Modèles d'ordre réduit pour les problèmes aux dérivées partielles paramétrés : approche couplée POD-ISAT et chainage temporel par algorithme pararéel

Dung Bui

ÉCOLE CENTRALE DES ARTS
ET MANUFACTURES
"ÉCOLE CENTRALE PARIS"

Thèse de doctorat

Discipline : Mathématiques Appliquées

présentée par

Dung Bui

Modèles d'ordre réduit pour les problèmes aux dérivées partielles paramétrés : approche couplée POD-ISAT et chainage temporel par algorithme pararéel

---

dirigée par Florian De Vuyst

Soutenue le 14 février 2014 devant le jury composé de :

M. Pierre Villon	UTC	Président
M. Christophe Prud'homme	Université Strasbourg	Rapporteur
M. Damien Tromeur-Dervout	Polytech Lyon I	Rapporteur
M. Pascal Laurent	ECP	Examinateur
M. Frédéric Abergel	ECP	Examinateur
M. David Neron	ENS Cachan	Examinateur
M. Michel Ravachol	Dassault Aviation	Invité
M. Florian De Vuyst	ENS Cachan	Directeur de thèse

École Central Paris
Grande Voie des Vignes
F-92 295 CHÂTENAY-MALABRY Cedex

Cette thèse est dédiée à Maman et Papa,
sans lesquels je n'aurais jamais vu le jour.

Remerciements

Ce travail de thèse, réalisé au laboratoire Mathématiques Appliquées aux Systèmes (MAS) et au Centre de Mathématiques et de Leurs Applications (CMLA), n'aurait pu être mené à bien sans l'aide des personnes qui je voudrais adresser toutes mon remerciement.

En premier lieu, je tiens à remercier mon directeur de thèse, Florian De Vuyst, pour la confiance qu'il m'accordée en acceptant d'encadrer ce travail doctoral, pour ses précieux conseils et pour toutes les heures qu'il a consacrées à diriger cette recherche.

Je tiens à remercier Christophe Prud'homme et Damien Tromeur-Dervout de m'avoir fait l'honneur de rapporter cette thèse. Je remercie tout particulièrement Christophe Prud'homme pour l'intérêt qu'il a porté à ce travail, pour ces remarques constructives et ses conseils. Je remercie également Pierre Villon, Pascal Laurent, Frédéric Abergel, David Neron et Michel Ravachol d'avoir accepté de fait partie de mon jury.

Je tiens à remercier tous les membres de l'équipe du projet CSDL, en particulier Mohamed Hamdaoui, pour non seulement sa collaboration scientifique, mais aussi pour son aide à la rédaction, pour les discussions de la vie quotidienne.

Je voudrais exprimer toute ma reconnaissance à Pascal, Hichem, Nesrine, Hinh, Thanh et Madame De Vuyst qui ont pris le temps de lire une grande partie de ce mémoire.

Je souhaiterais remercier tous les membres du laboratoire MAS, notamment Sylvie et Annie pour leur accueil chaleureux, leur humour. Mes remerciements pour l'ambiance de la vie agréable du laboratoire se dirigent également vers d'autres permanents, doctorants, post-doctorants parmi lesquels Nesrine, Cassio, Rania, Hichem,

Andrian, Victor, Antoine, Anthony, Nizar, Mario... pour les discussions durant les pauses-café, les repas, ainsi que pendant les matchs de baby-foot très amusants.

Mes remerciements vont aussi à tous mes camarades de la classe CLC47 qui m'ont permis d'oublier momentanément le travail dans les repas, soirées, sorties, voyages...pour leur soutien et leur encouragement tout au long de ma thèse.

En fin, je remercie du fond de mon cœur ma famille qui n'a jamais cessé de m'encourager et de croire en moi tout au long de mon cursus. Je remercie beaucoup ma femme pour son amour, sa compréhension et son soutien.

Modèles d'ordre réduit pour les problèmes aux dérivées partielles paramétrés : approche couplée POD-ISAT et chainage temporel par algorithme pararéel

Résumé

Cette thèse porte sur la conception des méthodes robustes de réduction d'ordre de modèles numériques de type Éléments Finis (EF) avec contrôle de la précision. La réduction d'ordre est en général nécessaire pour réduire drastiquement les temps de calcul et permettre ainsi une analyse paramétrique, une étude de faisabilité ou de performance de système (avion, unité de production, procédé complexe, etc). Dans cette étude, la technique de décomposition orthogonale aux valeurs propres (POD) sera utilisée pour construire des modèles réduits locaux. Informatiquement parlant, le "modèle" sera considéré comme une base de données de résultats de calcul avec capacité d'extrapolation et d'interpolation locale. Une stratégie adaptative pour stocker et accéder à la base de données est étudiée en étendant l'algorithme In situ Adaptive Tabulation (ISAT) proposé initialement par Pope. En fonction de l'usage et des exigences en précision des résultats, la base de données est enrichie en ligne (online) par des appels au modèle fin en respectant une précision spécifiée jusqu'à couvrir le domaine paramétrique entier, après quoi l'évaluation d'une solution devient très peu couteuse. L'approche couplée POD-ISAT proposée dans cette thèse fournit une méthode de réduction de modèle EF très performante. La méthodologie est évaluée sur un cas réel de conditionnement d'air en régime stationnaire de cabine d'avion dépendant de plusieurs paramètres de conception (température et vitesse d'entrée d'air, mode de ventilation personnalisée, conductivité thermique du fuselage, etc.). Pour les problèmes d'évolution en temps, nous explorons une piste de chainage de modèles et d'utilisation d'algorithme de parallélisation en temps tel que l'algorithme pararéel initialement proposé par Lions, Maday et Turinici (2001). Nous proposons ici une variante quasi-Newton de l'algorithme pararéel que nous appelons algorithme Broyden-pararéel. Il est appliqué au calcul de la diffusion d'un gaz dans la cabine d'avion. Cette thèse s'insère dans le cadre du projet CSDL (Complex System Design Lab, Fond Unique Interministériel) visant à développer une plate-forme logicielle multidisciplinaire pour la conception de systèmes complexes.

Mots clés décomposition orthogonale aux valeurs propres, in situ adaptive tabulation, réduction d'ordre, krigeage, éléments finis, semi-intrusive, système de contrôle de l'air, équations de Navier-Stokes incompressible, pararéel, calcul parallèle.

Reduced order models for parameterized partial differential problems : coupled approach POD-ISAT and temporal sequencing by parareal algorithm

Abstract In this thesis, an efficient Reduced Order Modeling (ROM) technique with control of accuracy for parameterized Finite Element solutions is proposed. The ROM methodology is usually necessary to drastically reduce the computational time and allow for tasks like parameter analysis, system performance assessment (aircraft, complex process, etc.). In this thesis, a ROM using Proper Orthogonal Decomposition (POD) will be used to build local models. The "model" will be considered as a database of simulation results with extrapolation capability and suitable local interpolation. An adaptive strategy to store and retrieve the database is studied by extending the algorithm In Situ Adaptive Tabulation (ISAT) originally proposed by Pope (1997). Depending on the use and the accuracy requirements, the database is enriched in situ (i.e. online) by call of the fine (reference) model and construction of a local model with an accuracy region in the parameter space. Once the trust regions cover the whole parameter domain, the computational cost of a solution becomes inexpensive. The coupled POD-ISAT, here proposed, provides a promising effective ROM approach for parametric finite element model. POD is used for the low-order representation of the spatial fields and ISAT for the local representation of the solution in the design parameter space. This method is tested on a Engineering case of stationary air flow in an aircraft cabin. This is a coupled fluid-thermal problem depending on several design parameters (inflow temperature, inflow velocity, fuselage thermal conductivity, etc.). For evolution problems, we explore the use of time-parallel strategies, namely the parareal algorithm originally proposed by Lions, Maday and Turinici (2001). A quasi-Newton variant of the algorithm called Broyden-parareal algorithm is here proposed. It is applied to the computation of the gas diffusion in an aircraft cabin. This thesis is part of the project CSDL (Complex System Design Lab) funded by FUI (Fond Unique Interministériel) aimed at providing a software platform for multidisciplinary design of complex systems.

Keywords proper orthogonal decomposition (POD), reduce order modeling (ROM), in situ adaptive tabulation (ISAT), kriging interpolation, finite element, semi-intrusive, imcompressible Navier-Stokes equation, air control system, parareal, parallel computing.

Table des matières

Table des figures

Liste des tableaux

Introduction

La simulation numérique des modèles physiques prend aujourd'hui une place importante dans de nombreuses branches de sciences et technologies industrielles. Le coût de la résolution numérique des problèmes non-linéaires peut parfois devenir inabordable. En particulier pour la conception d'un système complexe industriel, on utilise souvent des modèles Éléments Finis ou Volumes Finis de centaines de milliers, voire de millions de degrés de liberté. Malgré une forte évolution des capacités des ordinateurs, la visualisation d'une réponse qui dépend de paramètres de conception n'est pas si aisée, en raison du temps de calcul de la solution. Un système complexe se compose en général de plusieurs sous-systèmes, chacun étant représenté par un modèle numérique. La conception optimale est un lieu de forte interaction entre les disciplines intervenant dans la définition d'un système. Lors de la conception optimale d'un système, on distingue deux types de dépendances : des couplages entre physiques (les sorties d'une discipline servent d'entrées à d'autres disciplines, et des dépendances indirectes entre critères contradictoires), et des couplages dus à l'optimisation (la performance d'une discipline allant à l'encontre de la performance dans une autre discipline). Des formulations et algorithmes spécifiques d'optimisation collaborative sont alors nécessaires. Ils visent à créer des modèles pour chaque sous-système permettant d'estimer rapidement la solution correspondant à une nouvelle entrée. Ces modèles répondent à ces demandes : calcul rapide, erreur contrôlable, base de données portable.

Les méthodes de réduction de modèle semblent être une voie vers ces demandes. Ces méthodes exploitent le fait que la réponse d'un modèle ou d'une famille de modèles peut souvent être approchée avec une précision raisonnable par la réponse d'un modèle réduit, obtenu par projection du modèle initial sur une base réduite de fonctions de dimension inférieure de quelques ordres de grandeur à la dimension des modèles numériques fins sous-jacents.

Cette thèse s'inscrit dans le cadre du projet *Complex System Design Lab* piloté par Dassault Aviation. Le projet vise à mettre en place un environnement collaboratif complet d'aide à la décision pour la conception de systèmes complexes tout particulièrement durant la phase d'avant-projet. C'est en effet à ce niveau que l'uti-

lisation d'outils de simulation est le plus stratégique afin d'assurer une conception la meilleure possible.

L'objectifs de thèse :

- Proposer un algorithme robuste d'enrichissement adaptatif de modèles réduits locaux dans l'espace paramétrique pour des solutions éléments finis d'équations aux dérivées partielles paramétriques. Appliquer à la conception d'un système contrôle de l'air dans une cabine d'avion.
- Mettre en place, analyser et améliorer un algorithme parallèle en temps pour l'accélération de simulations numériques d'équations d'évolution.

Plan du manuscrit

Le travail de recherche s'organise de manière suivante :

Nous proposons au **premier chapitre** un cas industriel réel concernant la conception d'un système de contrôle de l'air dans une cabine d'avion. Notre travail est lié au projet CSDL dont l'objectif est la gestion et la manipulation de modèles numériques de systèmes complexes permettant l'aide à la décision en phase de conception avant-projet d'une cabine d'avion chez Dassault Aviation. Nous parlons d'abord du rôle important du système ECS et sa fonction dans une cabine d'avion commercial. Dans la phase d'avant-projet, il est important d'envisager un standard et des paramètres de conception. Nous évoquons donc quelques réglementations sur la qualité de l'air et définissons les éléments principaux influençant la qualité de l'air. Nous citons ensuite quelques travaux scientifiques récents concernent l'innovation du système ECS ainsi que le calcul numérique sur le flux d'air dans une cabine d'avion. Ensuite, nous présentons quelques applications de méthode de réduction de dimensionalité dans le contexte industriel. En particulier, l'application des modèles réduits dans le domaine de la visualisation, de l'optimisation et contrôle optimal sont étudiées.

Le **deuxième chapitre** présente des techniques de réduction de modèle dans la littérature. On peut classer ces techniques en deux catégories. La première se compose des modèles réduits dits "sans physiques sous-jacentes" ou "métamodèles" (*surrogate models* en anglais). Ces modèles réduits regroupent les surfaces de réponse polynômiales, les moindres carrés mobiles, les réseaux de neurones, les réseaux de fonctions à base radiale, krigeage, etc. La deuxième catégorie comprend des modèles qui tiennent compte de modèles physiques sous-jacents, appelés les méthodes de réduction d'ordre (Reduced Order Modeling ou ROM en anglais). Plus particulièrement, nous présentons le métamodèles de krigeage et quelques méthodes de réduction de dimensionalité, notamment la technique de POD-Galerkin, Bases Réduites, et Proper Generalized Decomposition (PGD).

Le **troisième chapitre** propose une nouvelle méthode de réduction de modèle, l'algorithme POD-ISAT, ce qui est la première partie de la contribution de thèse. Au début, on définit le cas test sous la forme mathématique avec des hypothèses prises en compte. Puis la méthodologie de l'algorithme POD-ISAT sera présentée en détail. Cette méthode est testée sur le cas test avec deux situations de conception, l'espace de paramètres est respectivement en dimension deux et quatre. Le résultat numérique consiste en l'estimation d'erreur relative (compare avec la solution de modèle fin) et analyse de speed-up de POD-ISAT. Nous comparons aussi l'efficacité de POD-ISAT avec un modèle de réduit POD standard.

Le **quatrième chapitre** concerne une technique pour résoudre un problème d'évolution. C'est une combinaison de la réduction de modèles et *l'algorithme pararéel*, un algorithme de calcul parallèle en temps. Plus précisément, l'algorithme pararéel qui permet un calcul parallèle d'un problème d'évolution temporelle fait appel à un solveur "grossier" prédicteur peu coûteux. Il est alors naturel d'envisager d'utilisation de modèles d'ordre réduit comme solveurs grossiers locaux en temps. Dans la première section du chapitre, nous parlons de l'état de l'art et d'analyse de l'algorithme pararéel. La deuxième section expose deux pistes pour construire et utiliser modèle réduit comme un solveur grossier de l'algorithme pararéel.

Le **cinquième chapitre** propose une nouvelle variante de l'algorithme pararéel nommée par l'algorithme Broyden-pararéel. La dérivation de cet algorithme vient du respect par construction la condition de quasi-Newton (CQN). Nous testons les deux algorithmes, pararéel original et Broyden-pararéel, sur deux applications. Une est la résolution des équations différentielles ordinaires pour le système Brusselator. L'autre est la résolution des équations aux dérivées partielles qui modélisent le courant du dioxyde de carbone (CO_2) dans la cabine d'avion sous l'effet de convection et diffusion.

Chapitre 1

Problématique industrielle

Sommaire

1.1 Système de contrôle de l'air dans une cabine d'avion

De nos jours, l'avion est un mode de transport essentiel, et parmi les plus confortables, en particulier pour les longues distances. Le nombre de passagers dans le monde augmente considérablement depuis des dizaines d'années et ne cesse de croître. En 30 ans, ce nombre s'est multiplié par 4, passant de 380 millions en 1970 à 1 460 millions en 1998 [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002]. Il y a de plus en plus des voyageurs âgés ainsi que des enfants et adultes souffrant de troubles médicaux (par exemple maladies cardio-vasculaires et pulmonaires). C'est pour cela que la qualité de l'air dans la cabine d'avion est prise en considération dans la phase de conception d'avions.

La cabine d'avion est un environnement fermé qui peut causer certaines gênes à ses occupants. En effet, l'altitude des avions en régime de croisière est d'environ 10000 mètres au-dessus du niveau de la mer où la pression et la température sont très faibles. En plus, l'air extérieur est moins humide, moins oxygéné. Les conditions sont les mêmes dans la cabine, où encore il y a des bruits, des vibrations, un manque de place et un risque de transmission de maladies.

Il devient primordial donc de s'intéresser aux risques sanitaires générés par cet environnement et plus particulièrement par la qualité de l'air en cabine d'avion. Selon les facteurs et les niveaux de contamination, la qualité de l'air peut avoir un impact sur la sécurité (assurée par l'équipage et la ventilation), sur la santé (maladies infectieuses) et sur le confort (température, odeurs, humidité relative...). La conception d'un système de contrôle de l'environnement (Environmental control system ou ECS) joue un rôle important dans ce cas. Ce système doit répondre à la fois à la qualité de l'air et à la contrainte économique (énergie consommée).

Tous les gros avions commerciaux utilisent le système ECS dont l'air extérieur est récupéré par des moteurs à propulsion (engine bleed air). La figure 1.1 montre un système ECS général. L'air comprimé (bleed air) est extrait par un moteur (engine) et distribué en grande partie dans le groupe de climatisation (air-conditioning pack), où l'air est comprimé et chauffé (200kPa, 200 °C). Une partie du fluide passe aux turbo refroidisseurs pour baisser sa température et sa pression (80kPa, 5 °C), puis s'écoule dans la chambre de mélange avant d'être injectée dans la cabine d'avion. Les ventilateurs de recirculation prélèvent une partie de l'air de la cabine, puis la poussent à la chambre de mélange à traverse un filtre. L'autre partie de l'air extérieur récupéré passe à la valve d'étranglement (throttling valve) permettant de donner un fluide de pression normale et de température conservé (80kPa, 200 °C). Cette dernière partie est contrôlée indépendamment et se mélange localement avec

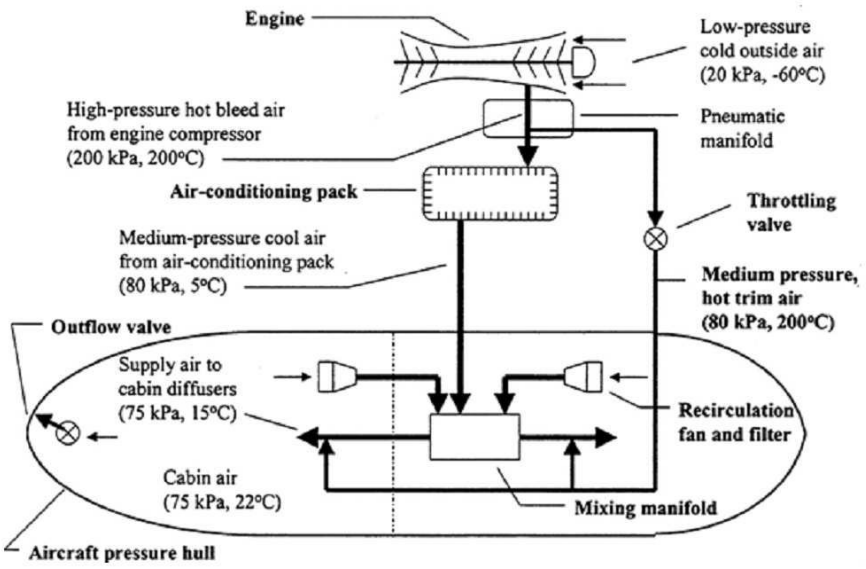

FIGURE 1.1 – Système contrôle environnement (extrait de [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002])

le fluide de la chambre de mélange au-dessus, avant d'être injecté dans la cabine des passagers. L'air récupéré par les moteurs est à une pression suffisante pour faire fonctionner les packs de climatisation et de pressurisation de la cabine. La pression précise de la cabine est maintenue par une ou plusieurs vannes d'échappement, qui régularisent automatiquement le flux d'air venu de l'extérieur de l'avion à l'environnement ambiant.

1.2 Quelques réglementations sur la qualité de l'air

Plusieurs comités se sont mis en place afin d'établir des normes et des directives pouvant être appliquées pour la qualité de l'air en cabine d'avion. Le tableau 1.1 présente quelques normes et directives utilisées et appliquées à la conception de système de contrôle de l'air de la cabine d'avion.

La FAA (Federal Aviation Administration) est une agence gouvernementale chargée des réglementations et des contrôles concernant l'aviation civile aux États-Unis. Elle dépend du Ministère des Transports des États-Unis. Actuellement, ses réglementations portent sur la qualité de l'air concernant l'Ozone (O_3), le monoxyde de carbone (CO), le dioxyde de carbone (CO_2), la ventilation, et la pression de la ca-

TABLE 1.1 – Les normes et directives de qualité de l'air

Norms et Directives	Facteurs environnementaux clés	Population/environnement
FAA-Airwothiness 1998	Quantité ou valeur acceptable de : CO, CO_2, O_3, pression	Passagers et équipages Cabine d'avion
ASHRAE 55-62 [ash, SDa, ash, SDb]	Confort thermique et ventilation pour une qualité acceptable de l'air intérieur	Population en général Bâtiments
ACGIH TLVs, 1998	Valeur limite des produits chimiques, agents physiques et agents biologiques	Employé Lieux de travail industriel
OSHA PELs, 1998	Valeur limite autorisée pour les produits chimiques toxiques	Employé Lieux de travail industriel

bine. Celles-ci sont définies dans la norme 14 CFR 25 (section 25, title 14 of the Code of Federal Regulations) et sont prévues comme des caractéristiques de conception d'avions qui sont certifiées par la norme 14 CFR 21 (attestant que le transporteur respecte la réglementation lors de la conception, de la construction et de l'exploitation de l'avion). La 14 CFR 21 est une norme opérationnelle et s'applique aux transporteurs aériens nationaux ainsi qu'internationaux.

L'ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) est une organisation technique internationale dans le domaine de génie thermique et climatique (chauffage, ventilation, air climatisé, production de froid). Fondée en 1894, elle tient une réunion annuelle. La norme ANSI/ASHRAE 62 (Ventilation pour la Qualité de l'Air) a pour objectif d'indiquer les taux de ventilation minimum et les conditions de qualité de l'air intérieur qui sont acceptables pour les occupants, tout en minimisant les effets potentiellement néfastes pour la santé. Cette norme vise à être le guide pour améliorer la qualité de l'air dans les édifices existants. En particulier, la norme ASHRAE 55 (Confort thermique) a pour but d'indiquer les combinaisons de facteurs environnementaux thermiques d'intérieur, et des facteurs personnels produisant des conditions environnementales thermiques acceptables pour une majorité d'occupants dans l'espace. Cette norme ne concerne pas les facteurs de l'environnement tels que la qualité de l'air, l'acoustique, l'éclairage ou d'autres polluants physiques, chimiques, ou biologiques qui peuvent affecter le confort ou la santé.

Pour la modélisation numérique, il convient de s'appuyer sur ces normes pour caractériser les intérêts de confort et de qualité de l'air.

1.3 Les éléments influençant le confort et la qualité de l'air dans la cabine d'avion

1.3.1 Pression de la cabine

Les avions commerciaux volent à une altitude entre 10 000 - 15 000 m, où l'air est assez sec, la température est de l'ordre de -60°C, et la pression barométrique basse (26 kPa). La pression dans la cabine est donc toujours inférieure à celle du niveau de la mer. Ces conditions ont deux conséquences : une diminution de la pression d'oxygène et une expansion des gaz dans le corps humain [Association, 2002]. La FAA stipule que la pression de la cabine ne doit pas être inférieure à 75 kPa, ce qui équivaut à la pression d'une altitude de 2440 m. Le but de la pressurisation est de maintenir la P_{O_2} (pression en oxygène) dans des limites acceptables. Les valeurs de P_{O_2} au niveau de la mer et à une altitude de 2440 m sont de 21 kPa et de 16 kPa respectivement.

1.3.2 Température et vitesse de l'air dans la cabine

Avec la pression, ces deux champs jouent un rôle important dans le système de contrôle de l'air. En particulier, ils sont indispensables pour étudier le confort des passagers lors de la phase de conception et d'exploitation d'un avion commercial.

La difficulté pour le contrôle de la température est que la perception de la température dépend des préférences individuelles. Chaque individu a une sensibilité particulière vis-à-vis de la température. Cela peut créer des difficultés pour les compagnies aériennes afin de répondre aux différentes attentes de confort des passagers. En plus, les équipages ont des exigences de température différentes de celles des passagers assis. Ils ont typiquement besoin de température plus fraîche pendant qu'ils travaillent et de température plus élevée pendant leur repos.

La température perçue par l'individu est influencée par la température aérienne directe, la température de la paroi (radiation) ainsi que la vitesse de l'air (à la fois la valeur moyenne et le niveau de variation). En outre, on préfère que l'environnement au niveau de la tête soit plus frais qu'aux pieds. Bien que cette demande de variante puisse être compensée par utilisation de couvertures et l'ajout ou non de vêtements, le but du concepteur est de s'assurer qu'une température acceptable soit fournie globalement dans tout l'espace de la cabine. Deux conditions principales doivent être considérées pour satisfaire ce but : la sélection de la température doit être flexible et la température doit être aussi homogène que possible à travers l'espace de la cabine dans les trois directions (longitudinale, verticale et horizontale).

Les champs de vitesse et de température dépendent des paramètres de conception

comme l'emplacement des injecteurs, la température de l'air en entrée, la fonction de ventilation individuelle, la propriété thermique de fuselage, la position des sièges, etc. Ils dépendent aussi de la phase de vol (roulage, décollage, croisière, atterrissage) qui produit le changement de température extérieure et des sources de chaleur intérieures.

A l'étape d'avant-projet pour un nouvel avion, on utilise des simulation CFD (Computational fluid dynamics) pour optimiser le courant d'air dans la cabine. Cela permet de faciliter la validation des systèmes de ventilation et de distribution au cours des essais. La distribution de la température et la vitesse d'air peuvent être prédites par le calcul CFD. Un courant d'air est considéré comme bon s'il assure un champ de température uniforme tout au long de la zone de contrôle particulière de la température.

En général, l'air diffusé dans la cabine d'avion est un mélange d'air neuf et d'air recyclé : 50% -50% dans le cas du Boeing 767 [Elwood H. Hunt et Tilton, 1995] et 60% d'air neuf - 40% d'air recyclé pour Airbus A310 et A340 [D., 1996]. Les normes (FAA et ASHRAE 62-1989) exigent que l'air injecté dans la cabine soit au moins à moitié composé d'air neuf. L'air neuf est extrait au niveau des moteurs ou dans certaines circonstances au niveau des groupes auxiliaires de puissance (ou APU c'est-à-dire Auxiliary Power Units). Le volume total de l'air est renouvelé environ toutes les 3 minutes [A., 1996] dans une cabine de taille standard, soit 20 renouvellements d'air par heure. En comparaison avec une norme du bâtiment, l'air est modifié de 1 à 2,5 fois par heure avec 75% de l'air recyclé [Hunt et Space, 1994].

La quantité d'air diffusé est définie par deux facteurs principaux, l'élimination du CO_2 et la distribution homogène de la température en cabine [D., 1996]. Le renouvellement de l'air sert en effet à contrôler les gradients de température, à prévenir les zones stagnantes d'air froid ou chaud et à dissiper les odeurs. Un passager en position assise a environ au moins 80 fois plus d'air qu'il ne lui est nécessaire. Il a besoin environ 6,8 l d'air par minute et donc 544 l d'air doit être fourni pour une personne par minute.

1.3.3 Humidité Relative (HR)

L'humidité relative, à une température donnée, est le rapport de la vapeur présente dans l'air sur la quantité de vapeur contenue dans cet air à la saturation. Une HR de l'ordre de 40% à 60% représente la valeur typique recommandée pour le confort des passagers, mais serait par contre défavorable pour la sûreté de l'avion. En effet, une trop forte HR entraînerait des phénomènes de condensation au niveau des parois, gorgeant ainsi les matériaux d'isolation d'eau, ce qui augmenterait la masse de l'avion, diminuerait les performances d'isolation et entraînerait le développement

de bactéries et de champignons , voire même la corrosion de la coque [ash, SDc].

À l'altitude de 10 000 m, l'air est extrêmement sec (HR de l'ordre de 2%), alors qu'au sol et dans les phases de vol intermédiaires, l'air doit être déshumidifié avant sa diffusion en cabine afin de prévenir une humidité excessive [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002]. À haute altitude, l'HR de l'air s'accroît progressivement après son passage dans le système de recirculation de l'air de la cabine grâce aux principales sources d'humidité présentes dans la cabine, c'est-à-dire la respiration et l'évaporation par sudation des passagers [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002]. L'évaporation d'eau à partir de la nourriture et des boissons peut également contribuer à une très légère augmentation de l'HR. Les valeurs d'HR en cabine peuvent varier de 5 à 35 % avec une moyenne se situant aux alentours de 20 %. Une valeur moyenne supérieure d'HR peut difficilement être obtenue en raison du fort taux de renouvellement de l'air avec l'air extérieur qui est extrêmement sec. Une longue exposition (supérieure à 3-4 heures) à cette faible HR est une source d'inconfort et peut entraîner une sécheresse de la gorge et des yeux [Hunt et Space, 1994, D., 1996, ash, SDc]. Malgré cela, la réglementation ASHRAE 55-1992 recommande le maintien du pourcentage d'HR entre 25 et 30 % pour des vols relativement longs, avec un minimum de 20 % (ASHRAE 161-2000 ; [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002]). Une HR inférieure à 20 % peut cependant être acceptée pour de courts trajets. Il faut noter qu'il existe un conflit entre le maintien de l'HR et le contrôle des contaminants, car augmenter l'HR, signifie une diminution de l'apport d'air extérieur et permet donc une accumulation des polluants [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002].

Le fait qu'il y ait une HR très faible en cabine présente cependant certains avantages tels que l'inhibition de la prolifération des champignons, la diminution de la viabilité de certaines bactéries ou encore la diminution de la perception de certaines odeurs. La faible humidité a par ailleurs l'inconvénient de favoriser le dépôt de particules respirables et ainsi par ce biais, la probabilité de survie des germes aéroportés est augmentée.

Pour le confort des passagers, il est possible d'augmenter l'HR à l'aide d'humidificateurs. La mise en place de ces systèmes a pour conséquence de générer une pénalité de poids liée à l'eau à transporter, de plus une telle zone humide est un terrain propice à la prolifération bactérienne [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002]. Certains humidificateurs ont malgré tout été développés et sont utilisés sur de petits appareils, mais ce sont des équipements coûteux qui peuvent générer un risque accru de transmission de

maladies infectieuses. De tels systèmes doivent être couplés à des assécheurs au niveau des parois de l'appareil. Une maintenance rigoureuse, des nettoyages et des désinfections des humidificateurs sont également nécessaires.

1.3.4 Contaminants de l'air

Tous les contaminants dans la cabine d'avion peuvent être classés selon deux grandes catégories : les particules et les gaz [PAPE, 2003]. Les particules consistent en des aérosols biologiques, minéraux, organiques, des fibres et graines de poussière et des plus grosses particules comme les matériaux fibreux dont la longueur excède 25 mm. Les molécules gazeuses sont le CO_2, CO, O_3, Composés Organiques Volatils (COV).

Les contaminants peuvent venir de l'extérieur de l'avion (air ambiant) ou de l'intérieur de l'avion (les occupants et la cabine elle-même). Ces sources de contamination peuvent être classées en 5 catégories :

- l'air ambiant ;
- l'air venant des moteurs ou air pressurisé ;
- les occupants ;
- les matériaux propres à l'avion ;
- les émissions liées aux opérations effectuées dans l'avion.

Dans la cabine, le CO2 est principalement généré par la respiration des passagers et de l'équipage et par les réfrigérateurs [D., 1996]. Différentes études montrent que les concentrations en CO_2 peuvent varier, selon le moment du vol, entre 293 à 4 238 ppm en cabine [Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002]. La teneur en CO_2 est utilisée comme indicateur de la qualité de l'air et donc du bon fonctionnement de la ventilation [Dumyahn *et al.*, 2000]. Durant un vol, la moyenne de CO_2 se situe entre 600 et 1 500 ppm et une étude effectuée par la National Aeronautics and Spatial Administration (NASA) montre que des concentrations inférieures à 5 000 ppm n'ont aucun effet sanitaire sur les passagers et l'équipage. Entre 5 000 et 30 000 ppm, il peut se produire des changements biochimiques comme le phénomène d'hypercapnie (augmentation de la pression en CO2 du sang artériel) qui entraîne une baisse du pH sanguin. Une concentration supérieure à 30 000 ppm peut provoquer des atteintes pathologiques allant des maux de tête, vertiges et troubles de la vue au coma voire à la mort [Hunt et Space, 1994].

1.4 Les recherches sur la conception optimale de systèmes ECS

En raison de l'environnement externe spécifique, la fonction principale de l'ECS est de préserver la vie des occupants de l'avion. À une altitude de croisière de 12 000 m, la pression ambiante peut être au bas niveau de 20kPa, la température est inférieure à -60 °C et le teneur en eau de l'air est presque zéro. Sans système de protection, les hommes ne seraient pas capables de survivre dans ces conditions.

Les ingénieurs concevant le système doivent s'assurer que le taux de variation de la pression et sa valeur minimale à l'intérieur de la cabine soient contrôlés pour éviter une dégradation physiologique des occupants. Une fois la protection de la vie assurée, le concepteur doit envisager la performance du système de chauffage et de climatisation ainsi que le système de contrôle du confort des occupants. La conception du confort est plus difficile que celle de la protection de la vie, parce que chaque occupant a une sensation différente vis-à-vis de l'environnement, surtout entre les les passagers et l'équipage.

La conception du système ECS a pour objectif non seulement de fournir un environnement confortable dans la cabine, mais aussi de minimiser la consommation d'énergie ou réduire la taille et le poids du système. Dans la littérature, plusieurs études traitent la conception de nouveaux systèmes ECS :

- Optimisation d'énergie consommée par le système ECS [Vankan, 2003, Leo et Pérez-Grande, 2005, Pérez-Grande et Leo, 2002].
- Optimisation d'énergie couplée avec la taille du système [Pérez-Grande et Leo, 2002].
- Distribution de l'air dans la cabine d'avion : [Zhang et Chen, 2007, Zhang *et al.*, 2010]

Toutes ces études nécessitent des informations sur le courant d'air dans la cabine. Ces informations sont très variables et dépendent des paramètres physiques (air injecté, air extérieur, systèmes d'exploitation électroniques ...). Généralement, il existe deux méthodes pour déterminer l'état de l'air dans la cabine d'avion :

1. sondages (mesurer les informations physiques comme la température, la vitesse, la densité des particules, etc, dans certaines positions intéressées) réalisées sur la maquette à l'échelle 1 :1 (en anglais, mock-up) [Yan *et al.*, 2009] ;
2. modèle numérique [Bianco *et al.*, 2009].

Des études expérimentales sont généralement considérées comme plus fiables, mais elles sont souvent très coûteux et prennent du temps, donc les mesures sont principalement utilisées pour fournir des données pour la validation des simulations numériques [Mo *et al.*, 2003, Garner *et al.*, 2004, Sun *et al.*, 2005, Zhang et Chen,

2007]. De plus, comme certaines caractéristiques de l'écoulement d'air dans les cabines ne sont pas connues, il est difficile d'identifier une méthode de mesure pour l'étudier. Un modèle numérique validé peut alors servir pour analyser de nombreux scénarios et trouver la meilleure conception du système ECS avec un faible coût. Comme la plupart des modèles numériques s'appliquent à un phénomène de fluide particulier, un modèle adapté à l'étude des distributions de l'air dans les cabines n'a pas été trouvé [Liu *et al.*, 2012].

Au cours des 15 dernières années, de nombreuses études expérimentales et numériques ont été réalisées pour étudier la distribution de l'air correcte dans la cabine. [Mo *et al.*, 2003] ont utilisé la vélocimétrie par images de particules (PIV) pour mesurer la distribution de l'air dans la cabine en l'absence de sièges. [Dechow *et al.*, 1997, Waters *et al.*, 2002] ont étudié la qualité de l'air dans la cabine en mesurant la quantité des contaminants, des particules et de la concentration de l'ozone sans analyse de la répartition de l'air dans le détail. [Garner *et al.*, 2004] ont mené une campagne de mesures sur la vitesse de l'air dans un Boeing 747 afin de valider un modèle numérique. Plus récemment, [Sun *et al.*, 2005] et [Zhang et Chen, 2007] ont effectué des mesures expérimentales par vélocimétrie dans un Boeing 767-300 en présence des mannequins. En suite, [Kühn *et al.*, 2009] a étudié les expériences sur la convection forcée et mélangée dans une maquette de cabine d'avion rempli de passagers. Les mesures de la vitesse et de la température ont été effectuées dans un plan de coupe de la maquette. Leur étude démontre que le champ d'écoulement dans les cabines est affecté par divers phénomènes de mécanique de fluide tels que l'interaction entre les jets d'air fournis, la poussée d'Archimède, et l'interactions entre des panaches thermiques. Tous ces paramètres influencent le champ d'écoulement à l'intérieur de la cabine.

Le modèle numérique CFD [Anderson *et al.*, 1995, Versteeg et Malalasekera, 2007, Hirsch, 2007] est un outil largement utilisé pour la conception d'un système ECS. Il résout l'ensemble des équations aux dérivées partielles (EDP) consistant en la conservation de la masse, de la quantité de mouvement (équation de Navier-Stokes), de l'énergie, de la concentration des espèces chimiques, ect. Les solutions sont peut-être des champs de température, de pression et de vitesse de l'air, ou de contaminants, d'humidité relative. Dans ce contexte, les modèles CFD utilisés sont de type RANS (Reynolds-Averaged Navier-Stokes) et LES (Large Eddy Simulation). Il existe plusieurs types pour le modèle RANS dont RNG $k-\epsilon$ [Yakhot et Orszag, 1986] et $k-\epsilon$ standard [Launder et Spalding, 1974] sont plus courants.

[Liu *et al.*, 2012] ont fait un état de l'art sur les études expérimentales et numériques concernant la distribution de l'air dans une cabine d'avion commerciale. Leurs études montrent que la mécanique des fluides numérique (en anglais, com-

putational fluid dynamics ou CFD) est devenue le plus populaire pour calculer des distributions de l'air dans une cabine d'avion. Parmi les modèles CFD, le RNG $k-\epsilon$ est plus utilisé que le modèle $k-\epsilon$ standard ; le modèle LES peut fournir des informatisions de fluide plus précises et détaillées, mais le temps de calcul est deux fois plus long que celui de deux modèles précédents ; le couplage de ces deux modèles LES/RANS [Jouvray et Tucker, 2005, Wang et Chen, 2010] est aussi utilisé et semble assez prometteur [Liu *et al.*, 2012].

1.5 Projet CSDL

Le projet CSDL (Complex Systems Design Lab) a pour objectif de mettre en place un environnement collaboratif complet d'aide à la décision pour la conception de systèmes complexes tout particulièrement durant la phase d'avant-projet. C'est en effet à ce niveau que l'usage des outils de simulation est le plus stratégique afin d'assurer une conception la meilleure possible :

- en explorant systématiquement l'ensemble des paramètres influents pour optimiser au mieux le système et générer le maximum d'innovation
- en estimant les risques et les incertitudes grâce à des analyses approfondies de critère de robustesse
- en disposant d'outils assurant la cohérence des différents niveaux de modélisation et permettant des prises de décision optimales par une analyse précise et interactive des résultats obtenus.

Par ailleurs, la complexité des systèmes considérés rend indispensable l'usage d'outils méthodologiques permettant une maîtrise des processus de conception. Ceux-ci doivent être conçus grâce à l'analyse des cas de tests représentatifs. Le projet CSDL s'appuie sur quatre cas tests couvrant un spectre très large :

- Conception d'un groupe moto propulseur thermique
- Conception d'un groupe moto propulseur électrique
- Optimisation de l'entrée d'air d'un engin supersonique
- **Conception d'un système de conditionnement d'air d'une cabine d'avion**

Tous ces cas sont représentatifs des processus complexes nécessaires pour l'analyse multidisciplinaire de systèmes complexes. Ces cas applicatifs servent de support à la mise en place des nouvelles méthodes et à l'amélioration des processus. En particulier, nous traitons dans ce travail seulement le quatrième cas test. L'objectif de ce dernier cas test est de fournir un environnement d'air sécurisé et confortable dans une cabine d'avion. La conception d'un système ECS est donc importante. Pour répondre à cet objectif, il faut que les spécialistes dans différents domaines

travaillent ensemble. Prenons par exemple, pour trouver un confort thermique, on a besoin d'un concepteur des équipements pour le système ECS, un spécialiste de mécanique de fluide, un spécialiste de confort thermique et un concepteur général pour travailler dans un espace collaboratif (voir 1.2).

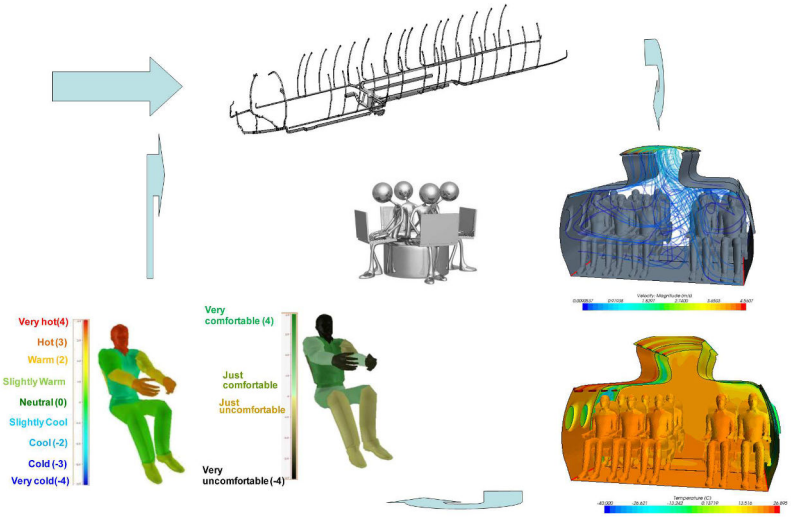

FIGURE 1.2 – Espace collaboratif pour la conception d'un système ECS (issues de [Davoodi et Greig, 2011, Chen, 2012])

Un challenge pour les concepteurs est le couplage entre des différentes disciplines. Les sorties d'une discipline servent d'entrées pour d'autres disciplines, et des dépendances indirectes entre critères contradictoires. Une partie importante de la conception est l'optimisation, ce problème est difficile pour chaque discipline, mais plus difficile pour une optimisation globale, puis que la performance d'une discipline allant à l'encontre de la performance dans une autre discipline. En plus, chaque spécialiste a besoin du temps de calcul ou de décision différents (par exemple, un ingénieur CFD demande une heure pour terminer un calcul du champ de température dans une cabine, alors qu'un spécialiste d'analyse de confort thermique prendre quelques seconds). Pour ces raisons, un modèle réduit pour chaque discipline est nécessaire.

En particulier, un modèle réduit CFD est le plus difficile mais aussi plus important. Ce modèle doit :

- être représentatif pour toute solution dépendant des paramètres de conceptions ;
- être possible d'enrichir quand on ajoute ou élargit l'espace de paramètres ;
- contrôler l'erreur de prédiction ;

– donner instantanément la solution ;
– construire une base de données portable.

La mise en place du modèle CFD ainsi que son modèle réduit seront présentés dans le chapitre 4.

1.6 Différents usages d'un modèle d'ordre réduit

1.6.1 Visualisation en ligne des champs de haute dimension

La visualisation joue un rôle important dans la plupart des domaines du calcul scientifique. En effet, l'analyse des données produites par les simulations numériques n'est généralement pas envisageable sans l'aide d'un outil de visualisation. Cet outil permet les utilisateurs de visualiser par exemple la solution d'un système d'EDP dans une salle de réalité virtuelle et d'interagir avec le calcul en temps réel ; ou de visualiser le mouvement des objets. Ceci ne peut être possible que si une approximation rapide de la solution physique est construite. La méthode de réduction de modèle est un parmi des techniques qui permettent de calculer en temps réel et visualiser les solutions physiques.

Les premières applications de la méthode de réduction de modèle pour la simulation des objets déformables dans le domaine infographie sont étudiées sur les objets éléments finis de déformation linéaire [Pentland et Williams, 1989, Kris K. Hauser, 2003]. D'ailleurs [Barbič et Popović, 2008, Barbič *et al.*, 2012] utilisent aussi la technique de réduction d'ordre pour accélérer leurs simulations afin d'animer le mouvement d'une structure déformable non linéaire. Ils testent sur plusieurs structures comme le pont, le rideau, le dragon, la mouette ou le fluide. Par exemple, pour visualiser le mouvement d'un pont après un accident, les auteurs imposent 240 déplacements différents (voir la figure 1.3) puis calculent la dynamique de pont par le modèle de référence. Les solutions obtenues sont utilisées pour construire un espace réduit. L'équation décrivant la dynamique de pont est projetée sur cet espace réduit. La déformation du pont est donc calculée rapidement. L'utilisateur peut interagir avec le pont en appliquant une force quelconque sur un point de structure, puis visualiser le mouvement du pont. [Riskin *et al.*, 2008, Chen *et al.*, 2009] a utilisé la technique POD pour visualiser le vol d'une chauve-souris. [James et Fatahalian, 2003] utilisent la technique ACP, non seulement pour calculer le déplacement des objets, mais aussi pour calculer le rendu des objets. Ils valident leur méthode sur le mouvement d'un dinosaure modélisé par le logiciel éléments finis ABAQUS avec 6499 éléments. Une fois le modèle réduit est construit, la vitesse de visualisation est environ 82 images par seconde.

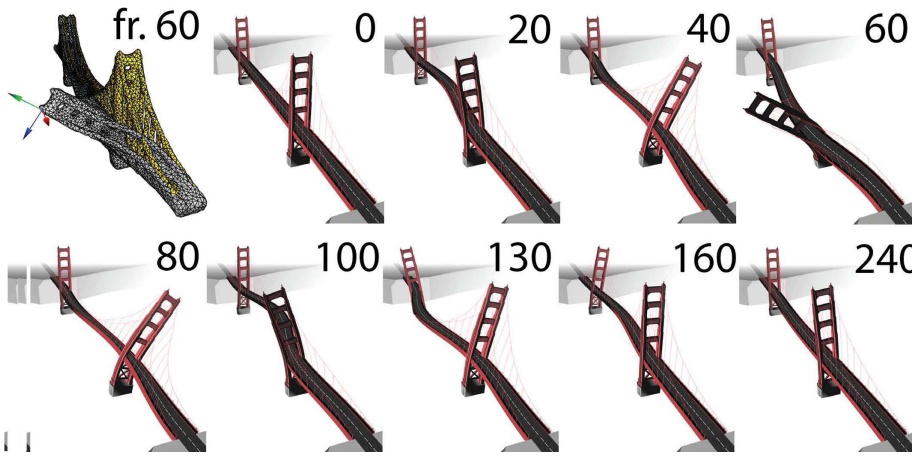

FIGURE 1.3 – Méthode pour l'édition interactive et la conception des animations d'objets déformables utilisant la technique POD [Barbič *et al.*, 2012]

1.6.2 Optimisation et conception optimale

L'optimisation et conception optimale est un outil fondamental dans tous les domaines de calcul scientifique et d'ingénierie. Le but est de maximiser ou de minimiser une fonction de coût sous quelques contraints à obtenir des variables optimales. Les variables sont des paramètres de conception, des variables internes de l'équation gouvernante ou des paramètres aux bords.

Dans le contexte industriel, il y a une grande quantité de problèmes d'optimisation, surtout le procès de conception de forme optimale pour un profil aérodynamique. Des problèmes classiques sont des optimisations d'un profil d'aile d'avion, de véhicule ou de coque de bateau dans un écoulement. Par exemple, pour un avion, on cherche le profil optimal d'aile d'avion pour augmenter sa portance en diminuant sa traînée aérodynamique.

Pour résoudre ces problèmes, on doit prendre en compte tous les phénomènes physiques de différentes disciplines qui sont généralement couplées. Cependant, le couplage entre les disciplines augmente significativement la complexité de la simulation numérique, tout en mobilisant des ressources importantes en mémoire et en temps de calcul.

Définition du problème couplé :

Lorsque les paramètres d'entrée et de sortie de plusieurs modèles sont indépendants, l'analyse de leur couplage doit être investiguée avec soin. Dans le cas de deux modèles M_1 et M_2, deux cas sont à considérer :

– Les modèles fortement couplés : les grandeurs et fonctions de couplages sont

exprimées dans un système unique :

$$\begin{cases} v_1 = M_1 \\ v_2 = M_2 \end{cases} ; \tag{1.1}$$

– Les modèles faiblement couplés : les modèles sont traités séparément, et interagissent uniquement par l'intermédiaire des paramètres de couplage.

Dans la figure 1.4, les paramètres v_{ij} indiquent les variables de sortie du modèle M_i utilisées comme entrées dans le modèle M_j. La cohérence des paramètres de couplage implique que l'ensemble (v_{12},v_{21}) est un point fixe, exprimant donc formellement que les conditions suivantes sont satisfaites :

$$v_{12} = M_1(M_2(v_{12})); \tag{1.2}$$
$$v_{21} = M_2(M_1(v_{21})). \tag{1.3}$$

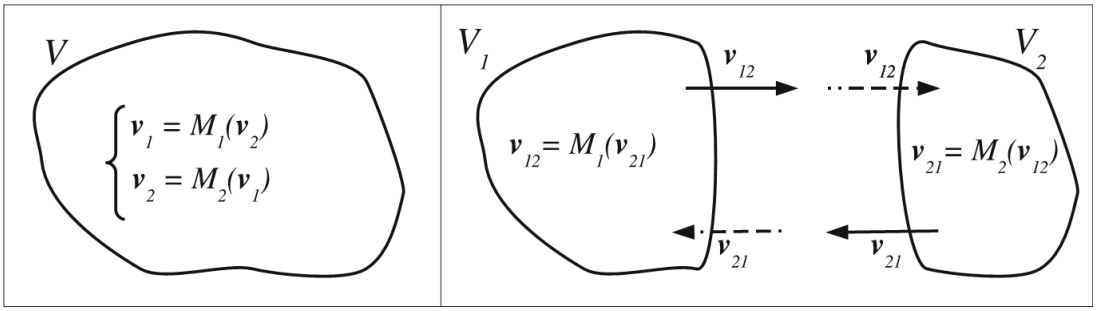

FIGURE 1.4 – Modèles M_1 et M_2 fortement couplés (à gauche) et faiblement couplés (à droite) [Filomeno Coelho *et al.*, 2009]

La première approche semble plus satisfaisante sur le plan mathématique, elle est souvent difficile à mettre en oeuvre parce que les modèles numériques s'appuient sur des logiciels commerciaux pour lesquels une approche intrusive est exclue. Dans ce cas, l'utilisation de modèles faiblement couplés constitue le seul recours. Cependant, trois aspects doivent être considérés :

– bien que les paramètres de couplage soient généralement des grandeurs physiques définies sur une interface commune $S_{interface}$ (voir figure 1.5). Ces paramètres sont souvent exprimés sur des maillages non conformes, voire sur des géométries non coïncidentes. Dans ce cas, un opérateur additionnel I_{ij} doit être introduit ;
– une grande quantité de données doit être échangée entre modèles, ce qui peut ralentir drastiquement l'analyse du système couplé. Pour pallier cet inconvénient, les stratégies de réduction sont indispensables ;

– étant donné que les réponses finales des modèles requises pour l'optimisation consistent habituellement en un petit nombre de valeurs (scalaires) issues d'un post-traitement, les métamodèles ou modèles approchés constituent souvent une solution économique, en particulier dans un processus d'optimisation coûteux en évaluation de fonctions.

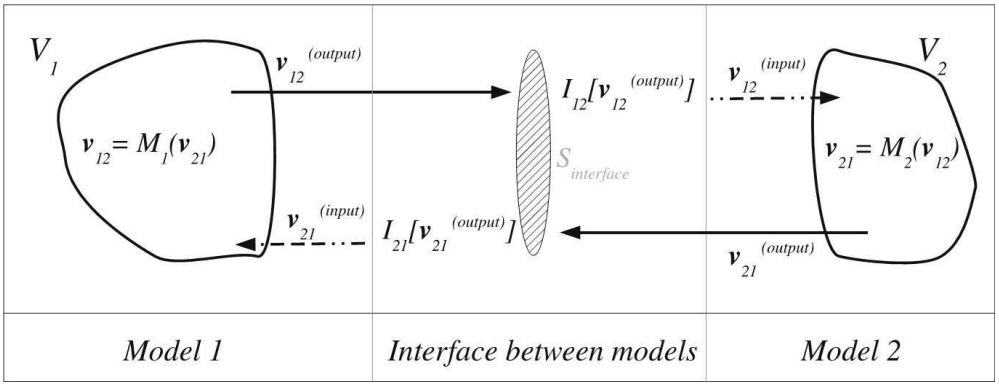

FIGURE 1.5 – Modèles M_1 et M_2 faiblement couplés par l'interface commune [Filomeno Coelho *et al.*, 2009]

Optimisation de problèmes multidisciplinaires :

On propose dans la littérature des stratégies qui se distinguent par leur gestion des paramètres de conception et de couplage. On les sépare généralement en deux catégories : méthodes mononiveaux et multiniveaux. Dans les méthodes à un niveau, une personne gère toutes les variables. Les techniques utilisées le plus couramment sont les méthodes MDF (*Multiple Discipline Feasible ou All-in-One*), IDF (*Individual Discipline Feasible*), et SAND (*Simultaneous ANalysis and Design ou All-at-Once*). Ces méthodes peuvent être délicates d'emploi puisqu'elles requièrent une gestion simultanée de toutes les disciplines. Donc, les stratégies multiniveaux sont proposées en séparant chaque jeu de paramètres en un niveau global et un niveau local (pour une discipline). Les méthodes le plus connues de cette approche sont BLISS (*Bi-Level Integrated System Synthesis*), CO (*Collaborative Optimization*) et DIVE (*Disciplinary Interaction Variable Elimination*).

