is due to the fact that the stopping criterion of the algorithm is based on \mathcal{E} , see Algorithm 5.3.

5.4.3 Adaptive estimation of the α -principal components subspaces

Borehole function

We consider the Borehole function

$$u(x_1, \dots, x_8) = \frac{2\pi x_3(x_4 - x_6)}{(x_2 - \log(x_1)) \left(1 + \frac{2x_7x_3}{(x_2 - \log(x_1))x_1^2x_8 + \frac{x_3}{x_4}} \right)}$$

which models the water flow through a borehole as a function of 8 independent random variables $X_1 \sim \mathcal{N}(0.1, 0.0161812)$, $X_2 \sim \mathcal{N}(7.71, 1.0056)$, $X_3 \sim \mathcal{U}(63070, 115600)$, $X_4 \sim \mathcal{U}(990, 1100)$, $X_5 \sim \mathcal{U}(63.1, 116)$, $X_6 \sim \mathcal{U}(700, 820)$, $X_7 \sim \mathcal{U}(1120, 1680)$, $X_8 \sim \mathcal{U}(9855, 12045)$.

We use polynomial approximation spaces $V_\nu = \mathbb{P}_p(\mathcal{X}_\nu)$, $\nu \in D$, with p adaptively chosen such that there is a negligible discretization error.

We construct the approximation of u in a tree-based tensor format with a balanced binary tree, using Algorithm 5.2.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-2.8; -1.3]	-1.8	[41; 51]	[512; 518]
-3	[-3.1; -2.7]	-2.9	[59; 72]	[569; 593]
-4	[-3.6; -3.2]	-3.4	[124; 131]	[695; 721]
-5	[-3.9; -3.5]	-3.7	[137; 146]	[770; 788]
-6	[-5.4; -3.9]	-4.7	[173; 200]	[880; 926]
-7	[-6.1; -5.4]	-5.8	[293; 377]	[1101; 1301]
-8	[-6.9; -6.5]	-6.7	[413; 451]	[1442; 1554]

(a) Without adaptive estimation of the principal components.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-3; -1.4]	-2.2	[45; 51]	[206; 212]
-3	[-3.2; -2]	-2.6	[54; 93]	[218; 264]
-4	[-4.3; -2.9]	-3.5	[99; 140]	[275; 320]
-5	[-5.4; -3.7]	-4.6	[150; 183]	[337; 370]
-6	[-5.6; -5]	-5.3	[196; 245]	[389; 440]
-7	[-6.2; -5.5]	-5.8	[293; 418]	[500; 638]
-8	[-7.9; -6.6]	-7.3	[468; 564]	[694; 791]

(b) With adaptive estimation of the principal components.

Table 5.2 – Borehole function. Approximation using interpolation as projections with a balanced binary tree, with a prescribed tolerance $\varepsilon_{pca} = \varepsilon$ for the estimation of the principal components. Confidence intervals for relative error $\varepsilon(u^*)$, storage complexity \mathcal{S} , number of evaluations n .

Table 5.2 shows that choosing $\varepsilon_{pca}^2 = \varepsilon^2$ and using interpolation does not provide an approximation error that is lower than ε . In this case, the adaptive strategy for the estimation of the principal components performs better (in the sense that the obtained approximation error is lower)

than in the non adaptive case and this with fewer evaluations.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-3.6; -3.4]	-3.5	[123; 127]	[699; 705]
-3	[-4.3; -3.5]	-3.9	[135; 156]	[769; 800]
-4	[-5.4; -4.9]	-5.2	[195; 208]	[876; 928]
-5	[-6.2; -5.5]	-5.8	[306; 365]	[1102; 1270]
-6	[-7.1; -6.6]	-6.8	[405; 450]	[1422; 1533]
-7	[-8.0; -6.9]	-7.5	[527; 603]	[1642; 1831]
-8	[-9.2; -8.1]	-8.7	[727; 815]	[2049; 2325]

(a) Without adaptive estimation of the principal components.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-4.4; -2.8]	-3.5	[105; 140]	[279; 323]
-3	[-5.3; -3.3]	-4.5	[159; 181]	[343; 370]
-4	[-5.5; -5.2]	-5.4	[206; 273]	[399; 471]
-5	[-6.0; -5.5]	-5.7	[326; 417]	[534; 636]
-6	[-7.9; -6.4]	-7.1	[482; 561]	[708; 791]
-7	[-9; -8.1]	-8.5	[641; 808]	[876; 1061]
-8	[-9.2; -8.7]	-8.9	[751; 939]	[1003; 1203]

(b) With adaptive estimation of the principal components.

Table 5.3 – Borehole function. Approximation using interpolation as projections with a balanced binary tree, with a prescribed tolerance for each $\alpha \in T$ $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ for the estimation of the principal components. Confidence intervals for relative error $\varepsilon(u^*)$, storage complexity \mathcal{S} , number of evaluations n .

Table 5.3 shows that for each $\alpha \in T$, choosing $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ and using interpolation provides a controlled approximation error (lower than ε), both when the principal components are adaptively determined or not. However using the adaptive strategy, strongly reduces the number of samples.

We can draw the same conclusions from Table 5.4. For each $\alpha \in T$, choosing $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ and using the boosted optimal weighted least-squares projection (even with $M = 100$ repetitions and subsampling) provides a controlled approximation error. Furthermore the adaptive strategy for principal components estimation strongly reduces the number of samples necessary to reach the desired accuracy. We also notice that the obtained error is about 10 times smaller than the desired ε .

In this particular example, we observe that for a given ε , the obtained error is smaller using the boosted least-squares projection than using the interpolation but the number of evaluations is also slightly larger.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-4.9; -4.4]	-4.6	[167; 169]	[1028; 1068]
-3	[-5.3; -4.6]	-4.9	[167; 194]	[1022; 1079]
-4	[-6.1; -5.2]	-5.7	[211; 230]	[1094; 1129]
-5	[-6.4; -6.1]	-6.2	[305; 349]	[1195; 1321]
-6	[-7.9; -7.1]	-7.5	[424; 476]	[1505; 1616]
-7	[-9; -7.7]	-8.3	[534; 617]	[1637; 1855]
-8	[-9.4; -9.1]	-9.2	[809; 909]	[2340; 2665]

(a) Without adaptive estimation of the principal components.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-4.9; -3.1]	-3.8	[141; 177]	[342; 379]
-3	[-5.9; -4.1]	-5.1	[182; 223]	[388; 428]
-4	[-6; -5.6]	-5.8	[232; 264]	[435; 476]
-5	[-6.5; -6.0]	-6.2	[335; 442]	[555; 675]
-6	[-8.8; -6.6]	-7.7	[518; 629]	[756; 871]
-7	[-9.2; -7.2]	-8.5	[718; 819]	[964; 1136]
-8	[-9.6; -8.9]	-9.2	[1046; 1297]	[1418; 1940]

(b) With adaptive estimation of the principal components.

Table 5.4 – Borehole function. Approximation using boosted least-squares as projections with a balanced binary tree, with a prescribed tolerance for each $\alpha \in T$ $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ for the estimation of the principal components. Confidence intervals for relative error $\varepsilon(u^*)$, storage complexity \mathcal{S} , number of evaluations n .

Anisotropic multivariate function

We consider the following function in dimension $d = 6$:

$$u(x) = \frac{1}{(10 + 2x_1 + x_3 + 2x_4 - x_5)^2} \quad (5.19)$$

defined on $\mathcal{X} = [-1, 1]^d$, equipped with the uniform measure.

We also consider the polynomial approximation spaces $V_\nu = \mathbb{P}_p(\mathcal{X}_\nu)$, with p chosen adaptively such that there is no discretization error ($p \leq 15$). We construct the approximation in a hierarchical tensor format with a balanced binary tree using Algorithm 5.2.

The conclusions for the Anisotropic function are similar to the one made for the Borehole function except for small desired tolerances ε .

Table 5.5 shows that choosing $\varepsilon_{pca}^2 = \varepsilon^2$ and using interpolation does not provide an approximation error that is lower than ε . In this case, the adaptive strategy for the estimation of the principal components performs better (in the sense that the obtained approximation error is lower) than in the non adaptive case and this with fewer evaluations.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-1.8; -0.8]	-1.4	[66; 70]	[468; 492]
-3	[-2.1; -1.6]	-1.9	[111; 132]	[586; 650]
-4	[-3.0; -2.3]	-2.7	[160; 201]	[715; 833]
-5	[-3.5; -3.1]	-3.3	[250; 284]	[944; 1080]
-6	[-4.5; -3.2]	-3.8	[343; 400]	[1194; 1449]
-7	[-5.2; -4.1]	-4.7	[590; 700]	[1597; 1999]

(a) Without adaptive estimation of the principal components.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-1.7; -0.7]	-1.3	[53; 74]	[180; 204]
-3	[-2.3; -1.6]	-1.9	[105; 153]	[241; 292]
-4	[-3.2; -1.8]	-2.5	[175; 211]	[313; 361]
-5	[-4.1; -3]	-3.6	[251; 365]	[416; 533]
-6	[-4.7; -3.8]	-4.2	[385; 490]	[545; 655]
-7	[-5.7; -4.1]	-4.8	[680; 875]	[702; 895]

(b) With adaptive estimation of the principal components.

Table 5.5 – Anisotropic function. Approximation using interpolation as projections with a prescribed tolerance for each $\alpha \in T$ $\varepsilon_{pca}^2 = \varepsilon^2$ for the estimation of the principal components. Confidence intervals for relative error $\varepsilon(u^*)$, storage complexity \mathcal{S} , number of evaluations n .

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-3.3; -2.1]	-2.8	[164; 185]	[708; 781]
-3	[-3.7; -2.9]	-3.3	[202; 263]	[814; 1046]
-4	[-4.8; -3.4]	-4.1	[333; 364]	[1137; 1348]
-5	[-5.3; -4.1]	-4.7	[450; 488]	[1707; 1852]
-6	[-6.4; -4.6]	-5.5	[566; 681]	[2012; 2657]
-7	[-6.7; -5.4]	-6	[855; 965]	[2658; 3243]

(a) Without adaptive estimation of the principal components.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-3.2; -1.7]	-2.4	[129; 205]	[269; 357]
-3	[-3.9; -2.5]	-3.2	[240; 391]	[395; 556]
-4	[-4.5; -3.3]	-3.9	[399; 540]	[557; 717]
-5	[-5.6; -4.3]	-4.9	[526; 843]	[705; 1042]
-6	[-6.3; -4.9]	-5.5	[758; 1025]	[959; 1223]
-7	[-7.3; -5.8]	-6.5	[1070; 1461]	[1124; 1520]

(b) With adaptive estimation of the principal components.

Table 5.6 – Anisotropic function. Approximation using interpolation as projections with a balanced binary tree, with a prescribed tolerance for each $\alpha \in T$ $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ for the estimation of the principal components. Confidence intervals for relative error $\varepsilon(u^*)$, storage complexity \mathcal{S} , number of evaluations n .

Table 5.6 shows that for each $\alpha \in T$, choosing $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ and using

interpolation does not provide a controlled approximation error for the small values of ε (i.e $\log(\varepsilon)$ lower than -4), both when the principal components are adaptively determined or not. However using the adaptive strategy, strongly reduces the number of samples.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-3.7; -2.3]	-3.2	[213; 232]	[759; 819]
-3	[-3.8; -2.8]	-3.3	[253; 292]	[837; 944]
-4	[-5.0; -3.4]	-4.2	[321; 408]	[981; 1275]
-5	[-5.1; -4.3]	-4.6	[426; 507]	[1353; 1692]
-6	[-5.8; -4.9]	-5.4	[551; 656]	[1823; 2329]
-7	[-6.7; -5.3]	-6.0	[735; 875]	[2851; 3791]

(a) Without adaptive estimation of the principal components.

$\log(\varepsilon)$	$\log(\varepsilon(u^*))$	$\log(\sqrt{\mathbb{E}(\varepsilon(u^*)^2)})$	\mathcal{S}	n
-2	[-3.6; -2.3]	-3	[193; 270]	[328; 403]
-3	[-5.0; -3.3]	-4.1	[309; 430]	[455; 579]
-4	[-4.9; -3.8]	-4.4	[385; 531]	[534; 697]
-5	[-6.2; -4.4]	-5.3	[588; 805]	[751; 985]
-6	[-6.7; -5.5]	-6.1	[827; 1268]	[1028; 1503]
-7	[-7.7; -6.2]	-7.0	[1203; 1861]	[1463; 2230]

(b) With adaptive estimation of the principal components.

Table 5.7 – Anisotropic function. Approximation using boosted least-squares as projections with a balanced binary tree, with a prescribed tolerance for each $\alpha \in T$ $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ for the estimation of the principal components. Confidence intervals for relative error $\varepsilon(u^*)$, storage complexity \mathcal{S} , number of evaluations n .

Table 5.7 shows that for each $\alpha \in T$, choosing $\varepsilon_{pca}^2 = \frac{\varepsilon^2}{(2(1+(1-\delta)^{-1}(1-\eta)^{-1}))^{l(\alpha)}(\#T-1)}$ and using the boosted optimal weighted least-squares projection (even with $M = 100$ repetitions and subsampling) provides a controlled approximation error. Furthermore the adaptive strategy for principal components estimation strongly reduces the number of samples necessary to reach the desired accuracy.

5.5 Conclusions

In this chapter, we have proposed an algorithm to construct the approximation of a function u in tree-based tensor format $\mathcal{T}_r^T(V)$ with $V = \bigotimes_{\nu=1}^d V_\nu$ some approximation spaces possibly selected adaptively. Using adaptive strategies for the control of the discretization error, the control of the α -ranks and the estimation of the principal components we are able to provide a controlled approximation of the function u , assuming we have a sufficiently high number of evaluations. The theoretical criteria used to control the approximation appear to be very pessimistic for two reasons:

- As underlined in Chapter 2, the constant of quasi-optimality C_1 of the boosted least-squares projection is loose compared to what we observe in practice.
- The proof of Theorem 5.9 leads to a bound with the constant C_1 to the power of the depth of the tree.

On the studied examples, these two observations turn out to be pessimistic. However, as this bound has been established for any function from L_μ^2 , some functions may indeed verify this bound (we have just not found them yet).

Appendix

An important question that is not fully answered yet is to be able to give the number of samples necessary for the algorithm 5.2 to construct an approximation with controlled precision (under some assumptions of the function class and the choice of the approximation tool $T_r^T(V)$). In this appendix we give some results on particular classes of functions that may be useful in future work to answer the question of the number of evaluations in totality.

Error bounds for Sobolev spaces

Corollary 5.10. *We consider $u \in W_\mu^{s,2}$.*

- *Assume that for all $\alpha \in T$, Q_{V_α} is the boosted optimal weighted least-squares projection verifying the condition from Lemma 5.3.*
- *Assume that for all $\alpha \in T \setminus D$, the empirical reconstruction error verifies,*

$$\mathbb{E}(\|Q_{V_\alpha}u - \mathcal{P}_{\hat{U}_\alpha^*}Q_{V_\alpha}u\|^2 | Q_{V_\alpha}u) \leq C_2 \mathbb{E}(\|Q_{V_\alpha}u - \mathcal{P}_{U_\alpha^*}Q_{V_\alpha}u\|^2 | Q_{V_\alpha}u),$$

where C_2 is a constant.

- *Also assume that for all $\alpha \in T \setminus D$, the α -rank r_α verifies*

$$r_\alpha \geq \left(\frac{\varepsilon}{\sqrt{2}}\right)^{-d_\alpha/s} \left(C_2(\#T - 1)(2C_1)^{l(\alpha)+1}C(s, d_\alpha)^2\right)^{d_\alpha/(2s)} \quad (\star)$$

with $C(s, d_\alpha)$ a constant depending on s and $d_\alpha = \min\{\#\alpha, d - \#\alpha\}$

- *For all $\nu \in \mathcal{L}(T)$, the approximation space V_ν is such that*

$$\|u - \mathcal{P}_{V_\nu}u\|^2 \leq m_\nu^{-2s} \|u\|_{W_\mu^{s,2}}^2, \quad (o)$$

and we assume that for $\nu \in \mathcal{L}(T)$, $m_\nu \geq \varepsilon^{-1/s}(\sqrt{d}(2C_1)^{l(\nu)+1})^{1/s}$.