Cependant, au-delà de leurs spécificités, ces méthodes doivent faire face à un double enjeu pour être applicables sur des cas industriels :

– les interconnexions entre les disciplines doivent être réduites.
– le nombre de simulations numériques de haute fidélité doit être le plus faible possible.

L'utilisation de modèles réduits permet de résoudre ces problèmes. On distingue deux catégories de méthode de réduction de modèle, une est la famille de métamodèles

généralistes (par exemple *krigeage, surfaces de réponse pronominales, réseaux de neurones, réseaux de fonctions à base radiale, moindres carrés mobiles, etc*), l'autre est la famille de technique de réduction en tenant compte de la physique (par exemple POD-*Proper Orthogonal Decomposition*).

Deux catégories de réduction de modèles en optimisation :

Dans la méthode d'optimisation multiniveaux BLISS, des modèles approchés par surfaces de réponse pronominales [Kodiyalam *et al.*, 2000, Park *et al.*, 2009] ou par krigeage [Agte, 2005] sont utilisés pour modéliser le comportement de chaque discipline au niveau global. D'autres métamodèles concernant la technique *moindres carrés mobiles* ont été employés en optimisation collaborative [Zadeh *et al.*, 2009] pour construire des approximations des contraintes multidisciplinaires, et plus particulièrement pour déterminer la région admissible de l'espace de conception.

Puis que les métamodèles généralistes ne concernent pas les informations physiques des problèmes étudiés. Récemment, les chercheurs s'intéressent à utiliser la deuxième catégorie de modèle réduit qui contient les informations physiques. On peut citer la méthode POD ; elle a été intégrée avec succès dans la méthode BLISS pour la conception de profils d'ailes [LeGresley, 2005]. Pour la même application, alors que [Filomeno Coelho *et al.*, 2008, Filomeno Coelho *et al.*, 2009] proposent la méthode à deux niveaux (*bi-level*) combinant la méthode POD avec la technique *moindres carrés mobiles* ; [Lassila et Rozza, 2010] utilisent la méthode *interpolation empirique* (ou RB-*Reduced Basis*). Cette dernière est aussi appliquée avec succès pour trouver la forme optimale d'une anastomose basée sur la minimisation de tourbillons dans l'artère coronaire [Manzoni *et al.*, 2012].

Chapitre 2

État de l'art de réduction de modèle

Sommaire

Dans ce chapitre nous présentons deux catégories de méthode de réduction de modèle. La première se compose des modèles réduits sans physique sous-jacente ou "métamodèles" (*surrogate models* en anglais). Ces modèles réduits regroupent les surfaces de réponses polynomiales, les moindres carrés mobiles, les réseaux de neurones, les réseaux de fonctions à base radiale, krigeage, etc. La deuxième catégorie comprend des modèles qui tiennent compte de termes physiques, appelés les méthodes de réduction d'ordre (Reduced Order Modeling ou ROM en anglais) qui s'appuient sur des calculs par éléments finis ou par volumes finis. Plus particulièrement, nous présentons le métamodèle de krigeage et quelques méthodes de réduction d'ordre, notamment la technique de POD-Galerkin, Bases Réduites, et Proper Generalized Decomposition (PGD).

2.1 Métamodèle de krigeage

Cette méthode provient des travaux des géostatisticiens. On cherche à prédire la distribution spatiale de minerais et de pétroles à l'aide de données fournies par un nombre limité de forages. Il faut aussi planifier le plus efficacement possible la situation des forages. Elle a été tout d'abord produite dans ce contexte précis par Krige, ingénieur minier sud-africain qui donne son nom à la méthode. Celle-ci a été ensuite fondée mathématiquement et étendue grâce aux travaux indépendants de [Matheron, 1963] et de [Gandin, 1963]. L'histoire de l'origine du krigeage est discutée par [Cressie, 1990].

La méthode est maintenant très populaire pour la construction de modèles réduits sur la base de simulations numériques [Simpson *et al.*, 2001]. En effet, par opposition aux méthodes d'approximation et de régression, elle fournit une technique d'interpolation répondant aux besoins des utilisateurs de simulations numériques qui font confiance aux données calculées par des modèles physiques détaillés à précision contrôlée.

L'idée de base du krigeage est de prédire la valeur de la variable régionalisée étudiée ($y(x)$) en un site non échantillonné (x) par une combinaison linéaire de m observations ponctuelles adjacentes $y(x_i)$, $i = 1, .., m$. L'estimation de variable $y(x)$ s'écrit :

$$\tilde{y}(x) = \sum_{i=1}^{m} \lambda_i(x) y(x_i). \tag{2.1}$$

où les poids λ_i sont déterminés de manière à minimiser la variance d'estimation résultante, en prenant en compte l'emplacement des points observés.

Il existe plusieurs types de krigeage dans la littérature [Cressie, 1993]. En général,

le krigeage s'écrit sous forme :

$$\tilde{y}(x) = \mu(x) + \delta(x),\ \forall x \in \Omega. \tag{2.2}$$

où $\mu(.)$ est la structure déterministe pour l'espérance de y(.) ; $\delta(.)$ est une fonction aléatoire stationnaire d'espérance nulle et de structure de dépendance connue. Pour formuler complètement le modèle, il faut spécifier la forme de la tendance $\mu(.)$. C'est en fait cette tendance qui précise le type de krigeage effectué. Les trois types classiques de krigeage sont :

- le *krigeage simple* si $\mu(x) = m$ est une constante connue,
- le *krigeage ordinaire* si $\mu(x) = \mu$ est une constante inconnue,
- le *krigeage universel* si $\mu(x) = \sum_{i=1}^{p} f_i(x)\beta_i$ est une combinaison linéaire de p fonctions de la position x.

En pratique, nous utiliserons DACE (Design and Analysis of Computer Experiments) [Lophaven *et al.*, 2002], une boîte à outils de Matlab développé à l'université technique du Danemark pour résoudre en krigeage. L'algorithme krigeage dans cet outil se base sur le krigeage universel. Nous présentons dans la suite quelques éléments de base sur la méthode krigeage universel dans le contexte de l'outil DACE.

2.1.1 Principe de krigeage universel

On réalise un plan d'expériences numériques qui consiste en un ensemble d'entrées X et de sorties correspondantes Y :

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_m^T \end{bmatrix}_{m \times n} \text{ avec } x_i \in \mathbb{R}^n; \qquad Y = \begin{bmatrix} y_1^T \\ \vdots \\ y_m^T \end{bmatrix}_{m \times q} \text{ avec } y_i \in \mathbb{R}^q. \tag{2.3}$$

À partir de ce plan d'expériences, un métamodèle permet de trouver une relation approchée de :

$$\begin{aligned} y(x) \quad &: \quad \mathbb{R}^n \to \mathbb{R}^q \\ &\qquad x \mapsto y(x) \end{aligned}$$

Pour simplifier, on suppose que la réponse y est un scalaire (c'est-à-dire $q = 1$). Le modèle krigeage s'écrit :

$$\tilde{y}(x) = \mathcal{F}(\beta, x) + z(x),\ \forall x \in \Omega. \tag{2.4}$$

où :

– $\mathcal{F}(\beta, x)$ est un modèle de régression étant la combinaison linéaire de p fonctions connues $f_i : \mathbb{R}^n \to \mathbb{R}$,

$$\mathcal{F}(\beta, x) = \sum_{i=1}^{p} f_i(x)\beta_i = \beta^T f(x). \tag{2.5}$$

Le vecteur de coefficients β est un paramètre de régression déterminé par la méthode des moindres carrés pondérés.

– $z(x)$ est un processus aléatoire ayant une espérance nulle et une covariance

$$cov[z(w), z(x)] = E[z(w)z(x)] = \sigma^2 \mathcal{R}(\theta, \omega, x). \tag{2.6}$$

σ est la variance du processus pour la réponse y ; $\mathcal{R}(\theta, \omega, x)$ est le modèle de corrélation de ω et de x avec le paramètre θ

On peut interpréter le krigeage universel (2.4) par deux composants. Le premier désigne une approximation globale déterminée par une régression de types moindres carrés. Le deuxième est une approximation locale qui assure que le krigeage est juste pour les points échantillonnés (estimateur sans biais). Ces deux parties interagissent l'une avec l'autre. La figure 2.1 présente un exemple unidimensionnel de modèle krigeage universel, dont la régression $\mu(x)$ est de l'ordre 2, et l'approximation locale $\delta(x)$ est le processus gaussien.

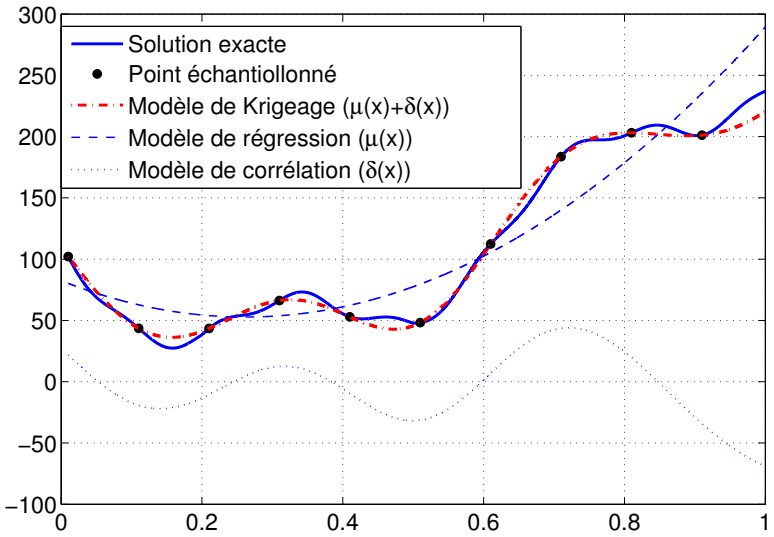

FIGURE 2.1 – Modèle de krigeage universel en 1D.

Dérivation de krigeage universel :

Supposons qu'on connaît les fonction $f(x)$ et $\mathcal{R}(\theta, \omega, x)$. On cherche β puis on

en déduit $z(x)$.

Notons $\mathbf{F} = \begin{bmatrix} f_1(x_1) & \cdots & f_p(x_1) \\ \vdots & \ddots & \vdots \\ f_1(x_m) & \cdots & f_p(x_m) \end{bmatrix}_{m\times p}$ et $\mathbf{R} = \begin{bmatrix} \mathcal{R}(\theta, x_1, x_1) & \cdots & \mathcal{R}(\theta, x_1, x_m) \\ \vdots & \ddots & \vdots \\ \mathcal{R}(\theta, x_m, x_1) & \cdots & \mathcal{R}(\theta, x_m, x_m) \end{bmatrix}_{m\times m}$

Pour x quelconque, on définit un vecteur de corrélation entre x et les points échantillonnés x_i :

$$r(x) = \begin{bmatrix} \mathcal{R}(\theta, x_1, x) \\ \vdots \\ \mathcal{R}(\theta, x_m, x) \end{bmatrix} \tag{2.7}$$

Le vecteur de régression β est déterminé par la méthode des moindres carrés pondérés pour satisfaire $\mathcal{F}(\beta, x) \simeq y(x)$:

$$\beta^* = (\mathbf{F}^T\mathbf{R}^{-1}\mathbf{F})^{-1}\mathbf{F}^T\mathbf{R}^{-1}Y. \tag{2.8}$$

On obtient ensuite l'estimateur du maximum de vraisemblance de variance :

$$\sigma^2 = \frac{1}{m}(Y - \mathbf{F}\beta^*)^T\mathbf{R}^{-1}(Y - \mathbf{F}\beta^*) \tag{2.9}$$

On considère une prédiction linéaire :

$$\tilde{y}(x) = c^T Y, \tag{2.10}$$

où $c = c(x) \in \mathbb{R}^m$ est la fonction à chercher telle que l'approximation de krigeage est sans biais, et que la variance de son erreur est minimum.

L'erreur entre le prédicteur linéaire et y est

$$\begin{aligned} \tilde{y}(x) - y(x) &= c^T Y - y(x) \\ &= c^T(\mathbf{F}\beta + Z) - (f(x)^T\beta + z) \\ &= c^T Z - z + (\mathbf{F}^T c - f(x))^T\beta \end{aligned}$$

où $Z = [z_1 ... z_m]^T$ est l'erreur du modèle de régression aux points échantillonnés. Pour que le krigeage soit un estimateur sans biais, on impose $(\mathbf{F}^T c - f(x))^T = 0$. Sous cette condition, l'erreur MSE (Mean Square Error) s'écrit :

$$\begin{aligned} \phi(x) &= \mathbf{E}[(\tilde{y}(x) - y(x))^2] \\ &= \mathbf{E}[(c^T Z - z)^2] \\ &= \mathbf{E}[z^2 + c^T Z Z^T c - 2c^T Z z] \\ &= \sigma^2(1 + c^T\mathbf{R}c - 2c^T r) \end{aligned} \tag{2.11}$$

On résout un problème de minimisation sous contrainte en appliquant la formulation lagrangienne :

$$L(c, \lambda) = \sigma^2(1 + c^T \mathbf{R} c - 2c^T r) - \lambda^T(\mathbf{F}^T c - f) \tag{2.12}$$

On obtient :

$$\tilde{y}(x) = r^T \mathbf{R}^{-1} Y - (\mathbf{F}^T \mathbf{R}^{-1} r - f)^T \underbrace{(\mathbf{F}^T \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^T \mathbf{R}^{-1} Y}_{\beta^*} \tag{2.13}$$

Finalement,

$$\tilde{y}(x) = f(x)^T \beta^* + r(x)^T \gamma^*. \tag{2.14}$$

où $\gamma^* = \mathbf{R}^{-1}(Y - \mathbf{F}\beta^*)$.

Pour le cas multi-dimensionnel ($q > 1$), la formule (2.14) est toujours applicable avec $\beta^*, \gamma^* \in \mathbb{R}^{p \times q}$.

Notons que, pour un plan d'expériences fixé, les matrices β^*, γ^* sont aussi fixées. L'approximation $\tilde{y}(x)$ de x quelconque peut être déterminée par la seule évaluation du vecteur $f(x) \in \mathbb{R}^p$ et $r(x) \in \mathbb{R}^m$ et deux opérations multiplicatives.

2.1.2 Modèle de régression

Selon l'évolution globale de la réponse, on peut choisir le modèle de régression de type polynôme avec l'ordre 0, 1 ou 2. Particulièrement, si l'on note $x(j)$ le $j$ème composant de x et n la dimension de x, on a une régression :

- ***constante***, $p = 1$:

$$f_1(x) = 1, \tag{2.15}$$

- ***linéaire***, $p = n + 1$:

$$f_1(x) = 1,\ f_2(x) = x(1), ...,\ f_{n+1}(x) = x(n), \tag{2.16}$$

- ***quadratique***, $p = \frac{1}{2}(n + 1)(n + 2)$:

$$\begin{aligned} &f_1(x) = 1 \\ &f_2(x) = x(1), ..., f_{n+1}(x) = x(n) \\ &... \\ &f_p(x) = x(n)^2. \end{aligned}$$

2.1.3 Modèle de corrélation

Nous considérons seulement des modèles de corrélation stationnaires, qui dépendent du paramètre θ et de la distance $d = \omega - x$. Les modèles s'écrivent sous la

forme :

$$\mathcal{R}(\theta, \omega, x) = \prod_{j=1}^{n} \mathcal{R}_j(\theta, \omega(j) - x(j)) \tag{2.17}$$

TABLE 2.1 – Fonctions de corrélation $\mathcal{R}_j(\theta, |\omega(j) - x(j)|)$ (Notons $\xi_j = \theta(j)|\omega(j) - x(j)|$).

exponentielle	$\exp(-\theta(j)\|\omega(j) - x(j)\|)$
gaussienne	$\exp(-\theta(j)\|\omega(j) - x(j)\|^2)$
linéaire	$\max\{0, 1 - \theta(j)\|\omega(j) - x(j)\|\}$
sphérique	$1 - 1.5\xi_j + 0.5\xi_j^3$, $\xi_j = min\{1, \theta(j)\|\omega(j) - x(j)\|\}$
cubique	$1 - 3\xi_j + 2\xi_j^3$, $\xi_j = min\{1, \theta(j)\|\omega(j) - x(j)\|\}$
spline	$\begin{cases} 1 - 15\xi_j + 30\xi_j^3 & \text{pour } 0 \leq \xi_j \leq 0.2 \\ 1.25(1 - \xi_j)^3 & \text{pour } 0.2 \leq \xi_j \leq 1 \\ 0 & \text{pour } \xi_j \geq 1 \end{cases}$

Le paramètre inconnu θ est estimé grâce au plan d'expériences en utilisant l'estimateur du maximum de vraisemblance, la méthode validation croisée ou la technique de variogram [Rasmussen et Williams, 2006, Stein, 1999, Cressie, 1993]. Particulièrement, l'outil DACE utilise la méthode proposée dans [Sacks *et al.*, 1989]. Le paramètre optimal est obtenu par une minimisation de :

$$\psi(\theta) = det(\mathbf{R})^{1/m}.\sigma(\theta)^2. \tag{2.18}$$

Revenons à l'exemple unidimensionnel pour le krigeage universel, on peut trouver dans la figure 2.2 ci-après que le paramètre θ joue un rôle important pour l'efficacité de krigeage. Voir [Lophaven *et al.*, 2002] pour plus d'informations concernant les propriétés des différents modèles de corrélation et le problème d'optimisation ci-dessus.

2.2 Méthodes de réduction d'ordre

Dans la suite, nous nous intéresserons aux méthodes de réduction d'ordre utilisées en mécanique de fluides. Parmi ces dernières, on peut citer la CVT (*centroidal Voronoi tessellation*) [Du *et al.*, 1999, Burkardt *et al.*, 2006] et la POD (Proper Orthogonal Decomposition) [Lumley, 1967]. Dans le cadre de cette thèse, nous nous intéressons en particulier à la méthode POD.

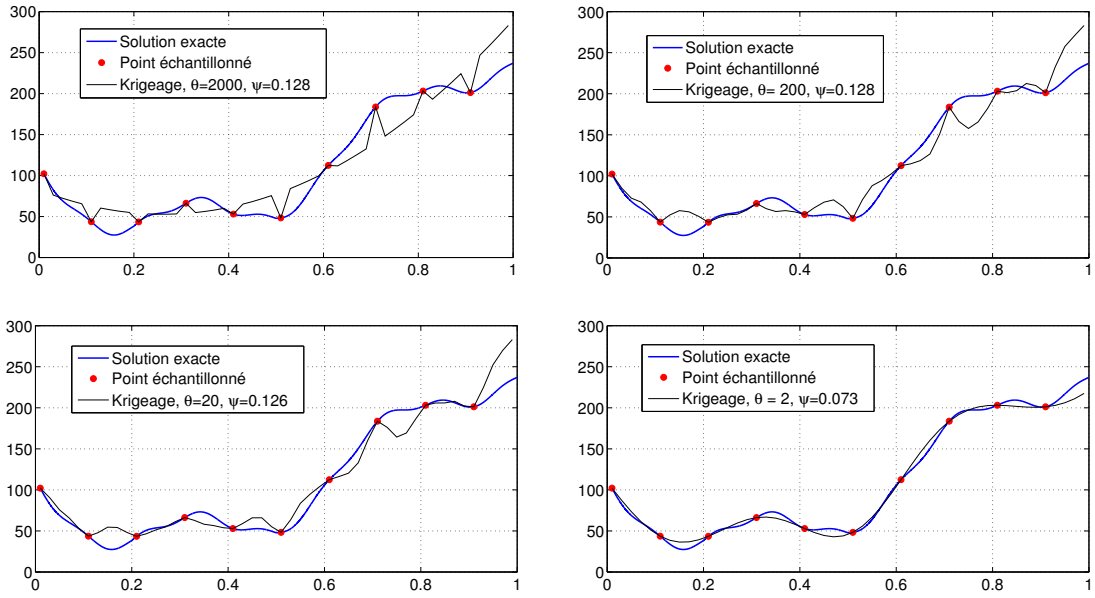

FIGURE 2.2 – Modèle Krigeage avec des différents paramètres de corrélation.

2.2.1 Décomposition orthogonale aux valeurs propres (POD)

La méthode POD est utilisée dans différents domaines. On la connaît sous différents noms : Analyse en Composantes Principales (ACP), Décomposition de Karhunen-Loève (KLD) ou Décomposition en Valeurs Singulières (SVD). La méthode POD a été présentée pour la première fois par [Lumley, 1967], puis largement utilisée dans le contexte des écoulements turbulents. D'autres applications existent dans le domaine du traitement des images [ROSENFELD, 1982, Kirby et Sirovich, 1990], de l'analyse des signaux [Algazi et Sakrison, 1969] ou de la compression des données [Andrews *et al.*, 1967], etc.

Supposons qu'on veut approcher un champ scalaire $\boldsymbol{u}(\theta, x)$ (qui vérifie un système d'équations) sur le domaine $D = [0,1]^p \times \Omega$ par une combinaison linéaire finie de fonctions de base $\boldsymbol{\Psi}^k(x)$.

$$\boldsymbol{u}(\theta, x) \simeq \sum_{k=1}^{K} a^k(\theta)\boldsymbol{\Psi}^k(x) \tag{2.19}$$

où x est une coordonnée spatiale, θ un paramètre ou coordonnée temporelle pour des problèmes dépendant du temps. On suppose que l'approche devient exacte si $K \to +\infty$.

La formule 2.19 n'est pas unique et dépend de la nature des fonctions $\boldsymbol{\Psi}^k(x)$. Il existe des méthodes classiques pour résoudre ce problème d'approximation en utilisant les fonctions de base *a priori* comme les séries de Fourier, les polynômes

de Legendre, les polynômes de Tchebychev, etc. Une autre méthode alternative, la décomposition orthogonale aux valeurs propres, permet de trouver des fonctions de base adaptées au problème. Les solutions $\boldsymbol{u}(\theta, x)$ appartiennent à un espace de Hilbert muni d'un produit scalaire $(.,.)$. Ces fonctions sont appelées modes POD et forment un système orthonormé :

$$(\boldsymbol{\Psi}^k(x), \boldsymbol{\Psi}^l(x)) = \delta_{kl}. \tag{2.20}$$

où $\delta_{kl} = \begin{cases} 1 & \text{si } k = l \\ 0 & \text{si } k \neq l \end{cases}$ est le symbole de Kronecker.

L'approximation définie par 2.19 est la projection de $\boldsymbol{u}(\theta, x)$ sur l'espace engendré par les modes $\boldsymbol{\Psi}^k, k = 1...K$. Dans ce cas les coefficients POD $a^k(\theta)$ sont les coordonnées de la projection de $\boldsymbol{u}(\theta, .)$ sur cet espace. Ces coefficients sont calculés par la projection de $\boldsymbol{u}(\theta, .)$ sur chaque mode POD, c'est-à-dire :

$$a^k(\theta) = \int_\Omega \boldsymbol{u}(\theta, x)\boldsymbol{\Psi}^k(x)dx. \tag{2.21}$$

En supposant qu'on connaît N solutions $\boldsymbol{u}(\theta^i, x)$ (par mesures expérimentales ou par calcul numérique *a priori*) pour différents paramètres θ^i, $i = 1...N$, l'approximation POD (2.19) équivaut à chercher des fonctions de base pour que la moyenne de N erreurs de projections au carré soit minimale.

Les tutoriels de cette méthode sont bien présentés par [Chatterjee, 2000] et [Cordier L., 2003]. Concernant les travaux pionniers, on pourra se reporter à [Lumley, 1967], [Holmes, 1990, P. Holmes, 1996] ou [Sirovich, 1987, Sirovich, 1989, Kirby et Sirovich, 1990]. Par ailleurs, [LIANG *et al.*, 2002] présentent trois méthodes différentes pour calculer des modes POD : KLD, SVD, et ACP.

Dans cette section, la méthode des clichés [Sirovich, 1987] pour le calcul des modes POD dans le cas discret est présentée. Elle utilise le théorème de décomposition en valeurs singulières (SVD) [Ciarlet, 1990]. Dans le cas discret, les solutions $\boldsymbol{u}(\theta^i, x)$ sont représentées par des champs d'éléments finis sous forme de vecteurs. Dans ce cadre, l'espace de Hilbert des solutions est un espace vectoriel euclidien $\mathbb{R}^d$, muni du produit scalaire $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \boldsymbol{u}^T\boldsymbol{v}$. On notera $\| \boldsymbol{u} \| = \sqrt{\langle \boldsymbol{u}, \boldsymbol{u} \rangle}$ la norme euclidienne associée.

On considère un ensemble $\mathcal{S}^{N_{exp}}$ de N_{exp} données vectorielles dans $\mathbb{R}^d$ correspondant à des champs de solutions d'un système d'EDP. Chaque solution $\boldsymbol{u}(\theta^i)$ est calculée par le code éléments finis pour un jeu de paramètres θ^i :

$$\mathcal{S}^{N_{exp}} = \left\{ \boldsymbol{u}^i = \boldsymbol{u}(\theta^i, x),\ i = 1, \ldots, N_{exp} \right\}. \tag{2.22}$$

Les modes POD seront les solutions du problème d'optimisation suivant : *étant*

donné un entier $K << N_{exp}$, trouver la famille de vecteurs $(\mathbf{\Psi}^k(x))_{k=1,...,K}$ qui réalise le minimum d'erreur de projection :

$$\min_{\substack{(\mathbf{\Psi}^1,...,\mathbf{\Psi}^K) \\ \langle \mathbf{\Psi}^k, \mathbf{\Psi}^\ell \rangle = \delta_{k\ell},\ 1 \leq k \leq \ell \leq K}} \frac{1}{2} \sum_{i=1}^{N_{exp}} \left\| \boldsymbol{u}^i - \sum_{k=1}^{K} \langle \boldsymbol{u}^i, \mathbf{\Psi}^k \rangle \, \mathbf{\Psi}^k \right\|^2 . \tag{2.23}$$

En supposant que les modes POD soient orthonormés, le problème 2.23 équivaut à rechercher :

$$\max_{\substack{(\mathbf{\Psi}^1,...,\mathbf{\Psi}^K) \\ \langle \mathbf{\Psi}^k, \mathbf{\Psi}^\ell \rangle = \delta_{k\ell}}} \frac{1}{2} \sum_{i=1}^{N_{exp}} \sum_{k=1}^{K} \langle \boldsymbol{u}^i, \mathbf{\Psi}^k \rangle^2 \tag{2.24}$$

On notera $\mathbf{X} \in \mathbf{M}_{dN_{exp}}$ la matrice constituée des vecteurs $\boldsymbol{u}^i$ en colonnes, c'est-à-dire :

$$\mathbf{X} = \begin{pmatrix} \boldsymbol{u}_1^1 & \boldsymbol{u}_1^2 & \cdots & \boldsymbol{u}_1^{N_{exp}} \\ \boldsymbol{u}_2^1 & \boldsymbol{u}_2^2 & \cdots & \boldsymbol{u}_2^{N_{exp}} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \boldsymbol{u}_d^1 & \boldsymbol{u}_d^2 & \cdots & \boldsymbol{u}_d^{N_{exp}} \end{pmatrix}$$

[Chatterjee, 2000] a montré que les modes POD sont exactement les K premiers vecteurs propres (ou K premières composantes principales en statistique) de la matrice $A = \mathbf{X}\mathbf{X}^T$, où les colonnes de la matrice $\mathbf{X}$ représentent les solutions $\boldsymbol{u}^i$. Pour des raisons de coût de calcul, ces vecteurs propres sont en fait dérivés des vecteurs propres υ^k et valeurs propres λ^k de la matrice $M = \mathbf{X}^T\mathbf{X}$ via la formule suivante (voir [VUYST et AUDOUZE, 2009] pour la démonstration) :

$$\mathbf{\Psi}^k = \frac{1}{\sqrt{\lambda^k}} \sum_{i=1}^{N_{exp}} (\upsilon^k)_i \boldsymbol{u}^i. \tag{2.25}$$

Il est important de noter que le rang de troncature K peut être calculé de façon à s'assurer que la projection de tous exemples $\boldsymbol{u}^i$ sur l'espace engendré par les vecteurs propres conduit à une erreur de projection inférieure à une certaine tolérance ϵ. Ainsi, pour tout critère de tolérance d'erreur $\epsilon > 0$, et en indexant les valeurs propres de sorte à avoir $\lambda_1 > \lambda_2 > ... > \lambda_{N_{exp}}$, le rang de troncature $K(\epsilon)$ est défini comme le plus petit entier tel que :

$$\frac{\sum_{k=K+1}^{N_{exp}} \lambda^k}{\sum_{k=1}^{N_{exp}} \lambda^k} \leq \epsilon. \tag{2.26}$$

Méthodologie de génération des modes POD :

1. Définir un ensemble de points θ^i, $i = 1, ..., N_{exp}$ dans l'espace de paramètres

$[0,1]^p$. On pourra pour cela utiliser un algorithme de remplissage d'espace standard tel que l'échantillonnage LHS (Latin Hypercube Sampling, voir annexe A). Calculer N_{exp} solutions éléments finis $\boldsymbol{u}^i = \boldsymbol{u}(\theta^i)$. Constituer l'ensemble de données :

$$\mathcal{S}_{exp}^N = \{\boldsymbol{u}^1, ..., \boldsymbol{u}_{exp}^N\}. \tag{2.27}$$

2. Calculer la matrice de corrélation $\mathbf{M}$ de taille $N_{exp} \times N_{exp}$, avec $M_{ij} = (\boldsymbol{u}^i, \boldsymbol{u}^j)_{H_0^1}$. Étant donné un critère d'erreur $\epsilon > 0$, calculer le rang de troncature $K = K(\epsilon)$ des modes POD à partir de (2.26).
3. Assembler les K modes POD orthonormés $\boldsymbol{\Psi}^1, ..., \boldsymbol{\Psi}^K$ à partir de (2.25).

Note : La norme pour définir POD joue aussi un rôle important. Certains travaux ont montré que la définition de POD dans l'espace Sobolev H^1 est meilleure que dans L^2 [Iollo *et al.*, 2000].

2.2.2 Réduction POD-Galerkin

La méthode POD-Galerkin est une combinaison entre la technique de décomposition orthogonale aux valeurs propres et la projection de Galerkin. Cette combinaison permet de réduire la dimension du modèle des systèmes dynamiques qui ont un espace de phase de dimension grande ou infinie. En principe, la POD-Galerkin équivaut à une application du principe variationnel de Galerkin en utilisant les modes POD comme des fonctions test.

Cette méthode est utilisée largement dans le domaine de la mécanique des fluides, en particulier pour résoudre l'équation de Navier-Stokes. La méthode de réduction de modèle en utilisant POD-Galerkin a été déjà étudiée avec succès pour le contrôle des écoulements instationnaires, bidimensionnels et laminaires. [Graham et Tang, 1999b, Graham et Tang, 1999a] ont mis en évidence l'intérêt mais aussi la difficulté qu'il y avait à définir des stratégies d'enrichissement des bases POD. [Graham et Tang, 2000a,Graham et Tang, 2000b] ont montré l'efficacité de cette méthode en comparant le résultat numérique avec la méthode des éléments finis ainsi que la méthode RB (Reduced Basis method). [Couplet *et al.*, 2005] ont proposé un algorithme robuste appliqué au contrôle des écoulements instationnaires à deux et trois dimensions. La recherche récente [Placzek *et al.*, 2011] élargit l'application de POD-Galerkin au problème d'écoulement compressible dans une structure déformée. En ce qui concerne l'estimation d'erreur ou la stabilité de POD-Galerkin, on peut consulter [Hinze et Volkwein, 2008, Iollo *et al.*, 2000, Galan delan Sastre et Bermejo, 2008, Ravindran, 2011, Kunisch et Volkwein, 2002].

La majorité des travaux récents ont abordé un problème d'évolution linéaire ou non-linéaire non paramétré. L'équation gouvernante est en général un système

d'EDP qui seront converties en EDO dont la variable est le temps (consulter par exemple [Rowley *et al.*, 2004], [Couplet, 2005], [NAGARAJAN, 2010] pour les détails). [Rambo et Joshi, 2007] ont abordé un problème paramétré dont les paramètres sont des conditions limites comme la température, la vitesse de l'air injecté, ou certaines propriétés de l'air, etc.

Dans cette section, une méthode POD-Galerkin pour le problème paramétré sera présentée. On notera :

- V : espace de Hilbert de fonctions régulières définies presque partout sur Ω^h et de produit scalaire $(.,.)_V$;
- V^h : espace éléments finis conforme sur V ;
- $\mathcal{N}$: un opérateur non linéaire agissant sur V.

On considère le problème variationnel non linéaire discret abstrait :
Trouver $\boldsymbol{u}^{\theta,h} \in V^h$ *telle que :*

$$(\mathcal{N}(\boldsymbol{u}^{\theta,h}), v^h)_V = (f^\theta, v^h)_{L^2(\Omega^h)} \quad , \forall v^h \in V^h \tag{2.28}$$

On suppose que le problème 2.28 admet une solution discrète unique dépendant continûment des paramètres θ. Ce problème sera résolu par une méthode des éléments finis. Le maillage est fin de sorte que la solution est considérée comme exacte.

Grâce à ce modèle éléments finis, on obtiendra un plan d'expérience en déterminant N solutions $\boldsymbol{u}^{i,h} = \boldsymbol{u}^{\theta^i,h}, i = 1, ..., N$. Puis, on calculera les fonctions de base POD $\boldsymbol{\Psi}^{1,h}, ..., \boldsymbol{\Psi}^{K,h}$ selon (2.25). La méthode POD-Galerkin consiste alors à résoudre le problème de plus petite dimension :
Trouver $\tilde{\boldsymbol{u}}$ *dans* $W^h = vect(\boldsymbol{\Psi}^{1,h}, ..., \boldsymbol{\Psi}^{K,h})$*, tel que :*

$$(\mathcal{N}(\tilde{\boldsymbol{u}}^{\theta,h}), w^h)_V = (f^\theta, w^h)_{L^2(\Omega^h)} \quad , \forall w^h \in W^h \tag{2.29}$$

où $\tilde{\boldsymbol{u}}^{\theta,h}$ s'écrit sous la forme :

$$\tilde{\boldsymbol{u}}(\theta, x) = \sum_{k=1}^{K} a^k(\theta) \boldsymbol{\Psi}^{k,h}(x) \tag{2.30}$$

En remplaçant $\tilde{\boldsymbol{u}}^{\theta,h}$ dans 2.29, et en choisissant K fonctions de test $w^h = \boldsymbol{\Psi}^{k,h} \in W^h (k = 1, ..., K)$, on obtiendra un système non linéaire à K équations de K variables $a^k(\theta)$:

$$\left(\mathcal{N}\Big(\sum_{k=1}^{K} a^k(\theta) \boldsymbol{\Psi}^{k,h}(x) \Big), \boldsymbol{\Psi}^{k,h} \right)_V = (f^\theta, \boldsymbol{\Psi}^{k,h})_{L^2(\Omega^h)} \quad , \forall k \in \{1, ..., K\} \tag{2.31}$$

Ou :

$$G_k(a(\theta)) = F_k(\theta), \quad \forall k \in \{1, ..., K\} \tag{2.32}$$

Si l'on note $\mathbf{G}(a(\theta))$ et $\mathbf{F}^\theta$, les vecteurs dans $\mathbb{R}^K$, ayant respectivement des composantes $G_k(a(\theta))$ et $F_k(\theta)$, l'équation (2.32) s'écrit sous la forme abstraite :

$$\mathbf{G}(a(\theta)) = \mathbf{F}^\theta, \tag{2.33}$$

Remarque :
Le schéma de POD-Galerkin est identique à la méthode des éléments finis en projetant la formulation variationnelle dans un espace de dimension finie. La différence entre les deux méthodes est que les fonctions de base qui constituent cet espace ne sont pas les mêmes. L'espace discret V^h de EF est engendré par les fonctions de forme : $V^h = vect(\Phi^{1,h}, ..., \Phi^{d,h})$ alors que W^h de POD-Galerkin est engendré par les modes POD : $W^h = vect(\mathbf{\Psi}^{1,h}, ..., \mathbf{\Psi}^{K,h}) \subset V^h$.

Remarquons que $K << d$, donc le coût de résolution de (2.33) de taille K est beaucoup moins cher qu'en utilisant le système d'équations de taille d dans la méthode des éléments finis.

La méthode POD-Galerkin présentée ci-dessus est appliquée au cas général (non linéaire, paramétré).

Approche intrusive, non-intrusive, semi-intrusive
Dans le cadre des projets industriels, on s'intéresse à distinguer les approches intrusives et non intrusives. En effet, pour l'analyse multidisciplinaire en phase d'avant-projet de conception, la méthodologie consiste à générer des modèles plus grossiers à partir de plan d'expériences obtenu par modèle EF fin. Les modèles EF fins sont souvent réalisés par des logiciels dont les utilisateurs ignorent le code écrit à l'intérieur (les EDPs, l'algorithme à résoudre...). Une approche pour construire un modèle grossier est intrusive si elle modifie le code EF. Au contraire, une approche non-intrusive utilise le code EF comme une boite noire. La deuxième méthode est donc intéressante dans le contexte industriel puisque les utilisateurs n'ont pas besoin de bien comprendre le code EF.

La méthode POD-Galerkin est une méthode intrusive puisqu'elle est basée sur une formulation variationnelle du problème aux dérivées partielles et par le choix d'un espace d'approximation particulier W^h engendré par les modes POD $\mathbf{\Psi}^{k,h}$.

Dans le chapitre suivant, nous proposons une méthode "semi-intrusive". C'est une méthode qui a besoin d'une petite modification de code EF pour l'évaluation de résidu à partir d'un champ "candidat" calculé en dehors du code EF. Si l'on suppose que le logiciel EF permet de consulter ce résidu, la méthode semi-intrusive devient non-intrusive.

2.2.3 Bases réduites et algorithme glouton ("greedy")

La méthode de bases réduites (RB) est une sorte de méthode de réduction de modèle, dont la méthode POD-Galerkin présentée ci-dessus est un cas particulier. La méthode RB a été proposée la première fois par [Almroth *et al.*, 1978] et [Nagy, 1979] pour l'analyse de structure. Elle est ensuite développée dans le contexte de mécanique de fluide [Peterson, 1989] et récemment synthétisée pour l'application d'EDP paramétré [Quarteroni *et al.*, 2011]. Le but général est de projeter l'équation gouvernante dans l'espace réduit sélectionné pour obtenir une équation avec une taille plus petite que celle de l'originale. L'espace réduit est constitué par une série finie de fonctions de base, qui peuvent être les bases réduites de Taylor, Larange, Hermite ou POD.

Dans cette section, nous nous intéressons à la méthode RB via estimation d'erreur *a posteriori* en utilisant l'algorithme glouton. Cette technique construit des bases réduites de manière adaptative et incrémentale. À partir d'une base réduite initiale, le paramètre dont l'approche est la pire, est identifié par une recherche complète (ou presque) dans l'espace de paramètres. La solution de référence du paramètre trouvé sera calculée par modèle fin puis elle sera utilisée pour enrichir les bases réduites pour l'itération suivante. Les itérations sont répétées jusqu'à ce que la précision de l'approche basée sur les bases réduites est trouvée. [Nair *et al.*, 2003] ont présenté un schéma numérique robuste de méthode glouton dans le contexte de régression avec une grande base de données existante. [Veroy *et al.*, 2003] ont introduit l'algorithme glouton dans le contexte de l'approximation par bases réduite (Reduced Basis method ou RB, voir [Rheinboldt, 1993]).

Dans le contexte de POD, [Bui-Thanh, 2007] a formulé l'approche glouton comme un problème d'optimisation sous contraintes sur l'équation gouvernante réduite. De même façon, [Audouze *et al.*, 2009] ont combiné leur modèle réduit de type POD avec l'algorithme glouton pour résoudre l'équation convection-réaction-diffusion. Cette méthode est aussi étendue au problème inverse par [Lieberman et Ghattas, 2010]. En particulier, [Haasdonk et Ohlberger, 2008] a proposé l'algorithme POD-Greedy en utilisant les modes POD comme des bases réduites. La convergence est étudiée dans le cas stationnaire paramétrique [Peter Binev et Wojtaszczyk, 2011] et révolutionnaire [Haasdonk, 2011].

L'algorithme glouton est appliqué à la construction d'un modèle réduit de l'équation Navier-Stokes incompressible stationnaire paramétrique [Veroy et Patera, 2005]. Il a été combiné avec l'estimation d'erreur 'a posteriori' pour l'EDP parabolique paramétrique et appliqué pour quelques problèmes de contrôle optimal, problème inverse [Grepl et Patera, 2005, Grepl, 2005].

Supposons que $\boldsymbol{u}(\theta)$ est la solution de problème EDP stationnaire paramétrique

résolue par le code d'éléments finis. Pour chaque θ^i donné on va trouver $\boldsymbol{u}^i = \boldsymbol{u}(\theta^i)$. La méthode RB permet de construire un sous-espace engendré par un ensemble $\{\boldsymbol{u}^i, i = 1...N\}$ sur lequel la projection de tout $\boldsymbol{u}(\theta)(\theta \in [0,1]^p)$ est maximum :

$$\mathcal{H}^N = vect(\boldsymbol{u}^1, ..., \boldsymbol{u}^N) \tag{2.34}$$

La projection de $\boldsymbol{u}$ sur $\mathcal{H}^N$ est définie par : $\pi^N_{\mathcal{H}^N}(\boldsymbol{u}) = \sum_{i=1}^N \alpha_i.\boldsymbol{u}^i$, où α_i sont des scalaires ou des coefficients d'interpolation linéaire. L'erreur de la projection dépend directement de l'ensemble $\boldsymbol{u}^i$, sauf si N tend vers l'infini où l'erreur va tendre vers zéros. Mais dans le contexte de réduction du coût de calcul et de stockage, on souhaite que N soit le plus petit possible en s'assurant que l'erreur de projection soit toujours acceptable. Pour résoudre ce problème, l'algorithme glouton est une stratégie pour sélectionner les bons $\boldsymbol{u}^i$ ou en autre terme pour chercher les bons paramètres θ^i :

1. Initialisation : On génère un petit ensemble initial S^0, puis construit $\mathcal{H}^0$ en utilisant la méthode RB
2. A la i ème itération, on cherche θ^i tel que :

$$\| \boldsymbol{u}^i - \pi^{i-1}_{\mathcal{H}^{i-1}}(\boldsymbol{u}^i) \| \geq \gamma \max_{\theta \in [0,1]^p} \| \boldsymbol{u}(\theta) - \pi^{i-1}_{\mathcal{H}^{i-1}}(\boldsymbol{u}(\theta)) \| \tag{2.35}$$

3. On calcule par éléments finis $\boldsymbol{u}(\theta^i)$, puis met à jour : $\mathcal{H}^i = \mathcal{H}^{i-1} \bigoplus \boldsymbol{u}^i$

où $\gamma \in (0,1]$ est une constante. Si $\gamma = 1$, l'algorithme est classique et l'erreur de projection est estimée directement par la solution exacte $\boldsymbol{u}(\theta)$. En pratique ce travail n'est pas réalisable puisque chaque itération d'optimisation du problème (2.35) doit appeler plusieurs calculs d'éléments finis. Le coefficient γ est donc utilisé pour une équivalence entre l'erreur de projection et un autre estimateur $r_i(\theta)$ qui satisfait : $c.r_i(\theta) \leq \| \boldsymbol{u}(\theta) - \pi^i_{\mathcal{H}^i}(\boldsymbol{u}(\theta)) \| \leq C.r_i(\theta)$ pour tout $\theta \in [0,1]^p$. Puis le problème (2.35) équivaut à chercher

$$\theta^i = argmax(r_{i-1}(\theta)). \tag{2.36}$$

Dans ce cas, l'erreur de projection $\| \boldsymbol{u}^i - \pi^{i-1}_{\mathcal{H}^{i-1}}(\boldsymbol{u}^i) \|$ satisfait (2.35) avec $\gamma = \frac{c}{C}$. L'estimateur r_i est le résidu de la formule varationnelle :

$$r_i = (\mathcal{N}(\tilde{\boldsymbol{u}}^{\theta,h}), v^h)_V - (f^\theta, v^h)_{L^2(\Omega^h)} \quad , \forall v^h \in V^h \tag{2.37}$$

où $\tilde{\boldsymbol{u}}^{\theta,h} = \pi^i_{\mathcal{H}^i}(\boldsymbol{u}(\theta))$.

Cette méthode est détaillée dans [Rozza *et al.*, 2007]. La méthode s'applique pour le cas bilinéaire a(u,v) ou pour les problèmes EDP paraboliques et elliptiques, mais elle est moins prometteuse pour le cas hyperbolique [Ngoc Cuong *et al.*, 2005].

2.2.4 Proper Generalized Decomposition (PGD)

Une autre méthode de réduction de modèle qui attire en ce moment l'attention des chercheurs est la PGD. Cette méthode, initialement appelée "radial time-space approximation", a été introduite pour la première fois par [Pierre, 1999] dans le contexte de la méthode LATIN (LArge Time INcrement method) pour réduire les coûts de calcul aussi bien en terme de stockage mémoire qu'en terme de temps de simulation. La méthode est étendue à plusieurs domaines ([Ammar *et al.*, 2006, Dumon *et al.*, 2011, Chinesta *et al.*, 2011, Bonithon *et al.*, 2011, Leygue et Verron, 2010, Beringhier *et al.*, 2010]). La PGD est une méthode 'a priori' puisqu'elle construit les fonctions de base sans connaître les informations de la solution. Ces fonctions sont enrichies au cours de chaque itération dont la solution approchée tend vers la solution exacte. On trouve la solution sous la forme suivante :

$$\tilde{\boldsymbol{u}}^k(x_1, x_2, ..., x_N) = \sum_{i=1}^{k} \alpha^i . \boldsymbol{\Psi}_1^i(x_1) . \boldsymbol{\Psi}_2^i(x_2) ... \boldsymbol{\Psi}_N^i(x_N). \tag{2.38}$$

où les variables $x_1, x_2 ... x_N$ peuvent présenter des variable spatiales, temporelles, ou des paramètres de contrôle du problème à résoudre (nombre de Reynold par exemple). Les fonctions de base normalisée $\boldsymbol{\Psi}_1^i, ..., \boldsymbol{\Psi}_N^i$ dépendent uniquement d'une variable et sont présentées par la formule discrète d'éléments finis. Les coefficients α^i sont des scalaires appelés les coefficients de pondérations. Les fonctions $\boldsymbol{\Psi}_j^i$, et les coefficients α^i sont calculés de manière itérative où les trois étapes suivantes sont nécessaires pour chaque itération n :

- l'étape d'enrichissement de la base consistant à calculer les fonctions $\boldsymbol{\Psi}_{j,j=1,..,N}^n$,
- l'étape de projection consistant à calculer n coefficients α^i,
- l'étape d'estimation de la convergence consistant à calculer le résidu du problème qui représente la précision de l'approche.

Les détailles pour calculer ces termes sont disponibles dans [Ammar *et al.*, 2006, Nouy, 2010]. Récemment, une application de PGD pour l'équation de Navier-Stokes a été réalisée par [Dumon *et al.*, 2011], où il a montré l'efficacité de cette méthode. Le speed-up est de l'ordre d'une dizaine de fois par rapport au calcul classique. Cependant, si l'espace de variable devient large (N est grand), la mis en place de la méthode PGD est un challenge. Cette méthode a donc encore besoin d'être étudiée pour les problèmes paramétrés de grande dimension [Chinesta *et al.*, 2010]. En plus, PGD est une approche intrusive, ce qui la rend moins intéressante dans le contexte industriel.

2.3 Conclusion

Au cours de ce chapitre, nous avons succinctement passé en revue les méthodes de réduction de modèle déjà connues dans la littérature. Parmi ces méthodes, le métamodèle de krigeage et la technique POD seront utilisés dans le chapitre 3 pour la construction d'une nouvelle méthode de réduction de modèle dans le contexte de conception du système ECS de la cabine d'avion. Elles seront donc présentées de façon plus détaillée. Les autres méthodes sont présentées pour donner une vue plus large de la méthode de réduction de modèle ainsi que les limites dans ces méthodes.

Dans le contexte de conception du système ECS, un modèle CFD numérique est nécessaire pour prédire les champs de vitesse et de température dans la cabine d'avion. Ces champs dépendent de plusieurs paramètres de conception. Le modèle CFD est normalement un modèle éléments finis ou volumes finis dont le degré de liberté est de l'ordre de centaines de milliers voire plusieurs millions. Donc, le modèle réduit à construire est une approximation de la relation entre les entrées (paramètres de conception) de petite dimension et les sorties de grande dimension. Le métamodèle de krigeage ainsi que les autres méthodes des surfaces de réponses ne sont pas capables d'approcher une telle relation.

Les méthodes de réduction d'ordre permettent de réduire la dimension de la sortie constituée de la combinaison des fonctions de base. La méthode POD-Galerkin présentée ci-dessus est applicable pour les cas nonlinéaires, paramétrés. Cependant, la précision de la solution et la stabilité du modèle réduit dépendent de la qualité du plan d'expériences et de nombre de modes POD pris en compte. Si l'espace de paramètres est assez large (la dimension > 3) et la sortie varie fortement selon les entrées (paramètres) dans certaines régions de l'espace de paramètres, il faut :

- enrichir le plan d'expériences de manière adaptative, la technique de DoCE n'est pas suffisant. Une stratégie "ad hoc" pour trouver un plan d'expériences optimal est nécessaire,
- augmenter le nombre de modes POD pris en compte. Ceci diminue l'avantage de la réduction.

La méthode RB et l'algorithme glouton permettent d'enrichir un plan d'expériences de manière adaptative en utilisant l'estimation *a posteriori*. Théoriquement, cette méthode se restreint aux problèmes EDP elliptiques et paraboliques. En plus, comme la méthode POD-Galerkin, quand le nombre de fonctions de base augmente l'efficacité de réduction diminue.

La méthode PGD est une méthode de réduction *a priori*. Elle n'a donc pas besoin de réaliser un plan d'expériences. Cependant, cette méthode a besoin d'être vérifiée pour les problèmes paramétrés de haute dimension.

Ces trois méthodes de réduction d'ordre sont intrusives. Les utilisateurs doivent

bien connaître les équations gouvernantes du problème ainsi que la méthode de réduction utilisée. Ceci les rend moins intéressantes dans le contexte industriel.

Pour ces raisons, la première contribution de cette thèse est de proposer une nouvelle méthode capable de répondre aux limites citées ci-dessus dans le contexte de la conception du système ECS.

Chapitre 3

Algorithme d'enrichissement adaptatif de bases données POD-ISAT

Sommaire

3.1 Introduction

Ce chapitre propose une méthode robuste pour réduire la complexité de modèles numériques de type EF tout en contrôlant l'erreur de l'approximation. En se basant sur les solutions calculées par éléments finis, un modèle paramétrique sera construit adaptativement pour remplacer le modèle fin de référence. Ce nouveau modèle permet de réduire le temps de calcul réel, et donne des solutions proches de la solution du modèle de référence avec une erreur inférieure à une tolérance admissible.

Dans cette étude la technique POD sera utilisée pour construire une base réduite de fonctions capables de représenter correctement les solutions du problème. Informatiquement parlant, le modèle sera vu comme une base de données de simulation avec capacité d'extrapolation et d'interpolation locale dans l'espace paramétrique. Une stratégie adaptative pour stocker et accéder à la base de données sera proposée en étendant l'algorithme In situ Adaptive Tabulation (ISAT) introduit par [Pope, 1997]. Cet algorithme aujourd'hui implanté dans le code Fluent [Gordon *et al.*, 2007] est utilisé efficacement dans plusieurs domaines, notamment combustion [Mazumder et ASME, 2007, Arsenlis *et al.*, 2006, Hedengren et Edgar, 2008] et continue à être perfectionné [Lu et Pope, 2009]. En fonction de l'usage et notamment de l'exigence en précision des résultats, la base de données pourra s'enrichir en ligne par des appels coûteux au code d'éléments finis. La combinaison des deux méthodes POD-ISAT est censée fournir une méthode de réduction de modèle EF très performante. La méthodologie sera évaluée sur un cas réel concernant un écoulement d'air dans une cabine d'avion décrit dans la section 3.2. La section 3.3 présente la méthode de réduction de modèle par POD et minimisation des résidus. La combinaison POD-ISAT ainsi que les résultats sont présentés dans la section 3.4.

3.2 Définition du cas test

3.2.1 Air conditionné dans une cabine d'avion

Nous nous sommes intéressés à la modélisation de la circulation d'air conditionné dans une cabine d'avion. On s'intéressera à l'écoulement de l'air dans une section passant par les sièges passagers pour simplifier le calcul de l'écoulement. Ce fluide est régi par les équations de Navier-Stokes incompressible avec prise en compte de la poussée d'Archimède due au changement de température (on considère l'approximation de Boussinesq). C'est-à-dire qu'on ajoute à la partie droite de l'équation de quantité de mouvement Navier-Stokes (3.2) un terme de poussée dépendant de la gravité et de la déviation de température $(T - T_0)$. L'équation de quantité de mouvement est donc couplée à l'équation de diffusion pour la température via ce terme

(équations (3.1)-(3.3)).

$$\nabla \cdot \boldsymbol{u} = 0 \quad \text{dans } \Omega, \tag{3.1}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} - \nu \Delta \boldsymbol{u} + \nabla p = \boldsymbol{g}\ (1 - \alpha(T - T_0)) \quad \text{dans } \Omega, \tag{3.2}$$

$$\frac{\partial T}{\partial t} + \boldsymbol{u} \cdot \nabla T - \nabla \cdot (\kappa \nabla T) = 0 \text{ dans } \Omega, \tag{3.3}$$

où $\boldsymbol{u}$, T, p sont respectivement le champ de vitesse, température et pression de l'air. Les coefficients ν, α, κ sont respectivement les coefficients de viscosité cinématique, de dilatation et de diffusion de l'air. Ω est le domaine où l'air s'écoule, autrement dit c'est la cabine où se trouvent des passagers.

Les conditions aux limites et les conditions initiales sont les suivantes :

$$\boldsymbol{u} = 0 \text{ sur } \Gamma_w, \boldsymbol{u} = \boldsymbol{u}_{in} \text{ sur } \Gamma_{in}, \tag{3.4}$$

$$\boldsymbol{u} = 0 \text{ à } t < 0, \tag{3.5}$$

$$\nu \Delta \boldsymbol{u} - p\boldsymbol{n} = 0 \text{ sur } \Gamma_{out} \tag{3.6}$$

$$T = T_{in} \text{ sur } \Gamma_{in},\ \frac{\partial T}{\partial \boldsymbol{n}} = 0 \text{ sur } \Gamma_{out},\ \kappa \frac{\partial T}{\partial \boldsymbol{n}} = \Phi(T - T_{ext}) \text{ sur } \Gamma_w. \tag{3.7}$$

$$T = T_0 \text{ à } t < 0, \tag{3.8}$$

où Γ_{in} correspond à l'endroit où on injecte l'air conditionné (injection au niveau du sol et au niveau des ventilateurs passagers) ; Γ_{out} est l'endroit où l'air conditionné quitte la cabine ; Γ_w est la partie restante du bord de fuselage ; T_{ext} est la température de l'air à l'extérieur de l'avion. La fonction $\Phi \in \mathcal{C}^\infty$ désigne le flux de chaleur au niveau du fuselage, donnée par la loi de refroidissement de Newton pour le transfert de chaleur par la convection [Burmeister, 1993] :

$$\forall x \in \mathbb{R}, \quad \Phi(x) = \frac{\kappa_f}{e} x, \tag{3.9}$$

où e est l'épaisseur du fuselage et κ_f est la conductivité thermique du fuselage.

En conditions réelles, l'ordre de grandeur de longueur L est de 1 m, la vitesse de fluide est de 1m/s et le coefficient de viscosité cinématique (à 300K) est de 1.57×10^{-5} m^2/s. Dans ce cas, le nombre de Reynolds est égal à :

$$Re = \frac{LU}{\nu} \approx 6.37 \times 10^4.$$

Le fluide est donc en régime turbulent. Mais par simplification, on suppose que l'écoulement est proche d'un écoulement laminaire et qu'il atteint un régime stationnaire. Ce que l'on obtient en augmentant la viscosité. En plus, le coefficient de diffusion thermique κ (à 300K et 1 atm) est de $2.22\,10^{-5}$ m^2/S, alors le nombre de

Péclet est :

$$Pe = \frac{LU}{\kappa} \approx 4.52 \times 10^4.$$

Le système est non linéaire et le processus de transport prédominant est la convection. Ce système couplé sera résolu par la méthode des éléments finis, le programme est écrit sous `FreeFem++3.10` [1]. Les solutions numériques discrètes $(\boldsymbol{u}, p, T)$ sont considérées comme les solutions exactes. On s'intéresse aux solutions où l'écoulement est quasiment stationnaire. Ces solutions dépendent des conditions aux limites ainsi que des propriétés du fuselage, qui sont des paramètres de conception (noté θ). Le cas test consiste à construire un modèle réduit permettant de prédire rapidement les solutions (output) pour un jeu de paramètres quelconques (input). Dans ce travail, on s'intéresse à construire un modèle réduit pour le champ de température (ouput=T) qui dépend de la température de l'air injecté, de la vitesse d'injection, de la conductivité thermique du fuselage.

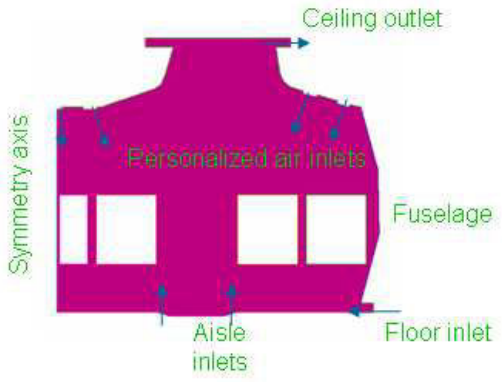

FIGURE 3.1 – Vue en coupe de la cabine avec les sièges et les niveaux d'injection d'air.

3.2.2 Résolution par la méthode des éléments finis

Pour résoudre le cas test ci-dessus par méthode des éléments finis, il est nécessaire de définir la formulation variationnelle (appelée également forme faible) à partir des équations (3.2) et (3.3). Soient q, τ les fonctions tests dans $H_0^1(\Omega)$, $\boldsymbol{v}$ la fonction test dans $[H_0^1(\Omega)]^2$. On suppose que Ω est un domaine avec une frontière lipschitzienne, $\boldsymbol{u}_{in} \in [H^1(\Gamma_{in})]^2$, $T_{in} \in H^1(\Gamma_{in})$ sur Γ_{in}, de sorte que $(\boldsymbol{u}, p, T)$ sont trouvées dans

1. http ://www.freefem.org/ff++/

$\boldsymbol{U}_{\boldsymbol{u}_{in}} \times L^2(\Omega) \times X_{T_{in}}$, où

$$\boldsymbol{U}_{u_{in}} = \left\{ \boldsymbol{v} \in [H^1(\Omega)]^2,\ \boldsymbol{v} = 0 \text{ sur } \Gamma_w,\ \boldsymbol{v} = \boldsymbol{u}_{in} \text{ on } \Gamma_{in} \right\},$$

$$X_{T_{in}} = \left\{ \tau \in H^1(\Omega),\ \tau = T_{in} \text{ sur } \Gamma_{in} \right\}.$$

On a deux équations variationnelles pour (3.2) et (3.3) :

$$\int_\Omega (\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}) \cdot \boldsymbol{v}\, dx + \nu \int_\Omega \nabla \boldsymbol{u} \cdot \nabla \boldsymbol{v}\, dx - \int_\Omega p \nabla \cdot \boldsymbol{v}\, dx + \int_\Omega \nabla \cdot \boldsymbol{u}\, q\, dx$$
$$- \int_\Omega (1 - \alpha(T - T_0))\, \boldsymbol{g} \cdot \boldsymbol{v}\, dx = 0, \tag{3.10}$$

$$\int_\Omega (\frac{\partial T}{\partial t} + \boldsymbol{u} \cdot \nabla T)\tau dx + \int_\Omega \kappa \nabla T \cdot \nabla \tau dx - \int_{\Gamma_w} \Phi\, (T - T_{ext})\tau\, dx = 0. \tag{3.11}$$

Puis, utilisant la méthode des caractéristiques [Pironneau, 1982] pour la dérivée particulaire, en discrétisant en temps, on obtient deux problèmes variationnels couplés :

$$\int_\Omega \frac{1}{\delta t}(\boldsymbol{u}^{m+1} - \boldsymbol{u}^m \circ X^m) \cdot \boldsymbol{v}\, dx + \nu \int_\Omega \nabla \boldsymbol{u}^{m+1} \cdot \nabla \boldsymbol{v}\, dx - \int_\Omega p^{m+1} \nabla \cdot \boldsymbol{v}\, dx + \int_\Omega \nabla \cdot \boldsymbol{u}^{m+1}\, q\, dx$$
$$- \int_\Omega (1 - \alpha(T^m - T_0))\, \boldsymbol{g} \cdot \boldsymbol{v}\, dx = 0, \tag{3.12}$$

$$\int_\Omega \frac{1}{\delta t}(T^{m+1} - T^m \circ X^m)\tau\, dx + \int_\Omega \kappa \nabla T^{m+1} \cdot \nabla \tau\, dx - \int_{\Gamma_w} \Phi\, (T^{m+1} - T_{ext})\tau\, dx = 0. \tag{3.13}$$

où $X^m(x) \simeq x - \boldsymbol{u}^m(x)\delta t$.

La méthode EF discrétise ces équations ci-dessus en remplaçant les vecteurs de l'espace infini par les fonctions de l'espace EF. Dans `freefem++`, notons `Th` le maillage (domaine approché de Ω) composé des triangles (voir figure 3.2), `Wh` l'espace de fonction polynomiale linéaire, `Vh` l'espace de fonction polynomiale quadratique. Les lignes de code pour la reconstruction du maillage, définition de l'espace EF et de vecteur EF ainsi que des paramètres sont les suivantes :

```
1 //--------Importer geometrie------------//
2 int outletbr = 1;
3 int wall     = 2;
4 int air4     = 3;
5 int air3     = 4;
6 int outlettr = 5;
7 int inlettl  = 6;
```

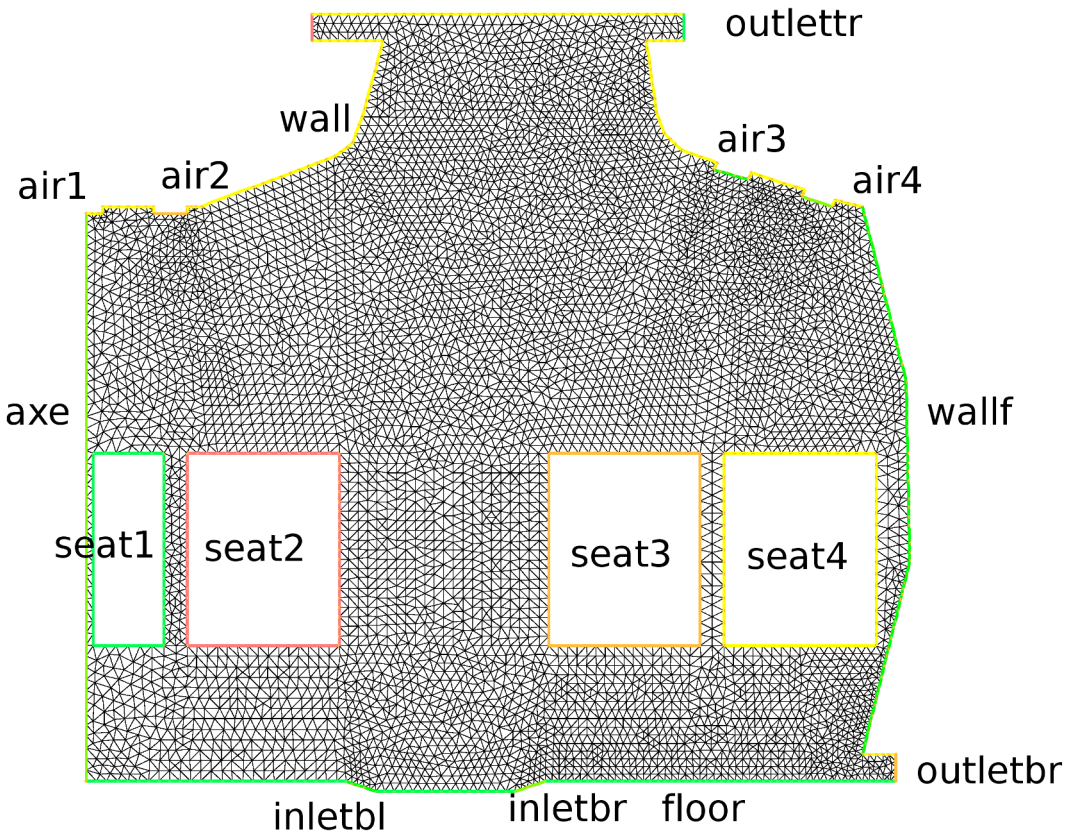

FIGURE 3.2 – Maillage fin généré par le code Freefem++

```
8  int air2      = 7;
9  int air1      = 8;
10 int axe       = 9;
11 int inletbl   = 10;
12 int seat1     = 11;
13 int seat2     = 12;
14 int seat3     = 13;
15 int seat4     = 14;
16 int inletbr   = 15;
17 int wallf     = 16;
18 int floor     = 17;
19 mesh Th = readmesh("TriangulationTh.msh");
20
21 //--------espace EF--------------------//
22 //
23 fespace Vh(Th, P2);
```

```
fespace Wh(Th, P1);
//
Vh u, uh, v, vh, uold, vold,uoldd,voldd;
Wh p, ph, pold;
Wh tp, tph, tpold,tpoldd;
//
u = 0; v = 0; // initially at rest
uold = 0; vold = 0;
//Input(parametre de conception)+Propriete du system//

real t=0;
real dt     ; // [sec]
real nu     ; // kinematic viscosity
real relax  ; // relaxation time [sec]
real tpinit ; // degree Celcius
real kappaa ; // thermal conductivity air (m2/s)
real kappaf ; // thermal conductivity fuselage
real thickn ; // Fuselage thickness
real inoutletbru ;// input velocity
real inoutletbrtp ; // input temperature
real thickfloor ; // floor thickness
real tpout  ; // external temperature
real gravity ; // [m s-2]
real alpha   ; // air dilatability
real eps = 1.d-11;
//
tp = tpinit;
tpold = tp;
int it=0;
```

Nous considérons `[u,v]` le champ de vitesse dans l'espace `Vh`, `p et T` les champs de pression et de température dans l'espace `Wh`. Les équations (3.12)-(3.13) peuvent s'écrire explicitement dans `freefem++` comme suit :

```
//-----Définition du problème de Navier-Stokes
problem ns([u,v,p], [uh,vh,ph], init=1) =
        int2d(Th)(u*uh/dt)
        -int2d(Th)(convect([uold, vold], -dt, uold)*uh/dt)
        +int2d(Th)(v*vh/dt)
        -int2d(Th)(convect([uold, vold], -dt, vold)*vh/dt)
        +int2d(Th)(nu*dx(u)*dx(uh) + nu*dy(u)*dy(uh))
```

```
        +int2d(Th)(nu*dx(v)*dx(vh) + nu*dy(v)*dy(vh))
        -int2d(Th)(p*dx(uh) + p*dy(vh))
        +int2d(Th)(dx(u)*ph + dy(v)*ph)
        +int2d(Th)(gravity*vh)
        -int2d(Th)(gravity*alpha*(tpold-tpinit)*vh)
        +on(wall, wallf, floor, u=0, v=0)
        +on(outletbr, u=-inoutletbru , v=0)
        +on(inlettl, u=0, v=0)
        +on(inletbl, u=-0.0, v=1.2)
        +on(inletbr, u=-0., v= 1.2)
        +on(axe, u=0)
        +on(air1, u= 0, v=-2.5)
        +on(air2, u= 1., v=-2.)
        +on(air3, u=-.5, v=-1.5)
        +on(air4, u=-0., v=-1.5)
        +on(seat1, seat2, seat3, seat4, u=0, v=0);
//Définition du problème thermique
problem thermal(tp, tph, init=1) =
        int2d(Th)(tp*tph/dt)
        -int2d(Th)(convect([uold, vold], -dt, tpold)*tph/dt)
        +int2d(Th)(kappaa*dx(tp)*dx(tph)+kappaa*dy(tp)*dy(tph
           ))
        -int1d(Th, wallf)(kappaf*tpout*tph/thickn)
        +int1d(Th, wallf)(kappaf*tp*tph/thickn)
        -int1d(Th, floor)(kappaf*tpout*tph/thickfloor)
        +int1d(Th, floor)(kappaf*tp*tph/thickfloor)
        +int1d(Th, outletbr)(inoutletbrtp*tph/eps)
        -int1d(Th, outletbr)(tp*tph/eps)
        +int1d(Th, air1,air2,air3, air4)(18*tph/eps)
        -int1d(Th, air1,air2, air3, air4)(tp*tph/eps)
        +int1d(Th, inletbl, inletbr)(25*tph/eps)
        -int1d(Th, inletbl, inletbr)(tp*tph/eps);
//220 itérations
for (int it=1; it < 221; it++) {
        t = t + dt;
        cout << "it = " << it << ", time = " << t <<endl;
        ns;
        uoldd=uold;voldd=vold;
    uold = u;   vold = v;
    thermal;
    plot([u, v],tp,viso=viso(0:viso.n-1), fill=1, value=1);
```

```
48
49      tpoldd=tpold;
50      tpold = tp;
51      plot(tp,fill=1);
52  }// FIN de la boucle en temps
```

Les équations (3.12)-(3.13) peuvent s'écrire formellement comme suit :

$$a(u^{m+1}, v) = L^{m+1}(v) \quad \forall v \in [H_0^1(\Omega)]^{2\,(ou\,1)} \tag{3.14}$$

où u représente des solutions (champs de vitesse, ou champs de température), a est un opérateur bilinéaire continu et coercif, L^{m+1} est un opérateur linéaire et continu. Après la discrétisation, le problème revient ensuit à résoudre l'équation linéaire de d équations à d inconnues :

$$AU^{m+1} = B^{m+1} \tag{3.15}$$

où A est appelé matrice de rigidité, symétrique, définie positive donc inversible, d est le nombre des nœuds de maillage. Dans le cas où le maillage est très fin, $(d >> 1)$, l'équation (3.14) devient très coûteuse à résoudre. Pour le présent cas test, le maillage est composé d'éléments triangulaires avec le nombre de degrés de liberté $d = 6382$. Le solveur pour résoudre les équation 3.14 est par défaut UMFPACK. Le pas de temps δt (`dt` dans `freefem++`) est fixé à 0.1. À chaque itération, l'équation de NS est d'abord résolue, en suite le champ de vitesse obtenu sert à résoudre l'équation thermique pour trouver la température. Supposons que l'écoulement atteint l'état stationnaire si :

$$sqrt(\int_\Omega (\boldsymbol{u}^{m+1} - \boldsymbol{u}^m) \cdot (\boldsymbol{u}^{m+1} - \boldsymbol{u}^m)\, dx) < 10^{-6}.$$

Après envrion 220 itérations, on trouve l'état stationnaire de fluide. Le temps réel d'un calcul est environ 1000 sec. Dans le cadre de cette thèse, on considère que les modèles éléments finis avec le maillage fin donne une solution exacte.

3.2.3 Définition d'un résidu

En supposant calculées des approximations $\tilde{\boldsymbol{u}} \in U_{u_{in}}$, $\tilde{p} \in L^2(\Omega)$ et $\tilde{T} \in X_{T_{in}}$ de $\boldsymbol{u}$,p et T respectivement, on peut définir un résidu à partir de l'équation (3.11)

et (3.11) à l'état stationnaire :

$$\begin{aligned}
&\forall(\boldsymbol{v}, q, \tau) \in [H_0^1(\Omega)]^2 \times L^2(\Omega) \times H_0^1(\Omega),\\
&R(\tilde{\boldsymbol{u}}, \tilde{p}, \tilde{T}, \boldsymbol{v}, q, \tau) = \int_\Omega (\tilde{\boldsymbol{u}} \cdot \nabla \tilde{\boldsymbol{u}}) \cdot \boldsymbol{v}\, dx + \nu \int_\Omega \nabla \tilde{\boldsymbol{u}} \cdot \nabla \boldsymbol{v}\, dx - \int_\Omega \tilde{p}\, \nabla \cdot \boldsymbol{v}\, dx\\
&+ \int_\Omega \nabla \cdot \tilde{\boldsymbol{u}}\, q\, dx - \int_\Omega (1 - \alpha(\tilde{T} - T_0))\, \boldsymbol{g} \cdot \boldsymbol{v}\, dx + \int_\Omega (\tilde{\boldsymbol{u}} \cdot \nabla \tilde{T})\tau\, dx\\
&+ \int_\Omega \kappa\, \nabla \tilde{T} \cdot \nabla \tau\, dx - \int_{\Gamma_w} \Phi\, (\tilde{T} - T_{ext})\tau\, d\sigma. \qquad (3.16)
\end{aligned}$$

Nous supposons que le code d'éléments finis ou de volumes finis permet de calculer la valeur du résidu pour toute solution approchée. D'une part, des nombreux codes de recherche permettent de calculer le résidu, par exemple Freefem++, d'un autre côté, quelques codes industriels ont évolué de telle manière que l'utilisateur peut interagir avec le solveur par un morceau de code en Fortran, C ou C++ (ou d'autres langages) qui permet de calculer le résidu d'une façon semi-intrusive. Notons que ce calcul est beaucoup moins coûteux qu'un calcul de la solution de modèle EF. Une analyse numérique concernant ce résidu est présentée à la section 3.3.2.

3.3 Méthode de réduction de modèle par POD et résidu

3.3.1 Formulation générale du modèle réduit

En général, la méthode de réduction de modèle (en anglais, ROM-reduced order modeling) par base réduite consiste à construire une série finie optimale de fonctions de base $(\boldsymbol{\Psi}^k(x))_{k=1,\dots,K}$ et déterminer les coefficients $a^k(\theta)$ correspondants. Le modèle réduit s'écrit sous la forme suivante :

$$\tilde{T}(\theta, x) = T^{lift,\theta}(x) + \sum_{k=1}^{K} a^k(\theta)\, \boldsymbol{\Psi}^k(x). \qquad (3.17)$$

T est un champ à déterminer (par exemple : champs de température dans la cabine), il dépend du paramètre θ et évolue selon la coordonnée spatiale x. La fonction $T^{lift,\theta}(x)$ est constante ou dépend linéairement de θ. Cette fonction vise à satisfaire la condition Dirichlet et la continuité de la solution approchée. Le temps de calcul de cette fonction est considéré comme négligeable. Dans ce travail, $T^{lift,\theta}(x)$ est solution d'équation de Laplace (3.18) avec la même condition Dirichlet dans le système d'équations principal.

$$\triangle T^{lift,\theta} = 0 \text{ dans } \Omega, \tag{3.18}$$

$$T^{lift,\theta} = T_0^{lift,\theta} \text{ sur } \Gamma_{in} \tag{3.19}$$

Remarque : L'équation ci-dessus sera résolue par Freefem++ sur le même maillage que celui du problème principal. Comme $T^{lift,\theta}$ est linéaire en θ, on va la résoudre seulement une fois puis la déduire instantanément en fonction de θ.

Les fonctions de base $\mathbf{\Psi}^k(x)$ peuvent être déterminées par plusieurs méthodes (POD, PGD, Krylov, RBM, série de Fourier, Polynômes...). Dans ce travail on utilise la méthode POD. Comme nous avons vu dans le chapitre 2, le principe de cette méthode est de construire à partir de la base de données fournie un ensemble de fonctions de base (également appelées "modes POD") permettant de reconstruire tout l'enregistrement de la base de données comme une combinaison linéaire de ces fonctions de base. De plus, ces fonctions de base sont ordonnées selon le montant d'information contenue dans chacune, de sorte que l'on peut agir sur la précision de l'approximation obtenue en effectuant une troncature des modes pour se passer des modes les moins importants en termes d'information.

Dans la méthodologie POD, les coefficients $a^k(\theta)$ peuvent être déterminés par plusieurs méthodes (POD-Galerkin, POD-Petrov-Galerkin, POD-RBF). Dans la section 3.3.2 une nouvelle méthode est proposée en utilisant le résidu éléments finis et l'approche de Krigeage.

3.3.2 L'approximation des coefficients en minimisant le résidu

On considère un ensemble $\mathcal{S}^{N_{exp}}$ de N_{exp} données vectorielles dans $\mathbb{R}^d$ qui sont des champs de températures $T(\theta^i)$ calculés pas le code d'éléments finis pour chaque jeux de paramètre θ^i :

$$\mathcal{S}^{N_{exp}} = \left\{ T(\theta^i) - T^{lift,\theta^i},\ i = 1, \ldots, N_{exp} \right\}. \tag{3.20}$$

La technique POD présentée dans le chapitre 3, permet de trouver les fonctions de base $(\mathbf{\Psi}^k(x))_{k=1,\ldots,K}$. La question est de trouver les coefficients $a^k(\theta)$ dans (3.17). Nous proposons ici deux niveaux d'approximation de ces coefficients, le premier étant le plus économique (en terme de coûts de calcul) et grossier, les suivants étant des corrections de celui-ci, nécessitant des calculs de résidus additionnels.

Le premier niveau d'approximation consiste à déterminer d'abord les $a_k^i := a^k(\theta^i)$ par projection de la solution exacte sur les modes POD :

$$a^k(\theta^i) = (T^i - T^{lift,\theta^i}, \mathbf{\Psi}^k), \quad (i = 1...N_{exp};\ k = 1...K) \tag{3.21}$$

puis de construire K métamodèles de la relation $\theta \rightarrow \tilde{a}^k(\theta) \simeq a^k(\theta)$. On peut déterminer aisément ces métamodèles en utilisant (3.21) et des techniques d'interpolation ou de régression : Méthode des moindres carrés (MLS ou Moving Least Square) [Lancaster et Salkauskas, 1981, Breitkopf *et al.*, 2005], réseaux de neurones (ANN ou Artificial Neural Networks) [Gérard, 2005], fonctions de base radiale (RBF ou Radial Basis Functions) [Buhmann, 2003, Audouze *et al.*, 2009], ou Krigeage [Cressie, 1990] par exemple.

Cependant, quand k est grand les coefficients a^k varient fortement en θ. Il est nécessaire d'enrichir la base d'apprentissages, autrement dit il faut augmenter N_{exp}. L'ensemble $\mathcal{S}^{N_{exp}}$ deviendra très grand dans ce cas-là. En plus, l'erreur de troncature combinée avec l'erreur d'approximation des coefficients peut mener à une erreur importante sur le champ complet.

Pour réduire la taille de $\mathcal{S}^{N_{exp}}$ et cette erreur, on propose d'utiliser la méthode des résidus (weighted residual method) comme **deuxième niveau d'approximation**. Cette méthode permet de mettre en place rapidement un modèle réduit et d'affiner les métamodèles $\tilde{a}^k$ (au sens d'un gain avec plus de précision).

Notons le vecteur $\mathbf{a} = (a^k)_{k\in\{1,...,K\}}$, étant donné un vecteur de paramètres θ, on peut chercher un modèle réduit $\tilde{T}(\theta)$ de forme :

$$\tilde{T}^\theta(\mathbf{a}) = T^{lift,\theta} + \sum_{k=1}^{K} a^k(\theta)\, \mathbf{\Psi}^k. \tag{3.22}$$

où $\mathbf{a} = \mathbf{a}(\theta)$ est choisi comme minimisant la norme $\mathcal{L}^2$ du résidu défini par (3.16) :

$$\min_{\mathbf{a}\in\mathbb{R}^K} \frac{1}{2} \parallel R(\tilde{T}^\theta(\mathbf{a})) \parallel^2 . \tag{3.23}$$

Dans la littérature, la technique de minimisation du résidu est utilisée par certaines auteurs [LeGresley, 2005, Bui-Thanh *et al.*, 2008a, Bui-Thanh *et al.*, 2008c, Carlberg *et al.*, 2011]. Le problème (3.23) peut être résolu au moyen d'un algorithme de recherche standard en petite dimension. Notons que les métamodèles $\tilde{a}^k(\theta)$ peuvent être utilisés pour fournir les premiers candidats au (3.23). Cela permet de réduire le temps du processus d'optimisation.

Un calcul de résidu d'éléments finis est beaucoup moins coûteux qu'un calcul d'éléments finis complet. Donc, le plan d'expérience utilisé en (3.21) peut être enrichi en minimisant le résidu pour d'autres valeurs de θ fixés ($\neq \theta^i$). Ces calculs étant indépendants, on peut donc les réaliser en parallèle sur une machine multi-cœur. Après d'avoir enrichi le plan d'expérience, on construit encore une fois K

métamodèles $\tilde{\tilde{a}}^k(\theta)$ plus précis.

On étudie les résidus pour le cas test dont l'espace de paramètres de conception est en dimension deux. Ces paramètres sont respectivement la conductivité thermique de fuselage (κ_f) et la température de l'air injecté (T_{in}) au niveau du "floor inlet". On réalise 200 calculs pour 200 valeurs différentes de paramètres répartis uniformément dans l'espace de conception. Puis, on calcule le résidu final de la résolution d'éléments finis, $R_0 := R(T^\theta)$. On constate que ces résidus ne sont pas toujours nuls, mais de l'ordre de 10^{-5} comme on peut le voir sur la figure 3.3 qui montre aussi une dépendance continue du résidu par rapport aux paramètres. Cette variation du résidu sur l'espace de conception reste raisonnable mais non négligeable, c'est pourquoi on normalise le résidu lors de la minimisation par un résidu de référence constant et égal à 10^{-5}. Désormais, on considère un résidu optimal relatif définit par :

$$\frac{\parallel R^{opt} \parallel}{\parallel R_0 \parallel} = \min_{\mathbf{a}\in\mathbb{R}^K} \frac{1}{2} \parallel R(\tilde{T}^\theta(\mathbf{a})) \parallel / \parallel R_0 \parallel \tag{3.24}$$

qui dépend de paramètres et du nombre de modes POD pris en compte n_{POD}. Puisqu'on ne connaît pas la valeur de R_0 à priori, on peut supposer que cette valeur est constante et égale à 10^{-5}.

Pour analyser la dépendance du résidu optimal par rapport aux paramètres et au nombre de modes POD n_{POD}, on réalise un petit plan d'expérience LHS de neuf solutions exactes. Puis, on construit les modes POD et on obtient un modèle réduit de premier niveau d'approximation. Ensuite, on fixe le paramètre $T_{in} = 25.3°\text{C}$ et fait varier uniformément κ_f en prenant 24 valeurs dans son intervalle de variation. Pour chaque point on calcul le résidu optimal relatif (voir (3.24)) qui correspond à l'erreur du modèle réduit amélioré ou de deuxième niveau (voir figure 3.4). On constate que la valeur relative du résidu optimal est plus grande quand n_{POD} est plus petit. Cette valeur tend vers 1 dans la zone où κ est grand, pour $n_{POD} = 3$ et $n_{POD} = 4$. Si on utilise quatre modes POD, et si le résidu est 8 fois inférieur à R_0, le modèle réduit donne une erreur acceptable sur la solution. Au contraire si on demande une valeur relative de résidu inférieur à 2 par exemple, il y a des zones de l'espace de conception où l'erreur est non acceptables, même si on augmente n_{POD} la courbe ne change pas. Ce problème vient du manque d'information de l'ensemble $\mathcal{S}^{N_{exp}}$. On peut résoudre cela en utilisant l'algorithme glouton repris par [Nair *et al.*, 2003, Audouze *et al.*, 2009, Veroy et Patera, 2005, Rozza et Veroy, 2007, Deparis, 2008, Bui-Thanh *et al.*, 2008b]. Le principe est de trouver θ_{greedy} où le résidu optimal est maximum, puis mettre à jour $\mathcal{S}^{N_{exp}}$ et les modes POD en rajoutant un calcul d'éléments finis de θ_{greedy}. Après plusieurs itérations on s'attend à ce que le résidu optimal soit inférieur à une valeur fixée par l'utilisateur. La base de donnés est donc enrichie de manière adaptative. Cependant, cette méthode demande beaucoup

de calculs pour minimiser en enrichir la base de données sans aucune garantie de diminuer efficacement le résidu dans certaines régions de l'espace de conception.

Pour cette raison on propose dans la section 3.4 une nouvelle technique de réduction de modèle : POD-ISAT.

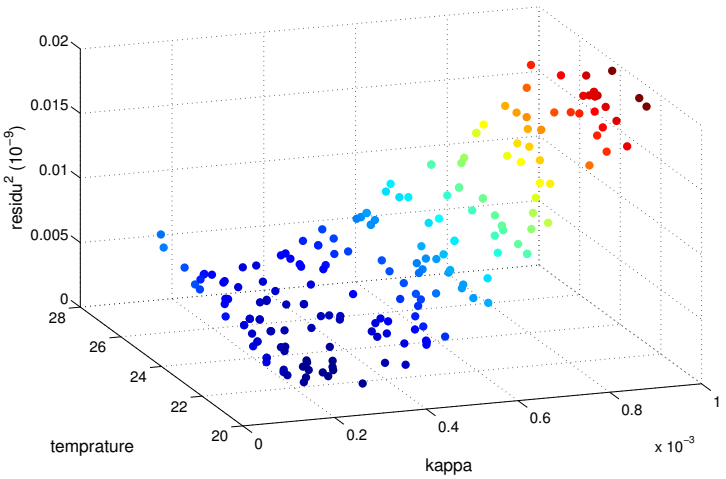

FIGURE 3.3 – Résidu de la solution exacte

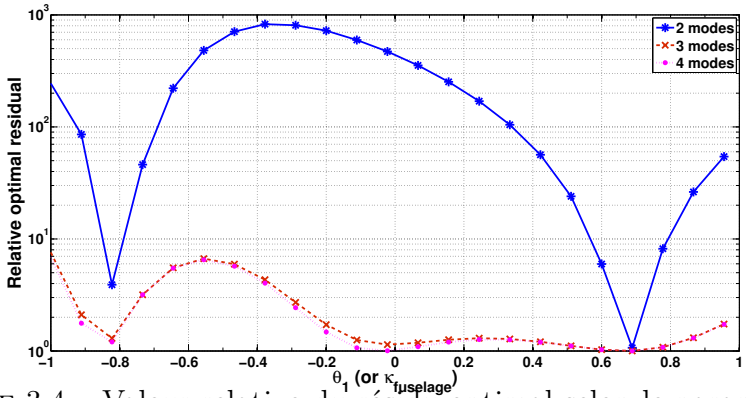

FIGURE 3.4 – Valeur relative du résidu optimal selon le paramètre κ

3.4 Modèle réduit POD-ISAT

3.4.1 Algorithme ISAT

In situ adaptive tabulation (ISAT), introduit par [Pope, 1997], est une méthode d'extraction et de stockage, qui fournit efficacement des approximations précises de fonctions de grande dimension numériquement coûteuses à évaluer. La méthode contrôle directement l'erreur d'approximation en ajoutant des régions de confiance multi-dimensionnelles dans l'espace des paramètres d'entrée. De cette façon, on peut

contrôler l'erreur d'approximation en la rendant inférieure à l'erreur tolérée. D'autre part, la base de données est construite séquentiellement de manière optimisée pour l'entrepôt et la fouille de données.

L'algorithme ISAT est un modèle réduit permettant de représenter une application à valeurs vectorielles

$$\begin{aligned} \boldsymbol{J} \quad &: \quad \mathbb{R}^p \longrightarrow \mathbb{R}^m \\ &\quad \boldsymbol{\theta} \mapsto \boldsymbol{J}(\boldsymbol{\theta}). \end{aligned}$$

Le vecteur $\boldsymbol{\theta}$ est appelé le vecteur d'entrée (input) tandis que le vecteur $\boldsymbol{J}(\boldsymbol{\theta})$ est appelé vecteur de sortie. En pratique, le vecteur $\boldsymbol{\theta}$ jouera le rôle d'un vecteur de paramètres de conception et $\boldsymbol{J}$ est un vecteur dont les composantes représentent des objectifs (ou des coûts).

Le modèle de données ISAT est un modèle réduit assimilé à une base de données dont chaque enregistrement (record), indicé par i, consiste en :

1. un vecteur d'entrée $\boldsymbol{\theta}^i$,
2. la valeur de la sortie $\boldsymbol{J}^i = \boldsymbol{J}(\boldsymbol{\theta}^i)$,
3. une matrice de sensibilité $\boldsymbol{A}^i = \frac{\partial J}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}^i)$,
4. un modèle local linéaire de forme

$$\tilde{\boldsymbol{J}}(\boldsymbol{\theta}) = \boldsymbol{J}^i + \boldsymbol{A}^i\,(\boldsymbol{\theta} - \boldsymbol{\theta}^i), \tag{3.25}$$

5. une région de confiance adaptative (appelée encore en anglaise, Ellipsoid of Accuracy ou EOA) représentée par la matrice des directions propres associées.

Tous ces enregistrements sont contrôlés par l'arbre de décision binaire (voir figure 3.5). Chaque enregistrement correspond à une feuille reliée par un nœud. Deux feuilles reliées au même nœud contiennent des paramètres θ considérés comme les voisins les plus proches. Un nœud contient l'information définissant l'hyperplan séparant les feuilles (correspondants aux points de la base de données dans l'espace des paramètres) reliées à ce noeud. Ceci permet de trouver rapidement le point de la base de données le plus proche du point de requête.

En pratique, pour un nouveau $\boldsymbol{\theta}$, l'algorithme ISAT recherche d'abord "le plus proche voisin" $\boldsymbol{\theta}^i$ du point $\boldsymbol{\theta}$ dans la base de données (cette étape s'appelle la récupération). Si $\boldsymbol{\theta}$ est dans la région de confiance, alors on peut utiliser le modèle réduit local comme approximateur ; sinon on calcule la valeur réelle $\boldsymbol{J}(\boldsymbol{\theta})$. Si $|\tilde{\boldsymbol{J}}(\boldsymbol{\theta}) - \boldsymbol{J}(\boldsymbol{\theta})| < \varepsilon$, alors on a affaire à un faux négatif et l'ellipsoïde de confiance est mise à jour pour inclure ce nouveau point (cette étape s'appelle la modification de

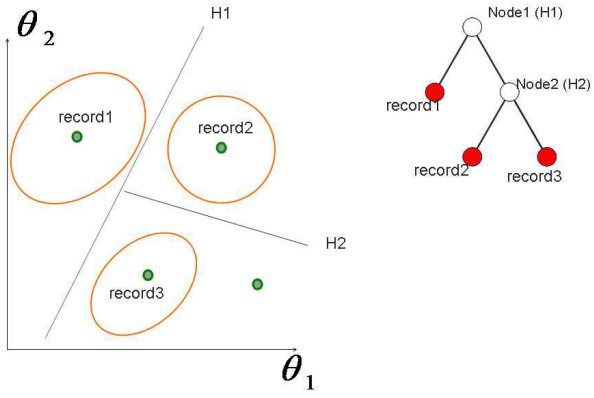

FIGURE 3.5 – Les enregistrements sont indicés dans l'arbre de décision

la région de confiance). Sinon, un nouvel enregistrement de centre $\boldsymbol{\theta}$ est créé dans la base de données, donc une région de confiance initiale se construit (cette étape est nommée l'addition de la région de confiance).

En bref, l'algorithme ISAT se compose 3 étapes principales :

1. récupération
2. addition
3. modification

3.4.1.1 Récupération (retrieve search)

L'objectif de l'étape "récupération" (retrieve search) est de déterminer toutes les EOAs qui couvrent un point θ^q donné dans l'espace des paramètres. S'il y a au moins une EOA qui couvre θ^q, la recherche est considérée comme "complète", sinon la recherche est considérée comme "incomplète". Une EOA qui couvre θ^q doit satisfait la condition suivante :

$$||(\theta^q - \theta^c)^T M(\theta^q - \theta^c)|| \leq 1,$$

où θ^c est le centre de l'EOA (c'est un point enregistré dans la base de données comme une feuille reliée à un noeud), et M est une matrice symétrique définie positive s'écrit aussi $M = U\Sigma^2 U^T$, avec U la matrice orthogonale dont les colonnes sont les vecteurs unités dans la direction des axes d'EOA, et Σ la matrice diagonale dont chaque élément $1/\Sigma_{ii}$ représente la longueur de $i^{ème}$ axe principal.

Pour simplification on pose $x = \theta - \theta^c$. L'information qui caractérise l'EOA se compose du point θ^c et de la matrice M. Dans la nouvelle version d'ISAT [Lu et Pope, 2009], le coût de la recherche des EOAs peut être réduit par une projection orthogonale de x et M sur un espace affine de dimension $n_a (1 \leq n_a < n_x)$.

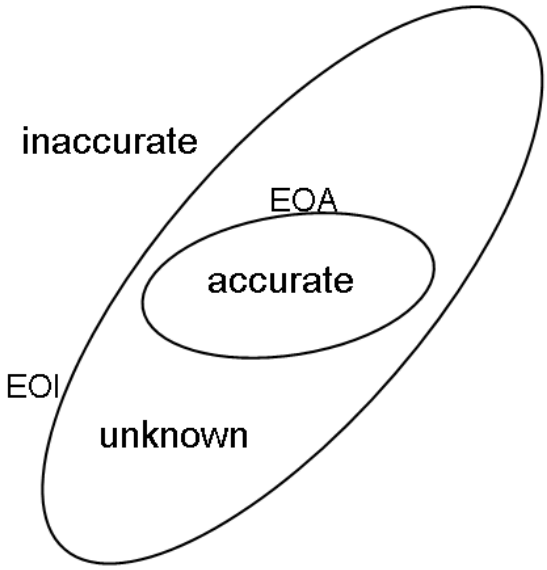

FIGURE 3.6 – L'ellipse de confiance(EOA) et l'ellipse de non confiance(EOI)

Le coût relatif est : $(n_a/n_x)^2 + p$, où p est la fraction des EOAs dont la projection dans l'espace affine contient la projection de θ^q. L'expérience [Lu et Pope, 2009] montre que la recherche est respectivement 3 et 7 fois plus rapide avec $n_x = 17$ et $n_x = 54$.

La méthode de recherche est plus performante dans la nouvelle version d'ISAT qui combine BT (binary tree) utilisée dans la première version d'ISAT avec 3 autres nouvelles méthodes MRU (most recent used), MFU (most frequently used), EBT (ellipsoidal binary tree).

3.4.1.2 Initialisation de la région de confiance (EOA et EOI)

A l'étape "addition" un nouveau point sera ajouté dans la base de données. Dans la nouvelle version d'ISAT, une information supplémentaire, l'ellipsoïde d'inexactitude (en anglaise, ellipsoid of inaccuracy ou EOI), est ajouté pour chaque point de la base de données. Cet ellipsoïde est concentrique à l'EOA, et tous les θ^q extérieurs à l'EOI sont dans une zone où le modèle d'approximation est considéré inexact. L'EOA et l'EOI correspondent respectivement aux positions inférieure et supérieure du bord de la région de confiance (voir figure 3.6). L'initialisation de l'EOA et de l'EOI sera respectivement présentée aux sections 3.4.1.2.1 et 3.4.1.2.2.

3.4.1.2.1 Initialisation de l'EOA Tout nouveau point $\theta^q = \theta^c$ ajouté à la base de données est associé à un EOA qui est construit à partir de deux tests simultanés : $||A^c(\theta - \theta^c)|| \leq \epsilon_{tol}$ (noté ellipsoïde1) et $||\theta - \theta^c||^2 \leq \alpha^2 \epsilon_{tol}$ (hypersphère).

Le premier test définit un ellipsoïde et le second une hypersphère. L'EOA initiale correspond à un ellipsoïde dont les valeurs propres sont calculées par la formule : $\lambda_i \equiv min(\lambda_{ellipsoïde1}, \alpha_{EOA}\epsilon^{1/2})$, où $\alpha_{EOA} = 0.1$.

3.4.1.2.2 Initialisation de l'EOI L'EOI est construit selon deux critères : Le premier est de considérer que l'erreur de l'approximation linéaire d'ISAT est d'ordre de $||\theta - \theta^c||^2$ et doit être inférieure à $\alpha_{EOI}^2\epsilon_{tol}$. Le deuxième critère est de supposer que θ^c n'appartient pas à l'EOI des autres point de la base de données, ou autrement dit les autres points de base données n'appartiennent pas à l'EOI de θ^c. Avec ces deux critères l'EOI initial est un ellipsoïde dont le demi-axe le plus long vérifie $r_{max} \leq \alpha_{EOI}\epsilon_{tol}^{1/2}$, et qui ne couvre pas d'autre point de base données.

3.4.1.3 Modification de la région de confiance

Quand l'étape de recherche d'un EOA a échoué, des points de la base de données (qui sont des centres d'EOA) sont choisis pour que leur EOA soit modifié. Chaque point candidat doit satisfaire les deux conditions suivantes : son EOI couvre θ^q et l'erreur de son modèle réduit local doit être inférieur à la tolérance prescrite par l'utilisateur. La modification de la région de confiance consiste à élargir l'EOA et à modifier (réduire ou élargir) l'EOI.

3.4.1.3.1 Elargissement de l'EOA (en anglais, EOA growth) [Hedengren et Edgar, 2008] ainsi que [Pope, 2008] ont bien expliqué comment on peut trouver un ellipsoïde de taille minimale qui couvre un ellipsoïde donné et un point θ^q donné. Le processus se décompose en trois transformations :

1. La transformation de l'EOA en une sphère unitaire :
   l'EOA donné est défini par :

   $$||L^T(x - c)|| \leq 1.$$

   où $x = \theta, c = \theta^c$, L est la factorisation de Cholesky de M.
   On définit une transformation de variable $y = L^T(x - c)$. L'EOA devient une sphère unitaire dans l'espace de cette variable (voir figure 3.7b). Avec la même transformation, θ^q devient $\theta_y^q = L^T(\theta^q - c)$.
2. Il existe une matrice de Householder H, qui transforme θ_y^q en un point sur le premier axe de variable z, où $z = H^T y = H^T L^T(x - c)$. Avec cette transformation l'EOA est toujours une sphère unitaire $z^T z \leq 1$. Afin de d'obtenir un ellipsoïde de taille minimale, EOA_{modif}, qui couvre à la fois l'EOA et θ^q, il suffit seulement de modifier l'EOA comme présenté dans la figure 3.7c. Cet

EOA modifié est recentré à nouveau à l'origine et est décrit par la formule suivante

$$z^T M_z z \leq 1$$

où

$$M_z = \begin{pmatrix} ||\theta_z^q||_2^{-2} & 0 & \cdots & 0 \\ 0 & 1 & \ddots & 0 \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

et

$$\theta_z^q = H^T \theta_y^q = H^T L^T (\theta^q - c)$$

3. En reprenant les variables originales x, l'équation de l'EOA$_{modif}$ est

$$(x-c)^T L H M_z H^T L^T (x-c) \leq 1$$

ou

$$(x-c)^T M_{modif} (x-c) \leq 1.$$

En bref, si l'on a un point θ^q extérieur d'une EOA associée à une matrice M et un centre θ^c, on peut trouver une EOA modifiée de taille minimale couvrant à la fois ancienne EOA et le point θ^q. Cette EOA modifiée est associée à un centre θ^c et une matrice $M_{modif} = LHM_z H^T L^T$.

3.4.1.3.2 Modification de l'EOI Un développement en série de Taylor montre que l'erreur du modèle réduit en θ est proportionnelle au carré de la distance $||\theta - \theta^c||$. Donc, si on connaît l'erreur $\epsilon(\theta^q)$ du modèle réduit calculé en θ^q, on peut trouver un point θ^I tel que l'erreur en ce point est égale à ϵ_{tol} :

$$\theta^I - \theta^c = \frac{\epsilon_{tol}^2}{\epsilon(\theta^q)^2} \cdot (\theta^q - \theta^c)$$

Ce nouveau point θ^I sera utilisé pour modifier l'EOI (voir figure 3.8). Une méthode de contraction de l'EOI courant vers un nouvel EOI de volume maximal qui couvre θ_I et est contenu dans l'ancien EOI est décrit dans [Pope, 2008].

3.4.1.4 Discussion

Le principal désavantage d'ISAT est qu'il faut connaître la matrice de sensibilité $\boldsymbol{A}^i$. [Hedengren et Edgar, 2008] ont proposé une régression linéaire pour approcher la matrice $\boldsymbol{A}^i$. [Varshney et Armaou, 2006] ont utilisé des différences finies avec une méthode de réduction de variance (Common Random Numbers ou CRN) [Dai,

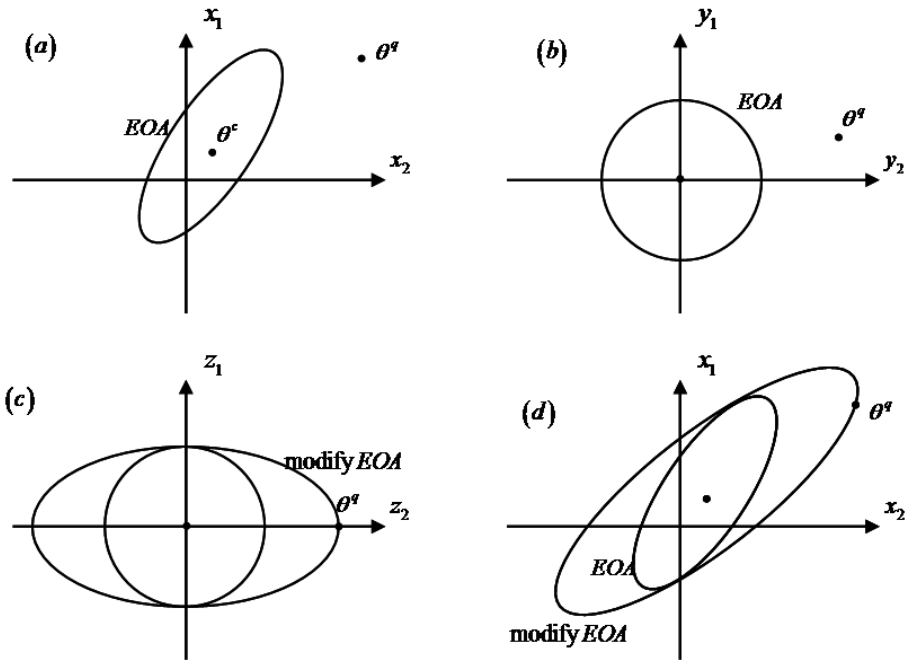

FIGURE 3.7 – (a) L'ellipsoïde initial et point θ^q dans l'espace x, (b) la transformation d'EOA en une sphère unitaire, (c) la rotation de de θ^q à un point sur l'axe z_2 et la modification d'EOA en EOA modifiée, (d) la modification d'EOA dans l'espace origine

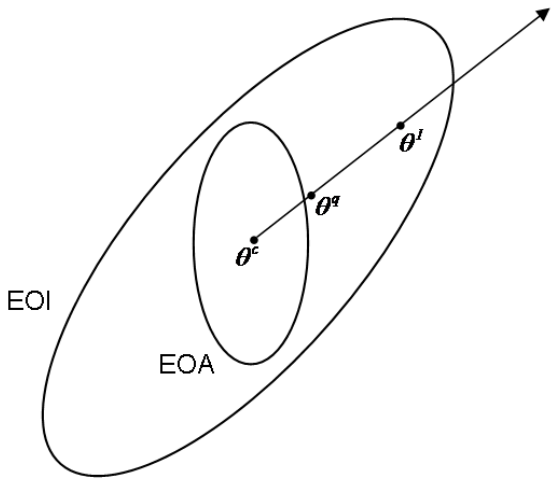

FIGURE 3.8 – Modification d'EOI

1997] pour calculer la matrice $\boldsymbol{A}^i$. Par ailleurs, ISAT est efficace dans le cas où la dimension de $\boldsymbol{J}$ est petite (≤ 50). Dans le cadre d'un calcul en dynamique des fluides, $\boldsymbol{J}$ peut être un champ de température de grande dimension (= le nombre de nœuds du maillage $> 10^6$).

Pour cette raison, on propose de combiner les deux méthodes POD et ISAT au sein de l'algorithme POD-ISAT. L'idée est de remplacer le modèle local linéaire de ISAT par le modèle réduit non linéaire basé sur la POD présenté dans la section (3.3). Chaque modèle POD est construit à partir des modes POD locaux. Le domaine (dans l'espace des paramètres) de chaque modèle local est limité par un ellipsoïde de confiance EOA. Cet ellipsoïde est construit une bonne fois pour toutes et n'est pas modifié comme dans ISAT. Puis, l'EOI n'est pas pris en compte.

3.4.2 Modèle réduit local de POD-ISAT

Supposons que la recherche d'un EOA qui couvre le nouveau paramètre θ^i a échoué et qu'on se trouve donc à l'étape "addition". La solution exacte T^i est déterminée par code d'éléments finis, puis un nouveau modèle réduit local centré en θ^i est construit par cinq étapes :

- Calcul de modes POD locaux
- Approximation au premier niveau de coefficients POD
- Approximation initiale d'un EOA
- Approximation au deuxième niveau de coefficients POD
- Correction de l'EOA

Ces étapes sont résumées dans la figure 3.9 et détaillées par les sections suivantes.

3.4.2.1 Modes POD locaux

D'abord, on calcule la solution $T(\theta^i)$ par le code d'éléments finis. Puis on détermine un plan d'expérience local dont les paramètres sont autour de θ^i. Pour cela on considère la boule de plus petit rayon $r_i^{N_{local}}$ qui contient N_{local} enregistrements de la base de données.

$$r_i^{N_{local}} = \arg\min_{r>0}\{\#\{\theta^j / \|\theta^j - \theta^i\| \leq r\} = N_{local}\} \tag{3.26}$$

Pour tous les points j dans cette boule, on constitue l'ensemble de données :

$$\mathcal{S}_{(i)}^{N_{local}} = \left\{T^j - T_{(i)}^{lift,\theta^j},\ j = 1, \ldots, N_{local}\right\}, \tag{3.27}$$

où $T_{(i)}^{lift,\theta^j} = T^{lift,\theta^j} + (T^i - T^{lift,\theta^i})$.