Then the error of approximation is bounded in expectation,

$$\mathbb{E}(\|u - u^\star\|^2) \leq \varepsilon^2 \|u\|_{W_\mu^{s,2}}^2, \quad (5.20)$$

Proof. Thanks to the two first hypotheses, it holds,

$$\mathbb{E}(\|u - u^\star\|^2) \leq C_2 \sum_{\alpha \in T \setminus D} \frac{1}{2} (2C_1)^{l(\alpha)+1} e_{r_\alpha}^\alpha(u)^2 + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} \|u - \mathcal{P}_{V_\alpha} u\|^2.$$

Proposition 4.6 states that

$$e_{r_\alpha}^\alpha(u)^2 \leq C_2 \left(C(s, d_\alpha) r_\alpha^{-s/d_\alpha} \|u\|_{W_\mu^{s,2}} \right)^2.$$

Using the hypothesis (\star) it comes,

$$e_{r_\alpha}^\alpha(u)^2 \leq \frac{1}{2} C_2^{-1} (\#T - 1)^{-1} (2C_1)^{-(l(\alpha)+1)} \varepsilon^2 \|u\|_{W_\mu^{s,2}}^2.$$

Replacing in the expression of the error bound, it comes

$$\mathbb{E}(\|u - u^\star\|^2) \leq \frac{1}{2} \varepsilon^2 \|u\|_{W_\mu^{s,2}}^2 + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} \|u - \mathcal{P}_{V_\alpha} u\|^2.$$

For the second term in the bound, using (o) and $m_\nu \geq \varepsilon^{-1/s}(\sqrt{d}(2C_1)^{l(\alpha)+1})^{1/s}$, it comes

$$\|u - \mathcal{P}_{V_\alpha} u\|^2 \leq (\#T - 1)^{-1} (2C_1)^{-(l(\alpha)+1)} \varepsilon^2 \|u\|_{W_\mu^{s,2}}^2.$$

All in all the final error bound becomes

$$\mathbb{E}(\|u - u^\star\|^2) \leq \varepsilon^2 \|u\|_{W_\mu^{s,2}}^2. \quad (5.21)$$

□

Error bounds for mixed Sobolev spaces

Corollary 5.11. *We consider $u \in W_{\mu, \text{mix}}^{s,2}$.*

- Assume that for all $\alpha \in T$, Q_{V_α} is the boosted optimal weighted least-squares projection verifying the condition from 5.3.
- Assume that for all $\alpha \in T \setminus D$, the empirical reconstruction error verifies

$$\mathbb{E}(\|Q_{V_\alpha} u - \mathcal{P}_{\hat{U}_\alpha^\star} Q_{V_\alpha} u\|^2 | Q_{V_\alpha} u) \leq C_2 \mathbb{E}(\|Q_{V_\alpha} u - \mathcal{P}_{U_\alpha^\star} Q_{V_\alpha} u\|^2 | Q_{V_\alpha} u),$$

where C_2 is a constant.

- Also assume that for all $\alpha \in T \setminus D$, the α -rank r_α verifies

$$r_\alpha \geq c(\varepsilon, d)(C(s, d_\alpha)(\#T - 1)^{-1}(2C_1)^{-l(\alpha)})^{1/2s} s^{-d+1} \varepsilon^{-1/2s} \log(\varepsilon^{-1})^{d-2} \quad (\star)$$

- For all $\nu \in \mathcal{L}(T)$ the approximation space V_ν is such that

$$\|u - \mathcal{P}_{V_\nu} u\|^2 \leq m_\nu^{-2s} \|u\|_{W_{\mu, \text{mix}}^{s, 2}}^2, \quad (o)$$

and we assume that for $\nu \in \mathcal{L}(T)$, $m_\nu \geq \varepsilon^{-1/s} (\sqrt{d}(2C_1)^{l(\nu)+1})^{1/s}$.

Then the error of approximation is bounded in expectation,

$$\mathbb{E}(\|u - u^\star\|^2) \leq \varepsilon^2 \|u\|_{W_{\mu, \text{mix}}^{s, 2}}^2, \quad (5.22)$$

Proof. The beginning of the proof is the same than for the corollary 5.10.

Thanks to the two first hypotheses, it holds

$$\mathbb{E}(\|u - u^\star\|^2) \leq C_2 \sum_{\alpha \in T \setminus D} \frac{1}{2} (2C_1)^{l(\alpha)+1} e_{r_\alpha}^\alpha(u)^2 + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} \|u - \mathcal{P}_{V_\alpha} u\|^2.$$

Proposition 4.7 states that

$$e_{r_\alpha}^\alpha(u)^2 \leq (C(s, d_\alpha) r_\alpha^{-s} \log(r_\alpha)^{s(d_\alpha-1)})^2 \|u\|_{W_{\mu, \text{mix}}^{s, 2}}^2, \quad d_\alpha = \min\{\#\alpha, d - \#\alpha\}.$$

Using the hypothesis (\star) it comes,

$$e_{r_\alpha}^\alpha(u)^2 \leq C_2^{-1} (\#T - 1)^{-1} (2C_1)^{-(l(\alpha)+1)} \varepsilon^2 \|u\|_{W_{\mu, \text{mix}}^{s, 2}}^2.$$

Replacing in the expression of the error bound, it comes

$$\mathbb{E}(\|u - u^\star\|^2) \leq \frac{1}{2} \varepsilon^2 \|u\|_{W_{\mu, \text{mix}}^{s, 2}}^2 + \sum_{\alpha \in \mathcal{L}(T)} \frac{1}{2} (2C_1)^{l(\alpha)+1} \|u - \mathcal{P}_{V_\alpha} u\|^2.$$

For the second term in the bound, using (o) and $m_\nu \geq \varepsilon^{-1/s} (\sqrt{d}(2C_1)^{l(\alpha)+1})^{1/s}$, it comes

$$\|u - \mathcal{P}_{V_\alpha} u\|^2 \leq (\#T - 1)^{-1} (2C_1)^{-(l(\alpha)+1)} \varepsilon^2 \|u\|_{W_{\mu, \text{mix}}^{s, 2}}^2.$$

All in all the final error bound becomes,

$$\mathbb{E}(\|u - u^\star\|^2) \leq \varepsilon^2 \|u\|_{W_{\mu, \text{mix}}^{s, 2}}^2. \quad (5.23)$$

□

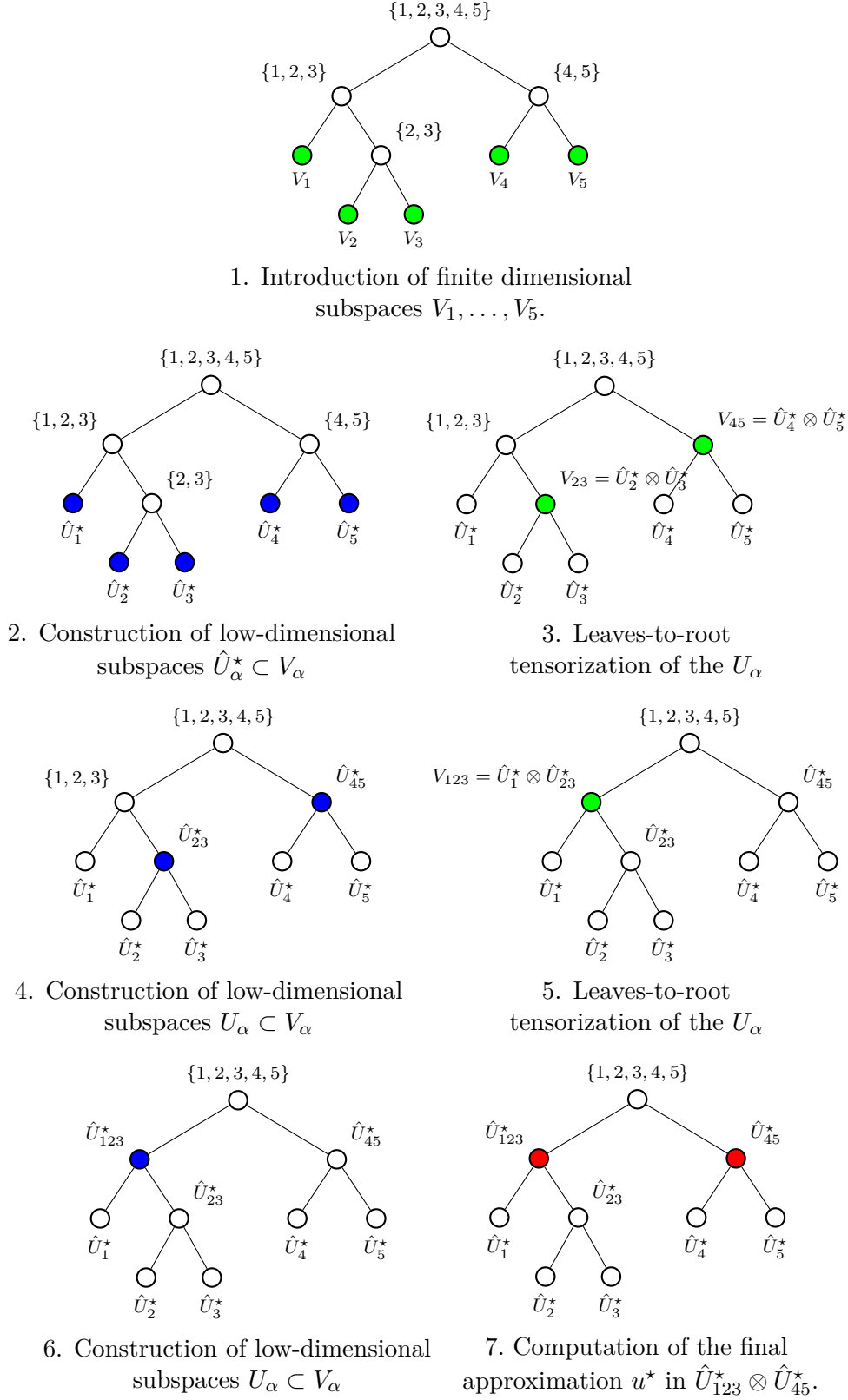


Figure 5.1 – Illustration of the leaves-to-root algorithm 5.2 for the construction of the approximation of a function in tree-based tensor format.

TREE ADAPTATION

Contents

6.1	Introduction	137
6.2	Estimation of α-ranks	139
6.2.1	Principle and algorithm	139
6.2.2	Numerical illustration	141
6.3	Leaves-to-root construction of the tree with local deterministic optimiza-	
	mizations	143
6.3.1	Max-mean rank strategy	143
6.3.2	Ballani and Grasedyck's strategy	145
6.4	Leaves-to-root construction of the tree with local stochastic optimiza-	
	tions	146
6.4.1	Principle and algorithm	146
6.4.2	Illustration of the algorithm	147
6.4.3	Illustration of the choices of the parameters	147
6.5	Adaptation of the tree with global stochastic optimizations	150
6.5.1	Principle and algorithm	150
6.5.2	Illustration of the algorithm	150
6.5.3	Illustration of the choice of the parameters	151
6.6	Numerical examples	153
6.7	Conclusions	157

6.1 Introduction

The choice of the tree may have a significant impact on the complexity required to reach a certain precision. In this chapter, we introduce different strategies for tree adaptation.

The purpose of the following example is to show the influence of the choice of a particular dimension partition tree T on the number of evaluations n necessary to compute the approximation

u^* with a desired precision ε .

We consider $\mathcal{X} = [-1, 1]^d$, equipped with the uniform measure and the function u , which is the sum of bivariate functions defined as follows,

$$u(x) = g(x_1, x_2) + g(x_3, x_4) + \dots + g(x_{d-1}, x_d), \text{ where } g(x_\nu, x_{\nu+1}) = \sum_{i=0}^3 x_\nu^i x_{\nu+1}^i. \quad (6.1)$$

We consider $V = \bigotimes_{\nu=1}^d \mathbb{P}_4(\mathcal{X}^\nu)$, so that there is no discretization error in $\mathcal{T}_r^T(V)$.

We compare the case when T is a balanced tree over $\{1, \dots, d\}$ and when it is a balanced tree over a permutation of $\{1, \dots, d\}$ (see respectively Figures 6.1 and 6.2 for an illustration in the case $d = 8$), the α -ranks are also represented respectively on Figures 6.1 and 6.2. We observe that the α -ranks are higher with the permuted tree.

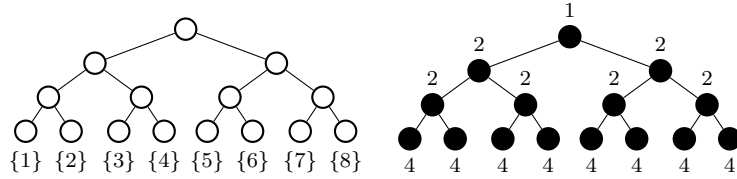


Figure 6.1 – On the left balanced tree over $\{1, \dots, d\}$ and on the right the α -ranks are noticed with the nodes for the sum of bivariate functions defined by (6.1) with $d = 8$.

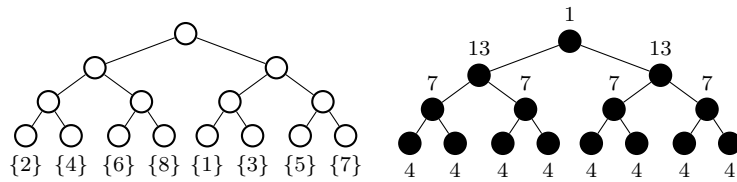


Figure 6.2 – On the left balanced tree over a permutation of $\{1, \dots, d\}$ and on the right the α -ranks are noticed with the nodes for the sum of bivariate functions defined by (6.1) with $d = 8$.

As explained in Section 5.3.3 from Chapter 5, the number of evaluations n necessary to get a desired accuracy ε using Algorithm 5.2 depends on the α -ranks required for such accuracy, and this for all $\alpha \in T$. For a given precision ε , the α -ranks of the leaves are independent of the choice of the tree T . To reduce the number of evaluations n we propose to look for strategies which reduce the α -ranks r_α , for all interior nodes $\alpha \in T \setminus \mathcal{L}(T)$. Several possibilities are considered,

- either we minimize the maximum of the α -ranks,
- or we minimize a functional which expresses the storage complexity of the tensors associated with the interior nodes

$$\mathcal{C}(r, T) = \sum_{\alpha \notin \mathcal{L}(T)} r_\alpha \prod_{\beta \in S(\alpha)} r_\beta. \quad (6.2)$$

As the number of possible trees T scales exponentially in the dimension d , finding the best tree (i.e. the tree that optimizes one of the above criteria) is a combinatorial problem and the global cost, due to the estimations of the α -ranks, for the search of the best tree among all the possibilities is not affordable in the context of costly evaluations.

In this chapter, we propose three different strategies to find a tree T with the objective of reducing the number of evaluations necessary to get a certain accuracy. First, we present two strategies that include tree optimization inside the algorithm for the construction of the approximation (Algorithm 5.2 presented in Chapter 5). As the algorithm goes from the leaves to the root, the number of possible trees (and thus the number of α -ranks to be evaluated) decreases sharply. This is why we propose both a stochastic strategy, (in Section 6.4) and a deterministic strategy which will explore a larger number of trees, (in Subsection 6.3.1). Then, we also propose in Section 6.5 a global stochastic strategy inspired from [48] which explores several dimension trees and selects the one minimizing the complexity functional given by Equation (6.2).

6.2 Estimation of α -ranks

6.2.1 Principle and algorithm

Performing tree optimization requires the estimations of α -ranks $r_\alpha(\varepsilon)$ for reaching a precision ε . These estimations require evaluations of the function u and this cost (denoted n_{optim}) should be reasonable, with respect to the number of evaluations required for constructing the approximation, denoted n . In [10], the authors propose a strategy based on an adaptive cross approximation technique [11] to estimate the α -ranks.

In the following algorithm, we propose a strategy to estimate the α -ranks $r_\alpha(\varepsilon)$ to achieve an empirical relative error ε , based on leave-one-out cross validations. To do this, we consider the matrix of the evaluations of u , $\mathbf{B}^\alpha = \{u(x_\alpha^l, x_{\alpha^c}^k) : 1 \leq l \leq z_\alpha, 1 \leq k \leq z_{\alpha^c}\}$ where $\{x_\alpha^l\}_{l=1}^{z_\alpha}$ are i.i.d samples of X_α and $\{x_{\alpha^c}^k\}_{k=1}^{z_{\alpha^c}}$ are i.i.d samples of X_{α^c} . We introduce $\mathbf{B}_{\setminus i}^\alpha$, the matrix \mathbf{B}^α without the column i , which admits a singular value decomposition

$$\mathbf{B}_{\setminus i}^\alpha = \sum_{k \geq 1} \sigma_{\alpha}^{\setminus i, k} \mathbf{v}_{\alpha}^{\setminus i, k} (\mathbf{v}_{\alpha^c}^{\setminus i, k})^T, \quad (6.3)$$

where $\sigma_\alpha^{\setminus i,k}$ are the singular values sorted in decreasing order, $\mathbf{v}_\alpha^{\setminus i,k}$ and $\mathbf{v}_{\alpha^c}^{\setminus i,k}$ are respectively the left and right singular vectors of $\mathbf{B}_{\setminus i}^\alpha$. For all $r \in \llbracket 1 ; \min(z_\alpha, z_{\alpha^c}) \rrbracket$, let $\mathbf{V}_{\setminus i,r}^\alpha$ be the matrix whose columns are $(\mathbf{v}_{\setminus i,1}^\alpha, \dots, \mathbf{v}_{\setminus i,r}^\alpha)$, the rank $r_\alpha(\varepsilon)$ is estimated as the minimal integer such that

$$\frac{1}{z_{\alpha^c}} \sum_{i=1}^{z_{\alpha^c}} \|\mathbf{B}_i^\alpha - \mathbf{V}_{\setminus i,r_\alpha(\varepsilon)}^\alpha (\mathbf{V}_{\setminus i,r_\alpha(\varepsilon)}^\alpha)^T \mathbf{B}_i^\alpha\|_2^2 \leq \varepsilon^2 \frac{1}{z_{\alpha^c}} \sum_{i=1}^{z_{\alpha^c}} \|\mathbf{B}_i^\alpha\|_2^2, \quad (6.4)$$

where \mathbf{B}_i^α denotes the i^{th} column of \mathbf{B}^α .