Puis on calcule les modes POD $(\boldsymbol{\Psi}_{(i)}^k(x))_{k=1,\ldots,K^i}$ à partir de ce plan d'expérience. Le rang de troncature K^i est choisi de façon à ce que l'énergie capturée par les

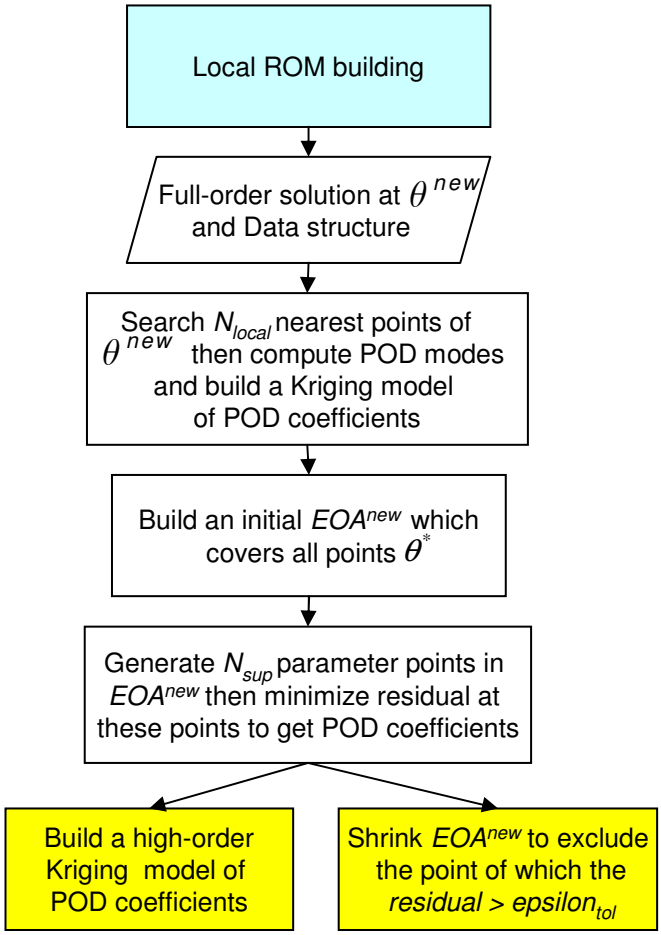

FIGURE 3.9 – Diagramme de construction d'un modèle local

K^i premiers modes soit supérieure à un seuil de confiance. En d'autres termes, l'erreur due à la projection sur la base POD est contrôlée localement via le rang de troncature K^i, ce qui est équivalent à fixer un seuil de tolérance local. K^i se varie pour chaque plan d'expérience. Cependant, au lieu de choisir adaptativement ce rang de troncature, on va le fixer ($K^i = K$) pour tous les cas. Le but est de simplifier l'algorithme, et de mettre en avant l'avantage d'utiliser des régions de confiance.

De la même façon que dans la section 3.3, une approximation locale s'énonce comme suit :

$$\tilde{T}^{\theta}_{(i)}(x) = T^{lift,\theta}_{(i)}(x) + \sum_{k=1}^{K} a^{k}_{(i)}(\theta)\, \mathbf{\Psi}^{k}_{(i)}(x). \tag{3.28}$$

Notons que si $\theta = \theta^i$, le deuxième terme du membre de droite de l'équation (3.28) doit disparaître ce qui impose que les cœfficients POD satisfassent $a^k_{(i)}(\theta^i) = 0$. L'ap-

proximation des coefficients POD est construite selon deux niveaux d'approximation de précision croissante.

3.4.2.2 Approximation au premier niveau de coefficients POD

En utilisant N_{local} échantillons voisins du point θ^i, on calcule les valeurs des coefficients POD en ces points :

$$a^k_{(i)}(\theta^j) = \left(T^j(x) - T^{lift,\theta^j}_{(i)}(x), \mathbf{\Psi}^k_{(i)}(x)\right);\ k = 1, \ldots, K;\ j = 1, \ldots, N_{local}. \tag{3.29}$$

Ensuite, on utilise la méthode de krigeage (voir section 2.1) pour construire K d'approximations au premier niveau de coefficients POD :

$$\begin{aligned} \tilde{a}^k_{(i)} \quad &: \quad \mathbb{R}^p \longrightarrow \mathbb{R} \\ &\theta \mapsto \tilde{a}^k_{(i)}(\theta), \quad k = 1, ..., K. \end{aligned} \tag{3.30}$$

Naturellement, ces méta-modèles sont validés pour les N_{local} points voisins de θ_i qui se trouvent à l'intérieur de la boule de rayon $r_i^{N_{local}}$. Au sein de cette boule, on va chercher une région de confiance autour de θ_i tel que l'erreur associée au modèle POD local (3.28) soit inférieure à un certain seuil de tolérance.

3.4.2.3 Approximation initiale de l'EOA

La région de confiance de chaque modèle local i est $\mathcal{E}(i)$, qui est définie comme :

$$\forall \theta \in \mathcal{E}(i), \quad \left\| R^{opt}(\tilde{T}_{(i)}(\theta), \cdot, \boldsymbol{u}^{\theta^i}) \right\|_{\mathcal{L}^2} \leq \varepsilon_{tol}. \left\| R_0 \right\|. \tag{3.31}$$

Supposons qu'il existe un ellipsoïde (appelé ellipsoïde de confiance ou EOA) appartenant à $\mathcal{E}(i)$. Un ellipsoïde dans $\mathbb{R}^p$ est défini par $\frac{(p^2+p)}{2}$ cœfficients. Donc, pour construire un EOA, on va chercher $M(> \frac{(p^2+p)}{2})$ points θ^* différents autour de θ^i satisfaisant la condition que $\|R^{opt}(\tilde{T}_{(i)}(\theta^*), \cdot, \boldsymbol{u}^{\theta^i})\|_{\mathcal{L}^2} = \varepsilon_{tol}. \|R_0\|$. La taille de l'EOA dépend du choix de la valeur de la tolérance ε_{tol}. Les points θ^* sont cherchés sous la forme $\theta^* = \theta^i + \alpha^* h$, où $\alpha^* \in \mathbb{R}$ et $h \in \mathbb{R}^p$ est un vecteur unité aléatoire. Cette étape peut prendre du temps si M est grand. Mais on peut paralléliser les calculs sur une machine multicoeur. À partir de ces θ^*, on va construire l'ellipsoïde de taille maximale qui ne contient pas ces points (voir figure 3.10) en utilisant successivement l'Ellipsoidal Toolbox pour MATLAB et la technique de modification des ellipsoïdes proposée par [Pope, 2008].

Le résidu en (3.16) dépend de la température et du champ de vitesse, en raison de la nature couplée du problème. Nous supposerons qu'une bonne approximation du

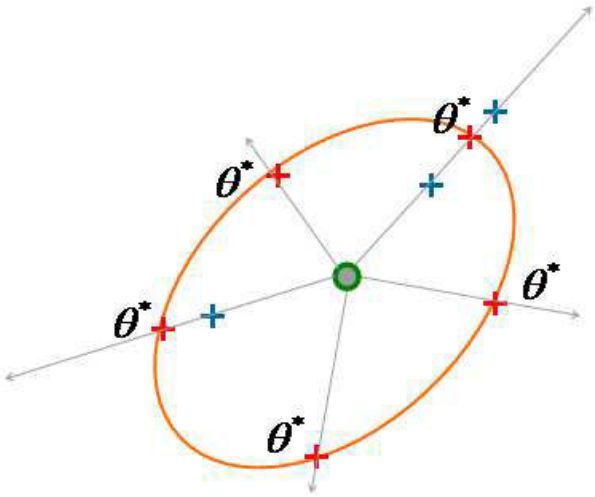

FIGURE 3.10 – construction d'une ellipse de confiance

champ de vitesse à l'intérieur de la région de confiance est donnée par $\tilde{\boldsymbol{u}}(\theta) = \boldsymbol{u}(\theta^i)$.

3.4.2.4 Approximation au deuxième niveau de coefficients POD

Une fois déterminé l'EOA, on génère un échantillon supplémentaire de coefficients POD selon des paramètres $(\theta^j)_{j=1,\dots,N_{sup}} \in \mathcal{E}(i)$. Ces coefficients sont calculés par :

$$(a^1_{(i)},\dots,a^K_{(i)})(\theta^j) = \arg\min_{(a^1_{(i)},\dots,a^K_{(i)})} \frac{1}{2}\left\| R\left(T^{lift,\theta^j}_{(i)}(x) + \sum_{k=1}^{K} a^k_{(i)}\,\boldsymbol{\Psi}^k_{(i)}(x),\cdot,\boldsymbol{u}^{\theta^i}\right)\right\|^2_{\mathcal{L}^2}. \tag{3.32}$$

puis on utilise cet échantillon combiné avec celui calculé par (3.29) pour reconstruire les interpolateurs krigeage de coefficients POD notés par $\tilde{\tilde{a}}^k_{(i)}(\theta)$.

3.4.2.5 Correction de l'EOA

L'EOA construit dans la section 3.4.2.3, ne vérifie pas strictement la condition (3.31). Grâce à l'échantillon calculé en (3.32) à la section précédente, on peut trouver les mauvais points dont le résidu optimal ne satisfait pas la condition (3.31). Dans ce cas-là, l'EOA est contracté afin d'enlever ces mauvais points en utilisant l'algorithme présenté dans la section 3.4.1.3. Cet algorithme permet de trouver une ellipse de volume maximale bornée par l'ancienne ellipse et par un point donné.

3.4.3 Résumé de l'algorithme POD-ISAT

L'algorithme POD-ISAT consiste en deux étapes principales. La première s'appelle "l'étape préliminaire hors ligne" de POD-ISAT et la deuxième s'appelle "l'étape d'enrichissement adaptatif" de POD-ISAT.

Au départ, l'algorithme POD-ISAT a besoin d'un plan d'expérience préliminaire $\mathcal{S}^{N_{init}} = \{T^j(\theta^j),\ j = 1, \ldots, N_{init}\}$. Les points θ^j sont choisis par la méthode LHS. En pratique on peut calculer les solutions d'éléments finis correspondant à chaque paramètre θ^j en parallèle puisqu'elles sont indépendantes. Cela permet d'accélérer la première étape. Puis à partir de ce plan d'expérience préliminaire, on construit les premiers modèles réduits locaux si nécessaire, pour chaque θ^j dans $\mathcal{S}^{N_{init}}$ (voir algorithme 1 et figure 3.11). Notons que si un point θ^j appartient à l'EOA d'un enregistrement existant, on a n'a pas besoin de construire un modèle réduit en ce point. Cependant, la solution correspondante déjà calculée est enregistrée dans la base de données pour servir à construire les modes POD locaux. À la fin de l'étape on aura une base de données incomplète avec un nombre d'enregistrement $n_{record} \leq N_{init}$.

À la deuxième étape quand on a une nouvelle entrée θ^q, si l'algorithme trouve dans la base de données incomplète une ellipse qui couvre ce point, la solution est instantanément donnée. Sinon, l'algorithme construit un nouvel enregistrement, et met à jour la base de données. Lorsque les EOAs dans la base de données couvrent tout l'espace des paramètres, on dit que le modèle réduit est complet (voir algorithme 2 et figure 3.11).

Données: Échantillon préliminaire du paramètre θ
Résultat: Base de données adaptative préliminaire

Calculer le plan d'expérience préliminaire :
$\mathcal{S}^{N_{init}} = \{T^j(\theta^j),\ j = 1, \ldots, N_{init}\}$
pour $i = 1$ **to** N_{init} **faire**
  si $n_{records} = 0$ **alors**
    Construire le premier modèle local en θ^i
  sinon
    Chercher EOA qui couvre θ^i grâce à l'arbre de décision.
    si *La recherche réussit* **alors**
      scénario=‘récupération’ ;
      On n'a pas besoin de stocker (T^i, θ^i) dans la base de données.
    sinon
      scénario=‘addition’ ;
      Construire un modèle réduit local ainsi que EOA en θ^i.
      Rajouter un nouvel enregistrement à la base de données.
    fin
  fin
fin

Algorithm 1: Étape préliminaire de POD-ISAT

Entrées: Nouveau θ^q appartenant à l'espace de paramètre
Sorties: Solution $\tilde{T}(\theta^q)$

Chercher l'EOA du point θ^i qui couvre θ^q grâce à l'arbre de décision de l'ISAT.
si *La recherche réussit* **alors**
    scénario='récupération' ;
    La solution est instantanément approchée par :
    $\tilde{T}^{\theta^q}_{(i)}(x) = T^{lift,\theta^q}_{(i)}(x) + \sum_{k=1}^{K} a^k_{(i)}(\theta^q)\, \mathbf{\Psi}^k_{(i)}(x)$.
sinon
    scénario='addition' ;
    Calculer la solution pour θ^q par modèle fin.
    Construire le modèle réduit local ainsi que EOA en θ^q.
    Rajouter un nouvel enregistrement à la base de données.
    La solution est exacte : $\tilde{T}^{\theta^q}(x) = T^{\theta^q}(x)$.
fin

Algorithm 2: Étape en ligne de POD-ISAT

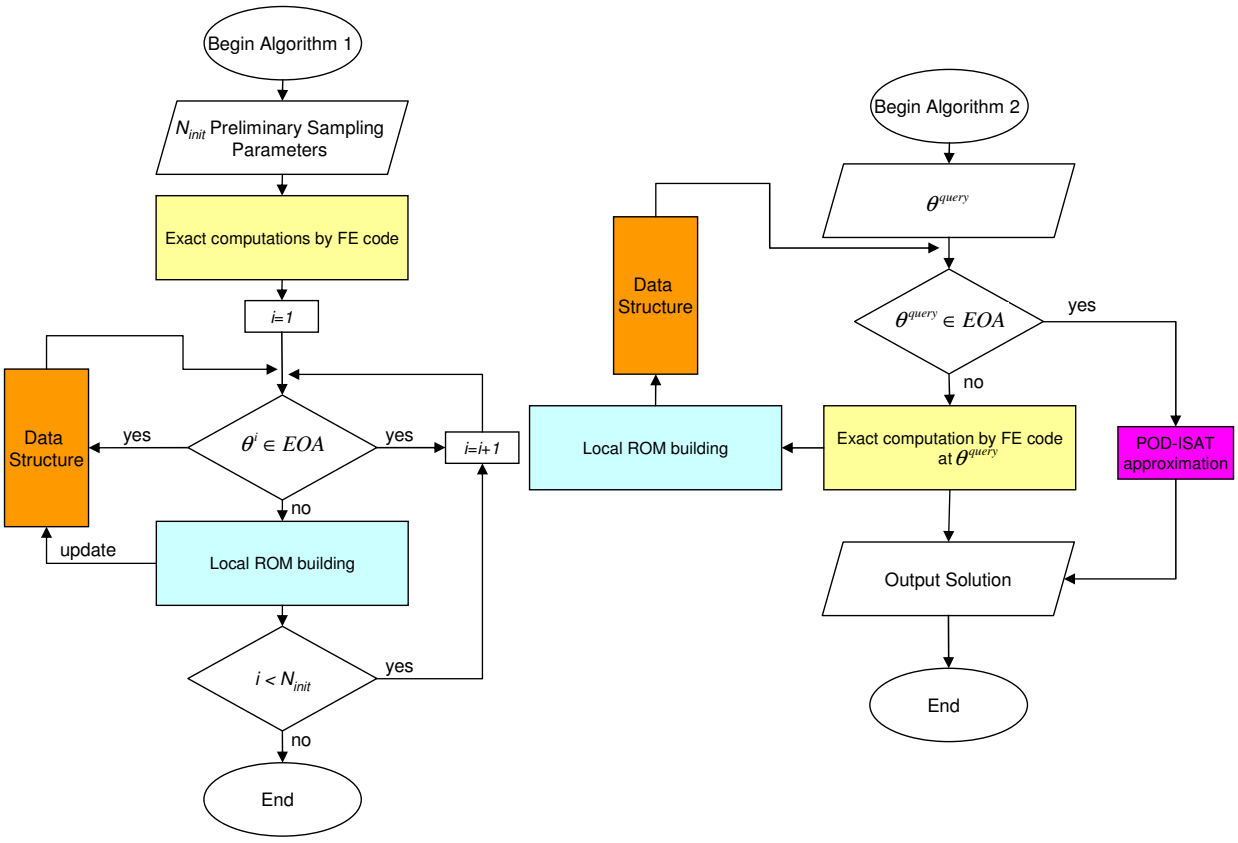

FIGURE 3.11 – Diagramme de l'algorithme POD-ISAT

3.5 Résultats numériques

La méthode POD-ISAT a été testée sur un cas test décrit dans la section 3.2. On considère deux cas différents selon la dimension de l'espace des paramètres. Le premier cas concerne un espace paramétrique d'ordre deux et le second concerne un espace paramétrique d'ordre quatre.

3.5.1 Conception d'un système de contrôle de l'air avec deux paramètres

L'espace des paramètres est de dimension 2. les deux paramètres sont la température de l'air injecté par la ventilation principale T_{in} et la conductivité thermique du fuselage κ_f. Ces deux paramètres varient dans les intervalles suivants :

$$\begin{aligned} T_{in} &\in [T_{in}^{min} = 21, \cdots, T_{in}^{max} = 28](^{\circ}\mathrm{C}), \\ \kappa_f &\in [\kappa_f^{min} = 10^{-4}, \cdots, \kappa_f^{max} = 10^{-3}](W \cdot m^{-1} \cdot K^{-1}). \end{aligned}$$

Les paramètres sont normalisés et dénotés par $\theta = [\theta_1 \quad \theta_2]^t$ où :

$$\theta_1 = \frac{T_{in} - \frac{1}{2}(T_{in}^{min} + T_{in}^{max})}{\frac{1}{2}(T_{in}^{max} - T_{in}^{min})},$$

$$\theta_2 = \frac{\kappa_f - \frac{1}{2}(\kappa_f^{min} + \kappa_f^{max})}{\frac{1}{2}(\kappa_f^{max} - \kappa_f^{min})},$$

L'air est initialement au repos et à température uniforme dans la cabine ($T_0 = 20\,^{\circ}\mathrm{C}$). L'air chaud est injecté en bas et au-dessus de la tête des passagers puis sort par le haute du fuselage. Après 220 itérations de calcul, l'écoulement est quasi stationnaire. La figure 3.12 présente quatre solutions d'éléments finis associées à quatre jeux de paramètres différents. Dix simulations préliminaires sont réalisées selon un plan d'expérience LHS. Les modes POD obtenus sont présentés sur la figure 3.13. Le nombre de modes K est fixé à 4 pour ce cas d'étude. Les propriétés du modèle réduit POD-ISAT sont résumées dans le tableau 3.1.

Étape préliminaire hors ligne : Au départ, 10 solutions sont calculées par le modèle éléments finis selon un plan d'expérience LHS. À la fin de cette étape, $n_{records} = 3$, ce qui veut dire que 3 modèles réduits locaux sont construits au niveau de trois points de l'espace des paramètres. Les 7 points restants se trouvent dans les régions de confiance des trois modèles réduits et donc on n'a pas besoin de construire de modèle réduit en ces points. Cependant, ces 7 solutions exactes sont enregistrées pour calculer ultérieurement les modes POD locaux.

Étape d'enrichissement adaptatif : On génère aléatoirement 190 points de requête

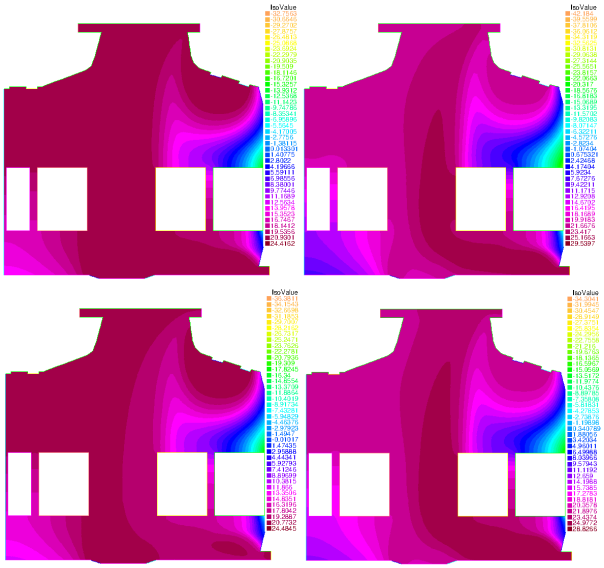

FIGURE 3.12 – Champs de température stationnaire dans la cabine pour chaque jeu de paramètre θ .

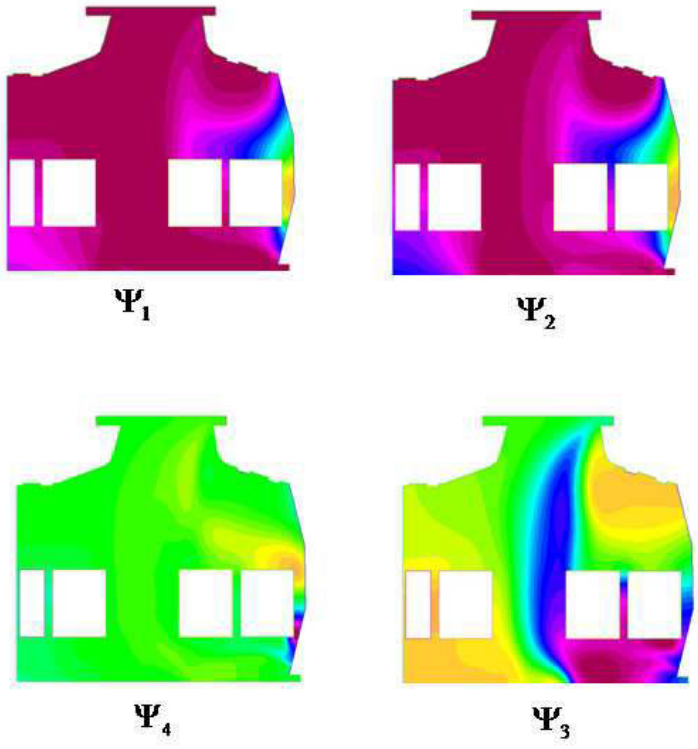

FIGURE 3.13 – Vue des quatre premiers modes POD.

TABLE 3.1 – Les propriétés du modèle réduit POD-ISAT

N_{exp}	10	Nombre de simulations EF initiales
ε^2_{tol}	2	Tolérance relative sur le résidu
N_{local}	9	Nombre de voisins pour construire la POD locale pour construire POD locale
K	4	Nombre de modes POD utilisés
p	2	Dimension de l'espace des paramètres

traités par l'algorithme 2. Quand le modèle POD-ISAT a recouvert tout l'espace de paramètre par des ellipses de confiance, $n_{records} = 9$, c'est-à-dire que dans la base de données finale, il y a neuf modèles réduits locaux. La figure 3.14 montre que les EOAs de ces modèles remplissent l'espace de paramètres normalisés. On peut aussi constater que la plupart des EOA sont étirés verticalement et plus petits dans la zone où κ_f est bas. Cela signifie que la conductivité thermique du fuselage a une influence plus forte sur la solution que la température injectée sur la solution, et que la solution varie plus fort dans la zone où κ_f est bas. Ce résultat reprend ce qui a été observé sur la figure 3.4 présentée dans la section 3.3.

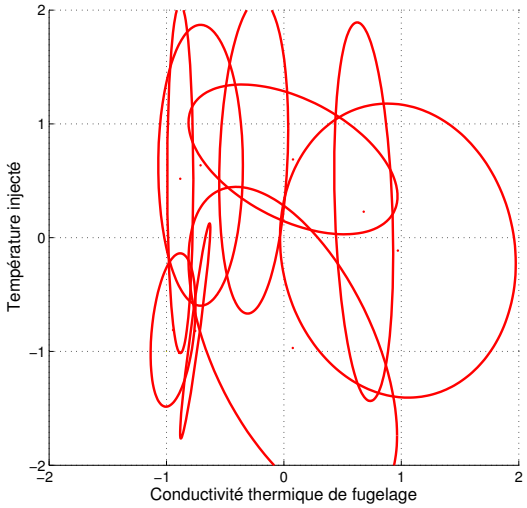

FIGURE 3.14 – EOAs

Notons que pendant l'exécution de l'algorithme POD-ISAT, 200 jeux de paramètres ont été générés (pour les deux étapes). Tous ces points sont calculés par le modèle d'éléments finis afin de vérifier l'approximation de POD-ISAT. On compare les solutions du modèle POD-ISAT avec celles obtenues par la méthode d'éléments

finis en utilisant l'erreur relative qui est définie sous la forme :

$$erreur \quad relative = \frac{\|\tilde{T}(\theta) - T(\theta)\|_{\mathcal{L}^2}}{\|T(\theta)\|_{\mathcal{L}^2}}. \tag{3.33}$$

La distribution de cette erreur sur l'espace de paramètre est présentée sur la figure 3.15, où on peut voir que l'erreur maximale est de 1.2×10^{-3}, tandis que la plupart sont de l'ordre de 10^{-4}. Les erreurs élevées sont plutôt aux bords du domaine où T_{in} est basse et k_f est élevée.

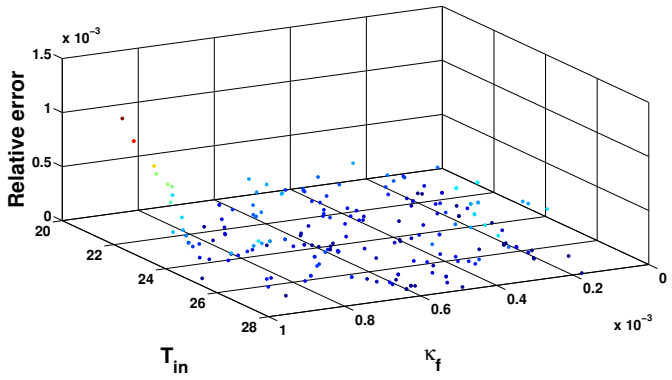

FIGURE 3.15 – L'erreur relative pour 200 essais de modèle réduit POD-ISAT

Par ailleurs, pour le même cas test, la même taille de base de données, le même nombre de modes POD n_{POD}, l'erreur d'un modèle réduit POD classique (sans régions de confiance) est calculée sur les 200 jeux de paramètres précédemment évoqués. La distribution de l'erreur est représentée sur la figure 3.16. On peut voir que l'erreur du modèle POD classique est plus élevée que celle du modèle POD-ISAT (à peu près 10 fois plus en général). Les erreurs élevées sont aussi aux bords du domaine. Ils n'apparaissent pas non seulement dans la zone où T_{in} est basse et k_f est élevée, mais dans la zone où T_{in} est basse et k_f est basse. L'évolution de l'erreur maximale selon le nombre de requêtes pour l'algorithme POD-ISAT et pour le modèle POD classique est présentée dans la figure 3.17. On peut trouver que l'algorithme POD-ISAT est beaucoup plus efficace.

3.5.2 Conception d'un système de contrôle de l'air avec quatre paramètres

Pour montrer la capacité de l'algorithme POD-ISAT à traiter un cas de dimension plus haute, nous ajoutons en plus deux paramètres de conception (voir figure 3.18). Le premier est la température $T_{in,a}$ de l'air injecté à Γ^1_{in}, et le deuxième est la température $T_{in,p}$ de l'air injecté à Γ^2_{in}. Ces paramètres varient de $T^{min}_{in,a} = T^{min}_{in,p} = 21°\text{C}$ à $T^{max}_{in,a} = T^{max}_{in,p} = 28°\text{C}$. Les paramètres sont normalisés et dénotés par $\theta =$

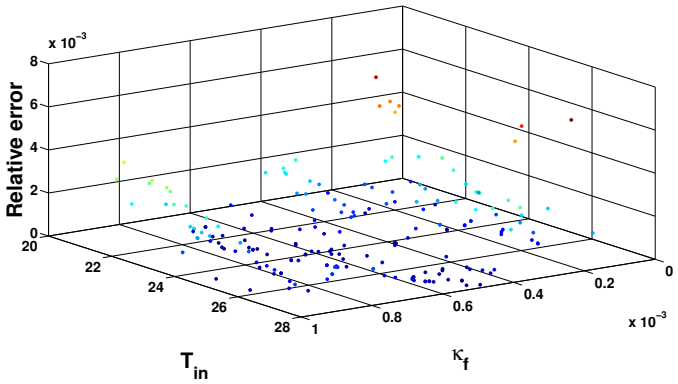

FIGURE 3.16 – L'erreur relative pour 200 essais du modèle réduit POD

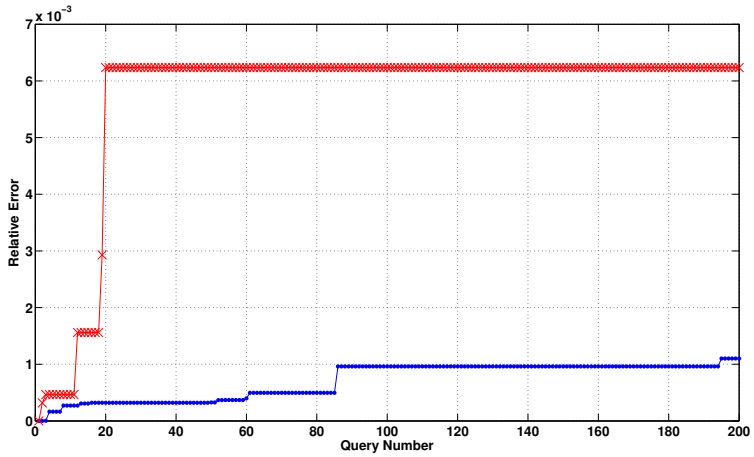

FIGURE 3.17 – Erreur maximale selon le nombre de requête de la solution calculée par POD (×) et par POD-ISAT (•) pour le cas de deux paramètres

$[\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4]^t$ où :

$$\theta_1 = \frac{T_{in} - \frac{1}{2}(T_{in}^{min} + T_{in}^{max})}{\frac{1}{2}(T_{in}^{max} - T_{in}^{min})},$$

$$\theta_2 = \frac{\kappa_f - \frac{1}{2}(\kappa_f^{min} + \kappa_f^{max})}{\frac{1}{2}(\kappa_f^{max} - \kappa_f^{min})},$$

$$\theta_3 = \frac{T_{in,a} - \frac{1}{2}(T_{in,a}^{min} + T_{in,a}^{max})}{\frac{1}{2}(T_{in,a}^{max} - T_{in,a}^{min})},$$

$$\theta_4 = \frac{T_{in,p} - \frac{1}{2}(T_{in,p}^{min} + T_{in,p}^{max})}{\frac{1}{2}(T_{in,p}^{max} - T_{in,p}^{min})},$$

Nous générons un plan d'expérience préliminaire de type LHS avec $N_{init} = 20$ calculs par modèle d'éléments finis. Au départ, l'algorithme 1 permet de construire

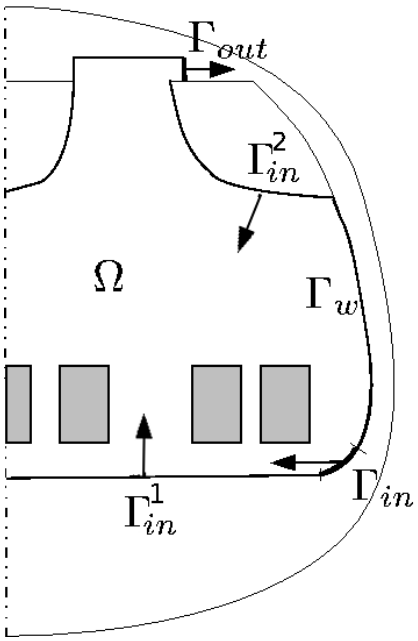

FIGURE 3.18 – Deux paramètres de conception ajoutés sont température de l'air injecté au couloir et au-dessus de passager

les premiers modèles réduits locaux en se fondant sur le plan d'expérience préliminaire. À la fin de cette étape, on obtient $n_{records} = 9$. Puis on utilise l'algorithme 2 qui examine 400 requêtes. Pour éviter la coïncidence des requêtes, on génère aussi ces points par la technique LHS pour couvrir l'espace des paramètres. À la fin de cette étape, on obtient $n_{records} = 28$ modèles réduits locaux. En bref, l'algorithme POD-ISAT a traité 420 jeux de paramètres. Les erreurs relatives selon le numéro de requête sont montrées sur la figure 3.19. Les points en noirs désignent les additions à la base de données, et les autres points appartiennent à l'une des 28 EOA déjà construites ("retrieve").

On évalue aussi pour les mêmes 420 jeux de paramètres le modèle POD classique avec un plan d'expérience de taille identique à celui de POD-ISAT. L'erreur maximale selon le numéro de la requête pour l'algorithme POD-ISAT et pour le modèle POD classique est présentée dans la figure 3.20.

3.5.3 Validation de modèle krigeage sur l'approximation des coefficients POD

3.5.3.1 Sélection de modèles

Dans la section 2.1 nous avons présenté le modèle de krigeage avec plusieurs types de modèles de régression et de corrélation. Le choix du binôme (modèle de régression, modèle de corrélation) détermine la qualité du modèle de krigeage. Nous utilisons

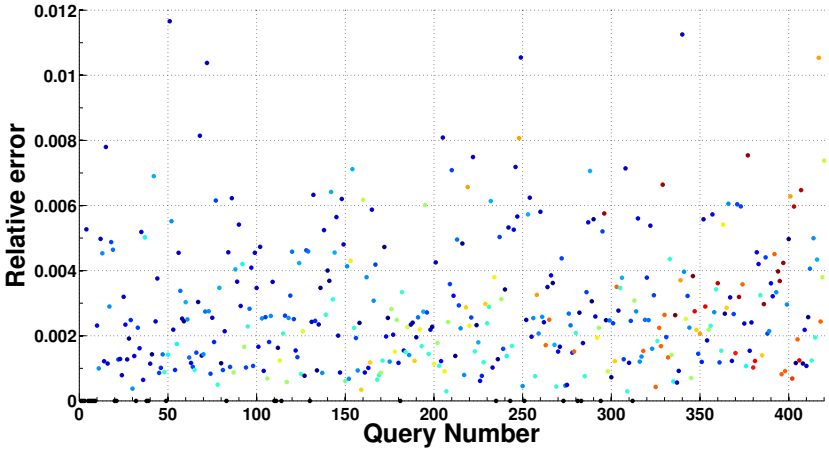

FIGURE 3.19 – Erreurs relatives des solutions calculées par POD-ISAT pour le cas de quatre paramètres

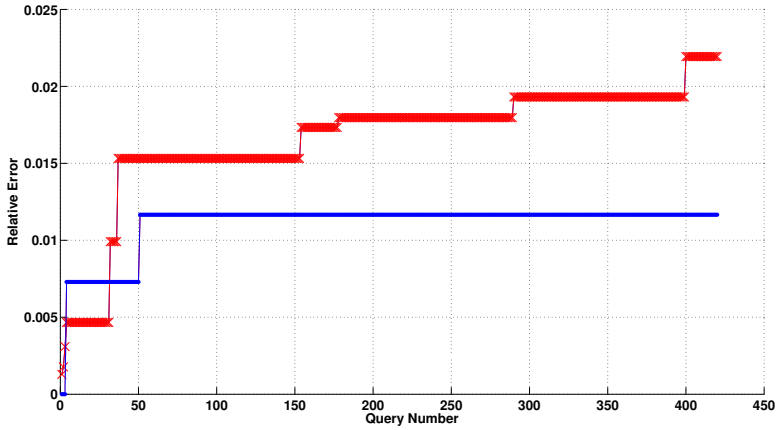

FIGURE 3.20 – Erreur maximale selon le nombre de requêtes de la solution calculée par POD (×) et par POD-ISAT (•) pour le cas de quatre paramètres

six modèles de krigeage $\mathcal{K}^i$, $i = 1, ..., 6$ associés à 6 binômes différents suivants :

- fonction constante + fonction gaussienne
- fonction linéaire + fonction gaussienne
- fonction quadratique + fonction gaussienne
- fonction constante + fonction exponentielle
- fonction linéaire + fonction exponentielle
- fonction quadratique + fonction exponentielle

Le meilleur binôme doit réaliser la plus petite estimation d'erreur définie par (3.34) : on considère alors qu'il est le plus adapté à la base d'apprentissage. La méthode la plus simple pour estimer l'erreur de généralisation est de diviser en deux l'ensemble des données disponibles en une base d'apprentissage $\mathcal{A}$ et une base de validation $\mathcal{V}$ et d'estimer l'erreur de généralisation par $\mathcal{E}(\mathcal{K}^i_{\mathcal{A}}, \mathcal{V})$ une fois l'appren-

tissage effectué, où $\mathcal{E}(\mathcal{K}^i_{\mathcal{A}}, \mathcal{V})$ est calculé par l'erreur quadratique moyenne :

$$\mathcal{E}(\mathcal{K}^i_{\mathcal{A}}, \mathcal{V})^2 = E\left((\tilde{y}(x)_{\mathcal{K}^i_{\mathcal{A}}} - y(x))^2\right) ; x \in \mathcal{V}. \tag{3.34}$$

Prenons par exemple l'étape "approximation au deuxième niveau de coefficients POD" pour construire un premier ROM dans le cas deux paramètres. On réalise un modèle de krigeage à partir d'un plan d'expériences de 40 observations générées par 40 calculs de (3.32). On divise ces données en $\mathcal{A}$ avec 35 observations et en $\mathcal{V}$ avec 5 observations. Une fois les six modèle $\mathcal{K}^i$ construis sur $\mathcal{A}$, les erreurs $\mathcal{E}(\mathcal{K}^i_{\mathcal{A}}, \mathcal{V})$ sont calculées et comparées. Le meilleur modèle est conservé et son estimation d'erreur sur la base de validation $\mathcal{V}$ est montré dans la figure 3.21. On peut voir que l'erreur est assez petite de l'ordre de 10^{-3}. Les fonctions approchées présentées dans la figure 3.22 montrent que les coefficients POD varient fortement de plus en plus selon l'ordre de mode POD.

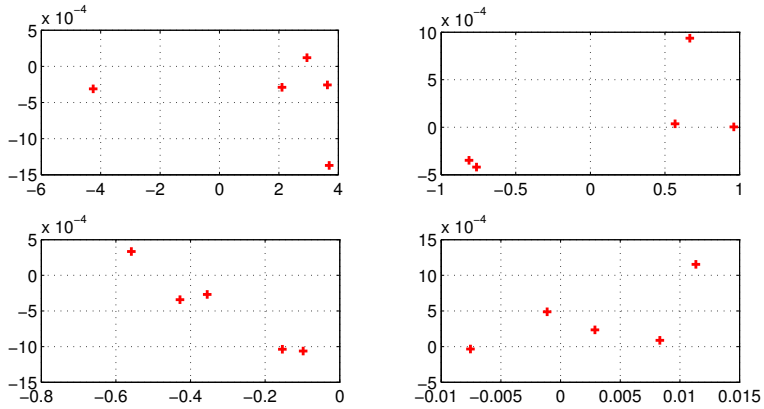

FIGURE 3.21 – Erreurs d'approximation (en abscisse verticale) pour les quatre premiers coefficients POD (en abscisse horizontal), relevées sur base de validation

L'estimation d'erreur n'est pas seulement pour sélectionner le meilleur modèle, mais aussi pour s'assurer que le nombre d'échantillons est suffisant.

3.5.3.2 Méthodes de validation croisée

Nous proposons dans cette section une autre méthodologie de validation de modèle nommée *validation croisée*. Celle-ci est plus coûteuse en temps de calcul consacré à l'apprentissage, mais elle permet d'estimer plus précisément la qualité du modèle et n'a pas besoin de base de validation $\mathcal{V}$.

L'idée principale de la validation croisée est d'enlever une observation $y(x_i)$ (revenons les notations dans la section 2.1), puis prédire cette valeur par le krigeage construit sur $m-1$ observations restées. Notons $\tilde{y}_{-i}(x_i)$ la valeur prédite de $y(x_i)$, et

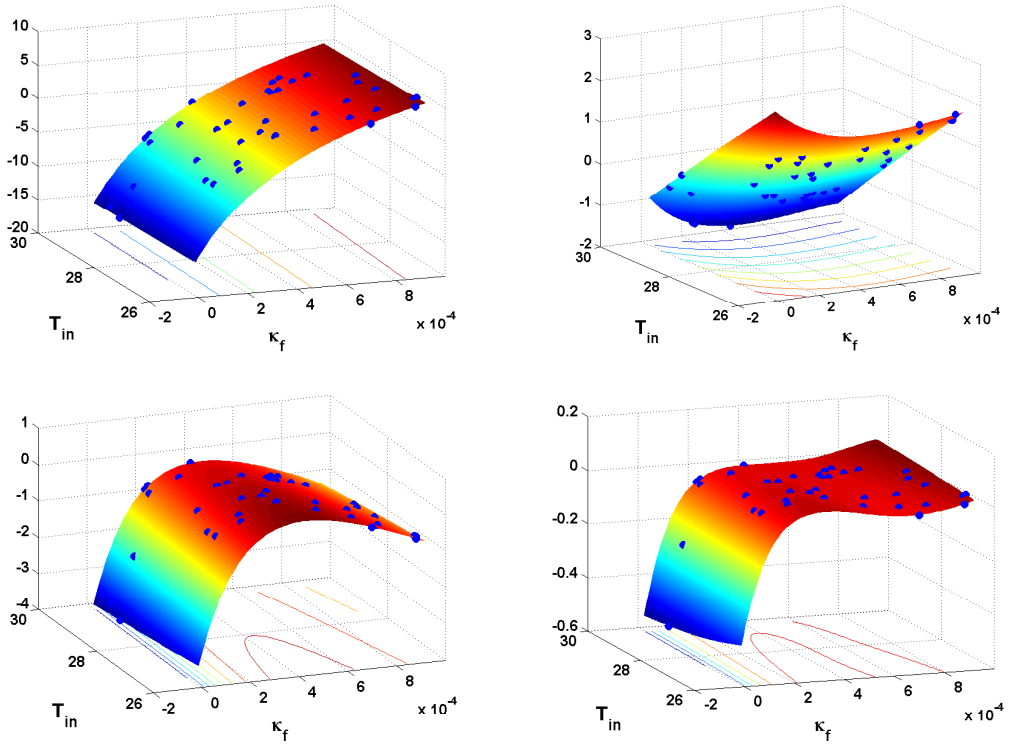

FIGURE 3.22 – L'approximation par krigeage des coefficients POD, $a^1(\theta)$ (en haut à gauche), $a^2(\theta)$ (en haut à droite), $a^3(\theta)$ (en bas à gauche), $a^4(\theta)$ (en bas à droite)

$\phi_{-i}(x_i)$ la variance (l'écart-type au carré) au point x_i (calculé par (2.12)). L'indice $-i$ désigne que l'observation i n'est pas prise en compte pour construire le krigeage. On définit une erreur normalisée (ou *standardized cross-validated residual*) :

$$e_i = \frac{\tilde{y}_{-i}(x_i) - y(x_i)}{\sqrt{\phi_{-i}(x_i)}} \tag{3.35}$$

Selon [Jones *et al.*, 1998], le modèle de krigeage est valide si les e_i doivent être à peu près dans l'intervalle $[-3, +3]$.

Nous reprenons l'exemple numérique dans la section précédente pour calculer les erreurs normalisées des modèles de krigeages pour chaque coefficient retenus précédemment suite à notre procédure de validation. Le plan d'expérience consiste en $m = 40$ observations. Les erreurs normalisées de ces observations montrées dans la figure 3.23 tombent dans l'intervalle $[-3, +3]$. Ceci indique que les modèles de krigeage retenus précédemment pour chaque coefficient sont bien construits sur la base de l'apprentissage.

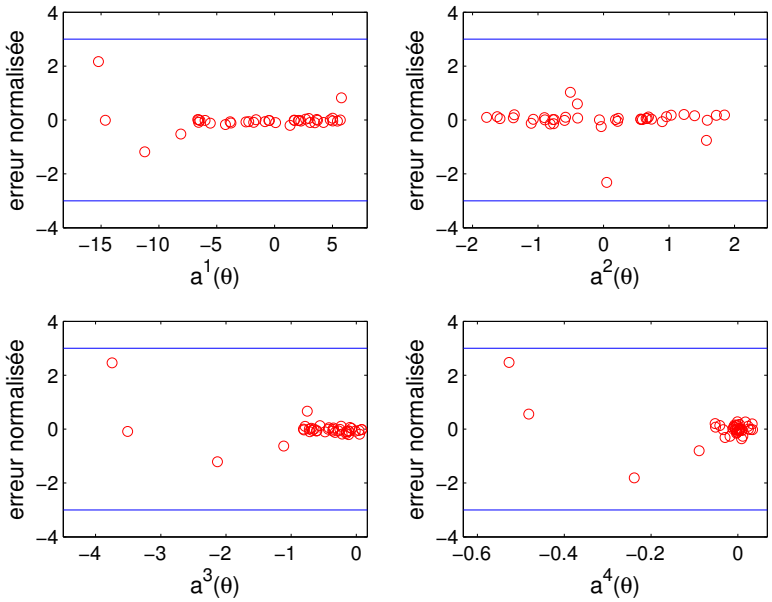

FIGURE 3.23 – Erreurs normalisées d'approximation pour les quatre premiers coefficients POD

3.5.4 Estimation du temps de calcul

Le coût de calcul CPU a été analysé. Le cas test a été réalisé sur un ordinateur Intel Core I7. Le modèle d'éléments finis est implanté sous `FreeFem++3.10`[2]. L'algorithme POD-ISAT est implanté sous `Matlab`[3]. Le temps total t_{tot} de calcul pour que les EOAs remplissent tout l'espace des paramètres de conception, se décompose en un temps de calcul hors ligne t_{off} et un temps d'enrichissement adaptatif t_{enr} ($t_{tot} = t_{off} + t_{enr}$). Les deux étapes de l'algorithme POD-ISAT font appel à des tâches élémentaires qui se déclinent en :

- le temps t_{fem} pour déterminer la solution exacte par le modèle d'éléments finis,
- le temps de la récupération t_{ret} pour chercher dans la base de données une bonne EOA et pour évaluer la solution approchée par modèle réduit local (3.28),
- le temps t_{eoa} pour construire une EOA dans la section 3.4.2.3.
- le temps t_{krg} pour établir des méta-modèles de krigeage de coefficients POD locaux dans la section 3.4.2.4.
- le temps pour établir des méta-modèles de krigeage dans la section 3.4.2.2 et pour corriger l'EOA dans la section 3.4.2.5 sont négligeable.

Ces coûts sont montrés dans le tableau 3.2 pour les cas à deux et quatre paramètres. Pour le cas test à deux paramètres, il faut $t_{tot} = 5.5$ heures pour évaluer 200 requêtes. Pour le cas test avec quatre paramètres, il faut $t_{tot} = 15.2$ heures pour évaluer 420

2. http ://www.freefem.org/ff++/
3. http ://www.mathworks.fr/products/matlab/

requêtes. Une fois que le modèle POD-ISAT a été complété, on observe qu'un calcul de POD-ISAT (étape de récupération) coûte 0.98sec (pour deux paramètres) et 1.1sec (pour quatre paramètres), lorsqu'un calcul éléments finis coûte 1000 sec. Le modèle réduit est donc 1000/0.98 ou 1000/1.1 $\sim 10^3$ fois plus rapide que modèle de référence éléments finis pour chaque calcul.

TABLE 3.2 – CPU costs

	Deux paramètres	Quatre paramètres
t_{eoa}	316 sec	408 sec
t_{krg}	95 sec	151 sec
t_{fem}	1000 sec	1000 sec
t_{ret}	0.98 sec	1.1 sec
t_{off}	11233 sec	25031 sec
t_{enr}	8466 sec	29621 sec

Il faut remarquer que $\epsilon_{tol} > 1$ peut varier selon les besoins de l'utilisateur. Si on augmente ϵ_{tol}, l'erreur du modèle POD-ISAT augmente, mais la base de données diminue, car la taille de EOA augmente. Donc, le temps pour compléter le modèle réduit diminue. Ce point est très important, puisque l'utilisateur peut manipuler l'erreur de modèle réduit avec la taille de la base de données. L'algorithme POD-ISAT est donc une technique robuste permettant d'évaluer rapidement ou de visualiser les solutions paramétriques avec une erreur contrôlable.

3.6 Conclusion

Dans ce chapitre, une technique de réduction de modèle adaptative, semi-intrusive, combinant POD et ISAT, a été présentée et testée pour la conception numérique d'air conditionné dans une cabine d'avion. Les résultats ont montré que cette technique est efficace et précise. Dans cette méthode, la précision est contrôlée à travers le résidu, mais pas directement sur la solution. Cela conduit par exemple à considérer des approches basées sur des estimateurs d'erreur *a priori* faisant directement intervenir la solution.

Il faut noter que dans ce cas d'application, le modèle réduit est établi pour la température. Le problème de minimisation du résidu était linéaire, cela permettait de trouver rapidement la solution. Dans le cas non linéaire, par exemple pour les champs vitesse, le calcul du résidu optimal peut augmenter significativement le temps de

calcul associé. L'utilisation d'une architecture de calcul parallèle peut être nécessaire pour la mise en oeuvre de cette méthode.

Dans le présent cas d'application on a supposé que le champ de vitesse ne change pas brutalement dans région de confiance ("trust region" en anglais). Dans d'autres cas si le champ de vitesse change vite on doit considérer que la vitesse et la température sont des solutions qu'il faut simultanément calculer. Le résidu se compose donc de deux parties : une est le résidu de l'équation de quantité de mouvement et l'autre est le résidu de l'équation de diffusion pour la température [Leblond *et al.*, 2011].

Dans le cadre de cette thèse, nous nous sommes limités à l'utilisation de l'algorithme POD-ISAT aux cas de problèmes stationnaires. Néanmoins, on peut probablement étendre la méthode aux cas des problèmes instationnaires en utilisant une technique suggérée par [Amsallem *et al.*, 2012]. Par ailleurs, nous nous sommes plutôt intéressés à résoudre un problème d'évolution en combinant une technique de calcul parallèle en temps avec la méthode réduction d'ordre. Cette idée est dévellopée dans le chapitre 4 suivant.

Chapitre 4

Problèmes d'évolution : Application de l'algorithme pararéel dans un cadre de modèle réduit

Sommaire

4.1 Introduction à l'algorithme pararéel

L'algorithme pararéel introduit par [Lions *et al.*, 2001], est une méthode efficace pour résoudre des équations différentielles ordinaires (EDO) et des équations aux dérivées partielles (EDP) dépendantes du temps, en utilisant des ordinateurs ayant plusieurs processeurs. Le principe est de décomposer le domaine temporel en plusieurs sous-domaines (ou intervalles) grossiers et d'effectuer des itérations sur la résolution de chaque sous-domaine jusqu'à ce que la solution converge vers la solution globale. Dans chaque sous-domaine temporel, la solution initiale est inconnue, sauf pour le premier intervalle où elle est déterminée par la condition initiale du problème global. A la première itération, ces inconnues sont déterminées par un propagateur grossier, puis on utilise un propagateur exact pour les calculs de chaque sous-domaine de manière parallèle. La différence entre la solution à la fin d'intervalle et celle au début d'intervalle suivant est utilisée pour corriger la prédiction à l'itération suivante.

L'algorithme a beaucoup attiré l'attention au cours des dernières années, en particulier en ce qui concerne les méthodes de décomposition de domaines [con, SD]. Il a été appliqué entre autre à la résolution des équations de Navier-Stokes [Fischer *et al.*, 2004], aux problèmes hyperboliques [Gander, 2008, Gander et Vandewalle, 2007, Farhat et Chandesris, 2003], aux mathématiques financières [Bal et Maday, 2002], aux problèmes d'interaction fluide-structure [Cortial et Farhat, 2007], à la simulation de réservoir [Garrido *et al.*, 2003], à la cinétique chimique [Blouza *et al.*, 2010, Engblom, 2008] et à la chimie quantique [Maday *et al.*, 2007].

L'algorithme pararéel a été révisé par [Bal et Maday, 2002] afin de le rendre applicable à la majorité des solveurs linéaires et non linéaires.

Simultanément, [Farhat et Chandesris, 2003] ont proposé une approche différente nommée PITA (Parallel Implicit Time-integration Algorithm). Il a été démontré que les algorithmes PITA et pararéel sont identiques dans le cas linéaire, mais différents dans le cas non linéaire [Cortial et Farhat, 2007]. Numériquement, les deux méthodes sont instables pour l'application de l'oscillateur linéaire. Le PITA original offre un potentiel significatif pour accélérer le temps de calcul sur des processeurs parallèles en cas de problèmes de premier rang hyperboliques (comme des applications à la dynamique de gaz). Cependant, pour les problèmes hyperboliques de second ordre (comme l'application à la dynamique des structures linéaires), la méthodologie PITA génère le phénomène de battement d'un parasite qui limite strictement son pas de temps de stabilité. C'est le même problème pour l'algorithme pararéel [Bal et Maday, 2002, Staff et Rønquist, 2003]. Pour le problème hyperbolique de second ordre, [Bal, 2005] a recommandé d'utiliser la dissipation artificielle pour rétablir la

stabilité numérique. Cependant, la dissipation artificielle est préjudiciable à la précision. [Cortial et Farhat, 2007] a proposé un nouveau algorithme PITA basé sur l'algorithme original et lié à l'algorithme pararéel. La solution d'EDO découlant de la semi-discrétisation des équations linéaires de second ordre des problèmes hyperboliques calculés par cette nouvelle méthode et par l'algorithme pararéel, est illustrée par deux exemples d'applications. Le premier est la vibration d'une structure simple de l'espace en trois dimensions avec 348 degrés de liberté. La seconde implique un modèle éléments finis d'une structure d'avion de combat avec 168799 degrés de liberté. Les résultats numériques ont montré une instabilité ainsi qu'un échec pour converger vers la solution séquentielle après trois itérations de l'algorithme pararéel. Au contraire, les résultats d'un même problème en utilisant le nouvel algorithme PITA, ont montré la convergence vers la solution séquentielle, celle-ci est réalisée en seulement deux itérations. Il est clair que le nouvel algorithme PITA converge plus rapidement que le pararéel en terme d'itérations, mais l'auteur n'a pas comparé le speed-up ou temps de calcul de chaque itération de ces deux méthodes.

4.1.1 Construction de l'algorithme pararéel

Cette section rappelle la construction de l'algorithme pararéel avec son interprétation comme une méthode de Newton. On souhaite résoudre le problème différentiel aux conditions initiales :

$$\begin{aligned} &\dot{u} = f(u), \quad t \in [0, T], \\ &u(0) = u^0. \end{aligned} \tag{4.1}$$

Soit ΔT un pas de temps grossier. On définit le propagateur exact $\mathbf{F}$ sur un intervalle de temps ΔT défini par $\mathbf{F}(v) = w(\Delta T)$ où w est la solution de

$$\begin{aligned} &\dot{w} = f(w), \quad t \in [0, \Delta T], \\ &w(0) = v. \end{aligned} \tag{4.2}$$

On désignera un propagateur grossier par $\mathbf{G}$ qui est un opérateur approché de $\mathbf{F}$. On note u_n des approximations de $u(n\Delta T)$, $n \in \{0, ..., N\}$, et calibre ΔT de façon que $N.\Delta T = T$. On note $U = (u_1, ..., u_N)^T$ et l'application du saut $R(U) = (u_1 - \mathbf{F}(u_0), ..., u_N - \mathbf{F}(u_{N-1}))^T$. On souhaite résoudre l'équation algébrique

$$R(U) = 0 \tag{4.3}$$

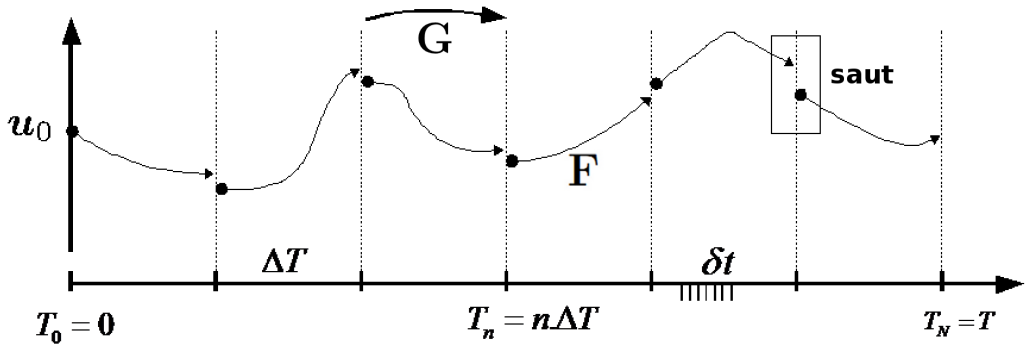

FIGURE 4.1 – La décomposition des domaines temporels

ayant la solution $U = [u(\Delta T), ..., u(N\Delta T)]$. L'algorithme de Newton-Raphson appliqué à $R(U) = 0$ s'écrit comme suit :

$$U^{k+1} = U^k - \left[DR(U^k)\right]^{-1} R(U^k), \quad k \geq 0$$

avec U^0 donné. La matrice jacobienne de R a une structure triangulaire creuse :

$$DR(U) = \begin{pmatrix} I & & & (0) \\ -D\mathbf{F}(u_1) & I & & \\ & \ddots & \ddots & \\ (0) & & -D\mathbf{F}(u_{N-1}) & I \end{pmatrix}.$$

On peut tirer avantage de cette structure triangulaire inférieure pour une écriture explicite séquentielle de l'itérée de l'algorithme de Newton. La $(n+1)$ième ligne du système

$$DR(U^k)\left[U^{k+1} - U^k\right] = -R(U^k)$$

s'écrit

$$-D\mathbf{F}(u_n^k)(u_n^{k+1} - u_n^k) + u_{n+1}^{k+1} - u_{n+1}^k = -\left[u_{n+1}^k - \mathbf{F}(u_n^k)\right].$$

On obtient ainsi l'algorithme

$$u_{n+1}^{k+1} = \mathbf{F}(u_n^k) + D\mathbf{F}(u_n^k)(u_n^{k+1} - u_n^k). \tag{4.4}$$

L'approximation pararéel consiste à approcher le terme $D\mathbf{F}(u_n^k)(u_n^{k+1} - u_n^k)$ par la méthode différence finie en ustilisant le propagateur grossier $\mathbf{G}$:

$$D\mathbf{F}(u_n^k)(u_n^{k+1} - u_n^k) \simeq \mathbf{G}(u_n^{k+1}) - \mathbf{G}(u_n^k), \tag{4.5}$$

L'algorithme pararéel s'écrit comme suit :

$$u_{n+1}^{k+1} = \mathbf{G}(u_n^{k+1}) + \mathbf{F}(u_n^k) - \mathbf{G}(u_n^k). \tag{4.6}$$

On peut interpréter que l'algorithme (4.6) rajoute une correction pour la prédiction du propagateur grossier $\mathbf{G}(u_n^{k+1})$. Cette correction est déterminée par la solution approchée de l'itération précédente. C'est la différence entre les solutions calculées par le proparateur exacte et le proparateur grossier sur le sous-domaine $[T_n, T_{n+1}]$, avec une condition initiale de u_n^k.

Remarque 1. *L'algorithme parareal (4.6) est une version générale mais pratique et applicable pour les cas linéaires ou non linéaires. Cette version est différente de la version originale introduite dans [Lions et al., 2001] qui est appliquée pour le cas EDP linéaire et EDP non-linéaire en utilisant la linéarisation (selon [Bal et Maday, 2002] et [Gander et Vandewalle, 2007] p.558).*

Remarque 2. *La différence entre les versions des algorithmes pararéels vient de l'approximation de la matrice jacobienne de $\mathbf{F}(u_n^k)$. En effet, ces matrices peuvent être calculées par la résolution de l'équation variationnelle :*

$$\left(D\mathbf{F}(u_n^k)\right)'(t) = f'(u)D\mathbf{F}(u_n^k), \quad D\mathbf{F}(u_n^k)(t = t_n) = I, \quad n = 0, 1, ..., N-1. \tag{4.7}$$

où I est la matrice d'identité. Si on résout ces équations avec les pas de temps grossier ΔT, on va obtenir le même algorithme que dans [Lions et al., 2001].
Puisque (4.7) est un système d'équation matricielle mais on a seulement besoin des vecteurs obtenus en appliquant les jacobiennes sur une direction $u_n^{k+1} - u_n^k$, il est plus efficace de résoudre le système d'équations vectorielles suivant :

$$v_n' = f'(u_n)v_n, \quad v_n(t_n) = u_n^{k+1} - u_n^k, \quad n = 1, ..., N-1. \tag{4.8}$$

pour obtenir :

$$v_n(t_{n+1}) = D\mathbf{F}(u_n^k)\left(u_n^{k+1} - u_n^k\right). \tag{4.9}$$

Cette approche est utilisée pour trouver l'algorithme PITA introduit par [Farhat et Chandesris, 2003]. PITA coïncide avec l'algorithme pararéel de [Lions et al., 2001], dans le cas linéaire.

Remarque 3. *On peut approcher la matrice jacobienne $D\mathbf{F}$ par la nouvelle méthode qu'on va proposer dans le chapitre 5. L'idée est de résoudre l'équation (4.3) par la méthode de Broyden. Dans ce cas, les matrices jacobiennes $D\mathbf{F}$ sont mis à jour pour chaque itération de l'algorithme pararéel. L'efficacité de la méthode est étudiée dans la section 5.3.*

4.1.2 Résumé de l'algorithme pararéel

L'algorithme pararéel est résumé par le pseudo code suivant :

- Étape 0 : On fournit une valeur initiale de U calculée par le propagateur grossier $\mathbf{G}$:

  pour $n = 0$ *à* $(N-1)$
  $u_{n+1}^0 = \mathbf{G}(u_n^0)$
  fin
  sachant que $u_0^0 = u_0$

Pour $k \geq 1$

- Étape 1 : Calculer en parallèle pour chaque intervalle $\mathbf{F}(u_n^k), n = 0, ..., (N-1)$.

- Étape 2 : Calculer séquentiellement les composants de U^{k+1} sachant que $u_0^{k+1} = u_0^k = u_0^0 = u_0$

  pour $n = 0 : (N-1)$
  Calculer et stocker $\mathbf{G}(u_n^{k+1})$
  Puis calculer $u_{n+1}^{k+1} = \mathbf{F}(u_n^k) + \mathbf{G}(u_n^{k+1}) - \mathbf{G}(u_n^k)$
  fin

  Arrêter quand $\| U^{k+1} - U^k \| \leq \epsilon$

Le schéma présenté par la figure 4.2 permet de mieux comprendre la procédure de l'algorithme pararéel.

4.1.3 Analyse de speed-up

Le speed-up désigne le gain de temps d'exécution de l'algorithme pararéel utilisant plusieurs processeurs par rapport à celui de l'algorithme séquentiel utilisant un processeur. Notons :

- $T_{\mathbf{F}}, T_{\mathbf{G}}$, le temps d'exécution respectivement du propagateur exact et du propagateur grossier sur un intervalle ΔT.
- $\tau_1 = NT_{\mathbf{F}}$, le temps d'exécution de l'algorithme séquentiel utilisant un processeur.
- τ_P temps d'exécution de l'algorithme pararéel utilisant P processeurs. Supposons que le temps de communication entre les processeurs est négligeable, on a : $\tau_P = NT_{\mathbf{G}} + K(\frac{T_{\mathbf{F}}N}{P} + T_{\mathbf{G}}N)$ (voir la figure 4.2).

Le speed-up est donc :

$$S_P := \frac{\tau_1}{\tau_P} = \frac{P}{\frac{T_{\mathbf{G}}}{T_{\mathbf{F}}}P + K\left(1 + \frac{T_{\mathbf{G}}}{T_{\mathbf{F}}}P\right)} \tag{4.10}$$

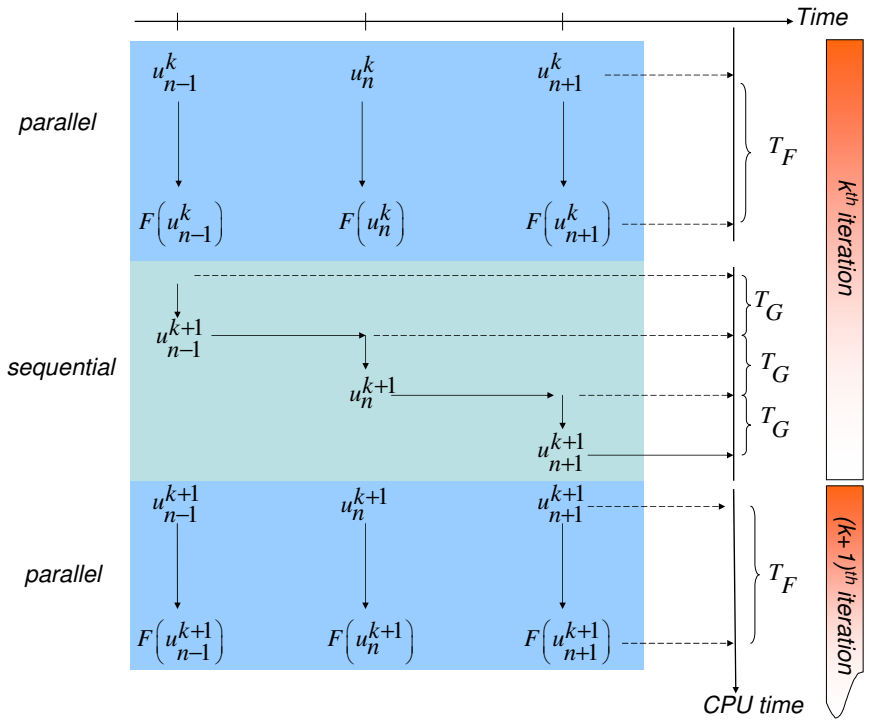

FIGURE 4.2 – Schéma de l'algorithme pararéel

On utilisera le nombre de processeurs (P) en fonction de nombre de sous domaine temporel (N), $P = N = \frac{T}{\Delta T}$. D'où $T_{\mathbf{F}} = \tau_1/P$. On remplace $T_{\mathbf{F}}$ dans (4.10), le speed-up à la K ème itération ne dépend que de P en supposant que $T_{\mathbf{G}}$ est indépendant.

$$S_P(P) = \frac{P}{K + (1+K)\,\frac{T_{\mathbf{G}}}{\tau_1} P^2} \tag{4.11}$$

La formule (4.11) permet de déterminer facilement le speed-up maximal $(S_P)_{max}$ ainsi que le nombre optimal de processeurs P_{opt} :

$$(S_P)_{max} = \frac{1}{2}\sqrt{\frac{\tau_1}{K\,(K+1)\,T_{\mathbf{G}}}}; \tag{4.12}$$

$$P_{opt} = \sqrt{\frac{K}{K+1}\frac{\tau_1}{T_{\mathbf{G}}}}; \tag{4.13}$$

Le cas idéal si $\frac{T_{\mathbf{G}}}{\tau_1}$ tend vers zéros, dans ce cas-là, selon (4.11) le speed-up est en ordre de $\frac{P}{K}$ et augmente linéairement avec le nombre de processeurs utilisés. Dans

le cas particulier, on suppose que si la fraction $\frac{\tau_1}{T_{\mathbf{G}}} = 10^3$, et $K = 3$ (où l'erreur de solution calculée par pararéel est acceptable), la figure 4.3 se montre la variation de speed-up selon le nombre de processeurs utilisés.

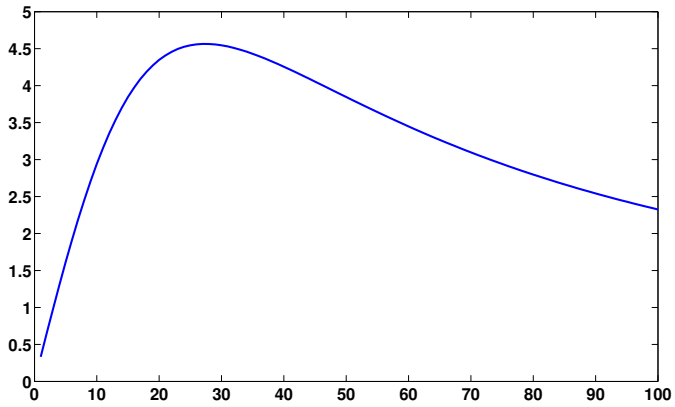

FIGURE 4.3 – Speed-up de pararéel selon le nombre de processeurs utilisés, où $K = 3$, $\frac{\tau_1}{T_{\mathbf{G}}} = 10^3$.

En réalité, P_{opt} dans la formule (4.13) n'est pas véritable parce que $T_{\mathbf{G}}$ dépend aussi de P et n'est pas identique sur tous les domaines. Mais cette formulation permet d'approcher et estimer le nombre optimal de processeurs utilisés. C'est utile pour déterminer le nombre de sous-domaines ainsi que le nombre de processeurs nécessaires. La courbe de la figure 4.3 montre que le speed-up augmente vite avant d'atteindre la valeur maximale, puis il diminue doucement selon P . Cela conduit le calculateur à choisir le nombre de processeurs au moins égale à celui estimé par la formule (4.13).

[Aubanel, 2011] a proposé une planification optimale des tâches sur l'algorithme pararéel original avec deux nouveaux algorithmes. Le premier se base sur le paradigme "manager-worker" des processeurs et sur la superposition de deux étapes séquentielles et l'étape parallèle. Le deuxième se base sur la distribution de calcul grossier sur tous les processeurs. Ce deuxième algorithme permet d'éviter le coût de stockage des données et de minimiser les informations d'échange entre les processeurs.

[Bal, 2003] a proposé et discuté en détail le speed-up et l'efficacité du système. En particulier, il a proposé un algorithme pararéel multi-niveaux pour profiter le nombre de processeurs disponibles ainsi que pour augmenter le speed-up de calcul.

4.1.4 Théorème de la convergence

Le théorème de la convergence de l'algorithme pararéel est présenté par [Lions *et al.*, 2001], puis développé dans le cas général par [Bal et Maday, 2002, Bal, 2005]. [Gander et Vandewalle, 2007, Gander et Petcu, 2008, Gander et Hairer, 2008, Gander, 2008] introduisent de nouveaux résultats sur le théorème de convergence puis les généralisent aux problèmes d'EDP.

Nous nous intéressons au théorème présenté dans [Gander et Hairer, 2008] avec les hypothèses suivantes :

- *Hypothèse 1 :* ΔT est suffisant petit pour que $\mathbf{G}_n$ soit une approximation de $\mathbf{F}_n$ avec l'erreur d'ordre p :

$$\mathbf{F}_n(u) - \mathbf{G}_n(u) = c_{p+1}(u)\Delta T^{p+1} + c_{p+2}(u)\Delta T^{p+2} + ..., \tag{4.14}$$

  où :

  + $\mathbf{F}_n(u)$, $\mathbf{G}_n(u)$ sont respectivement la solution fine et grossière de (4.1) sur l'intervalle $[T_n, T_{n+1} = T_n + \Delta T]$, avec valeur initiale $u(T_n) = u$.
  + Les coefficients $c_{p+1}, c_{p+2}, ...$ sont continûment différentiables.

  Si la fonction f dans (4.1) est suffisante régulière et $\mathbf{G}_n$ est un solveur de type Runge Kutta, la condtion (4.14) est vérifiée.
- *Hypothèse 2 :* Soit $\|\cdot\|$ une norme euclidienne, $\mathbf{G}$ remplit la condition de Lipschitz :

$$\|\mathbf{G}_n(u) - \mathbf{G}_n(v)\| \leq (1 + C_2\Delta T)\|u - v\| \tag{4.15}$$

- *Hypothèse 3 :* $\mathbf{G}_n(u)$ a une l'erreur borné par $C_3\Delta T^{p+1}$.

Théorème 4.1.4.1. Soit $\mathbf{F}_n(u)$ la solution exacte sur le sous-domaine temporel $[T_n, T_{n+1}]$, et $\mathbf{G}_n(u)$ est une solution approchée qui répond aux conditions (4.14) (4.15) (4.1.4). Alors, l'erreur de la solution calculée par l'algorithme pararéel (4.6) à l'itération k est bornée, et :

$$\begin{aligned} \|u(T_n) - u_n^k\| &\leq \frac{C_3}{C_1}\frac{(C_1\Delta T^{p+1})^{k+1}}{(k+1)!}(1 + C_2\Delta T)^{n-k-1}\prod_{j=0}^{k}(n-j) \\ &\leq \frac{C_3}{C_1}\frac{(C_1 T_n)^{k+1}}{(k+1)!}e^{C_2(T_n - T_{k+1})}\Delta T^{p(k+1)}. \end{aligned} \tag{4.16}$$

Preuve : En utilisant la définition de l'algorithme pararéel (4.6), on calcule l'erreur de la solution approchée à la $(k+1)$ème itération au moment T_{n+1}.

$$u(T_{n+1}) - u_{n+1}^{k+1} = \mathbf{F}_n(u(T_n)) - \mathbf{F}_n(u_n^k) - \mathbf{G}_n(u_n^{k+1}) + \mathbf{G}_n(u_n^k).$$

En rajoutant et soustrayant $\mathbf{G}_n(u(T_n))$ on obtient :

$$\begin{aligned} u(T_{n+1}) - u_{n+1}^{k+1} &= \mathbf{F}_n(u(T_n)) - \mathbf{G}_n(u(T_n)) && (4.17)\\ &\quad -\mathbf{F}_n(u_n^k) + \mathbf{G}_n(u_n^k) && (4.18)\\ &\quad +\mathbf{G}_n(u(T_n)) - \mathbf{G}_n(u_n^{k+1}). && (4.19) \end{aligned}$$

On utilise la première hypothèse (4.14) pour les deux premières lignes, et la deuxième hypothèse (4.15) pour la troisième ligne. On obtient la norme :

$$\|u(T_{n+1}) - u_{n+1}^{k+1}\| \leq C_1 \Delta T^{p+1} \|u(T_n) - u_n^k\| + (1 + C_2 \Delta T) \|u(T_n) - u_n^{k+1}\|.$$

Cela nous a conduit à étudier une suite récurrente de la forme :

$$\begin{aligned} e_{n+1}^{k+1} &= \alpha e_n^k + \beta e_n^{k+1}. && (4.20)\\ e_{n+1}^0 &= \gamma + \beta e_n^0, \quad avec \quad n \geq 0. && (4.21) \end{aligned}$$

Où $\alpha = C_1 \Delta T^{p+1}$, $\beta = 1 + C_2 \Delta T$, et $\gamma = C_3 \Delta T^{p+1}$ (selon la troisième hypothèse). Alors on a : $\|u(T_n) - u_n^k\| \leq e_n^k$ Supposons qu'on a une variable non nulle $\varepsilon < 1$, on va multiplier avec deux membres de (4.20) par ε^{n+1} puis sommer suivant n. On obtiendra une fonction génératrice

$$\sigma_k(\varepsilon) := \sum_{n \geq 0} e_n^k \varepsilon^n$$

qui vérifie

$$\begin{aligned} \sigma_{k+1}(\varepsilon) &= \alpha\varepsilon\sigma_k + \beta\varepsilon\sigma_{k+1}(\varepsilon)\\ \sigma_0(\varepsilon) &= \gamma \frac{\varepsilon}{1-\varepsilon} + \beta\varepsilon\sigma_0(\varepsilon) \end{aligned}$$

On en déduit σ_k sous la forme :

$$\sigma_k(\varepsilon) = \frac{\gamma \alpha^k \varepsilon^{k+1}}{(1-\varepsilon)(1-\beta\varepsilon)^k}$$

Comme $\varepsilon < 1$ et $\beta > 1$, on peut remplacer $(1-\varepsilon)$ par $(1-\beta\varepsilon)$, cela fait croître les σ_k. Puis, on utilise la formule du binôme négatif.

$$\frac{1}{(1-\beta\varepsilon)^{k+2}} = \sum_{j \geq 0} \binom{k+1+j}{j} \beta^j \varepsilon^j = \sum_{j \geq 0} C_j^{k+1+j} \beta^j \varepsilon^j.$$

On en déduit :

$$\sigma_k(\varepsilon) \leq \gamma\alpha^k \sum_{j\geq 0} C_j^{k+1+j}\beta^j\varepsilon^{k+1+j} = \gamma\alpha^k \sum_{n\geq 0} C_n^{k+1}\beta^{n-k-1}\varepsilon^n.$$

En comparant cette formule avec la définition de $\sigma_k(\varepsilon)$ on a :

$$e_n^k \leq \gamma\alpha^k\beta^{n-k-1}C_n^{k+1},$$

Ce qui conclut la preuve.

Pour l'analyse détaillée, l'auteur recommande de lire l'article de [Gander et Vandewalle, 2007]. Ils ont étudié la convergence séparément pour les EDO et EDP. Dans tous les cas, ils ont prouvé que l'algorithme pararéel (4.6) a une convergence superlinéaire sur un domaine temporel fini, et une convergence linéaire sur un domaine temporel infini. Cependant, les résultats numériques pour l'équation d'advection pure sur un domaine temporel borné montrent que l'algorithme pararéel converge linéairement. Dans un article récent [Gander, 2008], Gander a particulièrement développé le résultat de convergence pour l'équation de convection pure et l'équation d'onde de second ordre en utilisant la technique des caractéristiques. Pour ces applications, l'algorithme est performant si le propagateur grossier remplit certaines des hypothèses (voir [Gander, 2008]). Sinon, il faut modifier l'algorithme pararéel ou utiliser l'idée de la modification proposée dans [Farhat et Chandesris, 2003, Cortial et Farhat, 2007, Gander et Petcu, 2008].

Bal a prouvé dans [Bal, 2005] que l'algorithme pararéel permet de remplacer un solveur ($\mathbf{G}_n(u)$) d'ordre de précision p par un solver parallèle d'ordre de précision $p(k+1)$, en ajoutant une hypothèse suivante :
Hypothèse 4 : Supposons que la solution de (4.1) est stable dans le sens :

$$\|u(t)\| \leq C\|u_0\|, \tag{4.22}$$

Où C est une constante indépendante de u_0 et $t \subset (0, T)$.

Théorème 4.1.4.2. Sous les hypothèses 1-4, l'ordre de précision de l'algorithme pararéel (4.6) est $p \times k$, et :

$$\|u(T_n) - u_n^k\| \leq C(\Delta T)^{pk}\|u_0\|, \tag{4.23}$$

Où C est indépendante de ΔT et u_0.

Preuve : On peut justifier par l'induction. Pour $k = 0$, la formule (4.23) est vérifié avec hypothèse que le solveur grossier est en ordre de p.

On suppose pour k : $\|u(T_n) - u_n^k\| \leq C(\Delta T)^{pk}\|u_0\|$. On va justifier que c'est vrai pour $k+1$:

$$\begin{aligned}
\|u(T_{n+1}) - u_{n+1}^{k+1}\| &= \|\mathbf{F}_n(u(T_n)) - \mathbf{G}_n(u(T_n)) \\
&\quad -\mathbf{F}_n(u_n^k) + \mathbf{G}_n(u_n^k) \\
&\quad +\mathbf{G}_n(u(T_n)) - \mathbf{G}_n(u_n^{k+1})\| \\
&\leq C_1\Delta T^{p+1}\|u(T_n) - u_n^k\| + (1 + C_2\Delta T)\,\|u(T_n) - u_n^{k+1}\| \\
&\leq C_1(\Delta T)^{p(k+1)+1}\|u_0\| + (1 + C_2\Delta T)\,\|u(T_n) - u_n^{k+1}\|.
\end{aligned}$$

Il est alors un calcul de routine pour obtenir 4.23.

4.1.5 Différents choix de propagateur grossier

Le propagateur grossier $\mathbf{G}$ joue un rôle important dans l'efficacité de l'algorithme pararéel. Une méthode standard consiste à choisir $\mathbf{G}$ similaire à $\mathbf{F}$ avec le pas de temps (ou maillage temporel) plus grand.

Une autre méthode est de choisir $\mathbf{G}$ comme le propagateur fin $\mathbf{F}$ mais avec le maillage spatial grossier. Notons que cette méthode est cohérente avec celle standard pour la condition de stabilité. Cependant, elle est dédiée seulement aux problèmes EDP dont le domaine spatial est large. Dans ce cas-là, l'algorithme pararéel s'écrit sous la forme présentée par Fischer et al. [Fischer *et al.*, 2004] :

$$u_{n+1}^{k+1} = \Pi_H^h\mathbf{G}(\Pi_h^H u_n^{k+1}) + \mathbf{F}(u_n^k) - \Pi_H^h\mathbf{G}(\Pi_h^H u_n^k). \tag{4.24}$$

Où l'opérateur Π_H^h permet de projeter la solution de maillage grossier (H) au maillage fin (h). Au contraire, l'opérateur Π_h^H permet d'extraire la solution de maillage fin à maillage grossier.

La dernière méthode est de remplacer $\mathbf{G}$ par un modèle physique simplifié. C'est un cas particulier de la méthode qu'on va présenter dans la section 4.2. Le solveur grossier $\mathbf{G}$ sera remplacé par un modèle réduit.

4.2 Remplacement du propagateur grossier par un modèle réduit

L'efficacité de l'algorithme pararéel dépend crucialement de la qualité de propagateur grossier $\mathbf{G}$. En général, $\mathbf{G}$ doit donner des informations qualitatives sur la solution exacte mais aussi maintenir la rapidité. L'idée de remplacement d'un

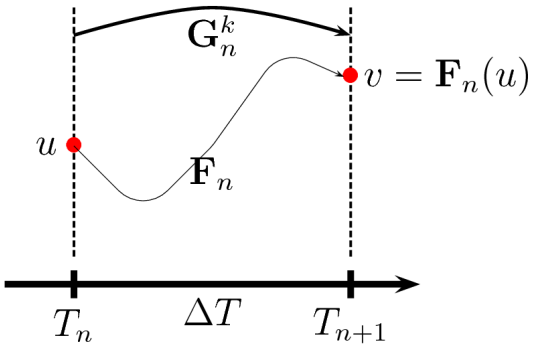

FIGURE 4.4 – Le solveur gorssier est un modèle réduit de solveur fin.

propagateur grossier $\mathbf{G}$ par un modèle réduit a beaucoup attiré l'attention.

Cette idée est proposée la première fois par [Baffico *et al.*, 2002], dans le contexte de la résolution du système différentiel pour dynamique moléculaire ab initio. Elle est aussi appliquée à résoudre des problèmes cinétiques chimiques [Maday, 2007, Blouza *et al.*, 2010]. Dans la littérature, il existe largement les modèles simples basés sur les propriétés physiques. Par exemple, le solveur grossier utilisé dans [Baffico *et al.*, 2002] se base sur l'approche d'ondes planes utilisant 16 fonctions de base. Comme la résolution de ce solveur est négligeable et l'algorithme converge à 4 ème itération en utilisant 30 processeurs, la vitesse de calcul gagne un facteur de $7.5 = 30/4$.

Dans cette section nous proposons deux pistes à combiner la méthode de réduction de modèle avec l'algorithme pararéel. La première vise au cas EDP non paramétré. Dans ce cas le modèle réduit se construit de fur à mesure grâce à l'information de solveur fin calculé à chaque itération. La deuxième sera appliquée pour le cas EDP paramétré. Le modèle réduit se construit antérieurement en étape "offline" grâce à un plan d'expérience numérique en utilisant le solveur fin.

4.2.1 Pour le problème EDP non paramétré

L'idée est d'améliorer le propagateur grossier grâce à des informations venant du calcul en parallèle des solveurs fins à l'itération précédente. Cette idée est déjà proposée par Charbel pour le nouveau PITA [Farhat et Chandesris, 2003, Cortial et Farhat, 2007] appliqué aux problèmes hyperboliques de dynamique de structure. Puis, Cortial a étendu l'algorithme pour qu'il puisse être appliqué au cas dynamique de structure non linéaire [Cortial et Farhat, 2009]. Gander a révisé cet algorithme dans le contexte de l'algorithme pararéel et de l'espace Krylov [Gander et Petcu, 2008] avec le nom "Krylov-subspace-en-hanced Parareal" ou KSE-Parareal (abbrégé par [Ruprecht et Krause, 2012]). Ces méthodes sont plus efficaces que l'algorithme

pararéel original pour résoudre le système dynamique de structure. Cependant, le nouveau PITA ainsi que KSE-Parareal ne sont pas encore appliqués ou faisables pour le problème de hyperbolique de fluide non linéaire [Ruprecht et Krause, 2012]. De plus, le temps supplémentaire pour renforcer le solveur grossier pour tous les sous-domaines temporels à chaque itération pararéel, va influencer sur le speed-up. Nous proposons dans cette section une autre méthode à construire le solveur grossier en utilisant la technique de réduction de modèle de type POD. Les coefficients POD sont déterminés par une régression. Cette technique nous permet de construire un solveur grossier avec un temps de calcul négligeable. D'autre part, l'information physique du système est conservée car les premiers modes POD capturent les informations les plus importantes.

Rappelons $\mathbf{F}$ le solveur fin et $\mathbf{G}$ le solveur grossier ayant le pas de temps grossier. Nous cherchons à construire un modèle réduit de $\mathbf{F}$ qui va remplacer le solveur grossier $\mathbf{G}$. Ce modèle réduit noté par $\mathbf{G}^{ROM}$ est mis à jour à chaque itération et construit à partir de plan d'expérience :

- $S_{in}^k = \{u_n^l; 1 \leq l \leq k, 0 \leq n \leq N\}$
- $S_{out}^k = \{\mathbf{F}(u_n^l); 1 \leq l \leq k, 0 \leq n \leq N\}$

La technique POD permet de construire à partir de ces deux ensembles deux espaces réduits entrée et sortie. Ces deux espaces sont associées aux L premiers modes POD $(\Psi_{in}^l)_{1\leq l\leq L}$ et $(\Psi_{out}^l)_{1\leq l\leq L}$ respectivement. On a :

$$vect\{\Psi_{in}^1, ..., \Psi_{in}^L\} \subset vect(S_{in}^k)$$
$$vect\{\Psi_{out}^1, ..., \Psi_{out}^L\} \subset vect(S_{out}^k),$$

où $vect()$ signifie "sous-espace vectoriel engendré". Les vecteurs u et v sont approchés par la projection sur ces deux espace :

$$u \simeq \sum_{l=1}^{L} a_{in}^l . \Psi_{in}^l,$$

$$v = \mathbf{F}(u) \simeq \sum_{l=1}^{L} a_{out}^l . \Psi_{out}^l$$

Grâce à un plan d'expérience obtenu après avoir utilisé $k \times N$ fois le solveur fin $\mathbf{F}$, on va établir L régressions (par exemple krigeage) pour L relations :

$$\tilde{a}_{out}^l \quad : \quad \mathbb{R}^L \longrightarrow \mathbb{R} \tag{4.25}$$

$$\{a_{in}^l; 1 \leq l \leq L\} \mapsto a_{out}^l, \quad l = 1, ..., L. \tag{4.26}$$

On a un modèle réduit de $\mathbf{F}$:

$$\mathbf{G}^{ROM}(u) \simeq \mathbf{G}^{ROM}(\sum_{l=1}^{L} a_{in}^{l}.\Psi_{in}^{l}) \simeq \sum_{l=1}^{L} \tilde{a}_{out}^{l}.\Psi_{out}^{l}. \tag{4.27}$$

L'algorithme pararéel est donc modifié comme le suivant :

- Étape 0 : On fournit une valeur initiale de U calculée par le propagateur grossier $\mathbf{G}$:

  pour $n = 0$ *à* $(N-1)$
  $u_{n+1}^{0} = \mathbf{G}(u_n^0)$
  fin
  sachant que $u_0^0 = u_0$

  Le plan d'expérience initial $(S_{in}^{k}, S_{out}^{k})$ est vide car le solveur $\mathbf{F}$ n'est pas encore utilisé.

Pour $k \geq 1$

- Étape 1 : Calculer en parallèle pour chaque intervalle $\mathbf{F}(u_n^k), n = 0, ..., (N-1)$. Puis mettre à jours les ensembles :

  $$S_{in}^{k} = S_{in}^{k-1} \cup \{u_n^k; 0 \leq n \leq N-1\}$$
  $$S_{out}^{k} = S_{out}^{k-1} \cup \{\mathbf{F}(u_n^k); 0 \leq n \leq N-1\}.$$

  Déterminer les modes POD $(\Psi_{in}^{l})_{1\leq l\leq L}$ et $(\Psi_{out}^{l})_{1\leq l\leq L}$.
  Mettre à jours les régressions $\tilde{a}_{out}^{l}$.

- Étape 2 : Calculer séquentiellement les composants de U^{k+1} sachant que $u_0^{k+1} = u_0^k = u_0^0 = u_0$.

  pour $n = 0 : (N-1)$
  Calculer et stocker $\mathbf{G}^{ROM}(u_n^{k+1})$
  Puis calculer $u_{n+1}^{k+1} = \mathbf{F}(u_n^k) + \mathbf{G}^{ROM}(u_n^{k+1}) - \mathbf{G}^{ROM}(u_n^k)$
  fin

  Arrêter quand $\| U^{k+1} - U^k \| \leq \epsilon$.

4.2.2 Pour le problème EDP paramétré

On redéfinit $\mathbf{F}$ par $\mathbf{F}^{\theta} = w(\Delta T)$ qui dépend du paramètre de conception $\theta \in [0,1]^p$, où w est la solution de :

$$\dot{w} = f^{\theta}(w), \quad t \in [0, \Delta T], \tag{4.28}$$

$$w(0) = v^{\theta}. \tag{4.29}$$

Notons que S^+, S^- sont des ensembles du plan d'expérience numérique obtenu par N_{exp} calculs exactes :

$$S^- = \{v^{\theta^1}, ..., v^{\theta^{Nexp}}\},$$
$$S^+ = \{\mathbf{F}^\theta(v^{\theta^1}), ..., \mathbf{F}^\theta(v^{\theta^{Nexp}})\}.$$

Supposons que : $\forall\theta \in [0,1]^p$, on a : $\tilde{v}^\theta \in vect(S^-)$ et $\tilde{w}(\Delta T) \in vect(S^+)$ sont respectivement des projections de v et $w(\Delta T)$ sur $vect(S^-)$ et $vect(S^+)$.

La méthode POD nous permet de reconstruire ces deux espaces par les espaces réduits basés sur les fonctions de base orthonormées :

$$vect(S^-) \simeq vect\{\Psi_1^-, ..., \Psi_{L^-}^-\},$$
$$vect(S^+) \simeq vect\{\Psi_1^+, ..., \Psi_{L^+}^+\}.$$

Les projections s'écrivent sous la forme :

$$\tilde{v}^\theta = \sum_{l=1}^{L^-} a_l^-(\theta).\Psi_l^-(x),$$
$$\tilde{w}^\theta(\Delta T) = \sum_{l=1}^{L^+} a_l^+(\theta).\Psi_l^+(x)$$

À l'inverse, on peut déterminer les coefficients POD par les projections :

$$a_l^-(\theta) =< \tilde{v}^\theta, \Psi_l^-(x) >, \quad l = 1, ...L^-, \tag{4.30}$$
$$a_l^+(\theta) =< \tilde{w}^\theta(\Delta T), \Psi_l^+(x) >, \quad l = 1, ...L^+ \tag{4.31}$$

Avec le plan d'expérience on peut construire L^+ métamodèles de la relation : $\{a_1^-, ..., a_{L^-}^-\} \to \tilde{a}_l^+ \simeq a_l^+, l = 1, ...L^+$. On peut obtenir aisément ces métamodèles en utilisant les techniques d'apprentissages : par exemple les moindres carrés mobiles, les réseaux de neurones, les réseaux de fonctions à base radiale, krigeage, etc.

Ensuite, on définit un propagateur grossier $\mathbf{G}^\theta$ approximation de $\mathbf{F}^\theta$:

$$\mathbf{G}^\theta(v^\theta) \simeq \mathbf{G}^\theta(\sum_{l=1}^{L-} a_l^-.\Psi_l^-) := \sum_{l=1}^{L+} \tilde{a}_l^+.\Psi_l^+ \tag{4.32}$$

Revenons à l'algorithme pararéel, on remplace les propagateurs $\mathbf{F}, \mathbf{G}$ par ceux définis ci-dessus $\mathbf{F}^\theta, \mathbf{G}^\theta$, où les signes $-$ et $+$ désignent le début et la fin d'intervalle $[T_n, T_{n+1}]$, $\quad T_{n+1} - T_n = \Delta T$.

Le propagateur grossier est construit grâce au plan d'expérience de l'étape prélimi-

naire qui consiste en :

- **Étape 1 :** étant donné une famille de N_{exp} points $\{\theta^i\}_{i=1,...,N_{exp}}$ générés par un algorithme de remplissage d'espace comme LHS (*Latin Hypercube Sampling*), on calcule par un solveur fin les solutions discrètes en temps et en espace $u_n(\theta) = u^h(\theta, t_n)$, où h désigne l'espace d'éléments finis, et $t_{n+1} = t_n + \delta t$. Puis, à partir de la discrétisation temporelle grossière de l'algorithme pararéel $\{T_0 = 0, T_1, ..., T_N\}$ sur l'intervalle $[0, T]$, on extrait de cette famille un ensemble de données :
$$S^n = \{u_n(\theta^1), ..., u_n(\theta^{N_{exp}})\}, \quad n = 1, ..., N. \tag{4.33}$$
Cette étape est réalisée en parallèle sur plusieurs processeurs.
- **Étape 2 :** On réalise une analyse en composantes principales de S^n, puis on en extrait des bases propres orthonormées $\{\Psi_1^n, ..., \Psi_{L_n}^n\}, n = 1, ..., N$, où L_n est le rang de troncature des modes POD pour chaque ensemble S^n. L_n est calculé de façon à assurer que toute projection de $u_n(\theta^i)$ sur l'espace engendré par les vecteurs propres conduit à une erreur de projection inférieure à une certaine tolérance ϵ.
- **Étape 3 :** On construit la base d'apprentissage $\{a_1^n, ..., a_{L_n}^n\}, n = 1, ..., N$. Puis, on établit les relations $\{a_1^n, ..., a_{L_n}^n\} \to \tilde{a}_l^{n+1}, l = 1, ..., L_{n+1}$, et une relation pour le premier intervalle $\theta \to \tilde{a}_l^1(\theta), l = 1, ..., L_1$.
- **Étape 4 :** On stocke seulement les modes POD et les métamodèles. Pour appliquer le propagateur grossier sur un intervalle $[T_n, T_{n+1}]$ avec la condition initiale $u(T_n) = u_n$, on détermine d'abord $a_l^n =< u_n, \Psi_l^n >$. Puis on appelle le $n^{ème}$ métamodèle pour déterminer $\tilde{a}_l^{n+1}, l = 1, ..., L_{n+1}$.

4.3 Conclusion

Dans ce chapitre, nous avons réintroduit l'algorithme pararéel par l'interprétation de la méthode quasi-Newton. L'accélération et la convergence de l'algorithme sont analysées. Ensuite, nous avons proposé de nouvelles méthodes pour accélérer l'algorithme pararéel en la combinant avec la méthode de réduction de modèles.

Nous avons montré par la théorie qu'il existe un nombre de processeurs critique où le speed-up de l'algorithme original est maximal. Ce nombre de processeurs peut être estimé si on connaît le temps d'exécution d'un propagateur grossier. La courbe théorique montre que si le nombre de processeurs utilisés dépasse la valeur critique, le speed-up de l'algorithme pararéel diminue ; l'efficacité de l'algorithme est donc contraire par le nombre de processeurs utilisés. Par ailleurs, quand on augmente le nombre de processeurs, la précision de l'approximation augmente. La question

est posée le nombre de processeurs à utiliser pour obtenir la meilleure efficacité de l'algorithme pararéel en supposant qu'on a assez de processeurs disponibles. La recherche sur cette question est une perspective assez intéressante.

Nous avons proposé deux pistes de modifications de l'algorithme pararéel en appliquant l'idée de l'utilisation des réductions d'ordre de modèles. Malheureusement, nous n'avons pas pu faire d'expériences numériques par manque de temps.

Chapitre 5

Broyden-pararéel : Nouvelle variante de l'algorithme pararéel

Sommaire

Dans le chapitre 4, nous avons introduit l'algorithme pararéel en se basant sur la résolution de l'équation $R(U) = 0$ par la méthode de Newton. Ici nous proposons une nouvelle variante de l'algorithme pararéel nommée Broyden-pararéel. La dérivation de cet algorithme vient de la résolution de l'équation $R(U) = 0$ par la méthode de quasi-Newton.

5.1 Approche de quasi-Newton

Les approches de quasi-Newton consistent à approcher la matrice jacobienne $DR(U^k)$ (ou son inverse) par une matrice plus simple à calculer, et qui possède la propriété de quasi-Newton (propriété de sécante).

Remplaçons donc $DR(U^k)$ par une matrice $\mathbb{A}^k$ (à construire) qui possède la propriété de sécante suivante :

$$\mathbb{A}^{k+1}(U^{k+1} - U^k) = R(U^{k+1}) - R(U^k). \tag{5.1}$$

La propriété de sécante s'appelle aussi la condition de quasi-Newton que l'on notera désormais (CQN). Étant donné la structure triangulaire creuse de $DR(U^k)$, on va chercher $\mathbb{A}^k$ sous la forme

$$\mathbb{A}^k = \begin{pmatrix} I & & & (0) \\ -A_1^k & I & & \\ & \ddots & \ddots & \\ (0) & & -A_{N-1}^k & I \end{pmatrix}$$

avec des petites blocs A_n^k à définir. La propriété de sécante bloc par bloc s'écrit

$$-A_n^{k+1}(u_n^{k+1} - u_n^k) + (u_{n+1}^{k+1} - u_{n+1}^k) = u_{n+1}^{k+1} - \mathbf{F}(u_n^{k+1}) - u_{n+1}^k + \mathbf{F}(u_n^k)$$

soit encore

$$A_n^{k+1}(u_n^{k+1} - u_n^k) = \mathbf{F}(u_n^{k+1}) - \mathbf{F}(u_n^k).$$

Il s'agit encore d'une condition de quasi-Newton que doit respecter chacune des matrices A_n^{k+1}. On reviendra plus loin sur la construction effective de telles matrices.

Pour le moment, écrivons explicitement la méthode de quasi-Newton

$$\mathbb{A}^k(U^{k+1} - U^k) = -R(U^k).$$

Bloc par bloc, ceci s'écrit

$$u_{n+1}^{k+1} = \mathbf{F}(u_n^k) + A_n^k(u_n^{k+1} - u_n^k).$$

Différents choix possibles de A_n^k vont conduire à différentes méthodes. Ils sont explicités dans la section suivante.

5.2 Dérivation de méthodes Broyden-pararéel

5.2.1 Broyden-pararéel sans solveur grossier

La méthode de Broyden consiste à construire une matrice de manière incrémentale (apprentissage) où l'incrément est une matrice de correction de rang 1 :

$$A_n^{k+1} = A_n^k + C_n^k$$

avec $C_n^k = v_n^k(w_n^k)^T$. En notant

$$\Delta u_n^k = u_n^{k+1} - u_n^k$$

et

$$\Delta \mathbf{F}_n^k = \mathbf{F}(u_n^{k+1}) - \mathbf{F}(u_n^k),$$

la condition de quasi-Newton s'écrit

$$A_n^{k+1}\Delta u_n^k = \Delta \mathbf{F}_n^k.$$

On obtient

$$C_n^k \Delta u_n^k = \Delta \mathbf{F}_n^k - A_n^k \Delta u_n^k := r_n^k$$

ou encore

$$v_n^k (w_n^k)^T \Delta u_n^k = r_n^k.$$

Le choix de Broyden consiste à choisir

$$w_n^k = \frac{\Delta u_n^k}{\|\Delta u_n^k\|}, \quad v_n^k = \frac{r_n^k}{\|\Delta u_n^k\|}.$$

Remarque 4. *Cette méthode de Broyden ne fait pas intervenir de propagateur grossier particulier. La performance de la méthode dépend de la qualité d'initialisation des matrices A_n^k par les A_n^0.*

5.2.2 Broyden-pararéel avec solveur grossier, version additive

La candidat connue des matrices A_n^k est :

$$A_n^{k+1} = \int_0^1 D\mathbf{F}(u_n^k + t(u_n^{k+1} - u_n^k))\, dt,$$

mais c'est difficile à calculer. Cela nous invite à rechercher des matrices de quasi-Newton de la forme

$$A_n^{k+1} = \int_0^1 D\mathbf{G}(u_n^k + t(u_n^{k+1} - u_n^k))\, dt + B_n^{k+1}$$

où $\mathbf{G}$ est un propagateur grossier de classe $\mathcal{C}^1$, et B^{k+1} est une matrice qui se construit de manière incrémentale selon les idées de Broyden :

$$B_n^{k+1} = B_n^k + C_n^k \tag{5.2}$$

où C_n^k est une matrice de rang 1. Par la formule de Taylor exacte avec le reste intégral

$$\mathbf{G}(u_n^{k+1}) - \mathbf{G}(u_n^k) = \int_0^1 D\mathbf{G}(u_n^k + t(u_n^{k+1} - u_n^k))\, dt\, (u_n^{k+1} - u_n^k),$$

la propriété de sécante définie par

$$A_n^{k+1}(u_n^{k+1} - u_n^k) = \mathbf{F}(u_n^{k+1}) - \mathbf{F}(u_n^k)$$

s'écrit comme suit :

$$\mathbf{G}(u_n^{k+1}) - \mathbf{G}(u_n^k) + B_n^{k+1}\Delta u_n^k = \Delta \mathbf{F}_n^k.$$

Notons

$$\Delta \mathbf{G}_n^k := \mathbf{G}(u_n^{k+1}) - \mathbf{G}(u_n^k),$$

en utilisant (5.2), il s'agit donc de construire la matrice de correction C_n^k telle que

$$C_n^k \Delta u_n^k = \Delta \mathbf{F}_n^k - \Delta \mathbf{G}_n^k - B_n^k \Delta u_n^k,$$

La méthode de quasi-Newton

$$u_{n+1}^{k+1} = \mathbf{F}(u_n^k) + A_n^k(u_n^{k+1} - u_n^k).$$

s'écrit dans ce cas

$$u_{n+1}^{k+1} = \mathbf{F}(u_n^k) + \mathbf{G}(u_n^{k+1}) - \mathbf{G}(u_n^k) + B_n^k(u_n^{k+1} - u_n^k).$$

On retrouve pour une partie le schéma pararréel standard plus une petite correction permettant de respecter la propriété de sécante.

Remarque 5. *Il est évident de considérer ici comme initialisation $B_n^0 = 0$. La première itération de la méthode est alors une itération de l'algorithme pararéel standard.*

5.2.3 Broyden-pararéel avec solveur grossier, version multiplicative

Une autre possibilité est de considérer la matrice suivante :

$$\int_0^1 D\mathbf{G}(u_n^k + t(u_n^{k+1} - u_n^k))\, dt$$

avec un correction multiplicative à gauche :

$$A_n^{k+1} = B_n^{k+1} \int_0^1 D\mathbf{G}(u_n^k + t(u_n^{k+1} - u_n^k))\, dt.$$

La propriété de sécante

$$\Delta\mathbf{F}_n^k = A_n^{k+1}\Delta u_n^k$$

s'écrit alors

$$\Delta\mathbf{F}_n^k = B_n^{k+1}\Delta\mathbf{G}_n^k.$$

La méthode Broyden-pararéel s'écrit dans ce cas

$$u_{n+1}^{k+1} = \mathbf{F}(u_n^k) + B_n^k\left(\mathbf{G}(u_n^{k+1}) - \mathbf{G}(u_n^k)\right).$$

Remarque 6. *Il est évident ici de considérer comme initialisation $B_n^0 = I$. La première itérée de la méthode est alors une itérée de l'algorithme pararéel standard.*

5.3 Applications

5.3.1 Résolution du système Brusselator

Nous reprenons un exemple qu'on a utilisé dans [Gander et Hairer, 2008] afin de valider notre code. Puis, on résout le même problème par l'algorithme Brodyen-pararéel. Les résultats sont présentés et comparés dans cette section. Il s'agit de

l'application des systèmes réaction chimique modélisée par un système d'équations différentielles :

$$\dot{x} = A + x^2 y - (B+1)x, \quad \dot{y} = Bx - x^2 y, \quad t \in [0, T], \tag{5.3}$$

$$x(0) = 0, \quad y(0) = 0. \tag{5.4}$$

Ces équations peuvent s'écrire sous la forme générale suivante : $\dot{u} = f(t, u), \quad t \in [0, T]$,
où :

$$u = \begin{pmatrix} x \\ y \end{pmatrix}$$

et

$$f(t, u) = \begin{pmatrix} A + x^2 y - (B+1)x \\ Bx - x^2 y \end{pmatrix}$$

Pour résoudre cette équation, on va utiliser la méthode Runge-Kutta d'ordre 4 (RK4) avec le pas de temps h, le schéma est donné par la formule suivante :

$$u_{n+1} = u_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

où

$$\begin{aligned} k_1 &= f(t_n, u_n) \\ k_2 &= f(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_1) \\ k_3 &= f(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_2) \\ k_4 &= f(t_n + h, u_n + hk_3) \end{aligned}$$

On choisit les paramètres $A = 1$, et $B = 3$, puis on résout l'équation 5.3 sur un intervalle temporel $[0, T = 12]$. Dans le contexte de l'algorithme pararéel, le solveur fin est choisi comme le RK4 avec le pas de temps $h = \Delta t = \frac{T}{640}$, ce qui a l'ordre de précision de 5.62×10^{-6}. Le solveur grossier est aussi RK4 mais avec le pas de temps plus grossier $h = \Delta T = \frac{T}{32}$, ΔT est aussi la taille d'un sous-domaine. Les solutions (x, y) calculées par ces deux solveurs et leurs erreurs en fonction du temps sont montrées dans la figure 5.1. Les résultats des cinq premières itérations de pararéel sont montrés dans la figure 5.2. Si l'on néglige le temps pour calculer le solveur grossier, l'algorithme pararéel avec 32 processeurs est 6 fois plus rapide qu'avec un seul processeur, lorsque l'erreur est de ordre de 10^{-6}. On trouve que ces résultats sont exactement les même ceux présentés dans [Gander et Hairer, 2008].