Estimating the α -ranks with a small precision ε may require many evaluations. It is important to underline that to perform tree optimization we do not need to know the exact value of $r_\alpha(\varepsilon)$ but we want to have an estimation accurate enough to detect whether $r_\alpha(\varepsilon)$ is high or not. Therefore, r_α is estimated with a coarse precision ε_c . The whole strategy is described in the Algorithm 6.1.

Algorithm 6.1 Determination of the ranks $r_\alpha(\varepsilon_c)$ of a function u

Inputs: coarse tolerance ε_c , function u , tuple α , product measure μ , n_α and n_{α^c}

Outputs: r_α and $cost = z_\alpha z_{\alpha^c}$

Generate z_α i.i.d samples $\{x_\alpha^i\}_{i=1}^{z_\alpha}$ from the measure μ_α

Generate $z_{\alpha^c} = 1$ sample $\{x_{\alpha^c}^j\}_{j=1}^{z_{\alpha^c}}$ from the measure μ_{α^c}

Evaluate the function u on the grid $\{(x_\alpha^i, x_{\alpha^c}^j) : 1 \leq i \leq z_\alpha, 1 \leq j \leq z_{\alpha^c}\}$ and set $\mathbf{B}^\alpha = (u(x_\alpha^i, x_{\alpha^c}^j))$

Set $r = 0$ and $\mathcal{E}(r) = \infty$

while $\mathcal{E}(r) > \varepsilon_c$ and $z_{\alpha^c} \leq n_{\alpha^c}$ **do**

Set $z_{\alpha^c} \leftarrow z_{\alpha^c} + 1$

Sample $\{x_{\alpha^c}^{z_{\alpha^c}}\}$ from the measure μ_{α^c}

Update $\mathbf{B}^\alpha = [\mathbf{B}^\alpha, \mathbf{b}^\alpha]$, with \mathbf{b}^α the vector corresponding to the evaluations of u on the grid $\{(x_\alpha^i, x_{\alpha^c}^{z_{\alpha^c}}) : 1 \leq i \leq z_\alpha\}$

while $(\mathcal{E}(r) > \varepsilon_c \text{ and } r < \min(n_\alpha, z_{\alpha^c}))$ **do**

Set $r \leftarrow r + 1$

for $i = 1, \dots, z_{\alpha^c}$ **do**

Determine the matrix $\mathbf{V}_{\setminus i,r}^\alpha$ of r left singular vectors of $\mathbf{B}_{\setminus i}^\alpha$.

end for

Set

$$\mathcal{E}(r)^2 = \frac{\sum_{i=1}^{z_{\alpha^c}} \|\mathbf{B}_i^\alpha - \mathbf{V}_{\setminus i,r}^\alpha (\mathbf{V}_{\setminus i,r}^\alpha)^T \mathbf{B}_i^\alpha\|_2^2}{\sum_{i=1}^{z_{\alpha^c}} \|\mathbf{B}_i^\alpha\|_2^2} \quad (6.5)$$

end while

end while

Set $r_\alpha = \min\{1 \leq k \leq r : \mathcal{E}(k) \leq \varepsilon_c\}$

6.2.2 Numerical illustration

In this section, to study the efficiency of Algorithm 6.1 we consider two numerical examples and we observe how well the algorithm estimates the α -ranks. The robustness of the algorithm is evaluated by repeating the algorithm 10 times and computing empirical confidence intervals of level 10% and 90% for the estimated $r_\alpha(\varepsilon)$ and the associated cost.

Sum of a bivariate functions (with separated variables).

We consider $\mathcal{X} = [-1, 1]^d$, with $d = 16$, equipped with the uniform measure and the function u defined by (6.1). Again, we consider $V = \bigotimes_{\nu=1}^d \mathbb{P}_3(\mathcal{X}^\nu)$, so that there is no discretization error. We want to estimate the α -ranks, for $\alpha = \{1, 2\}$, $\alpha = \{1, 3\}$ and $\alpha = \{1, 3, 5, 7\}$ and this for different values of the coarse precision ε_c . In this case, the α -ranks are exactly known and the results are presented in Table 6.1.

	$\alpha_{12} (r_{\alpha_{12}} = 2)$		$\alpha_{13} (r_{\alpha_{13}} = 8)$		$\alpha_{1357} (r_{\alpha_{1357}} = 16)$	
ε_c	r_α	$cost$	r_α	$cost$	r_α	$cost$
0.05	[1; 1]	[20; 20]	[1; 3]	[20; 40]	[3; 5]	[40; 60]
0.01	[1; 2]	[20; 30]	[3; 6]	[40; 70]	[7; 9]	[80; 100]
0.001	[1; 2]	[20; 30]	[6; 7]	[70; 80]	[9; 9]	[100; 100]
10^{-14}	[2; 2]	[30; 30]	[7; 8]	[80; 90]	[9; 9]	[100; 100]

(a) $d = 16, n_\alpha = n_{\alpha^c} = 10$

	$\alpha_{12} (r_{\alpha_{12}} = 2)$		$\alpha_{13} (r_{\alpha_{13}} = 8)$		$\alpha_{1357} (r_{\alpha_{1357}} = 16)$	
ε_c	r_α	$cost$	r_α	$cost$	r_α	$cost$
0.05	[1; 1]	[40; 40]	[1; 3.5]	[40; 90]	[3; 6]	[80; 140]
0.01	[1; 2]	[40; 60]	[3; 6]	[80; 140]	[8; 10.5]	[180; 230]
0.001	[2; 2]	[60; 60]	[6; 7]	[140; 160]	[12; 13]	[260; 280]
10^{-14}	[2; 2]	[60; 60]	[7; 8]	[160; 180]	[13; 14]	[280; 300]

(b) $d = 16, n_\alpha = n_{\alpha^c} = 20$

Table 6.1 – Illustration of Algorithm 6.1 for the sum of bivariate functions defined by (6.1) with $d = 16$.

Table 6.1a presents the results when $n_\alpha = n_{\alpha^c} = 10$. The α -ranks to reach ε are relatively well estimated when $\varepsilon_c \leq 0.01$. However, when $\varepsilon_c = 0.05$, we are barely able to distinguish the high ranks from the others. When $\varepsilon_c = 10^{-14}$, the rank is almost full regarding the values of n_α and n_{α^c} . Table 6.1b shows that the strategy can estimate ranks that are higher than 10, when $n_\alpha = n_{\alpha^c} = 20$, but the number of evaluations is also higher. In conclusion for this example, choosing $n_\alpha = n_{\alpha^c} = 10$ and $\varepsilon_c = 10^{-2}$ give a rather good distinction between high and low α -ranks.

Chain function.

We consider $\mathcal{X} = [-1, 1]^d$, with $d = 16$ equipped with the uniform measure and the function u , which is the sum of trivariate functions defined as follows,

$$u(x) = g(x_1, x_2, x_3) + g(x_2, x_3, x_4) + \dots + g(x_{d-3}, x_{d-2}, x_{d-1}) + g(x_{d-2}, x_{d-1}, x_d), \quad (6.6)$$

where $g(x_\nu, x_{\nu+1}, x_{\nu+2}) = \sum_{i=0}^2 x_\nu^i x_{\nu+1}^i x_{\nu+2}^i$. We consider $V = \bigotimes_{\nu=1}^d \mathbb{P}_3(\mathcal{X}^\nu)$, so that there is no discretization error. We want to estimate the α -ranks, for $\alpha = \{1, 2\}$, $\alpha = \{1, 3, 5, 7\}$ and $\alpha = \{1, 2, 3, 4, 5, 6, 7, 8\}$ and this for different values of the coarse precision ε_c .

	α_{12}		α_{1357}		$\alpha_{12345678}$	
ε_c	r_α	$cost$	r_α	$cost$	r_α	$cost$
0.05	[1; 1]	[20; 20]	[1; 1.5]	[20; 25]	[1; 1]	[20; 20]
0.01	[1; 3.5]	[20; 45]	[6; 8]	[70; 90]	[1.5; 4]	[25; 50]
0.001	[3.5; 5]	[45; 60]	[9; 9]	[100; 100]	[4; 6]	[50; 70]
10^{-14}	[5; 5]	[60; 60]	[9; 9]	[100; 100]	[6; 6]	[70; 70]

(a) $d = 16, n_\alpha = n_{\alpha^c} = 10$

	α_{12}		α_{1357}		$\alpha_{12345678}$	
ε_c	r_α	$cost$	r_α	$cost$	r_α	$cost$
0.05	[1; 1]	[40; 40]	[1; 1.5]	[40; 50]	[1; 1]	[40; 40]
0.01	[1.5; 4]	[50; 100]	[6.5; 10]	[150; 220]	[1; 4]	[40; 100]
0.001	[3; 5]	[90; 120]	[11.5; 13]	[250; 280]	[4.5; 6]	[110; 140]
10^{-14}	[5; 5]	[120; 120]	[13; 13.5]	[280; 290]	[6; 6]	[140; 140]

(b) $d = 16, n_\alpha = n_{\alpha^c} = 20$

Table 6.2 – Illustration of Algorithm 6.1 for the chain function defined by (6.6) with $d = 16$.

The conclusions that can be drawn for the chain function are similar to the one made for the sum of bivariate functions. Choosing $n_\alpha = n_{\alpha^c} = 10$ and $\varepsilon_c = 10^{-2}$ give a rather good distinction between the high and low α -ranks. Increasing n_α and n_{α^c} to 20 do not provide a better estimation of the α -ranks for $\varepsilon_c = 10^{-2}$ and imposing $\varepsilon_c = 10^{-3}$ is not competitive regarding the cost. The conclusion for this example is the same than with the sum of bivariate functions. In the following numerical examples, we thus choose $n_\alpha = n_{\alpha^c} = 10$ and $\varepsilon_c = 10^{-2}$.

6.3 Leaves-to-root construction of the tree with local deterministic optimizations

In this section, we present a leaves-to-root strategy, called max-mean rank algorithm, and the strategy from [10], where the tree T is adaptively constructed (with local deterministic optimization) during the algorithm 5.2 from Chapter 5.

6.3.1 Max-mean rank strategy

Let $\Lambda = \{\alpha_1, \dots, \alpha_l\}$ be a partition of $D = \{1, \dots, d\}$. For $\Lambda = \{\alpha_1, \dots, \alpha_l\}$ the algorithm will compare all the possible pairings denoted $\mathcal{K}(\Lambda)$ (detailed hereafter) and select the one which minimizes a certain rank-based criterion (also detailed hereafter). We consider $\mathcal{K}(\Lambda) = \{\beta = (\alpha_i, \alpha_j) : (i, j) \in \{1, \dots, l\}^2, i \neq j\}$ the set containing all possible pairings of two elements of Λ . The cardinal of $\mathcal{K}(\Lambda)$ is equal to $\frac{l(l-1)}{2}$. For each $\beta \in \mathcal{K}(\Lambda)$, we estimate all the β -ranks and evaluate $R_{\beta^c} = \frac{1}{\#\mathcal{K}(\Lambda \setminus \beta)} \sum_{\gamma \in \mathcal{K}(\Lambda \setminus \beta)} r_\gamma$, with $\Lambda \setminus \beta := \Lambda \setminus (\{\alpha_1\} \cup \{\alpha_2\})$. We select β which minimizes $\max\{r_\beta, R_{\beta^c}\}$. If several β are solutions of the minimization problem $\min_{\beta \in \mathcal{K}(\Lambda)} \max\{r_\beta, R_{\beta^c}\}$, β^* is selected at random among the minimizers and the strategy is not fully deterministic. Once a pair $\beta \in \mathcal{K}(\Lambda)$ is selected, the procedure continues with the set Λ private from the elements of the chosen pairs. The strategy is summarized in Algorithm 6.2.

Algorithm 6.2 Deterministic optimization of nodes pairing

Inputs: function to approximate u , measure μ , partition Λ of D .

Outputs: Partition Γ of D .

Set $\Gamma = \emptyset$.

while $\#\Lambda > 2$ **do**

For all $\beta = (\alpha_1, \alpha_2) \in \mathcal{K}(\Lambda)$, estimate the β -ranks $r_\beta = r_\beta(\varepsilon)$ with Algorithm 6.1 and set $R_{\beta^c} = \frac{1}{\#\mathcal{K}(\Lambda \setminus \beta)} \sum_{\gamma \in \mathcal{K}(\Lambda \setminus \beta)} r_\gamma$, with $\Lambda \setminus \beta := \Lambda \setminus (\{\alpha_1\} \cup \{\alpha_2\})$.

Choose $\beta^* = \arg \min_{\beta \in \mathcal{K}(\Lambda)} \max\{r_\beta, R_{\beta^c}\}$.

Set $\Lambda \leftarrow \Lambda \setminus (\{\alpha_1^*\} \cup \{\alpha_2^*\})$ with $\beta^* = (\{\alpha_1^*\} \cup \{\alpha_2^*\})$ and $\Gamma = \Gamma \cup \beta^*$.

end while

if $\#\Lambda = 2$ **then**

$\beta^* = \{\Lambda\}$

end if

$\Gamma = \Gamma \cup \Lambda$

The overall strategy that constructs the tree during the algorithm to construct the approximation in tree-based tensor format is given in Algorithm 6.3.

Algorithm 6.3 Adaptive construction of the tree with local optimization

Inputs: function to approximate u , measure μ , approximation spaces $V_\alpha, \alpha \in \mathcal{L}(T)$, tolerance ε , parameters relative to the rank estimation $\varepsilon_c, n_\alpha, n_{\alpha^c}$.

Outputs: the dimension tree T and the approximation u^*

Set $\Lambda = \{\{1\}, \dots, \{d\}\}$ and $T = \Lambda$

while $\#\Lambda > 1$ **do**

for $\alpha \in \Lambda$ **do**

if $\alpha \notin \mathcal{L}(T)$ **then**

 Set $V_\alpha = \bigotimes_{\beta \in S(\alpha)} \hat{U}_\beta^*$

end if

 Compute the estimation \hat{U}_α^* of the α -principal subspace of $\mathcal{Q}_{V_\alpha} u$ with relative reconstruction error ε , using Algorithm 5.1.

end for

 Determine a partition Γ of D by pairing elements of Λ thanks to Algorithm 6.2.

 Set $T \leftarrow T \cup \Gamma$ and $\Lambda \leftarrow \Gamma$

end while

Set $V_D = \bigotimes_{\alpha \in S(D)} \hat{U}_\alpha^*$

Compute $u^* = \mathcal{Q}_{V_D} u$

Set $T = T \cup D$.

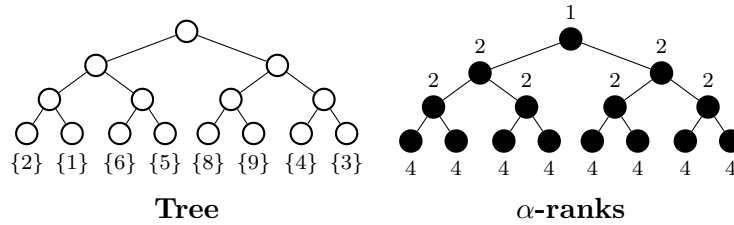
Illustration of the max-mean rank strategy

Figure 6.3 – Illustration of the max-mean rank algorithm for the sum of bivariate functions defined by (6.1) with $d = 8$, $\varepsilon_c = 10^{-2}$, $\varepsilon = 10^{-14}$, $n_\alpha = n_{\alpha^c} = 10$.

We consider again the sum of bivariate functions defined by (6.1) with $d = 8$. Figure 6.3 gives the tree constructed with the max-mean rank strategy for one run of the algorithm 6.3. This algorithm is not totally deterministic (because of the estimations of the α -ranks and the case where there are several minimizers of the optimality criteria). We observe that the max-mean rank algorithm finds the pairs which have the smallest α -ranks we can expect (as in the case of a balanced tree).

6.3.2 Ballani and Grasedyck's strategy

In [10], the authors present a method to select an appropriate tree using a subset of tensor entries without any a priori knowledge on the tree structure. Their goal is to minimize the storage complexity of a given tensor, which is closely related to the number of evaluations of the function u necessary to construct an approximation with a desired accuracy.