Les trois versions de l'algorithme Broyden-pararéel sont utilisées pour résoudre

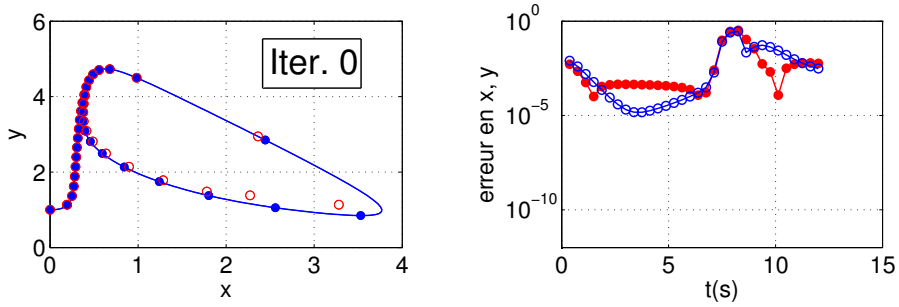

FIGURE 5.1 – À gauche : la solution de l'équation Brusselator calculée par le propagateur grossier (cercle rouge) et le propagateur exact (la ligne bleue et le point bleu). À droite : l'erreur en x (ligne bleue) et en y (ligne rouge) en fonction du temps.

l'équation Brusselator ci-dessus. Notons que le temps de calcul des matrices de correction C_n^k est négligeable. Le temps de calcul pour chaque itération de l'algorithme Broyden-pararéel est donc égale à celui de l'algorithme pararéel. La différence est la vitesse de convergence, qui est montrée dans la figure 5.3. On constate que les deux versions additives et multiplicatives de l'algorithme Broyden-pararéel sont plus efficaces que l'algorithme pararéel original. La version non-propagateur grossier est moins efficace, car elle converge plus lentement que les autres. Cependant, sa vitesse de convergence est aussi rapide que celle de la version multiplicative pour les deux premières itérations. Désormais, on utilise seulement la version multiplicative pour l'algorithme Broyden-pararéel.

5.3.2 Résolution des équations aux dérivées partielles (EDP)

5.3.2.1 Concentration de CO_2 dans la cabine d'avion

5.3.2.1.1 Problématique Dans une cabine d'avion, on cherche à connaître la distribution du CO_2, notamment sa quantité locale au niveau de la tête des passagers. Les calculs seront identiques pour étudier les concentrations des autres espèces chimiques ou contamination des virus, etc. Supposons qu'au début, CO_2 remplit uniformément dans la cabine. On injecte de l'air "pur" afin de recycler le mauvais air existant dans la cabine. La quantité de CO_2 produit par les passagers est pris en compte. On observe comment la concentration de CO_2 s'évolue. Il est raisonnable de considérer une équation de convection-diffusion pour le transport de l'espèce chimique c. On considère une diffusion pas trop petite pour que l'équation ne soit pas trop proche du régime hyperbolique, sachant que la turbulence de l'air ajoute une viscosité turbulente qui dilue le polluant. Étant donné une concentration chimique initiale uniforme, on aura un régime transitoire et on devrait atteindre à terme un régime permanent en un certain temps T où $c = 0$ ou est inférieur à

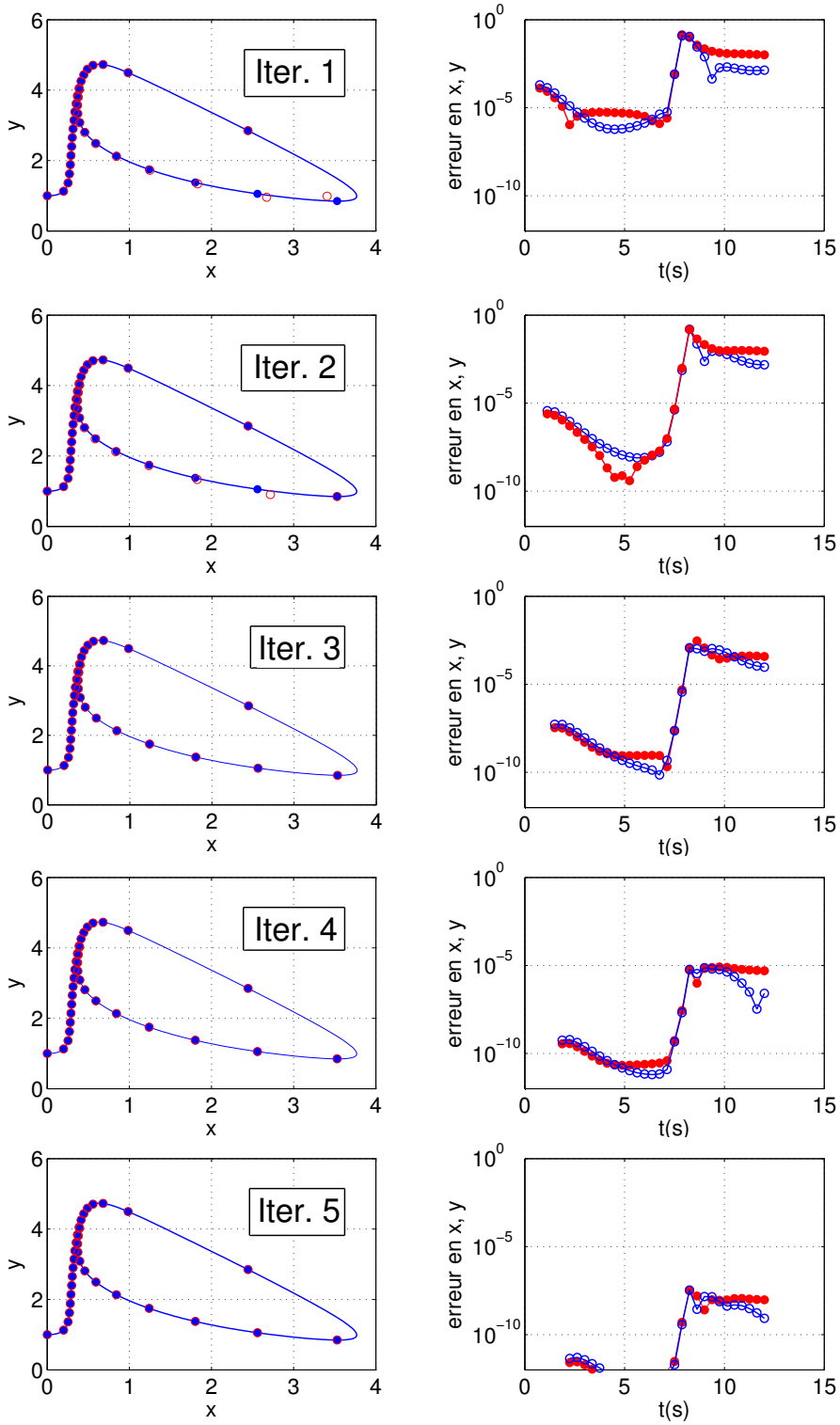

FIGURE 5.2 – À gauche : la solution de l'équation Brusselator calculée par l'algorithme pararéel à la k ème itération (cercle rouge) et par le modèle exact (ligne bleue et point bleu). À droite : l'erreur en x (ligne bleu) et en y (ligne rouge) en fonction du temps.

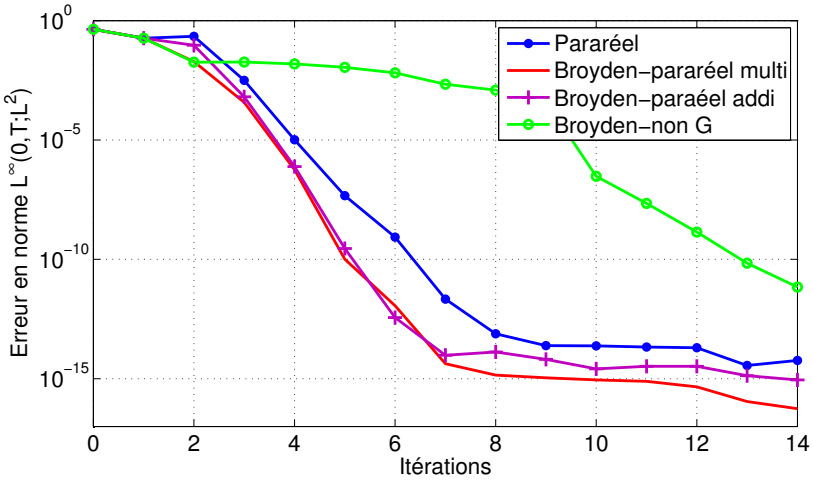

FIGURE 5.3 – Convergence de l'algorithme pararéel et de trois versions de l'algorithme Broyden-pararéel appliquées sur l'équation de Brusselator.

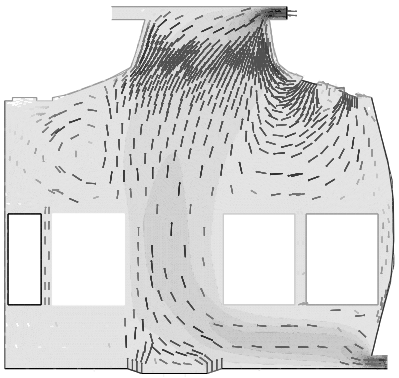

FIGURE 5.4 – Courant d'air dans la cabine d'avions

une valeur admissible. Cette valeur de T nous servira de temps final pour la fenêtre temporelle d'intérêt.

5.3.2.1.2 Modélisation

- **Partie fluide :** On utilise un champs de vitesse stationnaire $\boldsymbol{u} = \boldsymbol{u}(x)$ calculé du couplage de l'équation Navier-Stokes et l'équation diffusion de température (voir la figure 5.4).

- **Concentration chimique :** On résout un problème de convection-diffusion sur la concentration de CO_2 notée par $c = C_{CO_2}$ avec les sources produis par 3 passagers. Ces sources sont présentées par le membre de droite de l'équation, où ω_i une fonction indicatrice qui vaut 1 sur la surface remplie par la tête de

passager et vaut 0 à l'extérieur.

$$\partial_t c + \boldsymbol{u} \cdot \nabla c - \nabla \cdot (d \nabla c) = c_0 \Sigma_{i=1}^{3} (2 + sin(\alpha t) + sin(\beta t)) \omega_i \tag{5.5}$$

- Condition aux limites :

$$c = 0 \text{ sur } \Gamma_{\text{entrée}} \tag{5.6}$$

$$\nabla c \cdot n = 0 \text{ sur } \Gamma_{\text{sortie}} + \Gamma_{\omega} \tag{5.7}$$

- Condition initiale :

$$c(x, t = 0) = c_0/4 \text{ sur } \Omega \tag{5.8}$$

- Valeur des paramètres : $dt = 0.01, \quad \Delta T = 60 \times dt = 0.6, \quad N = 60, \quad \alpha = 1, \quad \beta = 2, \quad d = 10^{-2}$.

L'objectif est de résoudre l'équation (5.5) dont la solution intéressée est la concentration $c(x, t)$ sur tout l'espace Ω de cabine d'avion qui varie de $c(x, t = 0)$ à $c(x, t = T)$. Ayant calculé un champ de vitesse $\boldsymbol{u}$, l'équation (5.5) sera résolue par la méthode d'éléments finis grâce à logiciel Freefem++.

La précision de la solution dépend de la discrétisation temporelle. Supposons que la solution soit pertinente si le pas de temps pour le solveur fin est $dt = 0.01$. Comme le pararéel original, on utilise un solveur grossier qui est le même schéma du solveur fin mais avec un pas de temps $\Delta T >> dt$. Notons P, le nombre de processeurs utilisés, on peut choisir ΔT tel que : $P = \frac{T}{\Delta T} = \frac{\Delta T}{dt}$.

Pour visualiser l'évolution de la concentration chimique ainsi que la convergence de l'algorithme pararéel, on va observer la concentration de CO_2 au niveau de la tête des passagers. La figure 5.5 montre un exemple de la distribution de CO_2 à $t = 30$sec dans la cabine d'avion (à gauche) et sur une ligne transversale à la hauteur de la tête des passagers (à droite). Comme la solution évolue en temps et en espace (2D), on capture seulement la solution à chaque instant $t = 0, T_1, ..., T_N$ et à la hauteur de la tête des passagers. Dans ce cas, la concentration du dioxyde de carbone calculée par l'algorithme Broyden-Pararéel avec cinq premières itérations est montrée dans la figure 5.6. On estime l'erreur relative définie par la différence absolue entre la solution approchée et la solution exacte divisée par la norme L_2 de la solution exacte. Dans le cadre de cette thèse, on considère toujours que la solution exacte est calculée par le solveur fin.

L'algorithme pararéel original est aussi utilisé pour calculer cette application. On compare aussi la vitesse de la convergence des deux algorithmes (voir la figure 5.7). Le résultat montre que l'algorithme Broyden-pararéel est toujours plus efficace que

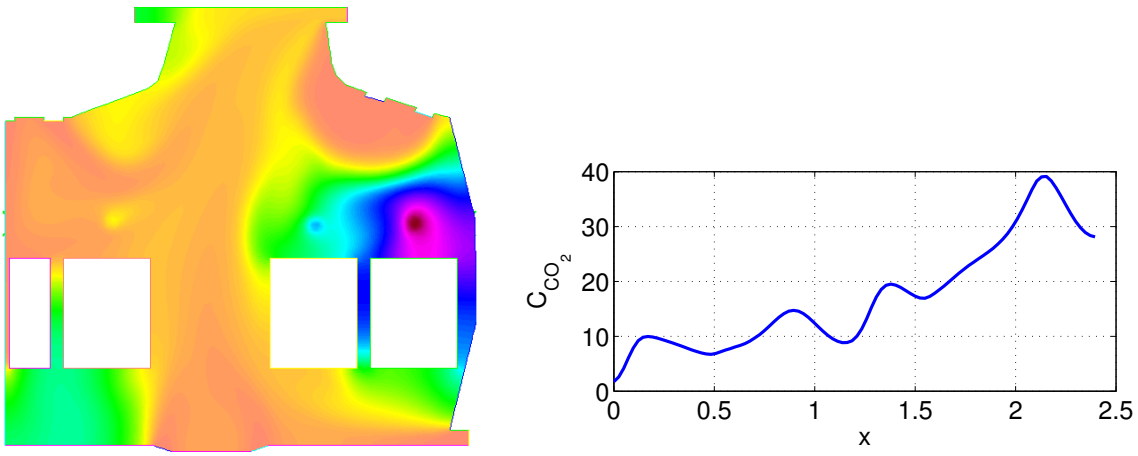

FIGURE 5.5 – À gauche : C_{CO_2} dans la cabine d'avion à $t = 30$sec. À droite : C_{CO_2} au niveau de la tête de passager.

celui original.

Concernant le speed-up de l'algorithme pararéel, nous avons mesuré le temps réel de calcul. Si on arrête le calcul de l'algorithme pararéel à 3ème itération et utilise 60 processeurs, on obtient un speed-up de 3.17. Comme nous l'avons étudiée dans la section 4.1.3, cette valeur varie selon le nombre de processeurs utilisés. En effet, la figure 5.8 montre le speed-up dépend du nombre de processeurs utilisées P. Cependant, quand on change P la vitesse de convergence change aussi (voir la même figure). La précision augmente légèrement selon P. Avec $P = 36$, l'algorithme est plus efficace puisque le speed-up est maximal ($= 3.43$) lorsque l'erreur est raisonnable.

5.3.2.2 Un cas non-linéaire où l'algorithme pararéel ne converge pas

Pour vérifier l'efficacité du nouvel algorithme dans le cas plus général, nous ajoutons un terme non-linéaire dans l'équation de la convection-diffusion. Prenons par exemple le problème suivant :

$$\partial_t c + \boldsymbol{u} \cdot \nabla c - \nabla \cdot (d \nabla c) + \alpha c^3 = c_0 sin(\beta t) \tag{5.9}$$

dans la cabine d'avion au-dessus avec :

- Condition aux limites :

$$c = 0 \text{ sur } \Gamma_{\text{entrée}} \tag{5.10}$$

$$\nabla c \cdot n = 0 \text{ sur } \Gamma_{\text{sortie}} + \Gamma_{\omega} \tag{5.11}$$

- Condition initiale :

$$c(x, t = 0) = c_0/4 \text{ sur } \Omega \tag{5.12}$$

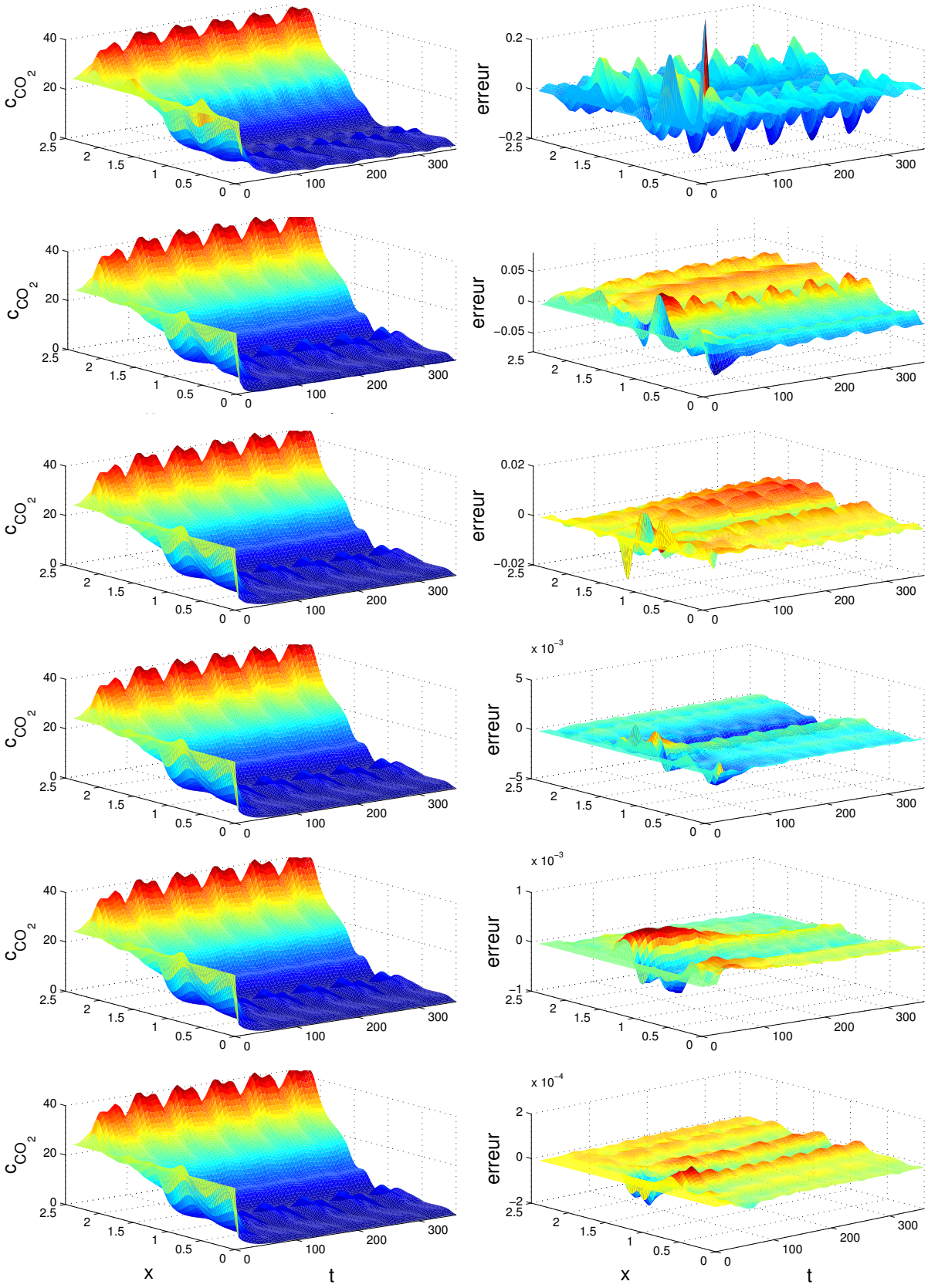

FIGURE 5.6 – À gauche : Distribution de CO_2 au niveau de la tête de passager en fonction du temps calculé par Broyden-pararéel. À droite : L'erreur relative correspondante.

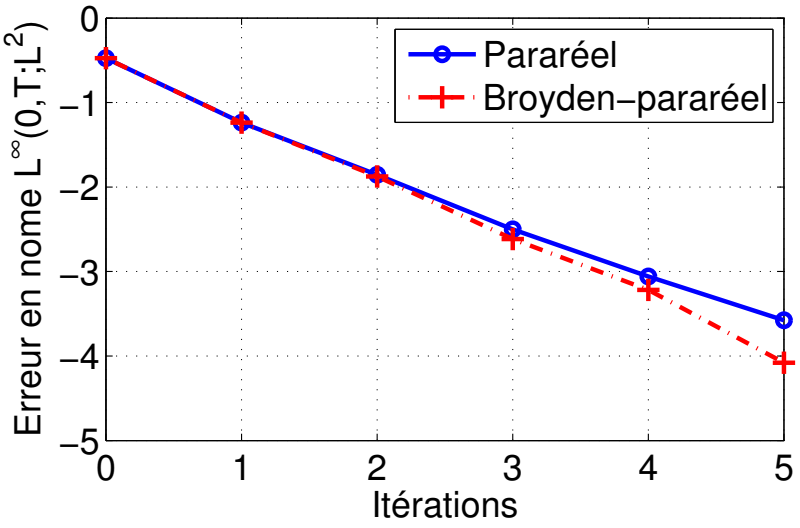

FIGURE 5.7 – Taux de convergence du Pararéel et Broyden-pararéel pour calculer C_{CO_2}.

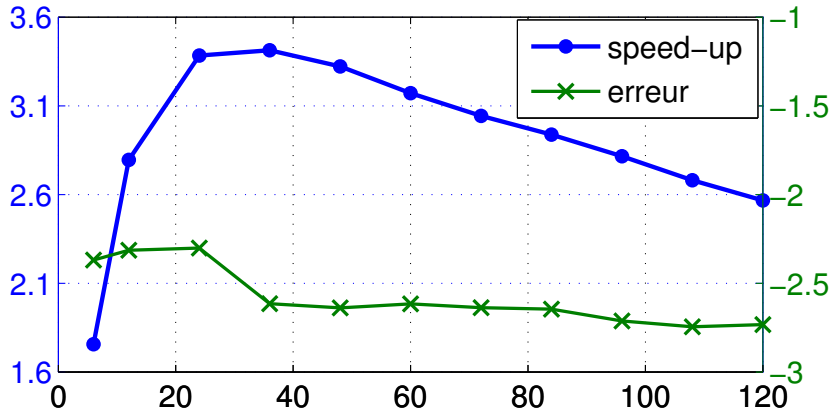

FIGURE 5.8 – Speed-up mesuré et log10 de l'erreur en norme $L^\infty(0,T;L^2)$ selon nombre de processeurs utilisés si on arrête à 3ème itération.

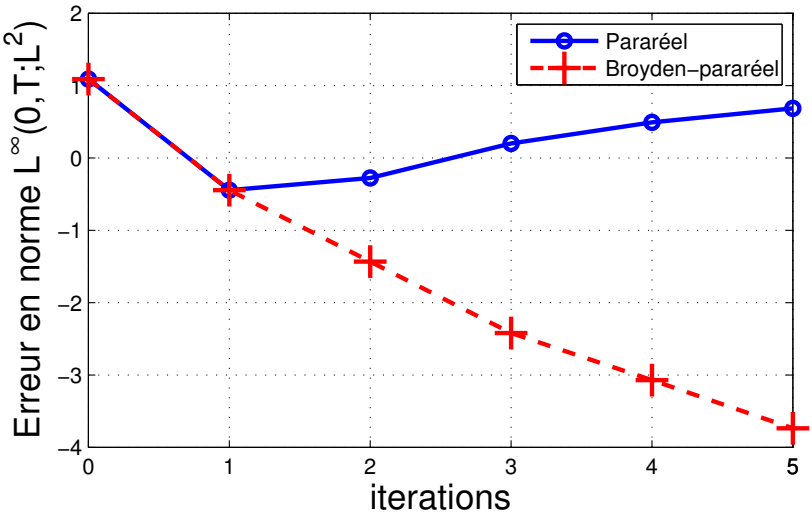

FIGURE 5.9 – Taux de convergence du Pararéel et Broyden-pararéel pour le cas non-linéaire.

- $dt = 0.01, \quad \Delta T = 20 \times dt = 0.2, \quad N = 24, \quad \alpha = 0.01, \quad \beta = 8.$

Les calculs sont réalisés sur 24 processeurs par les algorithmes pararéel et Broyden-pararéel. On maintient la même vitesse $\boldsymbol{u}$, et le cœfficient de diffusion d utilisées dans la section précédente. Après cinq premières itérations, l'algorithme pararéel original ne converge pas lorsque Broyden-pararéel converge (voir la figure 5.9).

5.4 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle variante de l'algorithme pararéel appelée l'algorithme Broyden-pararéel. La dérivation de cet algorithme vient du respect par construction de la condition de quasi-Newton (CQN). Ce nouvel algorithme est appliqué et comparé avec celui original, pour résoudre l'EDO non-linéaire, et en particulier l'EDP concernant la diffusion de CO_2 dans une cabine d'avion. Les calculs sont réalisés avec le code Freefem++-MPI au méso-centre de l'ECP équipé par une grande machine multi-cœurs.

L'algorithme Broyden-pararéel permet d'améliorer la vitesse de convergence lorsqu'il ne demande pas un calcul supplémentaire. Il est appliqué pour l'EDO ainsi que l'EDP. De plus, il peut converger en quelques itérations pour le problème d'EDP non-linéaire ce que l'algorithme pararéel original ne peut pas résoudre. Ce nouvel algorithme a un grand potentiel pour résoudre les problèmes non-linéaires.

Conclusions et perspectives

Conclusions

Cette thèse aborde de la conception de techniques de réduction d'ordre semi-intrusives pour l'approximation de solutions de problèmes aux dérivées partielles paramétrés. Ce travail s'insère dans le cadre du projet *Complex System Design Lab* piloté par Dassault Aviation. Le projet CSDL est un projet de type FUI (Fond Unique Interministériel) visant à développer une plate-forme logicielle multidisciplinaire pour la conception de systèmes complexes. Il regroupe 28 partenaires académiques et industriels.

Dans cette thèse, nous avons en particulier proposé une technique d'enrichissement adaptatif de modèles réduits locaux dans l'espace paramétrique, avec contrôle de l'erreur par régions de confiance. L'algorithme d'enrichissement adaptatif appelé POD-ISAT (Proper Orthogonal Decomposition - In Situ Adaptive Tabulation) sert à construire un modèle EDP stationnaire paramétrique. Nous l'avons ici appliqué à un problème de thermique, et plus précisément au calcul du champ de température dans une cabine d'avion, où la solution dépend de paramètres de conception, par exemple la conductivité thermique du fuselage, la température de l'air injecté, la vitesse d'entrée de l'air, etc.

Dans le cadre de problèmes d'évolution, nous avons aussi proposé et étudié une technique de modèle réduit tirant avantage de l'algorithme de parallélisation en temps appelé *algorithme pararéel*, initialement proposé par Lions, Maday et Turinici (2001). Cette technique s'appuie sur une variante de l'algorithme pararéel que nous appelons *algorithme Broyden-pararéel*. Il peut s'appliquer à des problèmes aux dérivées partielles. Il a été ici appliqué au calcul de la diffusion d'un gaz dans la cabine d'avion.

Algorithme POD-ISAT. C'est une méthode robuste de réduction d'ordre s'appuyant sur l'enrichissement d'une base de données de modèles réduits locaux dans l'espace paramétrique, permettant un accès rapide au modèle et une évaluation quasi-instantanée d'un champ de calcul. C'est une combinaison de la décomposi-

tion POD pour la représentation spatiale et de l'algorithme ISAT pour le domaine paramétrique (ici considéré comme un hypercube). Par ailleurs, nous utilisions des métamodèles généralistes de krigeage pour approcher précisément les coefficients des décompositions POD. Cet aspect est original par rapport aux méthodes de réduction d'ordre existantes dans la littérature. Il s'agit ici plutôt d'une méthode d'interpolation "enrichie" qui permet de calculer les coefficients POD par krigeage, alors que les méthodes plus classiques de projection telles que POD-Galerkin calculent les coefficients comme racines d'un système algébrique réduit, par algorithme de Newton par exemple avec l'évaluation délicate de matrices jacobiennes.

Pour l'algorithme POD-ISAT, nous avons besoin d'une première étape hors ligne avec la production d'un plan d'expérience numérique préliminaire. On construit alors une famille de modèles réduits locaux avec un ellipsoïde de précision (Ellipsoid of Accuracy ou EOA dans la dénomination ISAT) centré en un point d'expérience (une instance θ^i de paramètre).

La deuxième étape consiste à enrichir la base de données de manière adaptative et est essentielle pour couvrir tout le domaine paramétrique par recouvrement des régions de confiance. Selon l'usage et les besoins d'exploration du domaine paramétrique (requête par exemple), on calcule si besoin un nouvel enregistrement avec un nouveau modèle réduit local et une région de confiance associée. Pour résumer, les caractéristiques principales de POD-ISAT sont les suivantes :

- application aux problèmes EDP paramétriques (stationnaires),
- caractère faiblement intrusif de l'approche,
- rapidité d'accès et rapidité de reconstruction de la solution interpolée,
- contrôle de la précision par région de confiance,
- enrichissement adaptatif de la base de données.

L'algorithme a été testé avec succès sur un calcul de champs de température dans une cabine d'avion, dépendant des paramètres de conception. On a considéré deux cas différents de dimension d'espace paramétrique. Le premier cas concerne un espace paramétrique de dimension deux et le second un espace plus grand de dimension quatre. Pour couvrir tout le domaine paramétrique, on a constaté la génération d'une base de données de modèles de 9 et 28 modèles réduits locaux respectivement pour le premier et le deuxième cas. Les temps de calcul pour la génération complète de la base de données dans les deux cas sont de $2,35$ heures et $8,22$ heures respectivement. Une fois la base de données générée, la réponse à une requête est instantanée et permet comme attendu l'utilisation de champs CFD dans un contexte de visualisation décisionnelle, de travail collaboratif interactif, ou d'optimisation multidisciplinaire. En comparant avec le modèle réduit POD standard sans région de confiance, POD-

ISAT est plus efficace du point de vue de la précision et confirme l'intérêt de l'utilisation de régions de confiance. L'algorithme POD-ISAT dépend d'un paramètre de tolérance sur le résidu. Ceci peut être utilisé à profit pour la constitution de bases de données multi-niveaux. Si les besoins en précision sont faibles, on peut augmenter le seuil de tolérance. En principe les EOAs seront plus larges et la base de données sera donc plus légère.

Le cas test avec quatre paramètres présenté en section 3.5.2 génère une base de données de 28 modèles réduits locaux correspondant à 28 régions de confiance. Ce cas est encore suffisamment simple pour effectuer une recherche classique non optimisée pour trouver une région de confiance qui couvre le point de requête. Néanmoins, dans le cas où l'espace de paramètres est plus grand, on a besoin de milliers de régions de confiance. Le temps de recherche d'une ellipse va prendre plusieurs secondes voire même quelques minutes. Dans ce cas, l'algorithmique de recherche dans l'arbre de décision dans ISAT joue un rôle déterminant dans la performance de la recherche.

Dans le cadre de cette thèse, nous nous sommes limités à l'utilisation de l'algorithme POD-ISAT aux cas de problèmes stationnaires. Néanmoins, on peut probablement étendre la méthode aux cas des problèmes instationnaires en utilisant une technique suggérée par [Amsallem *et al.*, 2012].

Nous nous sommes plutôt intéressés à l'utilisation originale de l'algorithme pararéel dans un cadre de réduction de modèle, en générant une "chaîne" de modèles réduits locaux en temps. L'algorithme pararéel qui permet un calcul parallèle d'un problème d'évolution temporelle fait appel à un solveur "grossier" prédicteur peu coûteux. Il est alors naturel d'envisager d'utilisation de modèles d'ordre réduit comme solveurs grossiers locaux en temps.

À l'École Centrale Paris, nous avons porté l'algorithme pararéel sur une machine multi-cœurs (méso-centre de calcul) pour calculer rapidement la diffusion du CO_2 dans une cabine d'avion (à titre d'exemple). De plus, nous avons proposé un nouvel algorithme "Broyden-pararéel" qui respecte par construction la condition de sécante, appelée encore condition de quasi-Newton (CQN), permettant théoriquement la convergence super-linéaire de l'algorithme, quel que soit le modèle grossier. Le principe est de revisiter l'algorithme pararéel comme un algorithme de type quasi-Newton (Gander-Vandewalle) cherchant les racines d'un résidu de saut entre sous-domaines temporels. L'algorithme pararéel d'origine est modifié en ajoutant un terme correctif additionnel, additif ou multiplicatif de type Broyden. Les comparaisons entre l'algorithme original et la variante proposée ont été menées dans deux cas d'application EDO et EDP respectivement. Les résultats ont montré que l'algorithme Broyden-pararéel est en effet plus efficace.

Malheureusement, nous n'avons pas pu tester la chaîne complète de réduction

d'ordre de modèles spatio-temporels par manque de temps.

Perspectives

La construction des régions de confiance par ellipsoïdes (EOA, voir section 3.4.2.3) joue un rôle important dans la précision de POD-ISAT. La précision de EOA permet de générer un échantillon qualité pour construire un modèle de krigeage des coefficients POD. En plus, le temps, t_{eoa}, pour réaliser cette tâche est grand et augmente selon la dimension de l'espace paramétrique (voir tableau 3.2). La raison pour ce coût est l'étape d'optimisation unidimensionnelle sur une direction aléatoire h pour trouver les $\theta^* = \theta^i + \alpha^* h$ ce qui sert à construire EOA. On peut améliorer cette étape en déterminant d'abord la limite de α. Cette valeur sera déterminée par l'intersection entre l'EOI et vecteur $\theta^i + \alpha h$. L'algorithme POD-ISAT a bien été appliqué pour la conception d'un système ECS simplifié. Nous avons envisagé seulement le champ de température. Pour la poursuite des travaux, nous essayons de prendre en compte aussi le champ de vitesse, la présence des passagers ou des équipements électriques allumés et augmenter la dimension de l'espace de paramètres. En général, l'algorithme peut servir aux problèmes EDP paramétriques. Notamment, il peut être appliqué pour les simulations CFD stationnaire paramétrique. Donc, nous allons aussi l'appliquer pour la conception d'un système de contrôle de l'air dans un bâtiment ou un habitacle de voiture.

En ce qui concerne de l'algorithme pararéel, l'accélération de celui-ci est encore loin le but de résolution en temps réel. On peut probablement améliorer le speed-up ainsi que la vitesse de convergence en modifiant le propagateur grossier. Nous avons proposé deux pistes de modifications en appliquant l'idée de l'utilisation des réductions d'ordre de modèles. Une continuation de ces travaux sur ces pistes sera intéressante. Ces idées ont aussi besoin d'être validées par l'expérimentation numérique.

Bibliographie

[ash, SDa] *ASHRAE (1992) Thermal environmental conditions for human occupancy, ANSI/ ASHRAE 55-1992.American Society of Heating, Refrigerating and Air-Conditioning Engineers, Atlanta, GA.*

[ash, SDb] *ASHRAE (2001) Ventilation for acceptable indoor air quality, standard 62-2001. American Society of Heating, Refrigerating and Air-Conditioning Engineers, Atlanta, GA.*

[ash, SDc] *ASHRAE standard 161-2000(2000). Air quality within commercial aircraft.Committee review draft. Atlanta, GA.*

[con, SD] Domain decomposition methods in science and engineering. volume 40. Springer, Berlin Heidelberg-New York, 2003.

[A., 1996] A., B. (1996). Cabin air comfort. *FAST TECHNICAL MAGAZINE*, (50):22–29.

[Agte, 2005] Agte, J. (2005). A tool for application of bi-level integrated system synthesis to multidisciplinary design optimization problems. *In German Air and Space Congress*, numéro DGLR-2005-241.

[Algazi et Sakrison, 1969] Algazi, V. et Sakrison, D. (1969). On the optimality of the karhunen-loève expansion (corresp.). *Information Theory, IEEE Transactions on*, 15(2):319 – 321.

[Almroth *et al.*, 1978] Almroth, B. O., Stern, P. et Brogan, F. A. (1978). Automatic choice of global shape functions in structural analysis. *AIAA Journal*, 16:525–528.

[Ammar *et al.*, 2006] Ammar, A., Mokdad, B., Chinesta, F. et Keunings, R. (2006). A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *Journal of Non-Newtonian Fluid Mechanics*, 139(3):153 – 176.

[Amsallem *et al.*, 2012] Amsallem, D., Zahr, M. J. et Farhat, C. (2012). Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, 92(10):891–916.

[Anderson *et al.*, 1995] Anderson, J. *et al.* (1995). *Computational fluid dynamics*, volume 206. McGraw-Hill.

[Andrews *et al.*, 1967] Andrews, C., Davies, J. et Schwarz, G. (1967). Adaptive data compression. *Proceedings of the IEEE*, 55(3):267 – 277.

[Arsenlis *et al.*, 2006] Arsenlis, A., Barton, N., Becker, R. et Rudd, R. (2006). Generalized in situ adaptive tabulation for constitutive model evaluation in plasticity. *Computer Methods in Applied Mechanics and Engineering*, 196(1-3):1–13.

[Association, 2002] Association, A. M. (2002). *Medical Guidelines for Airline Passengers*. Aerospace Medical Association.

[Aubanel, 2011] Aubanel, E. (2011). Scheduling of tasks in the parareal algorithm. *Parallel Comput.*, 37(3):172–182.

[Audouze *et al.*, 2009] Audouze, C., De Vuyst, F. et Nair, P. B. (2009). Reduced-order modeling of parameterized pdes using time-space-parameter PCA. *IJNME*, 80(8):1025–1057.

[Baffico *et al.*, 2002] Baffico, L., Bernard, S., Maday, Y., Turinici, G. et Zérah, G. (2002). Parallel-in-time molecular-dynamics simulations. *Phys. Rev. E*, 66:057701.

[Bal, 2003] Bal, G. (2003). Parallelization in time of (stochastic) ordinary differential equations. *Preprint*.

[Bal, 2005] Bal, G. (2005). On the convergence and the stability of the parareal algorithm to solve partial differential equations. *In* Barth, T., Griebel, M., Keyes, D., Nieminen, R., Roose, D., Schlick, T., Kornhuber, R., Hoppe, R., Périaux, J., Pironneau, O., Widlund, O. et Xu, J., éditeurs : *Domain Decomposition Methods in Science and Engineering*, volume 40 de *Lecture Notes in Computational Science and Engineering*, pages 425–432. Springer Berlin Heidelberg.

[Bal et Maday, 2002] Bal, G. et Maday, Y. (2002). A "parareal" time discretization for non-linear pde's with application to the pricing of an american put. *In Recent developments in domain decomposition methods (Zürich, 2001), Vol. 23 of Lect. Notes Comput. Sci.Eng.*, pages 189–202. Berlin,Springer Verlag,Berlin.

[Barbič et Popović, 2008] Barbič, J. et Popović, J. (2008). Real-time control of physically based simulations using gentle forces. *ACM Trans. on Graphics (SIGGRAPH Asia 2008)*, 27(5):163 :1–163 :10.

[Barbič *et al.*, 2012] Barbič, J., Sin, F. et Grinspun, E. (2012). Interactive editing of deformable simulations. *ACM Trans. Graph.*, 31(4):70 :1–70 :8.

[Beringhier *et al.*, 2010] Beringhier, M., Gueguen, M. et Grandidier, J. (2010). Solution of strongly coupled multiphysics problems using space-time separated representations. application to thermoviscoelasticity. *Arch. of Comp. Methods in Engineering*, 17:393–401. 10.1007/s11831-010-9050-5.

[Bianco *et al.*, 2009] Bianco, V., Manca, O., Nardini, S. et Roma, M. (2009). Numerical investigation of transient thermal and fluidynamic fields in an executive aircraft cabin. *Applied Thermal Engineering*, 29(16):3418 – 3425.

[Blouza *et al.*, 2010] Blouza, A., Boudin, L. et Kaber, S. M. (2010). Parallel in time algorithms with reduction methods for solving chemical kinetics. *Communications in Applied Mathematics & Computational Science*, 5(2):241–263.

[Bonithon *et al.*, 2011] Bonithon, G., Joyot, P., Chinesta, F. et Villon, P. (2011). Non-incremental boundary element discretization of parabolic models based on the use of the proper generalized decompositions. *Engineering Analysis with Boundary Elements*, 35(1):2 – 17.

[Breitkopf *et al.*, 2005] Breitkopf, P., Naceur, H., Rassineux, A. et Villon, P. (2005). Moving least squares response surface approximation : Formulation and metal forming applications. *Computers & Structures*, 83(17-18):1411 – 1428. Advances in Meshfree Methods.

[Buhmann, 2003] Buhmann, M. D. (2003). *Radial Basis Functions*. Cambridge University Press.

[Bui-Thanh, 2007] Bui-Thanh, T. (2007). *Model-constrained optimization methods for reduction of parameterized large-scale systems*. Thèse de doctorat, Massachusetts Institute of Technology.

[Bui-Thanh *et al.*, 2008a] Bui-Thanh, T., Willcox, K. et Ghattas, O. (2008a). Model reduction for large-scale systems with high-dimensional parametric input space. *SIAM Journal on Scientific Computing*, 30(6):3270–3288.

[Bui-Thanh *et al.*, 2008b] Bui-Thanh, T., Willcox, K. et Ghattas, O. (2008b). MODEL REDUCTION FOR LARGE-SCALE SYSTEMS WITH HIGH-DIMENSIONAL PARAMETRIC INPUT SPACE. *SIAM JOURNAL ON SCIENTIFIC COMPUTING*, 30(6, SI):3270–3288. SIAM Conference on Computational Science and Engineering, Costa Mesa, CA, FEB 19-23, 2007.

[Bui-Thanh *et al.*, 2008c] Bui-Thanh, T., Willcox, K. et Ghattas, O. (2008c). Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications. *AIAA Journal*, 46(10):2520–2529.

[Burkardt *et al.*, 2006] Burkardt, J., Gunzburger, M. et Lee, H.-C. (2006). Pod and cvt-based reduced-order modeling of navier - stokes flows. *Computer Methods in Applied Mechanics and Engineering*, 196(1 - 3):337 – 355.

[Burmeister, 1993] Burmeister, L. C. (1993). *Convective heat transfer*. New York : Wiley, c1993, United-States of America, $2^n d$ édition. "A Wiley-Interscience publication.".

[Carlberg *et al.*, 2011] Carlberg, K., Bou-Mosleh, C. et Farhat, C. (2011). Efficient non-linear model reduction via a least-squares petrov-galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(2):155–181.

[Chatterjee, 2000] Chatterjee, A. (2000). An introduction to the proper orthogonal decomposition. *Current Science*, 78(7):808–817.

[Chen *et al.*, 2009] Chen, J., Kostandov, M., Pivkin, I. V., Riskin, D. K., Willis, D. J., Swartz, S. M. et Laidlaw, D. H. (2009). Visual analysis of dimensionality reduction for exploring bat flight kinematics in a virtual environment. *In Proceedings of the 15th Joint virtual reality Eurographics conference on Virtual Environments*, JVRC'09, pages 77–84, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

[Chen, 2012] Chen, K., K. S. H. T. G. D. e. a. (2012). Thermal comfort prediction and validation in a realistic vehicle thermal environment. SAE Technical Paper.

[Chinesta *et al.*, 2010] Chinesta, F., Ammar, A. et Cueto, E. (2010). Recent advances and new challenges in the use of the proper generalized decomposition for solving multidimensional models. *Archives of Computational Methods in Engineering*, 17:327–350. 10.1007/s11831-010-9049-y.

[Chinesta *et al.*, 2011] Chinesta, F., Ammar, A., Leygue, A. et Keunings, R. (2011). An overview of the proper generalized decomposition with applications in computational rheology. *Journal of Non-Newtonian Fluid Mechanics*, In Press, Corrected Proof:–.

[Ciarlet, 1990] Ciarlet, P. G. (1990). *Introduction à l'analyse numérique matricielle et à l'optimisation*. Masson,Paris.

[Committee on Air Quality in Passenger Cabins of Commercial Aircraft et Toxicology, 2002] Committee on Air Quality in Passenger Cabins of Commercial Aircraft, B. o. E. S. et Toxicology, N. R. C. (2002). *The Airliner Cabin Environment and the Health of Passengers and Crew*. The National Academies Press.

[Cordier L., 2003] Cordier L., B. M. (2003). Proper orthogonal decomposition : an overview. post-processing of experimental and numerical data Lecture series 2003-04, Von Karman Institute for Fluid Dynamics.

[Cortial et Farhat, 2007] Cortial, J. et Farhat, C. (2007). *Real-Time PDE-Constrained Optimization*, chapitre A Time-Parallel Implicit Methodology for the

Near-Real-Time Solution of Systems of Linear Oscillators, pages 115–143. Computational Science and Engineering.

[Cortial et Farhat, 2009] Cortial, J. et Farhat, C. (2009). A time-parallel implicit method for accelerating the solution of non-linear structural dynamics problems. *International Journal for Numerical Methods in Engineering*, 77:451–470.

[Couplet, 2005] Couplet, M. (2005). *Modélisation POD-Galerkine réuite pour le contrôle des écoulements instationnaires*. Thèse de doctorat, Université paris 13.

[Couplet *et al.*, 2005] Couplet, M., Basdevant, C. et Sagaut, P. (2005). Calibrated reduced-order pod-galerkin system for fluid flow modelling. *Journal of Computational Physics*, 207(1):192 – 220.

[Cressie, 1990] Cressie, N. (1990). The origins of kriging. *Mathematical Geology*, 22:239–252.

[Cressie, 1993] Cressie, N. A. C. (1993). *Statistics for Spatial Data, revised edition*. John Wiley & Sons, New York.

[D., 1996] D., C. (1996). Cabin air comfort. *FAST TECHNICAL MAGAZINE*, (19):4–11.

[Dai, 1997] Dai, L. (1997). Rate of convergence for derivative estimation of discrete-time markov chains via finite-difference approximation with common random numbers. *SIAM J. Appl. Math.*, 57(3):731–751.

[Davoodi et Greig, 2011] Davoodi, A. et Greig, I. (2011). Star-ccm+ for complex cae design problems. 25th European Workshop on Thermal and ECLS Software.

[Dechow *et al.*, 1997] Dechow, M., Sohn, H. et Steinhanses, J. (1997). Concentrations of selected contaminants in cabin air of airbus aircrafts. *Chemosphere*, 35(1-2):21–31. Experimental and Theoretical Approaches in Environmental Chemistry.

[Deparis, 2008] Deparis, S. (2008). Reduced basis error bound computation of parameter-dependent Navier-Stokes equations by the natural norm approach. *SIAM J. Num. A.*, 46(4):2039–2067.

[Du *et al.*, 1999] Du, Q., Faber, V. et Gunzburger, M. (1999). Centroidal voronoi tessellations : Applications and algorithms. *SIAM Rev.*, 41(4):637–676.

[Dumon *et al.*, 2011] Dumon, A., Allery, C. et Ammar, A. (2011). Proper general decomposition (pgd) for the resolution of Navier-Stokes equations. *Journal of Computational Physics*, 230(4):1387 – 1407.

[Dumyahn *et al.*, 2000] Dumyahn, T., Spengler, J., Burge, H. et Muilenburg, M. (2000). Comparison of the environments of transportation vehicles : results of two surveys. *ASTM special technical publication*, 1393:3–25.

[Elwood H. Hunt et Tilton, 1995] Elwood H. Hunt, Dr. Don H. Reid, D. R. S. et Tilton, D. F. E. (1995). Commercial airliner environmental control system engineering aspects of cabin air quality. *Fuel*, pages 1–8.

[Engblom, 2008] Engblom, S. (2008). Parallel in time simulation of multiscale stochastic chemical kinetics. Rapport technique 2008-020, Department of Information Technology, Uppsala University. Extended abstract to appear in Proceedings of ICNAAM 2008.

[Farhat et Chandesris, 2003] Farhat, C. et Chandesris, M. (2003). Time-decomposed parallel time-integrators : theory and feasibility studies for fluid, structure, and fluid-structure applications. *International Journal for Numerical Methods in Engineering*, 58(9):1397–1434.

[Filomeno Coelho *et al.*, 2008] Filomeno Coelho, R., Breitkopf, P. et Knopf-Lenoir, C. (2008). Model reduction for multidisciplinary optimization - application to a 2d wing. *Structural and Multidisciplinary Optimization*, 37:29–48.

[Filomeno Coelho *et al.*, 2009] Filomeno Coelho, R., Breitkopf, P., Knopf-Lenoir, C. et Villon, P. (2009). Bi-level model reduction for coupled problems. *Structural and Multidisciplinary Optimization*, 39:401–418.

[Fischer *et al.*, 2004] Fischer, P. F., Hecht, F. et Maday, Y. (2004). A parareal in time semi-implicit approximation of the navier-stokes equations. *In Proceedings of Fifteen International Conference on Domain Decomposition Methods*, pages 433–440. Springer Verlag.

[Galan delan Sastre et Bermejo, 2008] Galan delan Sastre, P. et Bermejo, R. (2008). Error estimates of proper orthogonal decomposition eigenvectors and galerkin projection for a general dynamical system arising in fluid models. *Numer. Math.*, 110:49–81.

[Gander, 2008] Gander, M. J. (2008). Analysis of the parareal algorithm applied to hyperbolic problems using characteristics. *Boletín de la Sociedad Española de Matemática Aplicada*, 42:21–35. ID : unige :6268.

[Gander et Hairer, 2008] Gander, M. J. et Hairer, E. (2008). Nonlinear convergence analysis for the parareal algorithm. *In* Langer, U., Discacciati, M., Keyes, D. E., Widlund, O. B., Zulehner, W., Barth, T. J., Griebel, M., Keyes, D. E., Nieminen, R. M., Roose, D. et Schlick, T., éditeurs : *Domain Decomposition Methods in Science and Engineering XVII*, volume 60 de *Lecture Notes in Computational Science and Engineering*, pages 45–56. Springer Berlin Heidelberg.

[Gander et Petcu, 2008] Gander, M. J. et Petcu, M. (2008). Analysis of a Krylov subspace enhanced parareal algorithm for linear problems. *In ESAIM : Proceedings*, volume 25, pages 114–129. edpsciences. org.

[Gander et Vandewalle, 2007] Gander, M. J. et Vandewalle, S. (2007). Analysis of the parareal time-parallel time-integration method. *SIAM J. Sci. Comput.*, 29(2):556–578.

[Gandin, 1963] Gandin, L. S. (1963). *Objective Analysis of Meteorological Fields*. Gidrometeorologicheskoe Izdatel'stvo (GIMIZ), Leningrad.

[Garner *et al.*, 2004] Garner, R., Wong, K., Ericson, S., Baker, A. et Orzechowski, J. (2004). Cfd validation for contaminant transport in aircraft cabin ventilation flow fields. Rapport technique, DTIC Document.

[Garrido *et al.*, 2003] Garrido, I., Espedal, M. S. et Fladmark, G. E. (2003). Fladmark, a convergence algorithm for time parallelization applied to reservoir simulation. *In in Proceedings of the 15th International Domain Decomposition Conference, Lect. Notes Comput. Sci*, pages 469–476. Springer.

[Gérard, 2005] Gérard, D. (2005). *Neural networks : methodology and applications*. Editions Eyrolles.

[Gordon *et al.*, 2007] Gordon, R., Masri, A., Pope, S. et Goldin, G. (2007). A numerical study of auto-ignition in turbulent lifted flames issuing into a vitiated co-flow. *Combustion Theory and Modelling*, 11(3):351–376.

[Graham et Tang, 1999a] Graham, W. R., P. J. et Tang, K. (1999a). Optimal control of vortex shedding using low-order models. part ii model based control. *International Journal for Numerical Methods in Engineering*, 44:495–976.

[Graham et Tang, 2000a] Graham, W. R., P. J. et Tang, K. (2000a). Reduced-order adaptive controllers for fluid flows using pod. *Journal of Scientific Computing*, 15:457–478.

[Graham et Tang, 2000b] Graham, W. R., P. J. et Tang, K. (2000b). A reduced order approach to optimal control of fluids using proper orthogonal decomposition. *International Journal for Numerical Methods in Fluids*, 34:425–448.

[Graham et Tang, 1999b] Graham, W. R., P. J. et Tang, K. Y. (1999b). Optimal control of vortex shedding using low-order models. part i-open-loop model development. *International Journal for Numerical Methods in Engineering*, 44:495–976.

[Grepl, 2005] Grepl, M. A. (2005). *Reduced-Basis Approximation and A Posteriori Error Estimation for Parabolic Partial Differential Equations.* Thèse de doctorat, Massachusetts Institute of Technology.

[Grepl et Patera, 2005] Grepl, M. A. et Patera, A. T. (2005). A posteriori error bounds for reduced-basis approximations of parametrized parabolic partial differential equations. *ESAIM : Mathematical Modelling and Numerical Analysis*, 39(01):157–181.

[Haasdonk, 2011] Haasdonk, B. (2011). Convergence rates of the pod-greedy method. Submitted to CRAS.

[Haasdonk et Ohlberger, 2008] Haasdonk, B. et Ohlberger, M. (2008). Reduced basis method for finite volume approximations of parametrized linear evolution equations. *ESAIM : Mathematical Modelling and Numerical Analysis*, 42(02):277–302.