Their strategy constructs a tree in a leaves-to-root strategy by successive clusterings of disjoint subsets of $D = \{1, \dots, d\}$. p is the number of elements gathered at the same time (which corresponds to the tree's arity) and it can be chosen to limit the number of possibilities which are explored. The clustering criterion is based on an estimation of the α -ranks. The reader is referred to [10] for further details on this method. The authors explain that when $p > 3$ the computational cost for the adaptive part is much higher than the cost necessary to compute the approximation in the tree-based tensor format. As we only consider pairings, in the strategies from Sections 6.3.1 and 6.4, we set $p = 2$ in all the numerical examples. Let us notice that this strategy explores a larger set of possibilities of trees than with the strategy from Subsection 6.3.1.

Remark 6.1. As for the max-mean rank strategy, the choice of the pairings is made on a minimization criterion. In the case where there are several minimizers, we choose it randomly and therefore the strategy is not totally deterministic.

Illustration of the Ballani and Grasedyck's strategy

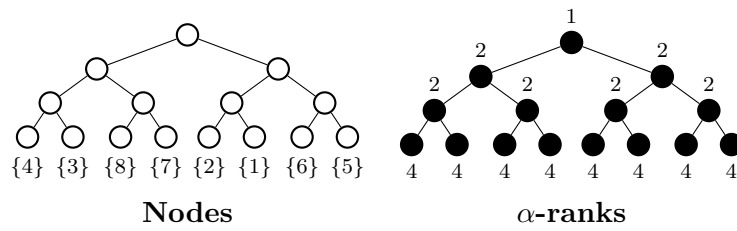


Figure 6.4 – Illustration of the algorithm from [10] for the sum of bivariate functions defined by (6.1) with $d = 8$, $\varepsilon_c = 10^{-2}$, $\varepsilon = 10^{-14}$, $n_\alpha = n_{\alpha^c} = 10$.

We consider again the sum of bivariate functions defined in (6.1) with $d = 8$. The constructed tree with the method from [10] is given by the Figure 6.4. This figure shows that the strategy from [10] is able to recover a tree with the lowest α -ranks we can expect for this example.

6.4 Leaves-to-root construction of the tree with local stochastic optimizations

6.4.1 Principle and algorithm

In this section, we present a leaves-to-root strategy, where the tree T is adaptively constructed during the algorithm 5.2 from Chapter 5.

Let $\Lambda = \{\alpha_1, \dots, \alpha_l\}$ be a partition of $D = \{1, \dots, d\}$. When $l = \#\Lambda$ is even, we consider $\mathcal{J}(\Lambda)$ the set of all partitions of Λ where each element has a cardinal equal to two. Each partition $\Gamma \in \mathcal{J}(\Lambda)$ contains thus $\frac{l}{2}$ elements. When $\#\Lambda$ is odd, we consider the set $\mathcal{J}(\Lambda) = \bigcup_{\alpha \in \Lambda} \bigcup_{\Lambda \in \mathcal{J}(\Lambda \setminus \{\alpha\})} \{\{\alpha\} \cup \Gamma\}$. Among all partitions of $\mathcal{J}(\Lambda)$, the aim is to find the one Γ which minimizes

$$\mathcal{C}_l(\Gamma) = \sum_{\beta \in \Gamma} r_\beta(\varepsilon) \prod_{\alpha \in \beta \cap \Lambda} r_\alpha(\varepsilon). \quad (6.7)$$

In practice, computing the function $\mathcal{C}_l(\Gamma)$ for all $\Gamma \in \mathcal{J}(\Lambda)$ requires a lot of α -ranks estimations and it is therefore not affordable. To minimize $\mathcal{C}_l(\Gamma)$, we propose a stochastic algorithm which finds a partition Γ associated to a minimal cost function $\mathcal{C}_l(\Gamma)$ among a limited set of different partitions. The principle is to compare a current partition Γ of $\mathcal{J}(\Lambda)$ with a new one Γ^* obtained from Γ by permuting two nodes selected according to a probability distribution defined hereafter, and to accept Γ^* if $\mathcal{C}_l(\Gamma^*) < \mathcal{C}_l(\Gamma)$.

The first node ν_1 is drawn in Λ according to the distribution

$$\mathbb{P}(\nu_1 = \alpha) \propto r_{P_\Gamma(\alpha)}(\varepsilon)^{\gamma_1}, \quad \text{where } P_\Gamma(\alpha) \text{ is the parent of } \alpha \text{ in } \Gamma. \quad (6.8)$$

A higher γ_1 increases the probability to select a node ν_1 whose parent in Γ has a high rank. Once the node ν_1 is selected, we consider the set $\Lambda \setminus (\{\nu_1\} \cup \{\nu_1^b\})$, where ν_1^b is the second element of the pair formed with ν_1 (in the case ν_1 is a singleton $\nu_1^b = \emptyset$), that is to say $P_\Gamma(\nu_1) = \nu_1 \cup \nu_1^b$. Then, we draw a second node ν_2 in $\Lambda \setminus (\{\nu_1\} \cup \{\nu_1^b\})$ according to the distribution

$$\mathbb{P}(\nu_2 = \alpha | \nu_1) \propto r_{P_\Gamma(\alpha)}(\varepsilon)^{\gamma_2}, \quad \text{where } \alpha \in \Lambda \setminus (\{\nu_1\} \cup \{\nu_1^b\}). \quad (6.9)$$

Again, a higher γ_2 increases the probability to select a node ν_2 whose parent in Γ has a high rank. If the permutation of the two nodes ν_1 and ν_2 decreases the cost function, then the two nodes are permuted. n_P successive random permutations of the nodes are performed according to this distribution. The last partition Γ is the one associated to the lowest cost function $\mathcal{C}_l(\Gamma)$ among all the visited partitions.

Algorithm 6.4 Optimization of nodes pairing

Inputs: function to approximate u , partition Λ of D , maximal number of iterations n_P , γ_1 , γ_2 .

Outputs: Γ

Choose randomly $\Gamma \in \mathcal{J}(\Lambda)$.

Calculate $\mathcal{C}_l(\Gamma)$ according to Eq. (6.7), with estimation of the α -ranks using Algorithm 6.1.

for $k = 1, \dots, n_P$ **do**

$\Gamma^* \leftarrow \Gamma$

 Draw ν_1 according to distribution (6.8) and then ν_2 according to distribution (6.9).

 Calculate $\mathcal{C}_l(\Gamma^*)$ according to (6.7), with estimation of the α -ranks using Algorithm 6.1.

if $\mathcal{C}_l(\Gamma^*) \leq \mathcal{C}_l(\Gamma)$ **then**

$\Gamma \leftarrow \Gamma^*$

end if

end for

To construct the final approximation, we use Algorithm 6.3, where the optimal pairings are determined with Algorithm 6.4.

6.4.2 Illustration of the algorithm

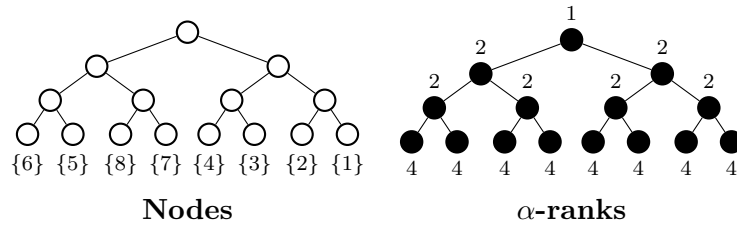


Figure 6.5 – Illustration of the stochastic algorithm for the sum of bivariate functions defined by (6.1) with $d = 8$, $\varepsilon_c = 10^{-2}$, $\varepsilon = 10^{-14}$, $n_\alpha = n_{\alpha^c} = 10$, $\gamma_1 = \gamma_2 = 6$ and $n_P = 2d$.

We consider again the sum of bivariate functions from (6.1) with $d = 8$. The constructed tree with the local stochastic algorithm is given by the Figure 6.5 for one run of the algorithm. This algorithm is stochastic both for the estimations of the α -ranks and the choice of the tree so the result may differ strongly for two different runs. We observe that for the sum of bivariate functions defined by (6.1), the stochastic algorithm finds all the pairs which have the smallest α -ranks we can expect (like for the balanced tree from Figure 6.1).

6.4.3 Illustration of the choices of the parameters

In this section, we want to study the influence of the choice of the parameters n_P , γ_1 and γ_2 in Algorithm 6.4 on the number of evaluations n necessary to compute the approximation and the

number of evaluations necessary for the α -ranks estimations n_{optim} . To do this, we consider two numerical examples: the sum of bivariate functions from (6.1), and the chain function from (6.6). As in the previous chapters, the error of approximation is defined by

$$\varepsilon(u^*) = \left(\frac{1}{n_{test}} \sum_{x \in \mathbf{x}_{test}} (u(x) - u^*(x))^2 \right)^{1/2}.$$

In practice, we choose $n_{test} = \#\mathbf{x}_{test} = 1000$. The robustness of the algorithm is evaluated by repeating the algorithm 10 times and by computing empirical confidence intervals of level 10% and 90% for the error of approximation $\log(\varepsilon(u^*))$, the storage complexity \mathcal{S} , the number of evaluations necessary to compute the approximation n , the number of evaluations necessary to estimate the α -ranks n_{optim} and the total number of evaluations n_{total} . To limit the number of choices, we consider $\gamma_1 = \gamma_2 = \gamma$. The higher n_P is, the more tree configurations are explored. The higher γ is, the more configurations avoiding high ranks are privileged in the random search.

Remark 6.2. n_{optim} denotes the number of evaluations of the function u that have been necessary to estimate the α -ranks with Algorithm 6.1 and n is the number of evaluations necessary to construct the approximation. We evaluate the performance of the algorithm for different choices of the parameters by comparing $n_{total} = n + n_{optim}$.

Local optimization						
γ	n_P	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{optim}	n_{total}
2	d	[-14.3; -13.9]	[4220; 15782]	[4988; 17045]	[4580; 6190]	[9748; 23190]
	$2d$	[-14.2; -13.7]	[3324; 11961]	[4013; 13090]	[6310; 8405]	[10288; 21035]
	$5d$	[-13.9; -13.4]	[2276; 8982]	[2882; 9990]	[9495; 11925]	[12377; 21290]
3	d	[-14.5; -14.0]	[4751; 13641]	[5585; 14803]	[4635; 5940]	[10813; 20658]
	$2d$	[-14.1; -13.8]	[3833; 13817]	[4574; 15025]	[5940; 7970]	[11014; 23285]
	$5d$	[-14.1; -13.5]	[2050; 7267]	[2636; 8171]	[7920; 10470]	[10872; 17641]
4	d	[-14.5; -14.1]	[4989; 11285]	[5853; 12375]	[4100; 5790]	[10348; 17689]
	$2d$	[-14.3; -13.9]	[3406; 9701]	[4092; 10755]	[4715; 7850]	[8742; 17697]
	$5d$	[-14; -13.7]	[1458; 8581]	[2016; 9589]	[6175; 10170]	[8772; 19611]
6	d	[-14.5; -13.9]	[4835; 13639]	[5671; 14841]	[3495; 5300]	[9398; 20092]
	$2d$	[-14.2; -13.6]	[3432; 10611]	[4128; 11718]	[4175; 6820]	[8069; 18328]
	$5d$	[-14; -13.5]	[1372; 6053]	[1901; 6924]	[4925; 8260]	[7616; 15803]

Table 6.3 – Sum of bivariate functions defined by (6.1) with $d = 24$. Approximation with local tree optimizations with prescribed tolerance $\varepsilon = 10^{-13}$, $\gamma_1 = \gamma_2 = \gamma$ and ranks estimations with $\varepsilon_c = 10^{-2}$, $n_\alpha = n_{\alpha^c} = 10$. Confidence intervals for relative errors $\log(\varepsilon(u^*))$, storage complexity \mathcal{S} , number of evaluations for the estimations of the α -ranks n_{optim} and the number of evaluations for the approximation n .

Table 6.3 presents the results for the sum of bivariate functions defined by (6.1). For small γ ($\gamma = 2$), a small n_P ($n_P = d$) is better (in the sense of the total number of evaluations n_{total}) than a high n_P . That means that the good partitions leading to small α -ranks are explored in the first possibilities. We also observe that for γ higher than 2, high n_P give better results than low n_P . For this particular example, the lowest total number of evaluations n_{total} is obtained with the highest γ ($\gamma = 6$), and the highest n_P ($n_P = 5d$). With this parametrization, the stochastic local strategy tends to a deterministic strategy avoiding the nodes with the highest α -ranks.

Local optimization						
γ	n_P	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{optim}	n_{total}
2	d	[-14.1; -13.6]	[12507; 18368]	[13511; 19545]	[7320; 8075]	[21226; 27145]
	$2d$	[-13.9; -13.2]	[13386; 18435]	[14448; 19579]	[10685; 11840]	[25292; 30844]
	$5d$	[-13.6; -12.9]	[11967; 17645]	[12992; 18772]	[16250; 17355]	[30082; 35307]
3	d	[-14; -13.8]	[12678; 18703]	[13688; 19841]	[7270; 8155]	[21229; 27636]
	$2d$	[-13.9; -13.6]	[11432; 17016]	[12388; 18154]	[10665; 11645]	[23138; 29674]
	$5d$	[-13.8; -13.3]	[10876; 18316]	[11844; 19514]	[15740; 17185]	[27584; 36516]
4	d	[-14.0; -13.7]	[13387; 16620]	[14386; 17707]	[7235; 8005]	[21961; 25262]
	$2d$	[-13.9; -13.5]	[13584; 16499]	[14583; 17653]	[10615; 11495]	[25338; 28758]
	$5d$	[-13.8; -13.4]	[12835; 16716]	[13845; 17833]	[15205; 16755]	[29688; 34375]
6	d	[-14.1; -13.8]	[12265; 19663]	[13249; 20804]	[7185; 8170]	[20434; 28629]
	$2d$	[-13.9; -13.4]	[12292; 18079]	[13205; 19301]	[9890; 10935]	[23790; 29366]
	$5d$	[-13.6; -13.1]	[12332; 17397]	[13327; 18554]	[14670; 16160]	[28947; 34119]

Table 6.4 – Chain function defined by (6.6) with $d = 16$. Approximation with local tree optimizations with prescribed tolerance $\varepsilon = 10^{-13}$, $\gamma_1 = \gamma_2 = \gamma$ and ranks estimations with $\varepsilon_c = 10^{-2}$, $n_\alpha = n_{\alpha^c} = 10$. Confidence intervals for relative errors $\log(\varepsilon(u^*))$, storage complexity \mathcal{S} , number of evaluations for the estimations of the α -ranks n_{optim} and the number of evaluations for the approximation n .

Table 6.4 presents the results for the chain function defined by (6.6). For each γ , a small n_P ($n_P = d$) is better (in the sense of the total number of evaluations n_{total}) than a high n_P . We also observe that high γ give slightly better results than low γ . But the differences between the γ are less marked than with the first function. However in this example low values of n_P give better results than high n_P .

According to the observations made in the two cases, for the next numerical examples we choose $\gamma = 6$ and $n_P = 2d$ (as a trade-off between the two examples).

6.5 Adaptation of the tree with global stochastic optimizations

In this section, we present a global stochastic optimization strategy to determine a relevant dimension tree before computing the approximation of the function u .

6.5.1 Principle and algorithm

We estimate the α -ranks for a given tree with Algorithm 6.1 and we search among a finite number of trees the one that minimizes the cost function from Equation (6.2). The set of trees is obtained by performing a random walk over the set of trees.

To explain the stochastic algorithm for tree optimization, we detail how to change a current tree T into a new tree T^* which decreases the cost function from Equation (6.2) with a relatively high probability. Concretely, the new tree T^* is obtained from T by permuting two nodes ν_1 and ν_2 . Considering a tree T , the first node ν_1 is drawn in $T \setminus \{D\}$ according to the distribution

$$\mathbb{P}(\nu_1 = \alpha | T) \propto r_{P_T(\alpha)}(\varepsilon)^{\gamma_1}, \quad \alpha \in T \setminus \{D\} \quad (6.10)$$

where $P_T(\alpha)$ denotes the parent of α in the tree T . A higher γ_1 increases the probability to select a node ν_1 with a high parent's rank. Once the node ν_1 is selected, we draw a second node ν_2 in $T \setminus \{D\}$ according to the distribution:

$$\mathbb{P}(\nu_2 = \alpha | T) \propto \begin{cases} d_T(\alpha, \nu_1)^{-\gamma_2} & \text{if } \alpha \cap \nu_1 = \emptyset \\ 0 & \text{otherwise} \end{cases} \quad (6.11)$$

with $\gamma_2 > 0$ and $d_T(\alpha, \nu) = l_T(\alpha) + l_T(\nu) - 2l_T(\mathcal{A}(\alpha, \nu))$ where $\mathcal{A}(\alpha, \nu)$ denotes the highest-level common ascendant of α and ν in T and $l_T(\alpha)$ is the level of α in T . If the permutation of the two nodes decreases the cost function, the new tree T^* is retained and the procedure is repeated, n_p times. The procedure is formalized in Algorithm 6.5.

6.5.2 Illustration of the algorithm

We consider again the sum of bivariate functions from (6.1) with $d = 8$. The constructed tree with the global stochastic algorithm is given by the Figure 6.6 for one run of the algorithm. This algorithm is stochastic both for the estimations of the α -ranks and the choice of the tree so the result may differ for another run. We observe that in this case the stochastic algorithm does not find all the pairs which have the smallest α -ranks we can expect (like for the balanced tree). However it tends to moderate α -ranks (the α -ranks are far from the high ranks from the balanced tree with

repeating the algorithm 10 times and by computing empirical confidence intervals of level 10% and 90% for $\log(\varepsilon(u^*))$, \mathcal{S} , n and n_{optim} . To limit the number of choices, we consider $\gamma_1 = \gamma_2 = \gamma$. The higher n_P is, the more tree configurations are explored. The higher γ is, the more configurations avoiding high ranks are privileged in the random search.

Remark 6.3. In the same manner than in Section 6.4, n_{optim} denotes the number of evaluations of the function u that have been necessary to estimate the α -ranks with Algorithm 6.1 and n is the number of evaluations necessary to construct the approximation. We evaluate the performance of the algorithm for different choices of the parameters by comparing the values of n_{total} (with $n_{total} = n + n_{optim}$).

Initial Tree = Permuted Balanced tree						
γ	n_P	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{optim}	n_{total}
2	d	[-14.0; -13.4]	[6066; 19394]	[6996; 20897]	[9130; 11765]	[17166; 31177]
	$2d$	[-14.1; -13.3]	[4496; 8810]	[5334; 9887]	[16460; 19585]	[21889; 28409]
	$5d$	[-14.3; -13.6]	[2162; 6101]	[2790; 7023]	[31790; 45720]	[34580; 50806]
3	d	[-14.0; -13.4]	[5448; 12027]	[6280; 13327]	[7820; 11640]	[14292; 23957]
	$2d$	[-14.1; -12.9]	[3776; 11753]	[4554; 12944]	[14780; 19930]	[19634; 31269]
	$5d$	[-14.3; -13.6]	[1962; 6199]	[2562; 7131]	[29655; 41855]	[32756; 46108]
4	d	[-13.8; -13.0]	[4966; 11946]	[5844; 13230]	[7460; 11250]	[14519; 22929]
	$2d$	[-14.2; -13.7]	[3931; 9046]	[4730; 10169]	[14840; 19145]	[20322; 26233]
	$5d$	[-14.2; -13.3]	[2327; 4739]	[2979; 5585]	[29695; 44000]	[33247; 48853]
6	d	[-13.8; -13.5]	[6628; 12939]	[7593; 14206]	[8115; 10840]	[16138; 24116]
	$2d$	[-14.1; -13.7]	[4432; 8082]	[5241; 9136]	[14190; 20655]	[21014; 29175]
	$5d$	[-14.2; -13.6]	[1895; 6824]	[2474; 7838]	[33690; 44265]	[36784; 50532]

Table 6.5 – Sum of bivariate functions defined by (6.1) with $d = 24$. Approximation with global tree optimization with prescribed tolerance $\varepsilon = 10^{-13}$ and rank estimation with $\varepsilon_c = 10^{-2}$ and $n_\alpha = n_{\alpha^c} = 10$. Confidence intervals for relative errors $\log(\varepsilon(u^*))$, storage complexity \mathcal{S} , number of evaluations for the estimations of the α -ranks n_{optim} and the number of evaluations for the approximation n .

Table 6.5 presents the results with the sum of bivariate functions. It shows that choosing $n_P \geq 2d$ implies a number of evaluations for the α -ranks estimations that it is high compared to n_{total} . This is due to the fact that the initial tree is a tree with high α -ranks and it requires numerous iterations to find a good tree. This observation is verified for all γ . However we notice that the the lowest n_{total} is given by $\gamma = 4$.

Initial Tree = Permuted Balanced tree						
γ	n_P	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{optim}	n_{total}
2	d	[-13.9; -13.2]	[10227; 18662]	[11140; 19848]	[7495; 10445]	[20845; 28548]
	$2d$	[-14.2; -13.5]	[7731; 19506]	[8577; 20628]	[13830; 17530]	[26107; 34658]
	$5d$	[-14.0; -13.4]	[7135; 18061]	[7986; 19206]	[30550; 43900]	[40821; 62602]
3	d	[-14.0; -13.3]	[10161; 19387]	[11148; 20760]	[5875; 7560]	[17498; 27965]
	$2d$	[-14.1; -13.5]	[10345; 20527]	[11401; 21777]	[11605; 13635]	[24142; 33382]
	$5d$	[-14.1; -13.5]	[11171; 18471]	[12150; 19539]	[23735; 29005]	[38135; 46313]
4	d	[-14.0; -13.1]	[11351; 20006]	[12300; 21387]	[5240; 6430]	[17865; 27472]
	$2d$	[-14.0; -13.3]	[11250; 19024]	[12210; 20329]	[7435; 10505]	[21490; 29054]
	$5d$	[-14.1; -13.5]	[10542; 20810]	[11498; 22194]	[17825; 20800]	[30478; 41723]
6	d	[-14.0; -13.3]	[11700; 20489]	[12654; 21682]	[4085; 5340]	[16924; 26202]
	$2d$	[-14.0; -13.6]	[10229; 17952]	[11129; 19147]	[6070; 8330]	[19003; 26535]
	$5d$	[-14.0; -13.3]	[11424; 18925]	[12356; 20053]	[10900; 14255]	[24326; 31610]

Table 6.6 – Chain function defined by (6.6) with $d = 24$. Approximation with global tree optimization with prescribed tolerance $\varepsilon = 10^{-14}$ and rank estimation with $\varepsilon_c = 10^{-2}$ and $n_\alpha = n_{\alpha^c} = 10$. Confidence intervals for relative errors $\log(\varepsilon(u^*))$, storage complexity \mathcal{S} , number of evaluations for the estimations of the α -ranks n_{optim} and the number of evaluations for the approximation n .

Table 6.6 presents the results for the chain function defined by (6.6). The conclusions that can be drawn are the same than for the strategy with local optimizations (see Table 6.4). For each γ , a small n_P ($n_P = d$) is better (in the sense of the total number of evaluations n_{total}) than a high n_P . We also observe that high γ give better results than low γ . However regarding the number of evaluations n for $\gamma = 6$, we conclude that for this function, avoiding pairings with the highest α -ranks is an efficient strategy that leads to trees with low α -ranks.

According to the observations made in the two cases, for the next numerical examples we choose $\gamma = 4$ for the following numerical experimentations.

Remark 6.4. Choosing a high value for γ tends to a deterministic strategy where we permute two nodes whose parents have the highest α -ranks. It is limiting the probability to explore trees very different from the initial one. However for these two examples, it seems relevant.

6.6 Numerical examples

In this section, we compare the different strategies for tree optimization, presented in this chapter for two numerical examples : the sum of bivariate functions defined by (6.1) and the chain function defined by (6.6). Among the leaves-to-root strategies, **d-LO** refers to the one with deterministic local optimizations (from section 6.3.1), **s-LO** refers to the one with stochastic local optimizations

(from section 6.4), **bg-LO** refers to the strategy proposed in [10] where the optimization criterion is a rank ratio. The global stochastic optimization strategy presented in Section 6.5 is denoted **s-GO**.

The choice of the parameters γ_1 , γ_2 and n_P for the **s-LO** strategy and for the **s-GO** strategy are heuristic, they are made after the observations from Tables 6.3, 6.4, 6.5 and 6.6. They will directly be specified in the examples.

These strategies are also compared with a random tree **RT** (with arity two) and a random balanced tree referred to as **RBT**. In these two cases, the overall number of evaluations of the function is dedicated to the computation of the approximation, such that $n_{\text{optim}} = 0$ and $n = n_{\text{total}}$.

Sum of bivariate functions

	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{total}
	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$
d-LO	[-15.1; -15; -14.8]	[354; 376; 406]	[485; 512; 545]	[2233; 2361; 2401]
s-LO	[-14.9; -14.7; -14.0]	[340; 529; 1360]	[468; 689; 1552]	[1243; 1699; 2752]
bg-LO	[-15; -14.7; -14.5]	[354; 376; 445]	[485; 512; 574]	[3239; 3372; 3611]
RBT	[-14.4; -14.3; -13.9]	[696; 925; 2198]	[858; 1150; 2432]	[858; 1150; 2432]
s-GO	[-14.9; -14.6; -14.2]	[390; 456; 791]	[521; 595; 954]	[1790; 2221; 2908]
RT	[-14.6; -14.3; -14.1]	[971; 1763; 2471]	[1166; 1987; 2745]	[1166; 1987; 2745]

Table 6.7 – Sum of bivariate functions defined by (6.1) with $d = 8$. Approximation with a prescribed tolerance $\varepsilon = 10^{-14}$, $\varepsilon_c = 10^{-2}$. q_{10}, q_{50}, q_{90} are the 10^{th} , 50^{th} and 90^{th} quantiles for relative error $\log(\varepsilon(u^*))$, number of evaluations n , for the **s-LO** strategy $\gamma = 6$ and $n_P = 2d$, for the **s-GO** strategy $\gamma = 4$ and $n_P = d$.

Table 6.7 shows that all the optimization strategies decrease the number of evaluations n necessary to compute the approximation with precision ε compared to a random tree **RT** or a random balanced tree **RBT**. But only the **s-LO** strategy has a total number of evaluations lower in average than the cost of a random tree **RT**. This is due to the fact that the dimension $d = 8$ is small and the α -ranks (even chosen randomly) remain moderate.

When the dimension d increases, the Table 6.8 corresponds to $d = 16$, we observe that all optimization strategies decrease the number of evaluations n necessary to compute the approximation with precision ε compared to a random tree **RT** or a random balanced tree **RBT**. For all the optimization strategies, the 90^{th} quantile of the total number of evaluations n_{total} is lower than the cost of a random tree **RT** or a random balanced tree **RBT**. In this case, the **s-LO** strategy is the most efficient as it decreases the three quantiles compared to the random trees. When the

	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{total}
	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$
d-LO	[-14.5; -14.2; -13.9]	[699; 1390; 3289]	[964; 1759; 3782]	[8118; 9739; 12427]
s-LO	[-14.4; -14.1; -13.7]	[949; 2011; 4167]	[1259; 2394; 4717]	[3874; 4979; 7802]
bg-LO	[-14.7; -14.5; -14.2]	[692; 729; 882]	[956; 993; 1165]	[11336; 11869; 12773]
RBT	[-14.0; -13.8; -13.2]	[7476; 10888; 14837]	[8148; 11657; 15651]	[8148; 11657; 15651]
s-GO	[-14.2; -14.0; -13.6]	[1174; 1830; 7008]	[1500; 2243; 7749]	[5756; 7036; 13499]
RT	[-14.0; -13.7; -12.7]	[5033; 11320; 36456]	[5600; 12154; 37635]	[5600; 12154; 37635]

Table 6.8 – Sum of bivariate functions defined by (6.1) with $d = 16$. Approximation with a prescribed tolerance $\varepsilon = 10^{-14}$, $\varepsilon_c = 10^{-2}$. q_{10}, q_{50}, q_{90} are the $10^{th}, 50^{th}$ and 90^{th} quantiles for relative error $\log(\varepsilon(u^*))$, number of evaluations n , for the **s-LO** strategy $\gamma = 6$ and $n_P = 2d$, for the **s-GO** strategy $\gamma = 4$ and $n_P = d$.

dimension increases, the interest of using optimizations strategies increases too.

	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{total}
	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$
d-LO	[-14.1; -14; -13.8]	[3160; 4843; 6339]	[3826; 5670; 7284]	[20866; 23859; 26077]
s-LO	[-14.2; -13.9; -13.6]	[2395; 5773; 8183]	[3035; 6685; 9177]	[7225; 12185; 15842]
bg-LO	[-14.3; -13.6; -13]	[1133; 1848; 2701]	[1623; 2415; 3300]	[25134; 27491; 29429]
RBT	[-13.6; -13; -12.5]	[16001; 22079; 46982]	[17305; 23634; 48815]	[17305; 23634; 48815]
s-GO	[-13.9; -13.8; -13.4]	[7471; 10319; 17918]	[8495; 11532; 19264]	[13135; 16089; 24299]
RT	[-13.7; -12.9; -12.2]	[15100; 22745; 32182]	[16418; 24269; 33793]	[16418; 24269; 33793]

Table 6.9 – Sum of bivariate functions defined by (6.1) with $d = 24$. Approximation with a prescribed tolerance $\varepsilon = 10^{-13}$, $\varepsilon_c = 10^{-2}$. q_{10}, q_{50}, q_{90} are the $10^{th}, 50^{th}$ and 90^{th} quantiles for relative error $\log(\varepsilon(u^*))$, number of evaluations n , for the **s-LO** strategy $\gamma = 6$ and $n_P = 2d$, for the **s-GO** strategy $\gamma = 4$ and $n_P = d$.

In Table 6.9, we see that choosing $d = 24$ highlights even more the advantages of performing tree optimization. The number of evaluations n necessary to compute the approximation is strongly reduced. In this case, for all optimization methods the 90^{th} quantile of the number of total evaluations is really lower than for both a random tree **RT** or a random balanced tree **RBT**. The two deterministic strategies (and in particular the **bg-LO** method) recover the best trees. However, the additional cost used to evaluate the α -ranks for the optimization which appears in n_{total} is not competitive for the 10^{th} and 50^{th} quantiles. The local stochastic strategy performs particularly well as the 90^{th} quantile of the number of evaluations is lower than the 10^{th} quantile of the number of evaluations necessary for a random tree **RT** and **RBT**.

Chain function.

	$\log(\varepsilon(u^*))$	\mathcal{S}	n	n_{total}
	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$	$[q_{10}; q_{50}; q_{90}]$
d-LO	[-14.1; -13.8; -13.6]	[8584; 16699; 19669]	[9409; 17654; 20698]	[25369; 33709; 36678]
s-LO	[-14.3; -13.9; -13.7]	[7733; 10812; 12189]	[8376; 11574; 12977]	[12211; 15487; 16987]
bg-LO	[-14.3; -14.1; -14]	[2204; 8267; 13762]	[2665; 9076; 14755]	[26450; 34031; 39490]
RBT	[-14; -13.8; -13.4]	[10647; 12420; 19681]	[11392; 13235; 20673]	[11392; 13235; 20673]
s-GO	[-14.1; -13.8; -13.5]	[7777; 11327; 14003.5]	[8532; 12098; 15008]	[11787; 15328; 18309]
RT	[-14.2; -13.8; -13.3]	[9063; 12262; 23263]	[9817; 13164; 24406]	[9817; 13164; 24406]

Table 6.10 – Chain function defined by (6.6) with $d = 19$. Approximation with a prescribed tolerance $\varepsilon = 10^{-14}$, $\varepsilon_c = 10^{-2}$. q_{10}, q_{50}, q_{90} are the $10^{th}, 50^{th}$ and 90^{th} quantiles for relative error $\log(\varepsilon(u^*))$, number of evaluations n , for the **s-LO** strategy $\gamma = 6$ and $n_P = 2d$, for the **s-GO** strategy $\gamma = 4$ and $n_P = 3d$.

Table 6.10 shows that all the optimization strategies decrease the number of evaluations n necessary to compute the approximation with precision ε compared to a random tree **RT** or a random balanced tree **RBT**. But the total number of evaluations for the deterministic optimization strategies is higher than the cost of a random tree. However with the stochastic strategies, we have a gain. This is due to the fact that whatever the tree is, the α -ranks (even chosen randomly) remain moderates, such the additional cost due to the α -ranks estimations may not be useful.

Overall conclusion for the examples

The numerical examples have shown that the interest of using optimization for the determination of the dimension tree T increases when the dimension d of the problem increases. When d is small the α -ranks may remain moderate and the additional evaluations for the α -ranks estimations are not useful.

With the choices we made for the parameters $(n_P, \gamma_1, \gamma_2)$ of the stochastic strategies, we observe that the deterministic strategies (**d-LO** and **bg-LO**) explore more possibilities for the dimension trees and therefore find the best trees (in the sense that the number of evaluations (n) is smaller). However the number of evaluations necessary to estimate the α -ranks (n_{optim}) is high. These choices for the parameters $(n_P, \gamma_1, \gamma_2)$ of the stochastic strategies is a good trade-off to both have a reasonable number of evaluations to compute the approximation n and moderate number of evaluations for the α -ranks estimations n_{optim} . Therefore they lead to a total number of evaluations n_{total} more competitive than computing directly the approximation on a random tree.