[Hedengren et Edgar, 2008] Hedengren, J. D. et Edgar, T. F. (2008). Approximate nonlinear model predictive control with in situ adaptive tabulation. *Computers & Chemical Engineering*, 32(4-5):706 – 714. Festschrift devoted to Rex Reklaitis on his 65th Birthday.

[Hinze et Volkwein, 2008] Hinze, M. et Volkwein, S. (2008). Error estimates for abstract linear-quadratic optimal control problems using proper orthogonal decomposition. *Computational Optimization and Applications*, 39:319–345. 10.1007/s10589-007-9058-4.

[Hirsch, 2007] Hirsch, C. (2007). *Numerical computation of internal and external flows : the fundamentals of Computational Fluid Dynamics*, volume 1. Butterworth-Heinemann.

[Holmes, 1990] Holmes, P. (1990). Can dynamical systems approach turbulence ? *In* Lumley, J., éditeur : *Whither Turbulence ? Turbulence at the Crossroads*, volume 357 de *Lecture Notes in Physics*, pages 195–249. Springer Berlin Heidelberg.

[Hunt et Space, 1994] Hunt, E. H. et Space, D. R. (1994). The airplane cabin environment. issues pertaining to flight attendant comfort. Montreal, Canada.

[Iollo *et al.*, 2000] Iollo, A., Lanteri, S. et Désidéri, J.-A. (2000). Stability properties of pod-galerkin approximations for the compressible navier-stokes equations. *Theoretical and Computational Fluid Dynamics*, 13:377–396. 10.1007/s001620050119.

[James et Fatahalian, 2003] James, D. L. et Fatahalian, K. (2003). Precomputing interactive dynamic deformable scenes. *ACM Trans. Graph.*, 22(3):879–887.

[Jones *et al.*, 1998] Jones, D. R., Schonlau, M. et Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *J. of Global Optimization*, 13(4):455–492.

[Jouvray et Tucker, 2005] Jouvray, A. et Tucker, P. G. (2005). Computation of the flow in a ventilated room using non-linear rans, les and hybrid rans/les. *International Journal for Numerical Methods in Fluids*, 48(1):99–106.

[Kühn *et al.*, 2009] Kühn, M., Bosbach, J. et Wagner, C. (2009). Experimental parametric study of forced and mixed convection in a passenger aircraft cabin mock-up. *Building and Environment*, 44(5):961 – 970.

[Kirby et Sirovich, 1990] Kirby, M. et Sirovich, L. (1990). Application of the karhunen-loeve procedure for the characterization of human faces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(1):103 –108.

[Kodiyalam *et al.*, 2000] Kodiyalam, S., Sobieszczanski-sobieski, J., Kodiyalam, S. et Sobieszczanski-sobieski, J. (2000). Bilevel integrated system synthesis with response surfaces. *AIAA Journal*, 38(8):1479–1485.

[Kris K. Hauser, 2003] Kris K. Hauser, Chen Shen, J. F. O. (2003). Interactive deformation using modal analysis with constraints. *In Graphics Interface*, pages 247–256. CIPS, Canadian Human-Computer Commnication Society.

[Kunisch et Volkwein, 2002] Kunisch, K. et Volkwein, S. (2002). Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM J. on Numerical Analysis*, 40(2):492–515.

[Lancaster et Salkauskas, 1981] Lancaster, P. et Salkauskas, K. (1981). Surfaces Generated by Moving Least Squares Methods. *Mathematics of Computation*, 37(155):141–158.

[Lassila et Rozza, 2010] Lassila, T. et Rozza, G. (2010). Parametric free-form shape design with pde models and reduced basis method. *Computer Methods in Applied Mechanics and Engineering*, 199(23-24):1583 – 1592.

[Launder et Spalding, 1974] Launder, B. et Spalding, D. (1974). The numerical computation of turbulent flows. *Computer Methods in Applied Mechanics and Engineering*, 3(2):269 – 289.

[Leblond *et al.*, 2011] Leblond, C., Allery, C. et Inard, C. (2011). An optimal projection method for the reduced-order modeling of incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 200(33-36):2507–2527.

[LeGresley, 2005] LeGresley, P. A. (2005). *Application of Proper Orthogonal Decomposition (POD) to Design Decompo- sition Methods.* Thèse de doctorat, Standford University.

[Leo et Pérez-Grande, 2005] Leo, T. J. et Pérez-Grande, I. (2005). A thermoeconomic analysis of a commercial aircraft environmental control system. *Applied Thermal Engineering*, 25(2-3):309 – 325.

[Leygue et Verron, 2010] Leygue, A. et Verron, E. (2010). A first step towards the use of proper general decomposition method for structural optimization. *ACME*, 17:465–472. 10.1007/s11831-010-9052-3.

[LIANG *et al.*, 2002] LIANG, Y., LEE, H., LIM, S., LIN, W., LEE, K. et WU, C. (2002). Proper orthogonal decomposition and its applications–part i : Theory. *Journal of Sound and Vibration*, 252(3):527 – 544.

[Lieberman et Ghattas, 2010] Lieberman, Chad, K. W. et Ghattas, O. (2010). Parameter and state model reduction for large-scale statistical inverse problems. *SIAM Journal on Scientific Computing*, 32.5:2523–2542.

[Lions *et al.*, 2001] Lions, J.-L., Maday, Y. et Turinici, G. (2001). Résolution d'edp par un schéma en temps "pararéel". a "parareal" in time discretization of pde's. *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, 332(7):661 – 668.

[Liu *et al.*, 2012] Liu, W., Mazumdar, S., Zhang, Z., Poussou, S. B., Liu, J., Lin, C.-H. et Chen, Q. (2012). State-of-the-art methods for studying air distributions in commercial airliner cabins. *Building and Environment*, 47(0):5 – 12. International Workshop on Ventilation, Comfort, and Health in Transport Vehicles.

[Lophaven *et al.*, 2002] Lophaven, S., Nielsen, H. et Søndergaard, J. (2002). *Aspects of the Matlab toolbox DACE*. Informatics and Mathematical Modelling, Technical University of Denmark, DTU.

[Lu et Pope, 2009] Lu, L. et Pope, S. B. (2009). An improved algorithm for in situ adaptive tabulation. *Journal of Computational Physics*, 228(2):361 – 386.

[Lumley, 1967] Lumley, J. L. (1967). The Structure of Inhomogeneous Turbulent Flows. *In* Yaglom, A. M. et Tatarski, V. I., éditeurs : *Atmospheric turbulence and radio propagation*, pages 166–178. Nauka, Moscow.

[Maday, 2007] Maday, Y. (2007). Parareal in time algorithm for kinetic systems based on model reduction. *In High-dimensional partial differential equations in science and engineering*, volume 41, pages 183–194. American Mathematical Society.

[Maday *et al.*, 2007] Maday, Y., Salomon, J. et Turinici, G. (2007). Monotonic parareal control for quantum systems. *SIAM J. Numer. Anal.*, 45:2468–2482.

[Manzoni *et al.*, 2012] Manzoni, A., Quarteroni, A. et Rozza, G. (2012). Shape optimization for viscous flows by reduced basis methods and free-form deformation. *International Journal for Numerical Methods in Fluids*, 70(5):646–670.

[Matheron, 1963] Matheron, G. (1963). Principles of geostatistics. *Economic Geology*, 58:1246–1266.

[Mazumder et ASME, 2007] Mazumder, S. et ASME, M. (2007). Modeling full-scale monolithic catalytic converters : challenges and possible solutions. *Journal of Heat Transfer*, 129:526.

[Mo *et al.*, 2003] Mo, H., Hosni, M. et Jones, B. (2003). Application of particle image velocimetry for the measurement of the airflow characteristics in an aircraft cabin. *TRANSACTIONS-AMERICAN SOCIETY OF HEATING REFRIGERATING AND AIR CONDITIONING ENGINEERS*, 109(2):101–110.

[NAGARAJAN, 2010] NAGARAJAN, K. K. (2010). *Analysis and control of self-sustained instabilities in a cavity using reduced order modelling*. Thèse de doctorat, Institut National Polytechnique de Toulouse.

[Nagy, 1979] Nagy, D. A. (1979). Modal representation of geometrically nonlinear behaviour by the finite element method. *Computers and Structures*, 10:683–688.

[Nair *et al.*, 2003] Nair, P., Choudhury, A. et Keane, A. (2003). Some greedy learning algorithms for sparse regression and classification with Mercer kernels. *JOURNAL OF MACHINE LEARNING RESEARCH*, 3(4-5):781–801. 18th International Conference on Machine Learning, WILLIAMSTOWN, MASSACHUSETTS, JUN 28-JUL 01, 2001.

[Ngoc Cuong *et al.*, 2005] Ngoc Cuong, N., Veroy, K. et Patera, A. (2005). Certified real-time solution of parametrized partial differential equations. *In* Yip, S., éditeur : *Handbook of Materials Modeling*, pages 1529–1564. Springer Netherlands.

[Nouy, 2010] Nouy, A. (2010). A priori model reduction through proper generalized decomposition for solving time-dependent partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 199(23-24):1603 – 1626.

[P. Holmes, 1996] P. Holmes, J.L. Lumley, G. B. (1996). *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, Cambridge, UK.

[PAPE, 2003] PAPE, H. L. (2003). *Etude des propriétés germicides de fibres de carbone active : application à la décontamination de l'air en cabine d'avion*. Thèse de doctorat, Université de limoges.

[Park *et al.*, 2009] Park, C., Joh, C.-Y. et Kim, Y.-S. (2009). Multidisciplinary design optimization of a structurally nonlinear aircraft wing via parametric modeling. *International Journal of Precision Engineering and Manufacturing*, 10:87–96.

[Pentland et Williams, 1989] Pentland, A. et Williams, J. (1989). Good vibrations : modal dynamics for graphics and animation. *SIGGRAPH Comput. Graph.*, 23(3):207–214.

[Peter Binev et Wojtaszczyk, 2011] Peter Binev, Albert Cohen, W. D. R. D. G. P. et Wojtaszczyk, P. (2011). Convergence rates for greedy algorithms in reduced basis methods. *SIAM J. on Mathematical Analysis*, 43(3):1457–1472.

[Peterson, 1989] Peterson, J. S. (1989). The reduced basis method for incompressible viscous flow calculations. *SIAM J. Sci. Stat. Comput.*, 10(4):777–786.

[Pierre, 1999] Pierre, L. (1999). *Nonlinear Computational Structural Mechanics. New Approaches and Non-Incremental Methods of Calculation*. Mech. Eng. Series. Springer-verlag.

[Pironneau, 1982] Pironneau, O. (1982). On the transport-diffusion algorithm and its applications to the navier-stokes equations. *Numerische Mathematik*, 38:309–332.

[Placzek *et al.*, 2011] Placzek, A., Tran, D.-M. et Ohayon, R. (2011). A nonlinear pod-galerkin reduced-order model for compressible flows taking into account rigid body motions. *Computer Methods in Applied Mechanics and Engineering*, 200(49-52):3497 – 3514.

[Pope, 1997] Pope, S. (1997). Computationally efficient implementation of combustion chemistry using in situ adaptive tabulation. *Combustion Theory and Modelling*, 1(1):41–63.

[Pope, 2008] Pope, S. (2008). Algorithms for ellipsoids. Rapport technique, Cornell University FDA.

[Pérez-Grande et Leo, 2002] Pérez-Grande, I. et Leo, T. J. (2002). Optimization of a commercial aircraft environmental control system. *Applied Thermal Engineering*, 22(17):1885 – 1904.

[Quarteroni *et al.*, 2011] Quarteroni, A., Rozza, G. et Manzoni, A. (2011). Certified reduced basis approximation for parametrized partial differential equations and applications. *Journal of Mathematics in Industry*, 1(1):1–49.

[Rambo et Joshi, 2007] Rambo, J. et Joshi, Y. (2007). Reduced-order modeling of turbulent forced convection with parametric conditions. *International Journal of Heat and Mass Transfer*, 50(3-4):539 – 551.

[Rasmussen et Williams, 2006] Rasmussen, C. et Williams, C. (2006). *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, MA.

[Ravindran, 2011] Ravindran, S. (2011). A reduced order approach to optimal control of fluids using proper orthogonal decomposition. *Numerical Methods for Partial Differential*, 27(6):1639–1665.

[Rheinboldt, 1993] Rheinboldt, W. C. (1993). On the theory and error estimation of the reduced basis method for multi-parameter problems. *Nonlinear Anal.*, 21:849–858.

[Riskin *et al.*, 2008] Riskin, D. K., Willis, D. J., Iriarte-Díaz, J., Hedrick, T. L., Kostandov, M., Chen, J., Laidlaw, D. H., Breuer, K. S. et Swartz, S. M. (2008). Quantifying the complexity of bat wing kinematics. *Journal of Theoretical Biology*, 254(3):604 – 615.

[ROSENFELD, 1982] ROSENFELD, A., K. A. C. (1982). *Digital picture processing*, volume 1&2. New York, Academic Press.

[Rowley *et al.*, 2004] Rowley, C. W., Colonius, T. et Murray, R. M. (2004). Model reduction for compressible flows using pod and galerkin projection. *Physica D : Nonlinear Phenomena*, 189(1-2):115 – 129.

[Rozza *et al.*, 2007] Rozza, G., Huynh, D. et Patera, A. (2007). Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods in Engineering*, 15:229–275. 10.1007/s11831-008-9019-9.

[Rozza et Veroy, 2007] Rozza, G. et Veroy, K. (2007). On the stability of the reduced basis method for Stokes equations in parametrized domains. *CMAME*, 196(7):1244 – 1260.

[Ruprecht et Krause, 2012] Ruprecht, D. et Krause, R. (2012). Explicit parallel-in-time integration of a linear acoustic-advection system. *Computers & Fluids*, 59(0):72 – 83.

[Sacks *et al.*, 1989] Sacks, J., Welch, W., Mitchell, T. et Wynn, H. (1989). Design and analysis of computer experiments. *Statistical science*, 4(4):409–423.

[Simpson *et al.*, 2001] Simpson, T. W., Poplinski, J. D., Koch, P. N. et Allen, J. K. (2001). Metamodels for Computer-based Engineering Design : Survey and recommendations. *Engineering With Computers*, 17:129–150.

[Sirovich, 1987] Sirovich, L. (1987). Turbulence and the dynamics of coherent structures. I - Coherent structures. II - Symmetries and transformations. III - Dynamics and scaling. *Quarterly of Applied Mathematics*, 45:561–571.

[Sirovich, 1989] Sirovich, L. (1989). Chaotic dynamics of coherent structures. *Physica D : Nonlinear Phenomena*, 37(1-3):126–145.

[Staff et Rønquist, 2003] Staff, G. A. et Rønquist, E. M. (2003). Rønquist, stability of the parareal algorithm. *In in Proceedings of the 15th International Domain Decomposition Conference, Lect. Notes Comput. Sci*, pages 449–456. Springer.

[Stein, 1999] Stein, M. (1999). *Interpolation of spatial data : some theory for kriging*. Springer Verlag.

[Sun *et al.*, 2005] Sun, Y., Zhang, Y., Wang, A., Topmiller, J., Bennet, J. et BESANT, R. (2005). Experimental characterization of airflows in aircraft cabins,

part i : Experimental system and measurement procedure. discussion. *ASHRAE transactions*, pages 45–52.

[Vankan, 2003] Vankan, W.J. | Kos, J. . L. W. (2003). Approximation models for multi-disciplinary system design - application in a design study of power optimised aircraft. Rapport technique, National Aerospace Laboratory NLR, Amsterdam, The Netherlands.

[Varshney et Armaou, 2006] Varshney, A. et Armaou, A. (2006). An efficient optimization approach for computationally expensive timesteppers using tabulation. *In* Gorban, A., Kevrekidis, I., Theodoropoulos, C., Kazantzis, N. et Öttinger, H., éditeurs : *Model Reduction and Coarse-Graining Approaches for Multiscale Phenomena*, pages 515–533. Springer Berlin Heidelberg.

[Veroy et Patera, 2005] Veroy, K. et Patera, A. T. (2005). Certified real–time solution of the parametrized steady incompressible navier–stokes equations : rigorous reduced–basis a posteriori error bounds. *International Journal for Numerical Methods in Fluids*, 47(8-9):773–788.

[Veroy *et al.*, 2003] Veroy, K., Prud'homme, C., Rovas, D. et Patera, A. (2003). A posteriori error bounds for reduced-basis approximation of parametrized noncoercive and nonlinear elliptic partial differential equations. *In Proceedings of the 16th AIAA Computational Fluid Dynamics Conference*, volume 3847.

[Versteeg et Malalasekera, 2007] Versteeg, H. et Malalasekera, W. (2007). *An introduction to computational fluid dynamics : the finite volume method*. Prentice Hall.

[VUYST et AUDOUZE, 2009] VUYST, F. D. et AUDOUZE, C. (2009). *Optimisation multidisciplinaire en mécanique 2*, chapitre 2, pages 73–122. LAVOISIER.

[Wang et Chen, 2010] Wang, M. et Chen, Q. Y. (2010). On a hybrid rans/les approach for indoor airflow modeling (rp-1271). *HVAC&R Research*, 16(6):731–747.

[Waters *et al.*, 2002] Waters, M., Bloom, T., Grajewski, B. et Deddens, J. (2002). Measurements of indoor air quality on commercial transport aircraft. *Proceedings Indoor Air*.

[Yakhot et Orszag, 1986] Yakhot, V. et Orszag, S. (1986). Renormalization group analysis of turbulence. i. basic theory. *Journal of Scientific Computing*, 1:3–51.

[Yan *et al.*, 2009] Yan, W., Zhang, Y., Sun, Y. et Li, D. (2009). Experimental and cfd study of unsteady airborne pollutant transport within an aircraft cabin mock-up. *Building and Environment*, 44(1):34 – 43.

[Zadeh *et al.*, 2009] Zadeh, P., Toropov, V. et Wood, A. (2009). Metamodel-based collaborative optimization framework. *Structural and Multidisciplinary Optimization*, 38:103–115.

[Zhang et Chen, 2007] Zhang, T. et Chen, Q. (2007). Novel air distribution systems for commercial aircraft cabins. *Building and Environment*, 42(4):1675–1684.

[Zhang *et al.*, 2010] Zhang, T. T., Yin, S. et Wang, S. (2010). An under-aisle air distribution system facilitating humidification of commercial aircraft cabins. *Building and Environment*, 45(4):907 – 915.

Annexe A

Méthode d'échantillonnage par hypercube latin

La méthode d'échantillonnage par hypercube latin (en anglais, Latin Hypercube Sampling ou LHS) a été introduite par Mac Kay, Conover et Beckman (1979) pour évaluer numériquement les intégrales multiples. Elle permet d'assurer la non-répétition de l'information au travers d'une bonne répartition des projections sur les axes factoriels. En pratique, les hypercubes latins sont très utilisés en génération d'un plan d'expériences numériques notamment pour leur simplicité d'usage et de construction.

Description :

Supposons qu'on veut générer un plan d'expérience de n point dans l'hypercube unité $[0,1]^p$. Le principe de la LHS est le suivant. On découpe chaque axe par n segments uniformes de taille de $\frac{1}{n}$. On obtient un maillage de n^p petits hypercubes (cellules) de même taille. Ensuite, on choisit n cellules parmi les n^p de telle manière que les projections de n cellules sur chaque axe ne coïncide pas. Enfin, on tire un point au hasard dans chaque cellule présélectionnée.

Algorithme LHS :

- Étape 1. On définit une matrice $\boldsymbol{\pi}$ de taille $n \times p$, dont chaque colonne est une mutation aléatoire de $1, 2, ..., n$. Chaque ligne de cette matrice correspond avec une cellule sélectionnée attachée à point $\frac{1}{n}\boldsymbol{\pi}(i,:)_{i=1,..,n}$ dans l'espace $[0,1]^p$ (voir figure A.1).
- Étape 2. Puis on génère une matrice U même taille que $\boldsymbol{\pi}$, dont chaque élément $U(i,j)$ est une valeur aléatoire de distribution uniforme sur $[0,1]$. Un LHS à

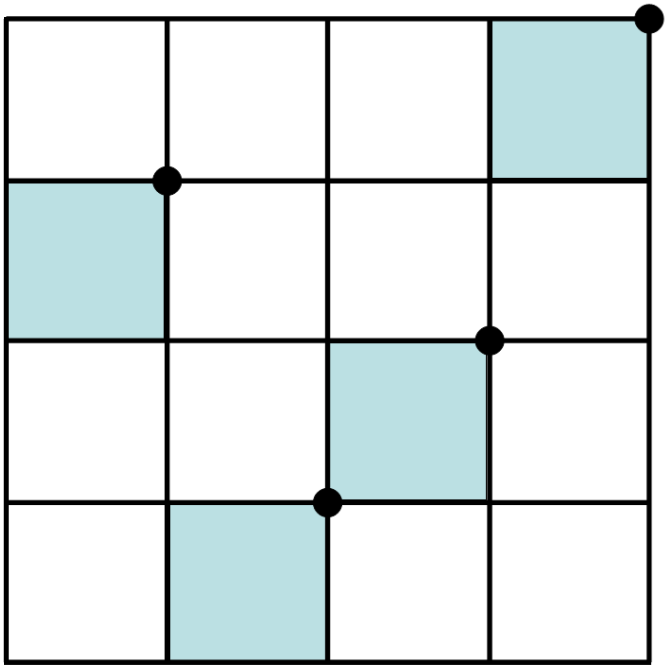

FIGURE A.1 – Les cellules sélectionnées dans l'espace $[0,1]^2$.

n point sur $[0,1]^p$, est défini par l'ensemble de points X^i tel que :

$$X^i = \frac{\boldsymbol{\pi}(i,:)^T + U(i,:)^T}{n},\ i = 1,..,n, \tag{A.1}$$

où on note M^T la transposée de la matrice M.
La figure A.2 présente un plan LHS de quatre points dans l'espace $[0,1]^p$, chaque point est sélectionné au hasard dans une cellule.

Remarques :

Les points de plan LHS possèdent la propriété intéressante d'être uniformément distribués sur les axes factoriels. Cependant, ceci n'assure pas l'uniformité des points sur tout le domaine, par exemple, dans le cas où les points sont placés sur l'une des diagonales du domaine. Si une fonction à approcher varie fortement dans l'autre diagonale, alors l'information donnée par ce plan d'expériences se réduit à un point au lieu de n points.

Pour pallier ce problème, plusieurs variantes de la méthode LHS ont été proposées dans la littérature. On peut chercher par exemple à améliorer le critère de corrélation. L'objectif est alors de construire un hypercube latin orthogonal pour les effets principaux, voire plus si possible. On peut citer Kenny (1998), Owen (1994). Une autre méthode basée sur les critères de distance, proposée par Johnson et al.

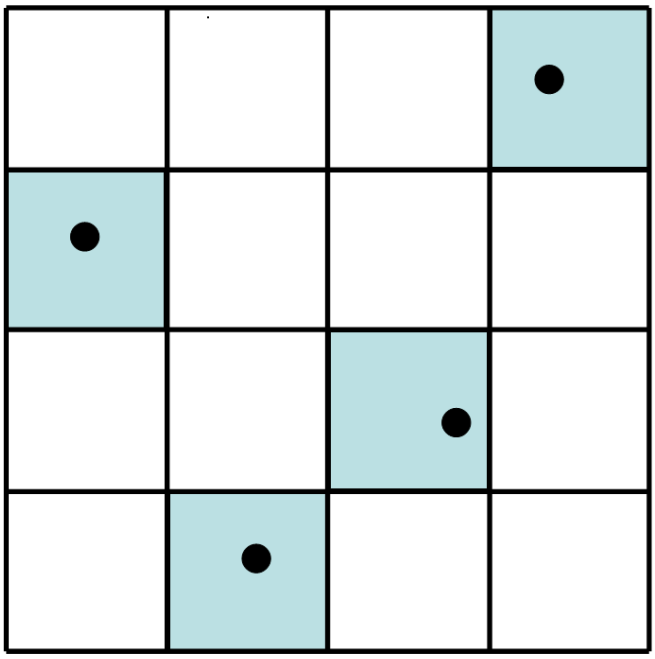

FIGURE A.2 – Plan LHS de quatre points dans l'espace $[0,1]^2$.

(1990) est largement utilisée. Elle est implémantée dans *Matlab* avec la commande *lhsdesign* en appliquant le critère *maximin*. Le principe est de maximiser la distance maximale entre les points de plan d'expériences.

Annexe B

Code de l'algorithme POD-ISAT

B.1 Algorithme POD-ISAT

```
% La routine principale de POD-ISAT
% Entrée: paramètre y0
% Sortie: solution y, scénario (addition ou récupération),
   identité d'EOA concerné
function [y,scenario,ell_number] = isatpod(y0)

%% POD-ISAT Global Variables
global records n_records pathdata path poissonA0 poissonA1 ;
global errors nexpi KPOD Nexp  newdata;
%% l'espace de paramètres
bord_param=load([ path '\bord_param.txt'],'w');
a=bord_param(:,1);
b=bord_param(:,2);
y0_red=(y0-(b+a)/2)./((b-a)/2);
n = size(y0,1);

%% Plot the query point
hold on;
figure(1)
scatter(y0_red(v1),y0_red(v2),20,'r','c','filled'); grid on;

title('The query point');
%initial of nearest point
filedata=nexpi;
%% Initialize POD-ISAT variables the first time through
if isempty(n_records),
```

```
    nexpi=1;
    isatpod_initialize(y0);
    figure(1)
    scatter3(y0_red(v1),y0_red(v2),30,'g','s','filled');
    title('Initialize ISAT');
    scenario=2; %addition
y = load([ pathdata '\' num2str(nexpi) '\Tp.txt' ],'w') ;
ell_number=0;
else
%% Recherche complète et optimal pour erreur mais non optimal
    pour le temps de recherche. RECOMANDER DE RECODER
isat_retrieval = 'n';
cri=1000;
for i=1:n_records
    nexpi=records(i).numberofEXP;
    load([pathdata '\' num2str(nexpi) '\ell_red.mat']);
    [vcent mshap] = parameters(E_eoa);
    if ((y0_red'-vcent')*mshap^-1*(y0_red-vcent))<cri,
        cri= (y0_red'-vcent')*mshap^-1*(y0_red-vcent);
        i_leaf=i;
    end
end
ell_number=0;
if cri<1
    isat_retrieval = 'y';
    ell_number=i_leaf;
end
%% Chercher les points les plus proches de point thetai
i_branch = isatpod_bt_search(y0);


%% Scenario 1: 'récupération'
if isat_retrieval == 'y',
    % Identifer la base donée 'la plus proche'
    nexpi=records(i_leaf).numberofEXP;
    % Calculer les coefficients POD
    coefak=akrom(y0);
    % Importer les mode POD locaux, solution EF, et Tlift de
       point nexpi
    psik=load([pathdata '\' num2str(nexpi) '\POD.txt'],'w');
    tpi = load([pathdata '\' num2str(nexpi) '\Tp.txt'],'w');
```

```
65      tppoissoni = load([pathdata '\' num2str(nexpi) '\poissontp
            .txt'],'w');
66      % Calculer Tlift pour y0
67      tppoisson=y0(2)*poissonA1+poissonA0;
68      % Calculer la solution approchée par POD-ISAT
69      tptilde = tpi-tppoissoni+tppoisson;
70       for  i=1:KPOD
71           tptilde=tptilde+coefak(i)*psik(:,i);
72       end
73       y=tptilde;
74       scenario=1;
75       figure(1); scatter(y0_red(v1),y0_red(v2),30,ell_number,'
            filled');
76       title('ISAT retrieval');
77         return;
78  else
79
80       disp('Rajoute d un nouveau enregistrement est en cours...
            ');
81
82  %% Scenario 2 : 'Addition'
83  %Calculer pour nouveau enregistrement la solution EF
84  newdata=text(y0_red(v1),y0_red(v2),...
85      [' newdata'] );
86  nexpi=filedata;
87  isatpod_integrate(y0); %lancer un calcul en EF
88  %Construire le modèle réduit local
89  ROM_ak; % routine importante
90  % Ajouter "leaf" à "b_tree" et nexpi à "records"
91  isatpod_add(nexpi,i_branch(1));
92  % Solution exacte est exporté
93  scenario = 2;
94  y = load([pathdata '\' num2str(nexpi) '\Tp.txt' ]) ;
95  % Turn the query point into a green square
96  figure(1);scatter(y0_red(v1),y0_red(v2),30,'g','s','filled');
97  title('ISAT addition');
98  end
99  end
```

B.2 Construction d'un modèle local

```
% Nom du fichier: ROM_ak.m
% Determiner le ROM de ak en optimisant le residu

function [] = ROM_ak %nak :=nombre des points où les ak sont
   calculés par l'optimisation de résidu
%%
global p Nlocal KPOD nexpi path pathdata nak  atol;
%%
%calcul POD local
modPOD(nexpi,Nlocal)
cd (path)
%
fid = fopen('nexpi.txt','w');
fprintf(fid, '%0.f \n', nexpi);
fclose(fid);

%% Approximation initiale de l'EOA.

parami=load([pathdata '\' num2str(nexpi) '\param.txt']);
dis=load([pathdata '\' num2str(nexpi) '\ptsvoisins.txt']);

bord_param=load([ path '\bord_param.txt'],'w');
a=bord_param(:,1);
b=bord_param(:,2);

% Déterminer les dicrections aléatoires.
rmax=2*min(dis(:,2));
nbpb=p*(p+1)/2+2+2+3;%nbpb=nombre des point au bord
paramred=-1+2*rand(p,nbpb);
paramred(:,1:p)=eye(p,p);
paramred(:,p+1:p+p)=-1*eye(p,p);
for i=1:nbpb
    paramred(:,i)=rmax*paramred(:,i)/norm(paramred(:,i));
end
param=paramred.*repmat(b-a,1,nbpb)+repmat(parami,1,nbpb);

paramred=(param-repmat((b+a)/2,1,nbpb))./repmat((b-a)/2,1,
   nbpb);
```

```
paramired=(parami-(b+a)/2)./((b-a)/2);

% Chercher des paramètres au bord de EOA initial
lpath=path;
lpathdata=pathdata;
latol=atol;
lnexpi=nexpi;
fac=1.;%facteur de sécurrite
parfor opt=1:nbpb
    deltatheta=param(:,opt)-parami;
    alpha=2.;
    res_atol=resthetareal(alpha,parami,deltatheta,opt,lpath,
       lpathdata,latol,lnexpi);
    res_atol
    if res_atol>0
        [alpha, res_atol] = fminbnd(@(alpha) restheta(alpha,
           parami,deltatheta,opt,lpath,lpathdata,latol,lnexpi
           ),0.0001,1.);
    end
    paramb(:,opt)=fac*alpha*deltatheta+parami;
end

parambred=(paramb-repmat((b+a)/2,1,nbpb))./repmat((b-a)/2,1,
   nbpb);

%Construction de EOA
%trier les point de bord plus proche
for i=1:nbpb %index
    dist(i,1)=i;
    dist(i,2)=sqrt((parambred(:,i)-paramired)'*(parambred(:,i
       )-paramired));
end
for i=1:nbpb %trier
    for j=(i+1):size(dist,1)
        if dist(j,2)<dist(i,2)
            temps=dist(i,:);
            dist(i,:)=dist(j,:);
            dist(j,:)=temps;
        end
    end
end
```

```
73
74  hold on
75  method='method1';
76  switch method
77      case 'method1'%elargir ellipse
78          % initialiser ellipse pD (tres petite) à visualiser
                dans l'espace reduit;
79          v1=1;v2=2;v3=3;
80          diaD=dist(1,2)^-2*ones(p,1) %for first ellipse which
                is minimum
81          D=diag(diaD)
82          new_eoa=D;
83          eoa=new_eoa;
84          ae = eoa(v1,v1);
85          be = eoa(v2,v2);
86          ce = 2*eoa(v1,v2);
87          E_eoa=ellipsoid(paramired,eoa^-1);
88
89          hold on;
90          fac=nbpb-16;%facteur de securite nbpb-16
91          for i=2:(nbpb-fac)
92              delta=parambred(:,dist(i,1))-paramired;
93              new_eoa = isatpod_growth(delta,new_eoa);
94              eoa =  new_eoa;
95              E_eoa=ellipsoid(paramired,eoa^-1);
96          end
97          [vcentred new_eoared]=parameters(E_eoa);
98          invnew_eoared=new_eoared^-1;
99
100     case 'method2' %'method2'construit une ellipse minimal
            qui contient les bord (il faut installer sedumi
            package)
101         E_eoa=ell_enclose(parambred);
102         [vcentred new_eoared]=parameters(E_eoa);
103         invnew_eoared=new_eoared^-1;%ATTENTION SHAPE MATRIX
104         %modify ellipsoid
105         for i=1:nbpb
106             if (parambred(:,i)'-vcentred')*invnew_eoared*(
                    parambred(:,i)-vcentred)<1
107                 invnew_eoared = isatpod_growth((parambred(:,i
                        )-vcentred),invnew_eoared);
```

```
108             end
109         end
110         E_eoa=ellipsoid(vcentred,invnew_eoared^-1);
111 end
112 %sauvegarder ellipse dans l'espace reduit
113 save([pathdata '\' num2str(nexpi) '\ell_red.mat'],'E_eoa');
114
115 %% 1.Generer alératoirement les points plus proche de point i
    , (profiter les
116 % points calculés audessus est recomandé).
117
118 %calcul bord de projection de l'ellipse sur un axe
119 Cor= eye(p);
120 for i=1:p
121     [cen shap] = parameters(Cor(:,i)'*E_eoa);
122     ebord(1,i)=cen-sqrt(shap); %borde min
123     ebord(2,i)=cen+sqrt(shap);%borde max
124 end
125 clear cen; clear shap;
126
127 %generer alétoirement
128 param_akred=ones(p,nak);
129 k=1;
130 while k <= nak,
131     param_akred(:,k) = ebord(1,:)'+(ebord(2,:)'-ebord(1,:)')
        .*rand(p,1);
132     if (param_akred(:,k)'-vcentred')*invnew_eoared*(
        param_akred(:,k)-vcentred)<1
133         k=k+1;
134     end
135 end
136 %  ell_plot(param_akred,'b*')
137
138 % param en vrai échelle
139 param_ak = param_akred.*repmat((b-a)/2,1,nak)+repmat((b+a)
    /2,1,nak);
140
141
142 %% 2.Optimiser les résidus -> obtenir les ak, residu_opt,
    cette etape on
```

```
% utilise calcul paralelle. Permet de calcul les ak plus vite possible
fvaltol=atol;
parfor nopt = 1:nak
    path1=[ pathdata '\' num2str(nexpi) '\Opt\' num2str(nopt) ] ;
    mkdir(path1);
    cd (path1);
    source=[ path '\residu_ak_opt.edp' ];
    copyfile(source,path1)
    %     source= [ path '\RunPoisson.edp'];
    %     copyfile(source,path1)
    source= [ path '\dlump.txt'];
    copyfile(source,path1)
    cd (path1);

    %.
    fid = fopen('param.txt','w');
    fprintf(fid, '%1.20e \n', param_ak(:,nopt)');
    fclose(fid);
    %pb poisson
    %     ! Freefem++ -nw -nowait RunPoisson.edp >> test.txt
    %calcul residuopt et akopt
    ! Freefem++ -nw -nowait residu_ak_opt.edp >> test.txt
    %input solution
    akexp(:,nopt)=load('ak.txt','w');
    fval(nopt)=load('resopt.txt','w');

    delete residu_ak_opt.edp;
    %     delete RunPoisson.edp;
    delete dlump.txt;

end
matlabpool close

%% modifier EOA
switch method
    case 'method1'
        for i=1:nak

```

```
181             %r      if fval(i)/res0rom(param_ak(:,i),path,
                   pathdata)> atol;
182             if fval(i)/res0rom(parami,path,pathdata)> atol;
183                 % if fval(i)> atol;
184                 delta=param_akred(:,i)-paramired;
185                 if (delta'*new_eoa*delta<1)
186                     new_eoa = isatpod_shrink(new_eoa,
                           param_akred(:,i),paramired,1)
187                     eoa =  new_eoa;
188                     E_eoa=ellipsoid(paramired,eoa^-1);
189                     %               plot(E_eoa);
190                     %               pause(2)
191                 end
192             end
193         end
194     case 'methode2'
195         n_bon=0;
196         for i=1:nak
197             if fval(i)/res0rom(param_ak(:,i),path,pathdata)<=
                    atol;
198                 n_bon=n_bon+1;
199                 parambon(:,n_bon)=param_ak(:,i);
200                 parambonred(:,n_bon)=param_akred(:,i);
201             end
202         end
203         E_eoa=ell_enclose(parambonred);
204         [vcentred new_eoared]=parameters(E_eoa);
205         invnew_eoared=new_eoared^-1;
206 end
207
208 save([pathdata '\' num2str(nexpi) '\ell_red.mat'],'E_eoa');
209
210 %% 3.Construire ROM de ak pour point i, modele krigea
211 NbBA = nak-5+1;
212 NbBT =5;
213 %
214 akoptapp(:,1:(NbBA-1))=akexp(:,1:(NbBA-1),1); akoptapp(:,NbBA
       )=0;
215 akopttest=akexp(:,NbBA:nak,1);
216
```

```
paramapp(:,1:(NbBA-1))=param_ak(:,1:(NbBA-1),1); paramapp(:,
    NbBA)=parami;
paramtest=param_ak(:,NbBA:nak,1);

bornesak=[min([akoptapp';akopttest']); max([akoptapp';
    akopttest'])];
bornesParam = [min([paramapp';paramtest']); max([paramapp';
    paramtest'])];

milieuak=(bornesak(1,:)+bornesak(2,:))/2;
largeurak=(bornesak(2,:)-bornesak(1,:))/2;

milieuParam = (bornesParam(1,:) + bornesParam(2,:))/2 ;
largeurParam = (bornesParam(2,:) - bornesParam(1,:))/2 ;

akappred=(akoptapp-repmat(milieuak',1,NbBA))./repmat(
    largeurak',1,NbBA);
aktestred=(akopttest-repmat(milieuak',1,NbBT))./repmat(
    largeurak',1,NbBT);
paramappRed = (paramapp - repmat(milieuParam',1,NbBA))./
    repmat(largeurParam',1,NbBA);
paramtestRed = (paramtest - repmat(milieuParam',1,NbBT))./
    repmat(largeurParam',1,NbBT);
%
akoptapp=akappred;
akopttest=aktestred;
%save borne param et ak
blocalparam=[milieuParam ; largeurParam];
path1=[pathdata '\' num2str(nexpi) '\blocalparam.txt']
fid = fopen(path1,'w');
fprintf(fid,'%12.8f  %12.8f\r\n',blocalparam);
fclose(fid);

blocalak=[milieuak; largeurak];
path1=[pathdata '\' num2str(nexpi) '\blocalak.txt']
fid = fopen(path1,'w');
fprintf(fid,'%12.8f  %12.8f\r\n',blocalak);
fclose(fid);

%Krigea
BA.inp = paramappRed;
```

```
251 BT.inp = paramtestRed;
252 % global p1           % Parametre de largeur de gaussienne
    pour SVR
253 lob = 0.01*ones(p,1);
254 upb = 5*ones(p,1);
255 %
256 Krig = cell(KPOD,1);
257 MSEminKrig = 100*ones(KPOD,1);
258 % Boucle sur les modes
259 for kmode=1:KPOD
260     BA.tar = akoptapp(kmode,:);
261     BT.tar = akopttest(kmode,:);
262     for essai = 1:6
263         theta = 1*ones(p,1);
264         switch (essai)
265             case 1
266                 [modelKrig_tmp, perf] = dacefit(BA.inp',BA.
                    tar',@regpoly0,@corrgauss,theta,lob,upb);
267             case 2
268                 [modelKrig_tmp, perf] = dacefit(BA.inp',BA.
                    tar',@regpoly1,@corrgauss,theta,lob,upb);
269             case 3
270                 [modelKrig_tmp, perf] = dacefit(BA.inp',BA.
                    tar',@regpoly2,@corrgauss,theta,lob,upb);
271             case 4
272                 [modelKrig_tmp, perf] = dacefit(BA.inp',BA.
                    tar',@regpoly0,@correxp,theta,lob,upb);
273             case 5
274                 [modelKrig_tmp, perf] = dacefit(BA.inp',BA.
                    tar',@regpoly1,@correxp,theta,lob,upb);
275             case 6
276                 [modelKrig_tmp, perf] = dacefit(BA.inp',BA.
                    tar',@regpoly2,@correxp,theta,lob,upb);
277         end
278         outKrig_tmp = predictor(BT.inp',modelKrig_tmp);
279         MSE_tmp = sqrt(mse(outKrig_tmp - BT.tar'));
280         if MSE_tmp < MSEminKrig(kmode)
281             Krig{kmode} = modelKrig_tmp;
282             MSEminKrig(kmode) = MSE_tmp;
283         end
284     end
```

```
285 end
286 path1=[ pathdata '\' num2str(nexpi) '\KrigeaROMak.mat'] ;
287 save(path1,'Krig');
```

Annexe C

Code de l'algorithme pararéel et Broyden-pararéel

C.1 Système brusselator

C.1.1 Fonction brusselator utilisant méthode Runge-Kutta (brus.m)

```
%% Résolution du système Brusselator par méthode Runge-Kutta
function uplus = brus(u,A,B,dt)
F = @(u, A, B) [A + (u(1,:).^2) .* u(2,:) - (B+1) .* u(1,:);
   B*u(1,:) - (u(1,:).^2) .* u(2,:)];
    k1 = F(u, A,B);
    k2 = F(u + 0.5*dt*k1, A,B);
    k3 = F(u + 0.5*dt*k2, A,B);
    k4 = F(u +     dt*k3, A,B);
    uplus = u + dt * (k1 + 2*k2 + 2*k3 + k4)/6;
```

C.1.2 Système brusselator résolu par l'algorithme pararéel

```
% Parareal Brusselator
clear all;
%% Parametre Brusselator;
A = 1;
B = 3;
T = 12;
N = 32; % Nombre de sous-domaine temporel $\Delta T$
```

```
M = 20; % Nombre de pas de temps sur chaque sous-domaine
DeltaT = T / N;
deltat = DeltaT / M;
u0 = [0; 1];

%% Calul par solveur fin
fukn(:,1) = u0 ;
t=zeros(N*M+1,1);
t(1)=0;
for n=2:(N*M+1)
    fukn(:,n) = brus(fukn(:,n-1),A,B,deltat);
end; % for n
fukng=zeros(2,N+1);
for n=1:(N+1)
    fukng(:,n)=fukn(:,(n-1)*M+1);
end
%% Calcul par l'Algorithme Parareal
%===Pour k=0
%calcul séquentiel par solveur grossier
ukn      = zeros(2, N+1);
gk_nprec = zeros(2,N+1);
ukn(:,1)=u0;
for n=2:N+1
    ukn(:,n)=brus(ukn(:,n-1),A,B,DeltaT);
end
gk_nprec=ukn;

for n=1:N+1
    e(:,n)=abs(ukn(:,n)- fukn(:,(n-1)*M+1)) ;
end;
normL2=sqrt(e(1,:).^2+e(2,:).^2);
norme(1)=max(normL2);
figure(2);
semilogy(e(1,:),'b');

hold on; grid on;
semilogy(e(2,:),'r');

%===Pour k>=1
%définir données sparse
for n=1:N
```

```
49      for j=1:M+1
50          ujnp(n).(['iterj' num2str(j)])=zeros(2,1);
51      end
52  end
53  ujnpnew=ujnp;
54
55  gknew_nprec=gk_nprec;
56  uknnew=ukn;
57  deltaG=ukn-ukn;
58
59  for k=1:5
60      for n=1:N %calcul parallel
61          ujnpnew(n).(['iterj' num2str(1)])=ukn(:,n);
62          for j=2:M+1
63              ujnpnew(n).(['iterj' num2str(j)]) = brus(ujnpnew(
                    n).(['iterj' num2str(j-1)]),A,B,deltat);
64          end;
65      end; % for n
66
67      for n=2:N+1 %correction séquentielle
68          gknew_nprec(:,n) = brus(ukn(:,n-1),A,B,DeltaT);
69          deltaG(:,n)=gknew_nprec(:,n) - gk_nprec(:,n);
70          gk_nprec(:,n)=gknew_nprec(:,n);
71          % Parareal :
72          uknnew(:,n) = ujnpnew(n-1).(['iterj' num2str(M+1)])
                + deltaG(:,n);
73          %      deltau(:,n)=uknnew(:,n)-ukn(:,n);
74          ukn(:,n)=uknnew(:,n);
75      end;
76
77      for n=1:N+1
78          e(:,n)=abs(ukn(:,n)- fukn(:,(n-1)*M+1));
79      end;
80      normL2=sqrt(e(1,:).^2+e(2,:).^2);
81      norme(k+1)=max(normL2);
82      figure(2)
83      semilogy(e(1,:),'b');
84      semilogy(e(2,:),'r');
85  end;
```

C.1.3 Système brusselator résolu par Broyden-Pararéel

```
%Brusselator  Broden-parareal version Multiplicative
%%
clear all;
% Parametre Brusselator Gander;
%
A = 1;
B = 3;
T = 12;
N = 32;
M = 20;
DeltaT = T / N;
deltat = DeltaT / M;
u0 = [0; 1];
%%%%%%%%calul fin
fukn(:,1) = u0 ;
t=zeros(N*M+1,1);
t(1)=0;
for n=2:(N*M+1)
    fukn(:,n) = brus(fukn(:,n-1),A,B,deltat);
end; % for n
%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%Broyden-Parareal%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
%%%%%%k=0
%calcul séquentielle
ukn      = zeros(2, N+1);
gk_nprec = zeros(2,N+1);
ukn(:,1)=u0; %les permiers Seeds
for n=2:N+1
    ukn(:,n)=brus(ukn(:,n-1),A,B,DeltaT);
end
gk_nprec=ukn;

for n=1:N+1
    e(:,n)=abs(ukn(:,n)- fukn(:,(n-1)*M+1)) ;
end;
    normL2=sqrt(e(1,:).^2+e(2,:).^2);
```

```
40      norme(1)=max(normL2);
41  figure(3);
42  semilogy(e(1,:),'b');
43  hold on; grid on;
44  %%
45  %%%%%%k=1
46  % Calcul en parallele (emulation) des solutions fines
47  for n=1:N
48      for j=1:M+1
49          ujnp(n).(['iterj' num2str(j)])=zeros(2,1); %définir
                données sparse
50      end
51  end
52  ujnpnew=ujnp;
53
54  for n=1:N
55      ujnp(n).(['iterj' num2str(1)])=ukn(:,n);
56      for j=2:M+1
57          ujnp(n).(['iterj' num2str(j)]) =brus(ujnp(n).(['iterj
                ' num2str(j-1)]),A,B,deltat);
58      end
59  end
60
61  gknew_nprec=gk_nprec;
62  uknnew=ukn;
63  deltaG=ukn-ukn;
64  deltau=ukn-ukn;
65  %%%correction
66  for n=2:N+1
67      gknew_nprec(:,n) = brus(ukn(:,n-1),A,B,DeltaT);
68      deltaG(:,n)=gknew_nprec(:,n) - gk_nprec(:,n);
69      gk_nprec(:,n)=gknew_nprec(:,n);
70      % Parareal formule
71      uknnew(:,n) = ujnp(n-1).(['iterj' num2str(M+1)])  +
            deltaG(:,n);
72      deltau(:,n) = uknnew(:,n)-ukn(:,n);
73      ukn(:,n)=uknnew(:,n);
74  end;%
75
76  for n=1:N+1
77      e(:,n)=abs(ukn(:,n)- fukn(:,(n-1)*M+1));
```

```
78  end
79  normL2=sqrt(e(1,:).^2+e(2,:).^2);
80  norme(2)=max(normL2);
81  figure(3);
82
83  semilogy(e(1,:),'b');
84  hold on; grid on;
85
86  %%%% Pour k>1
87  for n=1:N+1
88  Bkn(:,:,n)=eye(2,2);
89  end
90  for k=2:5
91      for n=1:N %calcul parallel
92          ujnpnew(n).(['iterj' num2str(1)])=ukn(:,n);
93          for j=2:M+1
94              ujnpnew(n).(['iterj' num2str(j)]) = brus(ujnpnew(
                    n).(['iterj' num2str(j-1)]),A,B,deltat);
95          end;
96      end;
97
98      for n=2:(N+1)
99          deltaF(:,n)=ujnpnew(n-1).(['iterj' num2str(M+1)])-
                ujnp(n-1).(['iterj' num2str(M+1)]);
100         Bkn(:,:,n)=Bkn(:,:,n)+max(1e-10,norm(deltaG(:,n)))
                ^-2*(deltaF(:,n)-Bkn(:,:,n)*deltaG(:,n))*deltaG(:,
                n)';
101     end
102     for n=2:N+1 %correction séquentielle
103         gknew_nprec(:,n) = brus(ukn(:,n-1),A,B,DeltaT);
104         deltaG(:,n)=gknew_nprec(:,n) - gk_nprec(:,n);
105         gk_nprec(:,n)=gknew_nprec(:,n);
106         % broyden version multiplicative ;
107         uknnew(:,n) = ujnpnew(n-1).(['iterj' num2str(M+1)])
                + Bkn(:,:,n)*deltaG(:,n);
108         deltau(:,n)=uknnew(:,n)-ukn(:,n);
109         ukn(:,n)=uknnew(:,n);
110     end
111     ujnp=ujnpnew;%update fine solution
112     for n=1:N+1
113         e(:,n)=abs(ukn(:,n)- fukn(:,(n-1)*M+1));
```

```
114     end
115     normL2=sqrt(e(1,:).^2+e(2,:).^2);
116     norme(k+1)=max(normL2);
117     figure(3)
118     semilogy(e(1,:),'b');
119 end; %for k
120 figure();
121 plot(norme,'r')
```