6.7 Conclusions

In this chapter, we proposed several optimization strategies for choosing a dimension partition tree T , which is adapted to the function we want to approximate, in the sense that the α -ranks of the approximation are as low as possible to get a certain accuracy. Deterministic strategies explore a large number of trees and are able to recover really good trees to reach low α -ranks. However this exploration is often too expensive compared to the number of evaluations necessary for the approximation of the function. In the presented cases, using these strategies leads to an overall number of samples which is better than the number of samples necessary for random trees with high α -ranks. However this overall number of samples is not competitive compared to the number of samples necessary to compute the approximation of a random tree in expectation.

Stochastic strategies (with a few exploration steps) are more competitive regarding the number of evaluations for the estimations of the α -ranks. However these strategies involves several numerical (and heuristics) parameters, which need to be tuned. Furthermore if the choices made for these examples are working relatively well we do not claim that this will be efficient for any function.

CONCLUSION

Conclusions of this thesis and future works

In this thesis, we have first proposed a new projection method onto a linear space called the boosted optimal weighted least-squares method, presented in Chapter 2. The proposed method constructs a weighted least-squares projection associated with random points sampled from a suitably chosen distribution. We obtained quasi-optimality properties (in expectation) for the weighted least-squares projection, with or without reducing the size of the sample by a greedy removal of points. The constant in the quasi-optimality property depends on the number of points selected by the greedy algorithm. The more points removed, the larger the error bound will be. Therefore, if the goal is an accurate control of the error, few points should be removed. On the contrary, if the goal is to reduce the cost while allowing a larger bound of the error, the maximum number of points may be removed from the sample, which in some cases leads to a number of samples equal to the dimension of the approximation space. As the size of the sample obtained after the greedy algorithm may differ from one run to another, it would be interesting to study the convergence properties of this greedy algorithm. Also the selection of the points to remove is not an optimization over the whole sample but a point-by-point greedy selection, therefore the resulting sample may not be the best over all possible combinations. It would be interesting to look for an optimal and efficient selection of the sub-sample with regard to the stability criterion (maybe using stochastic optimization for this combinatorial problem). The constant obtained for the quasi-optimality property is rather pessimistic regarding the numerical experiments. This suggests that our error bounds could be further improved.

For adaptive approximation a classical approach is to consider approximations in a sequence of nested approximation spaces. In Chapter 3, we proposed a method to construct a sequence of boosted least-squares projections associated to a nested sequence of approximation spaces, which reuses the samples from one approximation space to another. Even if the numerical results are promising, we still did not manage to obtain theoretical guarantees for the algorithm that constructs this sequence of boosted optimal weighted least-squares projections. In practice, the numerical examples show that the number of samples necessary to construct the adaptive boosted least-squares projection is close to the dimension of the approximation space (only observed).

The model class of tree-based tensor formats (presented in Chapter 4) is a prominent tool for

the approximation of high-dimensional functions. In this thesis, one objective was to develop algorithms that provide a controlled approximation while using as few evaluations as possible, (i.e close to the complexity of the approximation tool). In Chapter 5, we proposed an algorithm that constructs an approximation in tree-based tensor format which provides a controlled approximation for a sufficiently high number of evaluations of the function u . Using a leaves-to-root approach the algorithm constructs low-dimensional subspaces in which the function is projected, thanks to boosted optimal least-squares projections combined to principal component analyses. An important question that is not fully answered yet is to determine the number of samples necessary for the algorithm to construct an approximation with controlled precision (under some assumptions of the function class and the choice of the approximation tool). In practice, estimations of the low-dimensional subspaces are computed. As underlined in Chapter 5, the control of these estimations of the principal components is not guaranteed yet. First the assumptions that have to be made on the oblique projection of u are difficult to validate on concrete examples. Secondly, the condition on the number of samples necessary to reach a given error seems to be overestimated according to the numerical examples.

Chapter 6 was devoted to the presentation of adaptive strategies for the tree selection. We showed on examples that when the dimension d is high, it is worth spending some function's evaluations to estimate ranks and perform tree optimization. To this end, we propose deterministic strategies that explore a large number of trees and also stochastic strategies that explore a reduced number of trees, favouring the trees with low ranks. Deterministic strategies are only competitive to avoid very bad trees, while stochastic strategies seem to be a good trade-off between the cost devoted to the estimation of ranks and the number of function's evaluations needed to compute the approximation. However, these stochastic strategies involve several parameters which are chosen heuristically and we observe that parameters which limit the exploration (by driving the algorithm directly to trees without high ranks) are giving better results. It would be interesting to develop strategies that could tune these parameters along the exploration.

Further outlooks

Longer-term general prospects can also be considered. Solving bayesian inverse problems with a functional approach requires to approximate the posterior probability density function. One may think of approximating multivariate probability distributions in tree-based tensor formats. This would allow to easily extract information from the posterior (mean, moments) and also to sample from it using the technique of Appendix B.

This raises the question of imposing some constraints on the approximation (positivity integral, equal to one, monotonicity, ...).

The strategies for tree optimization proposed in this thesis are using only function's evaluations. In some situations we may have further information on the function (Sobol indices, equations satisfied by the function,...) that would lead to different optimization strategies using this knowledge.

This thesis focused on the construction of surrogate models which is an important step to solve Uncertainty Quantification problems. To solve more specific problems as model calibration, estimation of rare event's probability, sensitivity analysis, it would be interesting to combine the approaches of this thesis with state-of-the art Uncertainty Quantification algorithms (for example sequential approaches for rare event simulations,...).

In some situations, engineers need to add variables to the model during the study. A question immediately raises: how can we adapt our strategies in order to deal with the more complicated model while trying to reuse the function's evaluations already made.

Also in practice, we may encounter vector-valued outputs or inputs that are correlated, such situations would require new analyses.

BIBLIOGRAPHY

- [1] G. Migliorati F. Nobile R. Tempone A. Chkifa, A. Cohen. Discrete least-squares polynomial approximation with random evaluations - application to parametric and stochastic elliptic pdes. *ESAIM: Mathematical Modelling and Numerical Analysis*, 49:815–837, 2015.
- [2] M. Ali and A. Nouy. Approximation with tensor networks. Part i: Approximation spaces. *arXiv:2007.00118*, 2020.
- [3] M. Ali and A. Nouy. Approximation with tensor networks. Part ii: Approximation rates for smoothness classes. *arXiv:2007.00128*, 2020.
- [4] B. Arras, M. Bachmayr, and A. Cohen. Sequential sampling for optimal weighted least squares approximations in hierarchical spaces. *SIAM Journal on Mathematics of Data Science*, 1:189–207, 2019.
- [5] M. Bachmayr, A. Cohen, R. DeVore, and G. Migliorati. Sparse polynomial approximation of parametric elliptic pdes. Part ii: lognormal coefficients. *ESAIM Math. Model. Numer. Anal.*, 51(1):341–363, 2017.
- [6] M. Bachmayr, A. Cohen, and G. Migliorati. Sparse polynomial approximation of parametric elliptic pdes. Part i: affine coefficients. *ESAIM Math. Model. Numer. Anal.*, 51(1):321–339, 2017.
- [7] M. Bachmayr, A. Nouy, and R. Schneider. Approximation power of tree tensor networks for compositional functions. *In preparation*, 2020.
- [8] M. Bachmayr, R. Schneider, and A. Uschmajew. Tensor networks and hierarchical tensors for the solution of high-dimensional partial differential equations. *Foundations of Computational Mathematics*, 16:1423–1472, 2016.
- [9] J. Baglama, D. Calvetti, and L. Reichel. Fast Leja points. *Electronic Transactions on Numerical Analysis*, 7:124–140, 1998.
- [10] J. Ballani and L. Grasedyck. Tree adaptive approximation in the hierarchical tensor format. *SIAM J. Sci. Comput.*, 36(4):A1415–A1431, 2014.
- [11] M. Bebendorf. Approximation of boundary element matrices. *Numerische Mathematik*, 86(5):565–589, 2000.

-
- [12] R. Bellmann. *Adaptive Control Processes : A Guided Tour*. 1961.
- [13] J. Benasseni. Lower bounds for the largest eigenvalue of a symmetric matrix under perturbations of rank one. *Linear and Multi-linear Algebra*, 2011.
- [14] L. Bos and N. Levenberg. On the calculation of approximate fekete points: the univariate case. *Electronic Transactions on Numerical Analysis. Volume*, 30:377–397, 2008.
- [15] L. Bos, S. De Marchi, A. Sommariva, and M. Vianello. Computing Multivariate Fekete and Leja Points by Numerical Linear Algebra. *SIAM Journal on Numerical Analysis*, 48(5):1984–1999, 2010.
- [16] L. Bos, S. De Marchi, A. Sommariva, and M. Vianello. Adaptive Leja Sparse Grid Constructions for Stochastic Collocation and High-Dimensional Approximation. *SIAM Journal on Scientific Computing*, 36(6):A2952–A2983, 2014.
- [17] J.R. Bunch, C.P. Nielsen, and D.C. Sorensen. Rank-one modification of the symmetric eigenproblem. *Numer. Math.*, 31:31–48, 1978.
- [18] A. Chkifa, A. Cohen, and R. DeVore. Sparse adaptive taylor approximation algorithms for parametric and stochastic elliptic pdes. *ESAIM: Mathematical Modelling and Numerical Analysis*, 47(1):253–280, 2013.
- [19] A. Chkifa, A. Cohen, and C. Schwab. High-dimensional adaptive sparse polynomial interpolation and applications to parametric pdes. *Found. Comput. Math.*, 14:601–633, 2014.
- [20] A. Cohen. *Numerical Analysis of Wavelet Methods*, volume 32. 2003.
- [21] A. Cohen and A. Chkifa. *On the Stability of Polynomial Interpolation Using Hierarchical Sampling*, pages 437–458. 2015.
- [22] A. Cohen, A. Davenport, and D. Leviatan. On the stability and accuracy of least squares approximation. *Foundations of Computational Mathematics*, 13(3):819–834, 2013.
- [23] A. Cohen and R. DeVore. Approximation of high-dimensional pdes. *Acta Numerica*, 24:1–159, 2015.
- [24] A. Cohen and G. Migliorati. Optimal weighted least-squares methods. *SMAI Journal of Computational Mathematics*, 86(3):181–203, 2017.
- [25] A. Cohen, A. Nouy, G Keryacharian, and D. Picard. Accuracy of principal component spaces. *Running notes*, 2020.
- [26] N. Cohen, O. Sharir, and A. Shashua. On the expressive power of deep learning: A tensor analysis. *JMLR: Workshop and Conference Proceedings*, 49:1–31, 2016.

-
- [27] W. Dahmen. Wavelet and multiscale methods for operator equations. *Acta Numerica*, 6, 2000.
- [28] P.J. Davis. *Interpolation and approximation*. 1975.
- [29] R. DeVore. Nonlinear approximation. *Acta Numerica.*, pages 51–150, 1998.
- [30] R. DeVore, G. Kerkycharian, D. Picard, and V. Temlyakov. Approximation methods for supervised learning. *Foundations of Computational Mathematics*, 6:3–58, 2006.
- [31] L. Devroye. Non-uniform random variate generation. *Springer*, 1985.
- [32] A. Doostan and J. Hampton. Coherence motivated sampling and convergence analysis of least squares polynomial chaos regression. *Computer Methods in Applied Mechanics and Engineering*, 290(3):73–97, 2015.
- [33] P. Erdos, A. Kroo, and J. Szabados. On convergent interpolatory polynomials. *J. Approx. Theory*, 58:232–241, 1989.
- [34] O. Ernst, F. Nobile, C. Schillings, and T. Sullivan. Uncertainty quantification. *Oberwolfach Reports*, 16(1):695–772, 2020.
- [35] A. Falco and W. Hackbusch. On minimal subspaces in tensor representations. *Found. Comput. Math.*, 12:765–803, 2012.
- [36] A. Falco, W. Hackbusch, and A. Nouy. Tree-based tensor formats. *arXiv:1810.01262*, 2018.
- [37] A. Falco, W. Hackbusch, and A. Nouy. On the Dirac-Frenkel variational principle on tensor Banach spaces. *Foundations of computational mathematics*, 19(1):159–204, 2019.
- [38] V. Fedorov. *Theory of Optimal Experiments Designs*. 1972.
- [39] L. Fejér. Bestimmung derjenigen abszissen eines intervalles, für welche die quadratsumme der grundfunktionen der lagrangeschen interpolation im intervalle ein möglichst kleines maximum besitzt. *Annali della Scuola Normale Superiore di Pisa - Classe di Scienze*, Ser. 2, 1(3):263–276, 1932.
- [40] M. Fekete. Über die verteilung der Wurzeln bei gewissen algebraischen Gleichungen mit ganzzahligen Koeffizienten. *Mathematische Zeitschrift*, 1:377–402, 1923.
- [41] J.H. Friedman and W. Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76:817–823, 2009.
- [42] J.H. Friedman and J.W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, 23(9):881–890, 1974.

-
- [43] T. Gerstner and M. Griebel. Dimension-adaptive tensor-product quadrature. *Computing*, 71:65–87, 2003.
- [44] G.H. Golub. Some modified matrix eigenvalue problems. *SIAM Review*, 15(2):318–334, 1973.
- [45] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.
- [46] L. Grasedyck. Hierarchical singular value decomposition of tensors. *SIAM J. Matrix Analysis Applications*, 31:2029–2054, 2010.
- [47] L. Grasedyck, D. Kressner, and C. Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM-Mitteilungen*, 36(1):53–78, 2013.
- [48] E. Grelier, A. Nouy, and M. Chevreuil. Learning with tree-based tensor formats. *arXiv:1811.04455*, 2018.
- [49] E. Grelier, A. Nouy, and R. Lebrun. Learning high-dimensional probability distributions using tree tensor networks. 2019.
- [50] W.H. Greub. *Linear algebra*, volume 4th edition. New York, Springer-Verlag, 1981.
- [51] L. Guo, A. Narayan, L. Yan, and T. Zhou. Weighted Approximate Fekete Points: Sampling for Least-Squares Polynomial Approximation. *SIAM Journal on Scientific Computing*, 40(1):A366–A387, 2018.
- [52] C. Haberstick, A. Nouy, and G. Perrin. Boosted optimal weighted least-squares methods. *arXiv:1912.07075*, 2020.
- [53] W. Hackbusch. *Hierarchical Tensor Representation*. 2015.
- [54] W. Hackbusch and S. Kuhn. A new scheme for the tensor representation. *Journal of Fourier analysis and applications*, 15(5):706–722, 2009.
- [55] M. Hashemizadeh, J. Miller, M. Liu, and G. Rabusseau. Adaptive tensor learning with tensor networks. *arXiv:2008.05437*, 2020.
- [56] C.J. Hillar and L-K. Lim. Most tensors are np-hard. *Journal of the ACM*, 45, 2013.
- [57] B. Ibrahimoglu. Lebesgue functions and lebesgue constants in polynomial interpolation. *Journal of Inequalities and Applications*, 2016, 2016.
- [58] I.C.F. Ipsen and B. Nadler. Refined perturbation bounds for eigenvalues of hermitian and non-hermitian matrices. *SIAM J. Matrix Anal. Appl.*, 31(1):40–53, 2009.

-
- [59] B. Khoromskij. Tensors-structured numerical methods in scientific computing: Survey on recent advances. *Chemometrics and Intelligent Laboratory Systems*, 110:1–19, 2012.
- [60] T. Kolda and B. Bader. Tensor decompositions and applications. *SIAM Review*, 51:455–500, 2009.
- [61] A. Kolmogoroff. Über die beste annäherung von funktionen einer gegebenen funktionenklasse. *Ann. of Math.*, 37(2):389–434, 1936.
- [62] D. Kressner, M. Steinlechner, and A. Uschmajew. Low-rank tensor methods with subspace correction for symmetric eigenvalue problems. *SIAM J. Sci. Comput.*, 36(5):A2346–A2368, 2014.
- [63] Y. Liu and G. Xu. Widths and average widths of Sobolev classes. *Acta Math. Sci. Ser. B Engl. Ed.* 23, 12(5):178–184, 2003.
- [64] T.H. Luu, Y. Maday, M. Guillo, and P. Guérin. A new method for reconstruction of cross-sections using Tucker decomposition. working paper or preprint, 2017.
- [65] Y. Maday, N.C. Nguyen, A. Patera, and G. Pau. A general multipurpose interpolation procedure: the magic points. *Communications on Pure and Applied Analysis*, 8(1):383–404, 2009.
- [66] S. De Marchi. On leja sequences: some results and applications. *Applied Mathematics and Computation*, 152(1):621–647, 2004.
- [67] G. Migliorati. Adaptive polynomial approximation by means of random discrete least squares. *Lecture Notes in Computational Science and Engineering*, 103:547–554, 2015.
- [68] G. Migliorati. Adaptive approximation by optimal weighted least-squares methods. *SIAM J. Numer. Anal.*, 57(5):2217–2245, 2019.
- [69] G. Migliorati and F. Nobile. Analysis of discrete least squares on multivariate polynomial spaces with evaluations at low-discrepancy point sets. *Journal of Complexity*, 31(4):517 – 542, 2015.
- [70] G. Migliorati, F. Nobile, E. Schwerin, and R. Tempone. Analysis of discrete l2 projection on polynomial spaces with random evaluations. *Found. Comput. Math.*, 14(3):419–456, 2014.
- [71] G. Migliorati, F. Nobile, and R. Tempone. Convergence estimates in probability and in expectation for discrete least squares with noisy evaluations at random points. *Journal of Multivariate Analysis*, 142:167–182, 2015.
- [72] C. Milbradt and M. Wahl. High-probability bounds for the reconstruction error of pca. *Statistics and Probability Letters*, 161, 2020.

-
- [73] A. Narayan, J. Jakeman, and T. Zhou. A christoffel function weighted least squares algorithm for collocation approximations. *Mathematics of Computation*, 86(5), 2017.
- [74] F. Nobile, R. Tempone, and C.G. Webster. An anisotropic sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2411–2442, 2008.
- [75] F. Nobile, R. Tempone, and C.G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345, 2008.
- [76] A. Nouy. *Low-rank rethods for high-dimensional approximation and model order reduction*. In P. Benner and A. Cohen and M. Ohlberger and K. Willcox (eds.), *Model Reduction and Approximation: Theory and Algorithms*. SIAM, Philadelphia, PA, 2017.
- [77] A. Nouy. *Low-Rank Tensor Methods for Model Order Reduction*, pages 857–882. Springer International Publishing, Cham, 2017.
- [78] A. Nouy. Higher-order principal component analysis for the approximation of tensors in tree-based low rank formats. *Numerische Mathematik*, 141:743–789, 2019.
- [79] I. Oseledets and E. Tyrtyshnikov. Breaking the curse of dimensionality, or how to use SVD in many dimensions. *SIAM Journal on Scientific Computing*, 31(5):3744–3759, 2009.
- [80] I. Oseledets and E. Tyrtyshnikov. Tt-cross approximation for multidimensional arrays. *Linear Algebra and its Applications*, 432:70–88, 2010.
- [81] N.M. Radford. Slice sampling. *Ann. Stat.*, 31(3):705–767, 2003.
- [82] C.E Rasmussen and C.K.I Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.
- [83] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 1958.
- [84] R. Schneider and A. Uschmajew. Approximation rates for the hierarchical tensor format in periodic Sobolev spaces. *Journal of Complexity*, 30:56–71, 2014.
- [85] V. Silva and L-K. Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM Journal on Matrix Analysis and Applications*, 30, 2006.
- [86] A. Sommariva and M. Vianello. Computing approximate Fekete points by QR factorizations of Vandermonde matrices. *Computers & Mathematics with Applications*, 57(8):1324–1336, 2009.

-
- [87] T.J. Sullivan. *Introduction to Uncertainty Quantification*, volume 63. 2015.
- [88] G. Szego. *Orthogonal Polynomials*. Number vol. 23 in American Math. Soc: Colloquium publ. American Mathematical Society, 1939.
- [89] V.N. Temlyakov. Approximation of functions with bounded mixed derivative. *Trudy Matematicheskogo Instituta im. V. A. Steklova*, 178(5), 1986.
- [90] V.N. Temlyakov. Approximation of periodic functions with bounded mixed derivative. *Nova Science Publishers Inc. Commack NY*, 1993.
- [91] A.J. Tropp. User-friendly tail bounds for sums of random matrices. *Found. Comput. Math.*, 12(5):389–434, 2012.
- [92] O. Mula Y. Maday and G. Turinici. Convergence analysis of the generalized empirical interpolation method. *SIAM J. Numer. Anal.*, 2016.
- [93] T. Zhou, A. Narayan, and Z. Xu. Multivariate discrete least-squares approximations with a new type of collocation grid. *SIAM Journal on Scientific Computing*, 36(5):A2401–A2422, 2014.

APPROXIMATE FAST GREEDY ALGORITHM

In this paragraph, we present a computational strategy for the approximate fast greedy algorithm. As mentioned in Remark 2.5 from Chapter 2, the greedy subsampling is computationally costly. More precisely, for each removal of a point, the greedy subsampling requires as many computations of the norm $\|\mathbf{G}_{\mathbf{x}_{K \setminus \{k\}}^n} - \mathbf{I}\|_2$ as there are points in the sample from which we want to remove a point. Therefore we propose an approximate but faster greedy algorithm.

A.1 Computational strategy for the approximate fast greedy algorithm

For any k , we have

$$\mathbf{G}_{\mathbf{x}_{K \setminus \{k\}}^n} = \frac{\#K}{\#K-1} \mathbf{G}_{\mathbf{x}_K^n} - \frac{1}{\#K-1} w(x^k) \varphi(x^k) \otimes \varphi(x^k) \quad (\text{A.1})$$

and

$$\mathbf{G}_{\mathbf{x}_{K \setminus \{k\}}^n} - \mathbf{I} = \frac{\#K}{\#K-1} (\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}) - \frac{1}{\#K-1} w(x^k) \varphi(x^k) \otimes \varphi(x^k) + \frac{1}{\#K-1} \mathbf{I}. \quad (\text{A.2})$$

Denoting $\mathbf{B} = \frac{\#K}{\#K-1} (\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}) - \frac{1}{\#K-1} w(x^k) \varphi(x^k) \otimes \varphi(x^k)$, it holds

$$\|\mathbf{G}_{\mathbf{x}_{K \setminus \{k\}}^n} - \mathbf{I}\|_2 = \max(\lambda_1(\mathbf{B}) + \frac{1}{\#K-1}, -\lambda_m(\mathbf{B}) + \frac{1}{\#K-1}).$$

As the matrix \mathbf{B} is a rank-one update of the symmetric matrix $\frac{\#K}{\#K-1} (\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I})$, from [44] and [17], we can state that

$$\lambda_1(\mathbf{B}) \leq \frac{\#K}{\#K-1} \lambda_1(\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}), \quad (\text{A.3})$$

where $\lambda_1(\mathbf{B})$ and $\lambda_1(\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I})$ are respectively the maximal eigenvalues of \mathbf{B} and $\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}$.

Bounds on highest and lowest eigenvalues are obtained in [58] or [13]. Here we consider the lower

bound from [13] for the highest eigenvalue

$$\lambda_1(\mathbf{B}) \geq \frac{\#K}{\#K-1} \lambda_1(\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}) - \tilde{\rho}(\mathbf{q}_1^T \boldsymbol{\varphi}(x^k)),$$

where $\mathbf{q}_1 = \mathbf{Q}(:, 1)$ is the eigenvector of \mathbf{B} associated to its highest eigenvalue $\lambda_1(\mathbf{B})$, and $\tilde{\rho} = \frac{m}{\#K-1}$. To bound the quantity $-\lambda_m(\mathbf{B}) + \frac{1}{\#K-1}$, we use the fact that

$$\lambda_1(-\mathbf{B}) = -\lambda_m(\mathbf{B})$$

and the bound on $\lambda_1(-\mathbf{B})$

$$\lambda_1(-\mathbf{B}) \geq \frac{\#K}{\#K-1} \lambda_1(-(\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I})) + \tilde{\rho}(\mathbf{q}_1^T \boldsymbol{\varphi}(x^k)),$$

such that

$$\lambda_m(-\mathbf{B}) \geq -\frac{\#K}{\#K-1} \lambda_m((\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I})) + \tilde{\rho}(\mathbf{q}_1^T \boldsymbol{\varphi}(x^k)).$$

Instead of calculating $\|\mathbf{G}_{\mathbf{x}_{K \setminus \{k\}}^n} - \mathbf{I}\|_2$ for each k , we evaluate $\tilde{\rho}(\mathbf{q}_1^T \boldsymbol{\varphi}(x^k))$, this operation only involves matrix multiplications, and the knowledge of the spectrum of $\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}$, and we look for k_1 which minimizes $\frac{\#K}{\#K-1} \lambda_1((\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I})) - \tilde{\rho}(\mathbf{q}_1^T \boldsymbol{\varphi}(x^k))$ and k_2 which minimizes $-\frac{\#K}{\#K-1} \lambda_m((\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I})) + \tilde{\rho}(\mathbf{q}_1^T \boldsymbol{\varphi}(x^k))$. Then we choose

$$k^* \in \arg \min_{k \in \{k_1, k_2\}} \|\mathbf{G}_{\mathbf{x}_{K \setminus \{k\}}^n} - \mathbf{I}\|_2.$$

We may not find the minimizer over K , but in practice, we observe that this approach performs almost as well as the exact greedy and it is faster.

A.2 Complexity analysis

We compare these two subsampling methods in terms of computational cost. For the exact greedy subsampling, each time we remove a point from a l -sample, it requires l calculations of $\|\mathbf{G}_{\mathbf{x}^l} - \mathbf{I}\|_2$, which takes $\mathcal{O}(m^3)$ floating point operations. Furthermore, the computation of $\mathbf{G}_{\mathbf{x}^l}$ requires $\mathcal{O}(m^2 l^2)$ floating point operations. We start from $l = n$ and let the greedy subsampling runs until the stability condition is no longer verified to a sample of size k , summing over all withdrawn points it comes

$$\sum_{l=k+1}^n m^3 l + m^2 l^2 = \frac{m^3}{2} (n(n+1) - k(k+1)) + \frac{m^2}{6} (n(n+1)(2n+1) - k(k+1)(2k+1)).$$

m	Exact Greedy Subsampling				Fast Greedy Subsampling			
	$\ \mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}\ _2$	$\#K$	$\log(\varepsilon(u^*))$	t (sec)	$\ \mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}\ _2$	$\#K$	$\log(\varepsilon(u^*))$	t (sec)
6	[0.34; 0.68]	[6; 6]	[-0.9; -0.7]	[0.29; 0.43]	[0.18; 0.71]	[6; 6]	[-1; -0.6]	[0.03; 0.09]
11	[0.42; 0.76]	[11; 11]	[-2; -1.7]	[1.43; 1.48]	[0.41; 0.67]	[11; 12]	[-2.1; -1.9]	[0.16; 0.22]
16	[0.47; 0.87]	[16; 16]	[-2.8; -2.4]	[13; 15]	[0.60; 0.88]	[16; 17]	[-2.8; -2.3]	[0.52; 0.61]
21	[0.63; 0.87]	[21; 21]	[-4; -3.7]	[52; 54]	[0.74; 0.88]	[21; 23]	[-4; -3.6]	[1.6; 1.9]
26	[0.67; 0.88]	[26; 26]	[-4.6; -4.4]	[150; 204]	[0.64; 0.89]	[26; 29]	[-4.8; -4.4]	[3.5; 5.3]
31	[0.79; 0.89]	[31; 31]	[-5.7; -5.5]	[429; 464]	[0.73; 0.89]	[31; 34]	[-5.9; -5.5]	[5.8; 8.6]
36	[0.68; 0.87]	[36; 36]	[-6.5; -6.3]	[608; 1008]	[0.70; 0.87]	[36; 40]	[-6.7; -6.4]	[8.6; 12]
41	[0.77; 0.88]	[41; 42]	[-7.6; -7.4]	[948; 999]	[0.74; 0.89]	[42; 48]	[-7.8; -7.4]	[11; 13]

Table A.1 – Comparison of the performance between exact and fast greedy approaches, using $\delta = 0.9$ and $\eta = 0.01$ both for different m . Number of samples $\#K$, stability constant $\|\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}\|_2$ and CPU time t .

Assuming $k = cm$ with $c \geq 1$, the overall cost scales in

$$\mathcal{C}_E = \mathcal{O}(m^2 n^3).$$

The more points are withdrawn (c smaller), the sharper this bound is.

For the fast greedy subsampling method, each time we remove a point from a l -sample it requires one singular value decomposition of $\mathbf{G}_{\mathbf{x}^l} - \mathbf{I}$, which takes $\mathcal{O}(m^3)$ floating point operations. It also requires $\mathcal{O}(m^2 l)$ for the computation of $\mathbf{G}_{\mathbf{x}^l}$. Using the same assumptions than before, $k = cm$ with $c \geq 1$, the overall cost scales in

$$\mathcal{C}_F = \mathcal{O}(m^2 n^2).$$

The more points are withdrawn (c smaller), the sharper this bound is.

In regards to the assumptions made in the Theorem 2.5, we can say $n = \mathcal{O}(m \log(m))$ and therefore,

$$\mathcal{C}_E = \mathcal{O}(m^5 \log(m)^3) \text{ and } \mathcal{C}_F = \mathcal{O}(m^4 \log(m)^2).$$

A.3 Illustration

In the next table, we present the CPU computational times for the subsampling part, when using the technique presented in this section compared to an exact greedy approach, we also illustrate that its accuracy is the same by considering the example 2, with $\mathcal{X} = [-1, 1]$ equipped with the uniform measure and the function

$$u(x) = \frac{1}{1 + 5x^2}. \quad (\text{A.4})$$

The approximation space is $V_m = \mathbb{P}_{m-1} = \text{span}\{\varphi_i : 1 \leq i \leq m\}$, where the basis $\{\varphi_i\}_{i=1}^m$ is chosen as the Legendre polynomials of degree less than $m - 1$.

m	Exact Greedy Subsampling			Fast Greedy Subsampling		
	$\ \mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}\ $	$\#K$	t (sec)	$\ \mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}\ $	$\#K$	t (sec)
6	[0.32; 0.63]	[6; 6]	[0.28; 0.37]	[0.21; 0.85]	[6; 6]	[0.03; 0.07]
11	[0.48; 0.82]	[11; 11]	[1.45; 1.48]	[0.50; 0.77]	[11; 11]	[0.175; 0.22]
16	[0.61; 0.85]	[16; 16]	[13.9; 14.8]	[0.56; 0.87]	[16; 17]	[0.53; 0.6]
21	[0.65; 0.84]	[21; 21]	[60; 68]	[0.62; 0.86]	[21; 23]	[1.79; 2.17]
26	[0.72; 0.87]	[26; 26]	[173; 186]	[0.68; 0.88]	[26; 28]	[3.5; 4.2]
31	[0.79; 0.87]	[31; 32]	[354; 377]	[0.66; 0.87]	[32; 35]	[5.2; 6.5]
36	[0.74; 0.87]	[36; 37]	[584; 636]	[0.64; 0.88]	[36; 42]	[8.2; 8.8]
41	[0.73; 0.87]	[41; 42]	[993; 1041]	[0.74; 0.88]	[41; 44]	[11.9; 12.2]

Table A.2 – Comparison of the performance between exact and fast greedy approaches, using $\delta = 0.9$ and $\eta = 0.01$ both for different m . Number of samples $\#K$, stability constant $\|\mathbf{G}_{\mathbf{x}_K^n} - \mathbf{I}\|_2$ and CPU time t .

Indeed, looking at the CPU times in Table A.1 and Table A.2, we observe that with the fast methods, the computational times are divided by about $m \log(m)$.

SAMPLING OF MULTIVARIATE PROBABILITY DISTRIBUTION IN TREE-BASED TENSOR FORMATS

B.1 Probability distributions in tree-based format

Let \mathcal{X} be a subset of \mathbb{R}^d with a product structure $\mathcal{X} = \mathcal{X}_1 \times \dots \times \mathcal{X}_d$ and $\mu = \mu_1 \otimes \dots \otimes \mu_d$ be a product measure on \mathcal{X} . For each $\nu \in \{1, \dots, d\}$, let \mathcal{H}_ν be a space of univariate functions defined on \mathcal{X}_ν , we introduce an orthonormal basis of \mathcal{H}_ν , denoted $\{\varphi_{k_\nu}^\nu : k_\nu \in \Lambda^\nu\}$ and $m_\nu = \dim(\mathcal{H}_\nu) = \#\Lambda^\nu$. Let $\mathcal{H} = \mathcal{H}_1 \otimes \dots \otimes \mathcal{H}_d$, for a multi-index $k = (k_1, \dots, k_d) \in \Lambda^1 \times \dots \times \Lambda^d = \Lambda$, we let $\varphi_k = \varphi_{k_1}^1(x_1) \dots \varphi_{k_d}^d(x_d)$. Then the set of functions $\{\varphi_k : k \in \Lambda\}$ is a basis of \mathcal{H} and any function $u \in \mathcal{H}$ can be written,

$$u(x) = \sum_{k_1 \in \Lambda^1} \dots \sum_{k_d \in \Lambda^d} u_{k_1, \dots, k_d} \varphi_{k_1}^1(x_1) \dots \varphi_{k_d}^d(x_d) = \sum_{k \in \Lambda} u_k \varphi_k(x).$$

We consider ρ a probability distribution of a random vector $\mathbf{X} = (X_1, \dots, X_d)$, defined by its Radon-Nikodym derivative v with respect to the measure μ which is here a product measure, $\mu = \mu_1 \otimes \dots \otimes \mu_d$, i.e.

$$d\rho(x) = v(x)d\mu(x), \tag{B.1}$$

where v is in tree-based tensor format $\mathcal{T}_r^T(\mathcal{H})$. We assume for each $\nu \in \{1, \dots, d\}$, $d\mu_\nu$ is of the form $d\mu_\nu(x) = \mu_\nu(x)dx$, meaning that $d\mu_\nu$ is particularly continuous with respect to the Lebesgue measure and thus $d\mu(x) = d\mu_1(x_1) \times \dots \times d\mu_d(x_d)$. In this section, we are interested with the problem of sampling from such a multivariate probability density.

Remark B.1. Such multivariate probability densities are encountered in Chapter 5, where they are used to construct the approximation of u in tree-based tensor format (see Algorithm 5.2) as optimal sampling measure for the boosted optimal weighted least-squares projection. Also it may have been obtained by learning a high-dimensional probability distribution as in [49].

B.1.1 Marginal distributions

Let α denote a non-empty subset of $D = \{1, \dots, d\}$ and $\alpha^c = D \setminus \alpha$. We let $x_\alpha = (x_\nu)_{\nu \in \alpha}$, $\mu_\alpha = \otimes_{\nu \in \alpha} \mu_\nu$ and $\mathcal{X}_\alpha = \times_{\nu \in \alpha} \mathcal{X}_\nu$ and $x_{\alpha^c} = (x_\nu)_{\nu \in \alpha^c}$, $\mu_{\alpha^c} = \otimes_{\nu \in \alpha^c} \mu_\nu$ and $\mathcal{X}_{\alpha^c} = \times_{\nu \in \alpha^c} \mathcal{X}_\nu$.

The marginal distribution of $d\rho$ associated to the variables x_α is defined by

$$d\rho_\alpha(x_\alpha) = \int_{\mathcal{X}_{\alpha^c}} d\rho(x), \quad (\text{B.2})$$

and thus

$$d\rho_\alpha(x_\alpha) = \int_{\mathcal{X}_{\alpha^c}} v(x) d\mu(x) = f_\alpha(x_\alpha) d\mu_\alpha(x_\alpha). \quad (\text{B.3})$$

where

$$\begin{aligned} f_\alpha(x_\alpha) &= \int_{\mathcal{X}_{\alpha^c}} v(x) d\mu_{\alpha^c}(x_{\alpha^c}) = \int_{\mathcal{X}_{\alpha^c}} \sum_{k \in \Lambda} v_k \varphi_k(x) d\mu_{\alpha^c}(x_{\alpha^c}) \\ &= \sum_{k_\alpha \in \Lambda^\alpha} \varphi_{k_\alpha}^\alpha(x_\alpha) \sum_{k_{\alpha^c} \in \Lambda^{\alpha^c}} v_k \int_{\mathcal{X}_{\alpha^c}} \varphi_{k_{\alpha^c}}^{\alpha^c}(x_{\alpha^c}) d\mu_{\alpha^c}(x_{\alpha^c}) \\ &= \sum_{k_\alpha \in \Lambda^\alpha} \varphi_{k_\alpha}^\alpha(x_\alpha) \sum_{k_{\alpha^c} \in \Lambda^{\alpha^c}} v_k \prod_{\nu \in \alpha^c} \int_{\mathcal{X}_\nu} \varphi_{k_\nu}^\nu(x_\nu) d\mu_\nu(x_\nu). \end{aligned} \quad (\text{B.4})$$

Denoting $f_{\alpha, k_\alpha} = \sum_{k_{\alpha^c} \in \Lambda^{\alpha^c}} v_k \prod_{\nu \in \alpha^c} \int_{\mathcal{X}_\nu} \varphi_{k_\nu}^\nu(x_\nu) d\mu_\nu(x_\nu)$, for $k_\alpha \in \Lambda^\alpha$, we can write,

$$f_\alpha(x_\alpha) = \sum_{k_\alpha \in \Lambda^\alpha} f_{\alpha, k_\alpha} \varphi_{k_\alpha}^\alpha(x_\alpha). \quad (\text{B.5})$$

f_{α, k_α} is obtained by computing contractions of the tensor \mathbf{v} along the dimensions $\nu \in \alpha^c$ with vectors containing the integral of the basis function of spaces $\mathcal{H}_\nu, \nu \in \alpha^c$.

More precisely, let $\mathbf{h}^\nu \in \mathbb{R}^{\Lambda^\nu}$ denote the vector containing the integral of the basis functions of the space \mathcal{H}_ν ,

$$\mathbf{h}^\nu = (h_{k_\nu}^\nu)_{k_\nu=1}^{n_\nu} \text{ with } h_{k_\nu}^\nu = \int_{\mathcal{X}_\nu} \varphi_{k_\nu}^\nu d\mu_\nu(x_\nu).$$

Letting $\mathbf{h}^{\alpha^c} \in \mathbb{R}^{\Lambda^{\alpha^c}}$ be the tensor defined by $\mathbf{h}_{k_{\alpha^c}}^{\alpha^c} = \prod_{\nu \in \alpha^c} \mathbf{h}_{k_\nu}^\nu$, we have

$$f_{\alpha, k_\alpha} = \sum_{k_{\alpha^c} \in \Lambda^{\alpha^c}} v_k \prod_{\nu \in \alpha^c} \int_{\mathcal{X}_\nu} \varphi_{k_\nu}^\nu(x_\nu) d\mu_\nu(x_\nu) = \sum_{k_{\alpha^c} \in \Lambda^{\alpha^c}} v_{k_\alpha, k_{\alpha^c}} \mathbf{h}_{k_{\alpha^c}}^{\alpha^c}$$

The contraction of the modes k_{α^c} of \mathbf{v} and \mathbf{h}^{α^c} results in a tensor $f_{\alpha, k_\alpha} \in \mathbb{R}^{\Lambda^\alpha}$.

Proposition B.1. *Let v be a function in tree-based tensor format $\mathcal{T}_r^T(\mathcal{H})$ associated with a certain dimension tree T over $\{1, \dots, d\}$, the function $f_\alpha(x_\alpha)$ defined by $f_\alpha(x_\alpha) = \int_{\mathcal{X}_{\alpha^c}} v(x) d\mu_{\alpha^c}(x_{\alpha^c})$ is also in tree-based tensor format associated with the tree $T_\alpha = \{\gamma \in T : \gamma \subset \alpha\} \setminus \emptyset$ with root α .*

The goal is to sample from $d\rho$ and to do this, we present techniques to sample from $d\rho^*$.

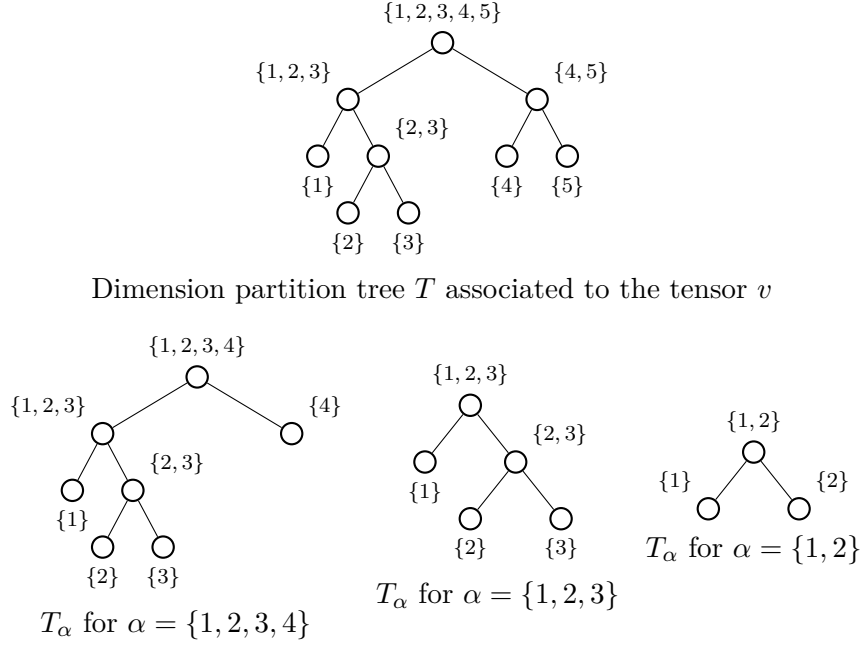


Figure B.1 – Examples of dimension partition trees

B.1.2 Conditional distributions

In this section, we introduce several notations and definitions that will be necessary for the multivariate algorithms.

We introduce the ordered set, $[\nu] = \{1, \dots, \nu\}$ and denote $x_{[\nu]} = (x_1, \dots, x_\nu)$. For all $2 \leq \nu \leq d$ the conditional distribution denoted $\rho_{\nu|[\nu-1]}$ of the random variable X_ν knowing $X_{[\nu-1]}$, and defined by

$$\rho_{\nu|[\nu-1]}(x_\nu | x_{[\nu-1]}) = \frac{\rho_{[\nu]}(x_{[\nu]})}{\rho_{[\nu-1]}(x_{[\nu-1]})}, \quad (\text{B.6})$$

where $\rho_{[\nu]}(x_{[\nu]})$ and $\rho_{[\nu-1]}(x_{[\nu-1]})$ denote the marginal distributions associated with the indices $\{1, \dots, \nu\}$ and $\{1, \dots, \nu-1\}$ respectively. Replacing the marginals $\rho_{[\nu]}(x_{[\nu]})$ and $\rho_{[\nu-1]}(x_{[\nu-1]})$ by their expressions from equation (B.3) and thanks to the product structure of μ , it holds

$$d\rho_{\nu|[\nu-1]}(x_\nu | x_{[\nu-1]}) = \frac{f_{[\nu]}(x_{[\nu]})}{f_{[\nu-1]}(x_{[\nu-1]})} d\mu_\nu(x_\nu). \quad (\text{B.7})$$

Remark B.2. If both $f_{[\nu]}$ and $f_{[\nu-1]}$ admit a representation in tree-based format, the ratio $\frac{f_{[\nu]}(x_{[\nu]})}{f_{[\nu-1]}(x_{[\nu-1]})}$ has not.

B.2 Sampling from multivariate probability distributions in tree-based tensor formats

In this section, we describe a sequential sampling technique which allows sampling from multivariate probability distributions. It relies on the fact that any multivariate probability distribution can be written,

$$d\rho^*(x_1, \dots, x_d) = d\rho_1^*(x_1) d\rho_{1|2}^*(x_2|x_1) \dots d\rho_{d|\{1, \dots, d-1\}}^*(x_d|x_1, \dots, x_{d-1}) \quad (\text{B.8})$$

where $d\rho_1^*(x_1)$ is the marginal of $d\rho^*$ associated to the variable x_1 , $d\rho_{\nu|[\nu-1]}^*(x_\nu|x_1, \dots, x_{\nu-1})$ is the conditional distribution of x_ν knowing $x_1, \dots, x_{\nu-1}$. The advantage of this technique is that we only have to sample from univariate densities. The principle of sequential sampling is to first sample x_1^k from $d\rho_1^*(x_1)$ and then for $2 \leq \nu \leq d$, successively sample x_ν^k from the conditional distribution of X_ν knowing $X_1 = x_1, \dots, X_{[\nu-1]} = x_{[\nu-1]}^k$. This way, at each step of the algorithm, we only have to sample from univariate densities, which are explicitly given. One may then rely on standard simulation methods such as rejection sampling, inverse transform sampling or slice sampling techniques, see section 3 or [31].

Algorithm B.1 Sequential conditional sampling for $\rho^*(x)$

Inputs: Function v such that $\rho(x) = v(x)\mu(x)$

Outputs: (x^1, \dots, x^n) sampled from $\rho(x)$.

for $k = 1, \dots, n$ **do**

 Compute $\rho_1(x_1)$ the marginal of ρ associated to the dimension 1

 Sample x_1^k from $x_1 \mapsto |\rho_1(x_1)|$ using one technique for univariate sampling.

for $\nu = 2, \dots, d$ **do**

 Compute $\rho_{[\nu]}(x_{[\nu]})$ the marginal of ρ associated to the indices dimension $1, \dots, \nu$.

 Compute $x_\nu \mapsto \rho_{\nu|[\nu-1]}(x_\nu|x_{[\nu-1]}^k) = \frac{\rho_{[\nu]}(x_{[\nu]})}{\rho_{[\nu-1]}(x_{[\nu-1]}^k)}$ the conditional distribution of X_ν knowing $X_1 = x_1^k, \dots, X_{\nu-1} = x_{\nu-1}^k$.

 Sample x_ν^k from $x_\nu \mapsto \rho_{\nu|[\nu-1]}(x_\nu|x_1^k, \dots, x_{\nu-1}^k)$ using one technique for univariate sampling.

end for

 Set $x^k = (x_1^k, \dots, x_d^k)$.

end for

Remark B.3. When working with an approximate density which may take negative values, the marginal and conditional densities may also take negative values. In practice, when working with a marginal and conditional density w with respect to Lebesgue measure we take the positive part of w .

Remark B.4. To avoid negative values in the approximated marginal ρ_α , we may replace it by its positive part $\langle \rho_\alpha \rangle_+$.

Titre : Approximation adaptative de fonctions en grande dimension avec des réseaux de tenseurs basés sur des arbres pour la quantification d'incertitudes.

Mot clés : approximation en grande dimension, approximation de faible rang, formats de tenseurs basés sur des arbres, analyse en composantes principales, stratégies adaptatives, moindres carrés pondérés.

Résumé : Les problèmes de quantification d'incertitudes des modèles numériques nécessitent de nombreuses simulations, souvent très coûteuses (en temps de calcul et/ou en mémoire). C'est pourquoi il est essentiel de construire des modèles approchés qui sont moins coûteux à évaluer. En pratique, si la réponse d'un modèle numérique est représentée par une fonction, on cherche à en construire une approximation.

L'objectif de cette thèse est de construire l'approximation d'une fonction qui soit contrôlée tout en utilisant le moins d'évaluations possible de la fonction. Dans un premier temps, nous proposons une nouvelle méthode basée sur les moindres carrés pondérés pour construire l'approximation d'une fonction dans un espace vectoriel. Nous prouvons que la projection vérifie une propriété de stabilité numérique presque sûrement et une propriété de quasi-optimalité en espérance. En pratique on observe que la taille de l'échantillon est plus proche de la dimension de l'espace d'approximation que pour les autres techniques de moindres carrés pondérées existantes. Pour l'approximation en grande dimension et afin

d'exploiter de potentielles structures de faible dimension, nous considérons dans cette thèse des approximations dans des formats de tenseurs basés sur des arbres. Ces formats admettent une paramétrisation multilinéaire avec des paramètres formant un réseau de tenseurs de faible ordre et sont ainsi également appelés réseaux de tenseurs basés sur des arbres. Dans cette thèse, nous proposons un algorithme pour construire l'approximation de fonctions dans des formats de tenseurs basés sur des arbres. Il consiste à construire une hiérarchie de sous-espaces imbriqués associés aux différents niveaux de l'arbre. La construction de ces espaces s'appuie sur l'analyse en composantes principales étendue aux fonctions multivariées et sur l'utilisation de la nouvelle méthode des moindres carrés pondérés. Afin de réduire le nombre d'évaluations nécessaires pour construire l'approximation avec une certaine précision, nous proposons des stratégies adaptatives pour le contrôle de l'erreur de discrétisation, la sélection de l'arbre, le contrôle des rangs et l'estimation des composantes principales.

Title: Adaptive approximation of high-dimensional functions with tree tensor networks for Uncertainty Quantification.

Keywords: high-dimensional approximation, low-rank approximation, tree-based tensor formats, principal component analysis, adaptive strategies, weighted least-squares.

Abstract: Uncertainty quantification problems for numerical models require a lot of simulations, often very computationally costly (in time and/or memory). This is why it is essential to build surrogate models that are cheaper to evaluate. In practice, the output of a numerical model is represented by a function, then the objective is to construct an approximation.

The aim of this thesis is to construct a controlled approximation of a function while using as few evaluations as possible. In a first time, we propose a new method based on weighted least-squares to construct the approximation of a function onto a linear approximation space. We prove that the projection verifies a numerical stability property almost surely and a quasi-optimality property in expectation. In practice we observe that the sample size is closer to the dimension of the approximation space than with existing weighted least-squares methods. For high-dimensional approximation, and in order

to exploit potential low-rank structures of functions, we consider the model class of functions in tree-based tensor formats. These formats admit a multilinear parametrization with parameters forming a tree network of low-order tensors and are therefore also called tree tensor networks. In this thesis we propose an algorithm for approximating functions in tree-based tensor formats. It consists in constructing a hierarchy of nested subspaces associated to the different levels of the tree. The construction of these subspaces relies on principal component analysis extended to multivariate functions and the new weighted least-squares method. To reduce the number of evaluations necessary to build the approximation with a certain precision, we propose adaptive strategies for the control of the discretization error, the tree selection, the control of the ranks and the estimation of the principal components.