C.2 Diffusion de CO_2 résolue par l'algorithme Broyden-pararéel

```
1  //***Input Maillage
2  int outletbr = 1;
3  int wall     = 2;
4  int air4     = 3;
5  int air3     = 4;
6  int outlettr = 5;
7  int inlettl  = 6;
8  int air2     = 7;
9  int air1     = 8;
10 int axe      = 9;
11 int inletbl  = 10;
12 int seat1    = 11;
13 int seat2    = 12;
14 int seat3    = 13;
15 int seat4    = 14;
16 int inletbr  = 15;
17 int wallf    = 16;
18 int floor    = 17;
19 mesh Th = readmesh("TriangulationTh.msh");
20 //*** Espace éléments finis
21 fespace Vh(Th, P2); //Difinition de l'espace de fonction
      éléments finis d'ordre 2
22 fespace Wh(Th, P1); //Difinition de l'espace de fonction
      éléments finis d'ordre 1
23 real dofs=Wh.ndof;
24 Vh u, v;
```

```
Wh c, cold, c0;
Wh C, Cold, C0;
c0=100.;  C0=c0;

{ifstream f("u.txt");
        for (int i=0; i<Vh.ndof; i++) {f >> u[](i);}
}

{ifstream f("v.txt");
        for (int i=0; i<Vh.ndof; i++) f >> v[](i);
}

//***Paramètres
int Nbpt=mpisize; //nombre de grand pas de temps = nombre de
   processeurs
int nbpt=60; //nombre de petit pas de temps pour un domain
int NBPT= nbpt*Nbpt ; //nombre de petit pas de temps pour
   toute domaine [0- T]
real dt=0.01;
real dT=dt*nbpt;

//*** Regions au niveau de la tete de passagers où CO_2
   apparait
real rad2=0.075/2.;
real xc2=1.595,yc2=1.15;  //head

real rad1=0.075/2.;
real xc1=0.545,yc1=1.15;  //head

real rad3=0.075/2.;
real xc3=2.115,yc3=1.15;  //head

func s1=rad1^-1*sqrt((x-xc1)^2+(y-yc1)^2);//source de CO2
func s2=rad1^-1*sqrt((x-xc2)^2+(y-yc2)^2);//source de CO2
func s3=rad1^-1*sqrt((x-xc3)^2+(y-yc3)^2);//source de CO2

func w1=1.*(s1<1.);//fonction indicatrice
func w2=1.*(s2<1.);//fonction indicatrice
func w3=1.*(s3<1.);//fonction indicatrice

//***Fonction Fin pour un pas de temps dt
```

```
63  func real[int] F(real[int] & c_in, real t)
64  {
65  cold[]=cin;
66  solve density (c, ch) =
67          int2d(Th)(c*ch/dt)
68          -int2d(Th)(convect([u, v], -dt, cold)*ch/dt)
69          +int2d(Th)(dif*dx(c)*dx(ch)+dif*dy(c)*dy(ch))
70          -int2d(Th)(c0*(2+sin(t)+sin(2*t))*w1*ch+c0*(2+sin(t)+
               sin(2*t))*w2*ch+c0*(2+sin(t)+sin(2*t))*w3*ch)
71          +on(outletbr,c=0.)
72          +on(air1, air2, air3,air4,c=0.)
73          +on(inletbl, inletbr,c=0.);
74  return c[];
75  }
76
77  //***Fonction Grossier  pour un pas de temps dT
78  func real[int] G(real[int] & C_in, real T)
79  {
80  Cold[]=Cin;
81  solve Density (C, Ch) =
82          int2d(Th)(C*Ch/dT)
83          -int2d(Th)(convect([u, v], -dT, Cold)*Ch/dT)
84          +int2d(Th)(dif*dx(C)*dx(Ch)+dif*dy(C)*dy(Ch))
85          -int2d(Th)(C0*(2+sin(T)+sin(2*T))*w1*Ch+C0*(2+sin(T)+
               sin(2*T))*w2*Ch+C0*(2+sin(T)+sin(2*T))*w3*Ch)
86          +on(outletbr,C=0.)
87          +on(air1, air2, air3,air4,C=0.)
88          +on(inletbl, inletbr,C=0.);
89          return C[];
90  }
91  //RESUME Broyden+parareal
92  /*
93  %%1 Iter 0:
94      %séquentielle G -> seed U^0_1---U^0_2--- --U^0_N.
95  %%2 Iter 1:
96      %Calcul parallel: F(U^0_n)
97      %Calcul B^0_n = 0
98      %Calcul séquentiel: U^1_(n+1)=F(U^0_n)+ [G(U^1_n)-G(U^0_n
           )]
99  %%3 Iter k+1 (grand itération)
100     %Calcul parallel: F(U^k_n)
```

```
101     %Calcul parallel: B^k_n = B^k_n + C^k_n
102     %Calcul sequentiel: U^(k+1)_(n+1)=F(U^k_n)+ [G(U^{k+1}_n)
           -G(U^k_n)]*B^k_n
103
104 */
105 //***Shema de Broyden-Parareal
106 /*
107 Input: G, F, U0=u0;
108 Output: U_parareal
109 */
110
111 //Définition des vecteurs
112
113 real[int,int] Uk(dofs,Nbpt+1),Ukplus(dofs,Nbpt+1),Gk(dofs,
       Nbpt),Gkplus(dofs,Nbpt); //seed & fonction grossier
114 real[int,int] ukn(dofs,(nbpt+1)*Nbpt),uknplus(dofs,(nbpt+1)*
       Nbpt);//solution fine
115 real[int,int] uknsub(dofs,(nbpt+1)); //solution provisoire
       pour un sous-domain temporaire
116
117 real[int,int] dG(dofs,Nbpt),dU(dofs,Nbpt+1),dGplus(dofs,Nbpt)
       ;
118 real[int,int] vk(dofs,Nbpt),wk(dofs,Nbpt),vkk(dofs,Kbig*Nbpt)
       ,wkk(dofs,Kbig*Nbpt);
119 real[int] vksub(dofs),wksub(dofs);
120
121 real[int] BdG(dofs);
122 real invNdu,const;
123 real[int] resv(dofs);
124
125 //Iteration k=0
126 //===========
127 real[int] cdinit(dofs); //condition initiale
128 cdinit=0.25*c0[];
129 Uk(:,0)  = cdinit;
130 ukn(:,0) = cdinit;
131 //Calcul séquentiel
132 for (int i=0;i<Nbpt;i++)
133 {
134 Gk(:,i)=G(Uk(:,i),T);
135 Uk(:,i+1)=Gk(:,i);
```

```
136 }
137 if (mpirank==0)
138 {ofstream f("Soluapproche"+0+".txt");
139 for (int i=0;i<dofs;i++){
140 for (int j=0;j<(Nbpt+1);j++) f<<Uk(i,j)<<" ";
141 f<<endl;}
142 //Iteration k=1;
143 //==============
144 //Caclcul parallel
145 uknsub(:,0)=Uk(:,mpirank);//initiale d'un sous-domaine
146 for (int j=0;j<nbpt;j++)
147         {
148         uknsub(:,j+1)=F(uknsub(:,j),t);
149         }
150 mpiAllgather(uknsub,ukn); //Rassembler tout les solution de
       sous-domain dan une matrice golobale
151
152 //Calcul séquentiel
153 Ukplus(:,0)   = Uk(:,0);
154 uknplus(:,0) = Uk(:,0);
155 dU(:,0)=0;
156 for (int i=0;i<Nbpt;i++)
157 {
158         Gkplus(:,i)    = G(Uk(:,i),T);
159         dG(:,i)        = Gkplus(:,i)-Gk(:,i);
160         Gk(:,i)        = Gkplus(:,i);
161         Ukplus(:,i+1)  = ukn(:,(i+1)*(nbpt+1)-1)+dG(:,i);
162         dU(:,i+1)      = Ukplus(:,i+1)-Uk(:,i+1);
163         Uk(:,i+1)=Ukplus(:,i+1);
164 }
165 if (mpirank==0)
166 {ofstream f("Soluapproche"+1+".txt");
167 for (int i=0;i<dofs;i++){
168 for (int j=0;j<(Nbpt+1);j++) f<<Uk(i,j)<<" ";
169 f<<endl;}
170 //Iteration k>1;
171 //==============
172 for (int k=1;k<Kbig;k++)
173 {
174 //Calcul parallel
175 uknsub(:,0)=Uk(:,mpirank);//initial d'un sous-domaine
```

```
176 for (int j=0;j<nbpt;j++)
177         {
178         uknsub(:,j+1)=F(uknsub(:,j),t);
179         }
180 //Calcul de la matrice de Broyden
181 invNdu=max(1e-10,(sqrt(dG(:,mpirank)'*dG(:,mpirank))))^-1; //
    '
182 resv=uknsub(:,nbpt)-Uk(:,mpirank+1);
183 vksub=invNdu*resv;
184 wksub= invNdu*dG(:,mpirank);
185
186 mpiAllgather(uknsub,ukn);//Rassembler tout les solution de
    sous-domain dan une matrice golobale
187 mpiAllgather(vksub,vk);
188 mpiAllgather(wksub,wk);
189
190 for (int i=0;i<Nbpt;i++)
191         {
192          vkk(:,(k-1)*Nbpt+i) = vk(:,i);
193          wkk(:,(k-1)*Nbpt+i) = wk(:,i);
194          }
195 //Calcul séquentiel
196 for (int i=0;i<Nbpt;i++)
197 {
198         Gkplus(:,i)=G(Uk(:,i),T);
199         dG(:,i)=Gkplus(:,i)-Gk(:,i); Gk(:,i)=Gkplus(:,i);
200         BdG=dG(:,i);
201                 //Broyden or not (si on supprimme 3 lignes
                    suivantes, l'algorithme devient le
                    pararéel classique)
202         for (int l=0;l<k;l++) {
203         const=dG(:,i)'*wkk(:,l*Nbpt+i); //'
204         BdG = BdG + const*vkk(:,l*Nbpt+i);
205                 }
206         //
207         Ukplus(:,i+1)=ukn(:,(i+1)*(nbpt+1)-1)+BdG;
208         dU(:,i+1)=Ukplus(:,i+1)-Uk(:,i+1);
209         Uk(:,i+1)=Ukplus(:,i+1);
210 }
211 if (mpirank==0)
212 {ofstream f("Soluapproche"+(k+1)+".txt");
```

```
213 for (int i=0;i<dofs;i++){
214 for (int j=0;j<(Nbpt+1);j++) f<<Uk(i,j)<<" ";
215 f<<endl;}
216 }
```

Annexe D

Article paru dans IJNME

INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2010; **00**:1–24
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme

POD-ISAT: an efficient POD-based surrogate approach with adaptive tabulation and fidelity regions for parametrized steady-state PDE discrete solutions

D. BUI[1,2,*], M. HAMDAOUI[1], F. De VUYST[2]

[1] *Laboratoire de Mathématiques Appliquée aux Systèmes (MAS), École Centrale de Paris, Grande voie des vignes, 92295 Châtenay-Malabry, France*

[2] *Centre de Mathématiques et de Leurs Applications (CMLA), UMR 8536, École Normale Supérieure de Cachan, 61, avenue du Président Wilson 94235 Cachan cedex, France*

SUMMARY

A combination of Proper Orthogonal Decomposition (POD) analysis and In Situ Adaptive Tabulation (ISAT) is proposed for the representation of parameter-dependent solutions of coupled partial differential equations (PDE) problems. POD is used for the low-order representation of the spatial fields and ISAT for the local representation of the solution in the design parameter space. The accuracy of the method is easily controlled by free threshold parameters that can be adjusted according to the user needs. The method is tested on a coupled fluid-thermal problem: the design of a simplified aircraft air control system. It is successfully compared to the standard POD: while the POD is inaccurate in certain areas of the design parameters space, the POD-ISAT method achieves accuracy thanks to trust regions based on residuals of the fluid-thermal problem. The presented POD-ISAT approach provides flexibility, robustness and tunable accuracy to represent solutions of parametrized PDEs. Copyright © 2010 John Wiley & Sons, Ltd.

Received . . .

KEY WORDS: Multiphysics Problems, Reduced-Order Modeling (ROM), Semi-Intrusive, In-Situ-Adaptive-Tabulation (ISAT), Proper Orthogonal Decomposition (POD), Trust-Regions, Residuals Minimization, Aircraft air control system.

1. INTRODUCTION

Computational tools are today a success factor in Engineering Design. Finite Elements or Finite Volumes codes become very efficient in the evaluation of criteria at given design points. However, the 'full' exploration of the design space is still a difficult task. Indeed, in spite of the democratization and computational power increase of high performance computing tools, the design space parameter exploration still needs an excessive amount of time. This is especially true for high dimensional design parameter spaces that suffer from the curse of dimensionality. Alternative solutions are the use of metamodels or low-order optimal bases that show both accuracy and computational efficiency for particular classes of problems (e.g. linear and elliptic problems). The earliest methods for model reduction belong to the field of structural dynamics, where the dynamic analysis of structures (eigen frequencies, frequency response functions, etc.) is of interest. In addition to the mode displacement reduction method and extensions thereof

*Correspondence to: Dung Bui, Laboratoire de Mathématiques Appliquée aux Système, École Centrale de Paris, Grande voie des vignes, 92295 Châtenay-Malabry, France.
†E-mail: dung.bui@ecp.fr, buidung84@gmail.com

[1], methods such as component mode synthesis methods [2],[3] started to emerge in the 1960s. Later on, efficient methods such as LATIN (LArge Time INcrement) methodology [4] or Hyper-reduction reduced-order modeling methods [5],[6] have been specifically developed for non-linear problems in structural dynamics. These methods relied on building low-order bases of functions to approximate the parametrized PDE solutions. Many well-known reduction methods such as Proper Orthogonal Decomposition (POD) [7],[8], Reduced Basis Method (RBM) [9],[10], Goal-oriented approaches [11],[12], Proper Generalized Decompositions (PGD) [13] rely on low-order bases and have been widely used in several fields to approximate the solutions of parametrized PDE solutions. But, in spite of the active research in the model reduction area, the capture of the evolutions of the solution in function of the parameters is still a challenging task for models relying on low-order bases, because the reduced space can require a high number of basis functions to provide accurate global approximations. A way to deal with this issue is to adapt the low-order bases to parameter changes. To tackle the problem, Amsallem *et al.* proposed in [14],[15] an interpolation scheme on manifolds and in [16] a method to build local reduced order models for non-linear systems, Bergmann *et al.* [17] recomputed low-order bases within trust regions, Dihlmann *et al.* proposed an algorithm for adaptive time domain partitioning for non-linear evolution problems and Gu *et al.* [18] proposed a method to build reduced order models by projecting on non-linear piecewise linear manifolds. The present paper deals with the design of reduced order modeling techniques that the foreseen application is a simplified aircraft air control system depending on inflow and exterior conditions. Navier-Stokes equations are coupled with a thermal equation by means of a buoyancy force (Boussinesq approximation). In this work, an innovative approach called POD-ISAT is presented. It combines Proper Orthogonal Decomposition for the representation of the spatial fields and In Situ Adaptive Tabulation (ISAT) for the local representation of the solution in the design space. This leads to a set of local reduced-order models whose fidelity is controlled by means of trust regions (TR). In Pope's ISAT model [19, 20], ellipsoids of accuracy (EOA) are used and adapted during the learning process of the table. Here we rather use a threshold criterion on fluid-thermal model residuals. The whole algorithm is detailed in the paper and numerical results show the efficiency of the approach.

The remaining sections of the paper are organized as follows. In sections 2 and 3, we describe a real application of (simplified) design of aircraft Air Control System (ACS), motivating the need of efficient numerical tools of analysis. Then we present in section 4 the methodology of Reduced Order Modeling (ROM) and refer to the state-of-the-art literature. In section 5, a new ROM method – the so-called POD-ISAT – is presented. Numerical experiments are presented in section 6. Some interesting results of residual depending on the design parameters and on the number of used POD basis functions are analysed. The accuracy and efficiency (speedup efficiency) of POD-ISAT are shown. We compare also the accuracy of this method and the POD. Concluding remarks and perspectives are discussed in the last section.

2. NEEDS OF HIGH-PERFORMANCE DESIGN ANALYSIS TOOLS: THE EXAMPLE OF AIR AIRCRAFT CABIN COMFORT ANALYSIS

Aircraft cabin comfort is today a commercial key factor in aircraft design. The airline industry suppliers of aircraft have made great efforts to satisfy passengers needs, as cabin comfort plays an important role in airline ticket sales: improving aircraft comfort attracts more passengers. Moreover, it has been reported that international air travel include potential risks associated with airborne disease transmission [21, 22]. On a flight from Hong Kong to Beijing on the 15^{th} of March 2003, 22 passengers out of 120 were infected with the Severe Acute Respiratory Syndrome (SARS) [23]. Therefore, it is essential to pay attention to the quality of the air of the cabin to ensure that it is comfortable for all passengers. Airplanes are equipped with environmental control systems (ECSs) that provide suitable indoor environments. The ECS consists of a complex set of air treatment and heat exchange systems, sensors and controllers and is part of the aircraft's electric, pneumatic and hydraulic systems. Its behaviour significantly depends on the airflow and heat exchange in the aircraft cabin.

DOI: 10.1002/nme

An important element of passenger satisfaction is thermal comfort. The major factors involved in thermal comfort are the air temperature, relative humidity, air refreshment rate, heat radiation and clothing. The research about this problem is initiated by Franger [24], and then improved by Kok *et al.* [25] in the framework of the European project FACE (Friendly Aircraft Cabin Environment). In these works, the numerical simulation of the three major phenomena that play a significant role in thermal comfort were considered: the human response to its thermal environment which is also known as thermal regulation, the actual movement of air and heat inside aircraft cabins due to natural and/or forced convection, and heat transfer due to radiation. For the first one, the thermal regulation model of Tanabe [26] is used. For the second one a CFD model based on a weakly compressible Reynolds-averaged Navier-Stokes (RANS) formulation is used.

Note that a CFD model is a system of non-linear equations that can be time consuming to solve. Moreover, the air flow in cabin depends not only on the human body thermal and heat radiation but on the air supplied by ECS. Ideally, one would like to explore the parameter design space, in order to find or choose attractive designs, or to perform optimization according to some criteria and constraints. In this case we face a high-dimensional problem where each design point requires a CFD solution. Therefore, a reduced order modeling method for parametrized CFD models is required.
A CFD model numerically solves a set of partial differential equations (PDEs) including conservation of mass, momentum, energy, and possibly chemical species convection-diffusion-reaction equations and turbulence model equations. The solutions are the field of air velocity, air temperature, the concentrations of water vapour and contaminants, and turbulence parameters in an aircraft cabin. These solutions depend on the boundary and initial conditions. Liu et al. [27] summarized all the numerical studies on the flow in an aircraft cabin published in the past two decades. The most used CFD models are RANS models and Large Eddy Simulation (LES) models. It is well known that these simulations are time consuming as they require solving for the turbulent features of the flow which means fine meshes and turbulent models to use.
Generally, aircraft cabin airflow temperature depends on many parameters which are almost boundary conditions of CFD models:

- the heat source of passenger body, of galley switched on to prepare hot meals, and of electric device;
- external environment: temperature and pressure;
- inflow temperature;
- inflow velocity;
- position and orientation of inflow air;
- fuselage thermal conductivity.

In the next section, we shall discuss a simple CFD model coupling the stationary Navier-Stokes equation and thermal diffusion equation that takes into account several design parameters.

3. MATHEMATICAL SETTING

3.1. Fluid model

We are interested in the modeling of stationary air circulation and heating conditions in an aircraft cabin. For the sake of simplicity, the flow is supposed two-dimensional and the domain of interest is the cross-section of the fuselage (see figure 1). The air is seen as an incompressible fluid but we take into account buoyancy Archimedes forces due to air dilatability by heating. So the stationary Navier-Stokes equations with the Boussinesq approximation are considered. At the right hand side of the Navier-Stokes momentum equation (2) appears a buoyancy term depending on the gravity $\boldsymbol{g}$ and the temperature deviation $(T - T_0)$ from the nominal temperature T_0 (fixed at 300 K). The Navier-Stokes equations are coupled, through this buoyancy term, with a thermal equation that governs the evolution of the temperature of the fluid (equations (1)-(3)). The coefficient κ is the

DOI: 10.1002/nme

thermal diffusivity of the air.

$$\nabla \cdot \boldsymbol{u} = 0 \quad \text{in } \Omega, \tag{1}$$

$$\boldsymbol{u} \cdot \nabla \boldsymbol{u} - \nu \Delta \boldsymbol{u} + \nabla p = \boldsymbol{g}\ (1 - \alpha(T - T_0)) \quad \text{in } \Omega, \tag{2}$$

$$\boldsymbol{u} \cdot \nabla T - \nabla \cdot (\kappa \nabla T) = 0 \text{ in } \Omega, \tag{3}$$

In realistic conditions, the reference length L is 1 m, the characteristic speed U is 1 m/s and the kinematic viscosity of the air at 300 K is 1.57×10^{-5} m^2/s so that the Reynolds number is equal to

$$Re = \frac{LU}{\nu} \approx 6.37 \times 10^4.$$

For such a Reynolds number, the flow regime is naturally turbulent. Computational mesh sizes are not relevant to capture turbulence effects with accuracy. A realistic model should include a turbulence model, e.g. Reynolds-averaged Navier Stokes (RANS) equations. Turbulent viscosities generally transform an unsteady turbulent flow into a steady flow for statistically Favre-averaged quantities. For the sake of simplicity, we do not take into account any turbulence model here.

The thermal diffusivity of air at 300 K and 1 atm is 2.22×10^{-5} m^2/s, we get a Péclet number

$$Pe = \frac{LU}{\kappa} \approx 4.52 \times 10^4.$$

meaning that the thermal convection-diffusion problem dominated by convection. Furthermore, the air thermal expansion coefficient α is $3.43 \times 10^{-3} K^{-1}$ and the maximal temperature variation ΔT is 10 K thus the Archimedes number Ar is

$$Ar = \frac{g\alpha \Delta T L^3}{\nu^2 Re} \approx 0.2,$$

which means that the flow is dominated by forced convection.

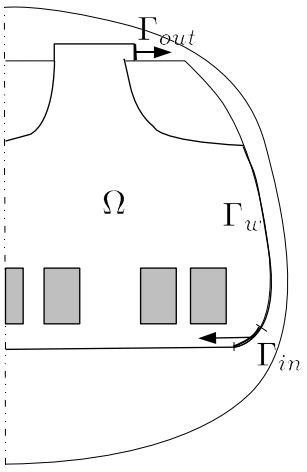

Figure 1. Spatial geometry and domain boundaries (half domain with symmetry axis)

3.2. Boundary conditions

Let us now consider the boundary conditions. The cabin boundary is denoted Γ. It is divided into three parts: the inflow boundary Γ_{in}, the outflow Γ_{out} and the wall boundary Γ_w. For the fluid, no slip boundary conditions is used on Γ_w, velocity is imposed at the inflow and constant pressure is given at the outflow:

$$\boldsymbol{u} = 0 \text{ on } \Gamma_w,\ \boldsymbol{u} = \boldsymbol{u}_{in} \text{ on } \Gamma_{in} \text{ and } p = 0 \text{ on } \Gamma_{out}. \tag{4}$$

DOI: 10.1002/nme

For thermal boundary conditions, we used Dirichlet boundary conditions on Γ_{in} with imposed inflow temperature T_{in}. The heat loss at the walls is expressed by inhomogeneous Fourier boundary conditions. The boundary heat flux may depend on the difference between the wall temperature and the exterior temperature. Finally, homogeneous Neumann boundary conditions are written at the outflow:

$$T = T_{in} \text{ on } \Gamma_{in}, \ \frac{\partial T}{\partial n} = 0 \text{ on } \Gamma_{out}, \ \kappa \frac{\partial T}{\partial n} = \Phi(T - T_{ext}) \text{ on } \Gamma_w, \tag{5}$$

where T_{ext} is the external temperature fixed at 223 K which corresponds to the nominal external temperature at a standard cruise altitude of 11000 meters. Possibly, if interior boundaries are defined (like seats for example), then homogeneous Neumann boundary conditions are imposed. The whole system is non-linear and the dominating phenomenon is the convection (because of the large Reynolds and Péclet numbers).

3.3. Residual definition

It is assumed that the domain boundaries are Lipschitz continuous, $\boldsymbol{u}_{in}, T_{in} \in H^{1/2}(\Gamma_{in})$ on Γ_{in}, $\Phi \in \mathcal{C}^\infty$, so that $(\boldsymbol{u}, p, T)$ are searched in $U_{u_{in}} \times L^2(\Omega) \times X_{T_{in}}$, where

$$U_y = \left\{ \boldsymbol{v} \in [H^1(\Omega)]^2, \ \boldsymbol{v} = 0 \text{ on } \Gamma_w, \ \boldsymbol{v} = \boldsymbol{y} \text{ on } \Gamma_{in} \right\},$$

$$X_y = \left\{ \tau \in H^1(\Omega), \ \tau = y \text{ on } \Gamma_{in} \right\}.$$

The low value of the Ar number indicates that the heat transfer is dominated by convection and that the Boussinesq term that appears in (2) can be neglected. Then, according to some approximate candidates $\tilde{\boldsymbol{u}} \in U_{u_{in}}$ and $\tilde{T} \in X_{T_{in}}$ the following residual is defined:

$$\forall \tau \in X_0, \quad R(\tilde{T}, \tau, \tilde{\boldsymbol{u}}) = \int_\Omega (\tilde{\boldsymbol{u}} \cdot \nabla \tilde{T}) \tau \, dx - \int_\Omega \kappa \, \nabla \tilde{T} \cdot \nabla \tau \, dx - \int_{\Gamma_w} \Phi \, (\tilde{T} - T_{ext}) \tau \, d\sigma. \tag{6}$$

The function $\Phi \in \mathcal{C}^\infty$ that defines the heat flux at the fuselage is given by the Newton's law of cooling for convective heat transfer [28], which gives:

$$\forall x \in \mathbb{R}, \quad \Phi(x) = \frac{\kappa_f}{e} x, \tag{7}$$

with e the fuselage thickness and κ_f the fuselage thermal conductivity. We assume that, the FE code or FV code allows to compute the residual of any solution. On one hand, many research codes allow residual computation, for example Freefem++[†]. On the other hand, many industrial codes have evolved in such a way that the user can interact with the solver through hand-written pieces of code in Fortran, C or C++ (or other languages) which allows the computation of residuals at minimum programming cost in a semi-intrusive fashion. The cost of computing the residual is cheaper than the computation of one solution.

3.4. Design parameters

As stated in section 2, an important element of passenger satisfaction is thermal comfort whose evaluation rely on the use of CFD models that can be time consuming, which motivates the need for surrogate models of CFD based temperature fields. In this paper, we are interested in building surrogate models of the temperature field $T(\theta, x)$, the velocity field will not be reduced as well. Remember that the temperature field depends on many parameters as noted in section 2. For our first reference problem (see section 6.1), we will keep as design parameters the inflow temperature T_{in} and the fuselage's thermal conductivity κ_f. For our second reference problem (see section 6.2), we will keep as design parameters the inflow temperature T_{in}, the inlet temperature $T_{in,a}$ at the aisle Γ^1_{in}

[†]http://www.freefem.org/ff++/

DOI: 10.1002/nme

(see figure 12), the private inlet temperature $T_{in,p}$ above the passenger head Γ^2_{in} (see figure 12) and the fuselage's thermal conductivity κ_f. Using dimensionless parameters $\theta_i \in [-1,1]$, $i = 1,\dots,p$, we are looking for the family of fluid-thermal solutions $\left(\boldsymbol{u}^\theta = \boldsymbol{u}(\theta,.),\, T^\theta = T(\theta,.)\right)_{\theta\in[-1,1]^p}$.

4. ROM METHODOLOGY: GENERAL ASPECTS AND RELATED WORKS

4.1. Low-order modeling approach

A Reduced Order modeling (ROM) approach for partial differential equations consists of a low-order representation of the solution by help of few optimal basis functions, possibly adapted according to some enrichment process. Considering for example the parametrized temperature field $T^\theta = T(.,\theta)$, reduced-order models are searched in the form

$$T^\theta(x) = T^{lift,\theta}(x) + \sum_{k=1}^{K} a_k(\theta)\, \boldsymbol{\Psi}^k(x). \tag{8}$$

The function $T^{lift,\theta}$ is a lifting function aimed at satisfying some boundary conditions (especially Dirichlet BC), possibly depending on θ but quite easy to compute (for example the solution of a linear Stokes Problem). The family $(\boldsymbol{\Psi}^k)_{k=1,\dots,K}$ is the 'optimal' basis. The truncation rank K is expected to be rather small, let us say 10. The expansion coefficients $a_k(\theta)$ are functions depending on the vector parameter $\theta \in [-1,1]^p$. In a ROM methodology, there are two main steps: the design of the basis functions $\boldsymbol{\Psi}^k$ and the learning process of the $a_k(\theta)$.

4.2. Snapshot POD

In the snapshot POD approach [8, 29, 7], some snapshot fields $(T^i)_{i=1,\dots,N}$ are computed according to a Design of Computer Experiment (DoCE) from a parameter space sampling $\{\theta^i\}_i$. Then, the POD basis spans the best linear subspace able to represent the snapshot solutions:

$$\min_{\substack{(\boldsymbol{\Psi}^1,\dots,\boldsymbol{\Psi}^K)\\ (\boldsymbol{\Psi}^k,\boldsymbol{\Psi}^\ell)=\delta_{k\ell},\ 1\le k\le \ell\le N}} \frac{1}{2}\sum_{i=1}^{N} \left\| T^i - T^{lift,\theta^i} - \sum_{k=1}^{K} (T^i - T^{lift,\theta^i}, \boldsymbol{\Psi}^k)\, \boldsymbol{\Psi}^k \right\|^2, \tag{9}$$

which is equivalent to solving the problem of maximum correlation:

$$\max_{\substack{(\boldsymbol{\Psi}^1,\dots,\boldsymbol{\Psi}^K)\\ (\boldsymbol{\Psi}^k,\boldsymbol{\Psi}^\ell)=\delta_{k\ell},\ 1\le k\le \ell\le N}} \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K} (T^i - T^{lift,\theta^i}, \boldsymbol{\Psi}^k)^2. \tag{10}$$

In the POD methodology, the $(\boldsymbol{\Psi}^k)$ are said to be an empirical basis because of the empirical choice of the snapshot set (see [30] for a recent analysis on the optimal location of the snapshots).

4.3. Reduced basis and PGD

Reduced Basis Methods or RBM [9, 10] are sequential approaches where the basis is enriched during an iterative learning process. At a given iteration (k) involving k modes, a $(k+1)th$ mode Ψ^{k+1} is searched as a best corrector direction corresponding to the the worst case location in the parameter domain. This is a kind of 'min-max' algorithm. RBM involves easy-to-compute accuracy estimators for specific problems and involve a projection step of the underlying equations on the low-order basis; we refer to the literature ([31, 32, 33]) for this issue. Because of the iterative enrichment process, RBM belongs to the family of greedy algorithms. Let us emphasize that the RBM analysis framework is mainly restricted to elliptic problems. Another and recent approach which knows an increasing interest is the Proper Generalized Decomposition or PGD, introduced by Ladevèze in the context of the LATIN method for reducing computational costs. Then it is

DOI: 10.1002/nme

extended and used in many fields of applications ([13, 34, 35, 36, 37, 38]). PGD is also a greedy algorithm where the variables are separated. From a level-k model $\tilde{T}^{(k)}(\theta,.)$, a higher-fidelity model $\tilde{T}^{(k+1)}(\theta,.)$ is searched in the form

$$\tilde{T}^{(k+1)}(\theta,.) = \tilde{T}^{(k)}(\theta,.) + a_1^{(k+1)}(\theta_1)\, a_2^{(k+1)}(\theta_2) \ldots a_p^{(k+1)}(\theta_p)\, \mathbf{\Psi}^{(k+1)}(x), \tag{11}$$

where the one-dimensional functions $a_1^{(k+1)}(\theta_1)$, $a_2^{(k+1)}(\theta_2)$ $\ldots$ $a_p^{(k+1)}(\theta_p)$ and the spatial model $\mathbf{\Psi}^{(k+1)}(x)$ are searched in an optimal way, for example by a variational principle and a Galerkin projection, see references [13, 39] for more details. Although very promising, PGD still needs investigation especially for parameter problems. It is unclear from the numerical analysis point of view what is the truncation rank K for a given error criterion. Moreover, PGD for the moment is an intrusive approach, what can be a shortcoming in a practical large-scale industrial context. PGD also needs more developments in the case of coupled problems.

4.4. Physics based metamodels

Our paper aims at developing a semi-intrusive approach based on FE or FV code without modification to the reference code. So we refer to a formalism of semi-intrusive physics-based metamodeling proposed by Nair [40] which will be used in this work. This metamodeling technique was then extended by Audouze *et al.* [41] using POD method for spatial approximation within RBF interpolation of POD coefficients. They proposed two variants of ROM algorithms. The first one is based on both spatial and parameter space POD decompositions. The second one uses only POD decompositions in spatial space and models the coefficients of the decomposition as general functions of the parameter vector θ. The principle is to determine in offline stage the coefficients $a_k(\theta)$ in (8) by minimizing a function based on the residual returned by the fine solver as done in [16]. We assume that this computation is far less expressive than the fine FE computation. Indeed, the rank K is expected to be small, typically in the range $[4, 10]$, whereas the FE dimension is of order $10^5 - 10^6$ for a two-dimensional problem, and $10^6 - 10^7$ for a three-dimensional one. A classical approach to determine the POD coefficients $a_k(\theta)$ is to use a design of computer experiments (DoCE) in order to sample the parameter space, then to interpolate the obtained data set using metamodeling methodology (kriging or RBF for instance). Note that for a parallel computer architecture, all these sample computations can be done in parallel. For some problems or applications, the spatial structure of the solution may strongly depend on the parameters meaning the spatial components strongly change within the parameter space. To get a global model with all the spatial structure, this would involve a higher rank truncation K (say between 20 and 90 as illustration) thus reducing the ROM efficiency. In general case, there are large zones of parameter space where the solution shape does not change strongly. Our idea is to build a local adaptive POD ROM for zones centred on a sample in the parameter space. In each zone, we only need some POD modes to capture enough information. We here suggest an In Situ Adaptive Tabulation (ISAT) approach [20] combined with the POD for low-order spatial representation. The so-called POD-ISAT algorithm is an easy-to-implement semi-intrusive approach that can be used in our opinion in an industrial context.

5. THE POD-ISAT ALGORITHM

The determination of POD modes by the method of snapshots [8] relies on the prior computation of some accurate finite element solutions. A very popular physics-based metamodeling technique, namely the POD-Galerkin approach, consists of carrying out the approximation on the full Finite Element vector fields using POD modes and Galerkin projection [42]. POD-Galerkin approaches are based on a low-dimensional projection of the PDE equations so that it is accurate in some sense and the physics is included into the equations. However, it is known that the POD-Galerkin method may become unstable for convection-dominated problems, and stabilization/up-winding fixes are required. Moreover, these approaches are fully intrusive: the computational code must be accessible

DOI: 10.1002/nme

in order to build the reduced system to solve. In the present work, we rather use a semi-intrusive approach which consists of growing a database of fine solutions with some interpolation process and control of the accuracy. In some sense, it is a set of local reduced order models with trust regions. The accuracy of the model is controlled via trust regions which prevents from tuning the truncation rank of the low-order bases as can be done in [43]. The reduction of both space and parameters is performed by a combination of the POD method with the ISAT algorithm [19, 20], the goal being to cover the whole design parameter with local surrogate models enclosed in trust regions.

5.1. The ISAT algorithm

The purpose of the ISAT algorithm [19] is to tabulate a function $f(x)$:

$$\begin{aligned} f \quad &: \quad \mathbb{R}^p \longrightarrow \mathbb{R}^m \\ &\quad\ \theta \mapsto f(\theta). \end{aligned}$$

where θ and f are respectively input vector and output vector. In the context of this work, the input θ is the design parameter vector and the output is the temperature field in the cabin. Given a query, θ^q, ISAT returns $\tilde{f}(\theta^q)$, an approximation to $f(\theta^q)$. Then we define $\varepsilon = \|\tilde{f}(\theta^q) - f(\theta^q)\|$ the approximation error.
An essential aspect of ISAT is that the table is built up, not in a pre-processing stage, but *in situ* as the simulation is being performed. An other important aspect is that ISAT can control the size of the table so that the approximation error is less than a given tolerance value noted by ε_{tol}. At the beginning, the table is empty. Then the entries are added as needed based on the queries θ^i and on its output $f(\theta^i)$ computed by the fine simulation. Each entry is considered as a leaf of a binary tree. The i^{th} leaf includes:

- its location θ^i;
- the function value $f^i = f(\theta^i)$;
- a jacobian matrix A^i, which has components;

$$A^i_{kl} = \frac{\partial f^k}{\partial \theta_l}(\theta^i)$$

- The i^{th} region of accuracy (ROA) of the leaf in parameters space.

The matrix A^i is used to construct a local linear approximation. Given a query θ^q, the approximation to $f(\theta)$ based on the i^{th} leaf is defined as:

$$\tilde{f}(\theta^q) = f^i + A^i(\theta^q - \theta^i) \tag{12}$$

The ROA is defined to be the connected region containing θ^i, in which the error of the linear approximation is less than the specified tolerance ε_{tol}. In original ISAT, the ROA is approximated by an ellipsoid, called the ellipsoid of accuracy (EOA) whose center is θ^i. The EOA is initialized and it may subsequently be modified (or grown) as additional information about the ROA is generated. Then, there are 3 different scenarios:

- **Retrieve attempt**: For a new point query θ^q, the ISAT algorithm will identify a leaf such that the EOA covers this point. If such a leaf is found, the linear approximation to $f(\theta^q)$ based on that leaf is returned.
- **Growth attempt**: If the first attempt is unsuccessful, then $f(\theta^q)$ is directly evaluated by reference simulation. Some number of leaves that in some sense are close to θ^q are selected for *growth attempt*. Based on each of these selected leaves, the error ε in the linear approximation to $f(\theta^q)$ is evaluated, and if it is less than ε_{tol} then the leaf's EOA is grown to cover θ^q and $f(\theta^q)$ is returned. The new EOA will be the minimum ellipsoid which covers the old EOA and θ^q (see [44] for more detail).
- **Add**: If the growth attempt is unsuccessful, then the new leaf containing θ^q is added to the binary tree.

DOI: 10.1002/nme

In this way, the ISAT is as an efficient storage and retrieval method. The new entry is stored in the table as needed and is indexed by the leaf of a binary tree. A new stored leaf's EOA is distinguished from previous EOAs by a cutting plane. The information of theses cutting planes is stored in the nodes of the binary tree. Thanks to these ones, ISAT finds out rapidly the EOAs that cover or are nearest to the query point in the parameter space. Furthermore, it is worth noting that the ISAT algorithm becomes efficient when it is needed to evaluate the output $f(\theta)$ very many times. Indeed, during the table building, ISAT requires the fine evaluation of the output. But, once the table is built up covering the whole design parameter space, ISAT becomes highly efficient at computing the output with the prescribed tolerance threshold as no fine computation is required. The speed-up is then equal to the ratio between the CPU time needed for a fine computation and the CPU time needed for a retrieve. Typical values of speed-up for ISAT applications are of the order of 1000.

5.2. Modified ISAT algorithm

In this paper modifications have been made to the original ISAT algorithm [19] to adapt it to our problem.

5.2.1. Derivatives – The biggest limitation of ISAT is the requirement of gradient matrix. Hedengren [45] provided a method using linear regression to approach the gradient matrix. Varshney and Armaou [46] used finite differences with common random numbers to compute the gradient matrix. However, the accuracy of computing this matrix will likely impact the efficiency of ISAT. In this paper, as the approximation model is based on local POD bases, no gradient or Hessian matrix are used to build the EOAs, improve the local POD model or estimate the error [20].

5.2.2. EOA growth and correction – Furthermore, Lu *et al.* [20] noticed that the growth step can induce inaccurate trust regions and proposed strategies to correct these errors. For simplicity reasons and as the strategies developed in [20] are beyond the scope of this paper, the growth step is disabled. Moreover, [20] proposed methods to accelerate the retrieve step, check the error and correct inaccurate EOAs. A correction method of the trust region based on ellipsoid shrinking [44] (maximum volume algorithm) is used: if the EOA encompasses points that do not satisfy the tolerance error, a maximum volume EOA that do not contain these points is generated. As our goal is to cover the design parameter domain with the smallest number of trust regions the maximum volume has been adopted.

5.2.3. Combination with POD – In addition, the ISAT proved its performance in the case of low dimensional outputs ($\leq$50). In the context of CFD, the output may be a velocity field or a temperature field which are high dimensional (= number of nodes $> 10^6$). The POD is one of the best widely used method to reduce the dimensionality of CFD fields. For these reasons, this paper proposes a combination of ISAT and POD to build a reduced order model applicable to the design of aircraft air control systems.

5.3. Design of Computer Experiment (DoCE) –

The parameter space sampling is an important issue for the accuracy of any metamodel. Commonly used DoCE procedure include Latin Hypercube Sampling (LHS), U-designs [47] and Lattice Design [48]. In this work, we used LHS method to build our DoCEs. Let N_{init} be the number of design sites in the parameter space chosen according to a LHS design procedure. After computing the exact solutions (e.g. the temperature fields) for these design sites by a FE code, an initial snapshot set $\mathcal{S}^{N_{init}}$ is formed as follows:

$$\mathcal{S}^{N_{init}} = \left\{ T^i,\ i = 1, \ldots, N_{init} \right\}. \tag{13}$$

This preliminary LHS design step allows us to explore the design space while creating neighbours for each snapshot. This neighbourhood is used to initialize the building of local low-order POD

DOI: 10.1002/nme

bases. This step is mandatory in our method as no derivative information is provided contrary to the original ISAT method of Pope [19].

5.4. *Boundary conditions treatment* –

Furthermore, the boundary conditions are an important issue in reduced order modeling. A quite easy way to compute a regular lifting function that satisfies the boundary conditions (especially Dirichlet BC), is to solve a linear Laplace problem. For each snapshot of $\mathcal{S}^{N_{init}}$, defined by a solution T^i corresponding to the vector of design parameters θ^i ($T^i = T(\theta^i)$), a lifting function T^{lift,θ^i} is computed as the solution of the following problem:

$$\triangle T^{lift,\theta^i} = 0 \quad \text{in } \Omega, \tag{14}$$

$$T^{lift,\theta^i} = T_{in} \text{ on } \Gamma_{in}, \ \frac{\partial T^{lift,\theta^i}}{\partial n} = 0 \text{ on } \Gamma_{out}, \ \kappa \frac{\partial T^{lift,\theta^i}}{\partial n} = \Phi(T^{lift,\theta^i} - T_{ext}) \text{ on } \Gamma_w \tag{15}$$

Note that the solution linearly depends on the Dirichlet boundary condition T_{in}. To reduce the computational cost, this equation is solved once by a FE method for the boundary condition $T_{in} = 1$. The corresponding solution, called $T^{lift,1}$ is stored. Then by linearity the solution of (14,15) for an arbitrary T_{in} is simply $T^{lift,\theta^i} = T_{in}\, T^{lift,1}$.

5.5. *Local form of the POD-ISAT ROM*

Assume that we are at the addition step, so the current query parameter θ^i becomes a new entry for the table, say the ith entry. The corresponding solution T^i (e.g. temperature field) is computed by the FE model. Then a new local reduced order model is built up at θ^i in five main steps:

- Computation of local POD modes (see section 5.5.1),
- Initial interpolation model of POD coefficients (see section 5.5.2),
- Building of an initial trust region (see section 5.5.3),
- Final interpolation model of POD coefficients (see section 5.5.4),
- Building of the final trust region (see section 5.5.5).

These steps are summarized in the figure 2. The adaptive enrichment process is aimed at covering the whole design domain. Below we provide the whole details of the algorithm, especially the construction of the trust region.

5.5.1. Local POD modes – Firstly, we need to compute the good POD modes, then build up a local approach for the neighbour of θ^i. The local POD modes are computed using a sample consisting in N_{local} nearest (in the sense of the euclidean norm in the parameter space) neighbours of θ^i:

$$\mathcal{S}_{(i)}^{N_{local}} = \left\{ (T^j - T_{(i)}^{lift,\theta^j}),\ j = 1, \ldots, N_{local} \right\}, \tag{16}$$

with $T_{(i)}^{lift,\theta^j} = T^{lift,\theta^j} + (T^i - T^{lift,\theta^i})$ the concentrated lifting function. The N_{local} samples $\mathcal{S}_{(i)}^{N_{local}}$ are selected among the fine finite element solutions recorded in the ISAT database.
The truncation order K^i is chosen so that the energy captured by the K^i first modes is higher than a confidence threshold. In other word, the projection error is expected to be less than a tolerance value. However, for the sake of clarity, we will fix $K^i = K$ for all local ROMs. Rather than varying the local truncation rank K^i adaptively, we preferred to fix it in order to highlight the benefit of using trust regions to perform surrogate modeling of computational fields. This choice obviously implies that the extent of the trust region depends on the truncation rank of the reduced basis. The adaptive tuning of the local truncation rank K^i could improve the performances of the method at the price of increasing the number of basis functions that can become large. This motivates our choice of fixing the truncation rank and focusing on the use of trust regions. The local form of ISAT-POD

DOI: 10.1002/nme

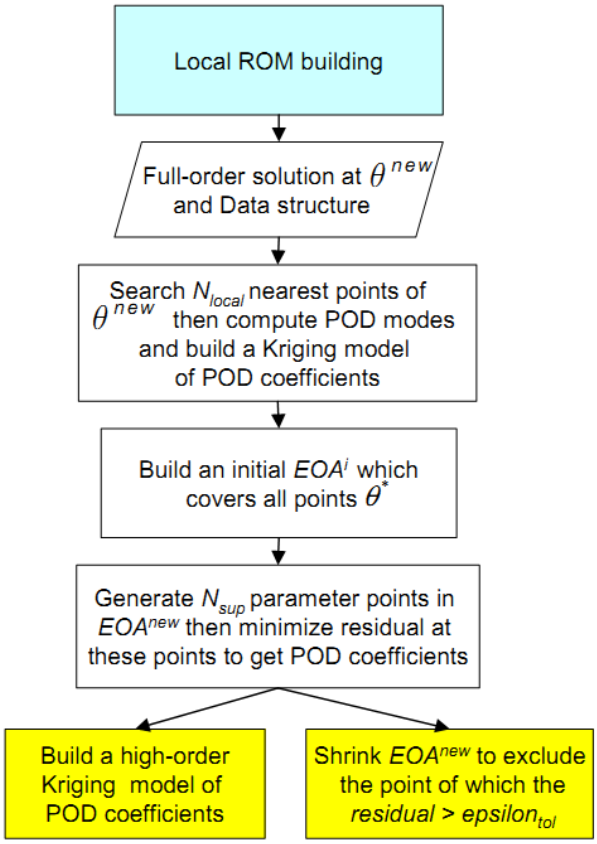

Figure 2. Details of local ROM building where θ^{new} corresponds to θ^i in the algorithm 1 and to θ^{query} in the algorithm 2.

reads as follows:

$$\tilde{T}^{\theta}_{(i)}(x) = T^{lift,\theta}_{(i)}(x) + \sum_{k=1}^{K} a^k_{(i)}(\theta)\, \mathbf{\Psi}^k_{(i)}(x), \tag{17}$$

where the local POD coefficients $a^k_{(i)}$ depends on the design parameters θ and θ^i.

5.5.2. Initial local POD coefficients– Generally, the local POD coefficients $a^k_{(i)}$ are unknown at θ. Note that we can compute these ones at the N_{local} sampling points θ^j as the coefficients of POD projection:

$$a^k_{(i)}(\theta^j) = \left(T^j(x) - T^{lift,\theta^j}_{(i)}(x), \mathbf{\Psi}^k_{(i)}(x)\right); k = 1, \ldots, K; j = 1, \ldots, N_{local}. \tag{18}$$

Using (18), the coefficients $a^k_{(i)}(\theta)$ can be interpolated or approximated by standard robust methods (Moving Least Square (MLS) [49, 50], artificial neural networks (ANN) [51], radial basis functions (RBF) [41] or Kriging approaches [52]). In this paper we used the kriging technique provided by the DACE Toolbox [‡] to approximate the K POD coefficients.

$$\begin{aligned} \tilde{a}^k_{(i)} \quad &: \quad \mathbb{R}^p \longrightarrow \mathbb{R} \\ &\theta \mapsto \tilde{a}^k_{(i)}(\theta), \quad k = 1, ..., K. \end{aligned} \tag{19}$$

[‡] http://www2.imm.dtu.dk/~hbn/dace/

DOI: 10.1002/nme

5.5.3. Initial Trust region building – Assume that a new database record is currently constructed. We need to characterize a Trust Region around θ^i. Assuming that the POD coefficients are continuous functions in the parameter space, there is a region $\mathcal{E}(i)$ such that the residual field satisfies

$$\forall \theta \in \mathcal{E}(i) \quad \left\| R(\tilde{T}_{(i)}(\theta), \cdot, \boldsymbol{u}^{\theta^i}) \right\|_{\mathcal{L}^2} \leq \varepsilon_{tol}. \left\| R_0 \right\|, \tag{20}$$

where $\varepsilon_{tol} \ll 1$, and $\|R_0\|$ is a reference residual.
For computational conveniences, the trust region $\mathcal{E}(i)$ is searched as an ellipsoid, as initially proposed by Pope [19]. The so-called ellipsoid of accuracy (EOA) in $\mathbb{R}^p$ is uniquely defined by $\frac{p^2+p}{2}$ points. The EOA is determined by finding out $M > \frac{p^2+p}{2}$ different θ^* in the vicinity of θ^i such that $\|R(\tilde{T}_{(i)}(\theta^*), \cdot, \boldsymbol{u}^{\theta^i})\|_{\mathcal{L}^2} - \varepsilon_{tol}. \|R_0\|$ is minimized. The size of the EOA obviously depends on the choice of tolerance threshold ε_{tol}. The sample points θ^* are searched in the form $\theta^* = \theta^i + \alpha^* h$, with $\alpha^* \in \mathbb{R}$ and $h \in \mathbb{R}^p$ is a fixed unit vector. By randomly choosing M different vectors h, M points θ^*_h are computed in parallel as the M one-dimensional minimization problems are independent (see Figure 3). So this algorithm is particularly suited for today's many-core computer architectures.
In the expression of the residual (6) used for the minimization problems, the velocity field associated with the temperature field appears. We assume that a good approximation of this velocity field within the ellipsoid of accuracy is given by the velocity field at θ^i i.e. at the center of the ellipsoid of accuracy. This hypothesis seems reasonable as the ellipsoid of accuracy is built by minimizing the $\mathcal{L}^2$ norm of the expression of the residual. Furthermore, the evaluation of the $\mathcal{L}^2$ norm of the residual for any value of θ^* needs the local POD model described by (17) giving access to $\tilde{T}_{(i)}(\theta^*)$. The local POD coefficients of the model (17) are unknown at θ^*, they are determined by minimizing $\mathcal{L}^2$ norm of the residual at fixed θ^*:

$$(a^1_{(i)}, \ldots, a^K_{(i)})(\theta^*) = \arg \min_{(a^1_{(i)}, \ldots, a^K_{(i)})} \frac{1}{2} \left\| R\left(T^{lift,\theta^*}_{(i)}(x) + \sum_{k=1}^{K} a^k_{(i)} \, \boldsymbol{\Psi}^k_{(i)}(x), \cdot, \boldsymbol{u}^{\theta^i} \right) \right\|^2_{\mathcal{L}^2}. \tag{21}$$

This residuals minimization approach has been used by many authors in the literature [53],[54], [55], [56]. In summary, for a given value of θ^*, the local POD coefficients are determined using (21), then the local POD model (17) is evaluated and the $\mathcal{L}^2$ norm of the residual (6) is computed. Practically we have used the Matlab Ellipsoidal Toolbox [§]. This toolbox allows to build an ellipsoid which forms the contour of the boundary points θ^*_h.

5.5.4. Final local POD coefficients – The local representation (17) is valid only in a trust region denoted by $\mathcal{E}(i)$ defined as an ellipsoid centred at θ^i. Once this trust region is determined, a supplementary random sampling $(\theta^j)_{j=1,\ldots,N_{sup}} \in \mathcal{E}(i)$ within the EOA is generated and the local POD coefficients $a^k_{(i)}(\theta^j)$, $k \in [1, \ldots, K]$, $j \in [1, \ldots, N_{sup}]$ are computed by (21). This optimization problem can be solved by means of standard search algorithms in low dimension. We used a standard optimization algorithm to make the implementation easy. The initial guess for $a^k_{(i)}$ is important as it conditions the efficiency of the optimization. One can choose the initial guess as follows: $a^k_{(i)}(\theta^i) = 0$, $k \in [1, \ldots, K]$. Another way is to initialize with the interpolated values of the coefficients using (19). Then, using these coefficients, an interpolation (kriging for example) of $a^k_{(i)}(\theta)$ noted $\tilde{\tilde{a}}^k_{(i)}(\theta)$ is built one more time.

Thus, the local ROM attached to the point θ^i is completely defined by:

- the formula (17),
- the trust region $\mathcal{E}(i)$,
- the POD coefficients kriging interpolation model $\tilde{\tilde{a}}^k_{(i)}(\theta)$, $\quad k = 1, ..., K$.

Then this new record can be added to the database.

[§] http://www.mathworks.com/matlabcentral/fileexchange/21936-ellipsoidal-toolbox-et

DOI: 10.1002/nme

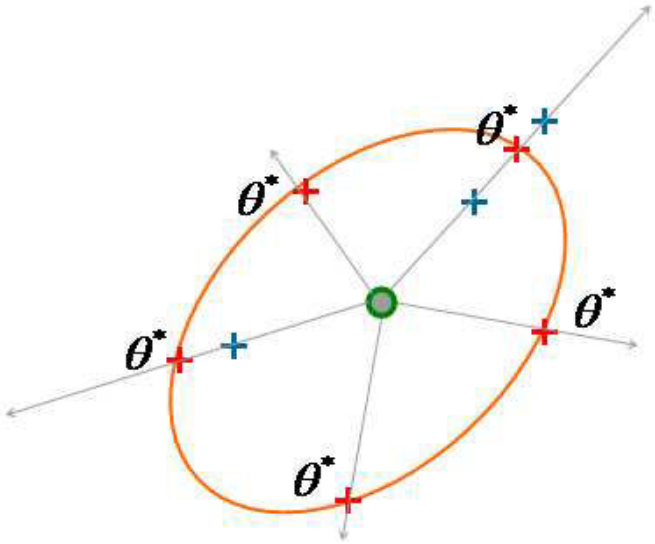

Figure 3. Trust region building (for each random direction, one finds out a point θ^* such that $\|R\| - \varepsilon_{tol}.\|R_0\| \leq 0$, this point will be used to define the boundary of EOA)

5.5.5. Final Trust region building – The initial trust region (see section5.5.3) is sought as an ellipsoid. It is worth noticing that this ellipsoid may encompass points that violate the inequality (20) that defines the trust region. Thus, it not strictly guaranteed that the residuals are less than the given threshold within the ellipsoid. Once sampling points are generated inside the initial EOA to build a more accurate kriging interpolation model $\tilde{\tilde{a}}^k_{(i)}$ (see section5.5.4), the EOA boundary is adapted so that only the points with residuals less than the given threshold are kept. The ellipsoid describing the EOA is shrunk using the algorithm of maximum volume algorithm described at section 11.1 of [44].

5.6. Summary of POD-ISAT algorithm

The POD-ISAT algorithm consists of 2 stages. The first one is called the offline preliminary stage and the second one is the adaptive enrichment building stage. Initially, POD-ISAT algorithm needs a preliminary sampling $\mathcal{S}^{N_{init}} = \left\{T^i,\ i = 1, \ldots, N_{init}\right\}$. The sampling parameters are generated by a space-filling sampling method (for example Latin Hypercube Sampling, LHS) and the corresponding (fine) solutions are computed by Finite Elements (or Finite Volume methods). In practice, these independent computations can be done in parallel.
Thanks to this initial sampling, the corresponding local ROMs for each θ^j in $\mathcal{S}^{N_{init}}$ are built if needed using algorithm 1. Let us highlight that in the preliminary stage each point of the initial sampling is saved. If it belongs to the trust region of an existing record, we do not need to build the local ROM at this point, but we can save it in the database for the purpose of local POD construction (see algorithm 1 and figure 4). At the end of this preliminary stage, the database consists of $n_{record} \leq N_{init}$ local ROMs. Let us mention that the order of the samples in the initial sampling $\mathcal{S}^{N_{init}}$ impacts the number $n_{records}$ of built EOA which can be interesting to devise an algorithm to choose the samples in order to reduce the number of records and increase the covered design space parameter volume, but it is beyond the scope of this paper.
In the next stage, the sample within its local ROM will be built up adaptively and added in the database (see algorithm 2 and figure 4). For the new query point θ^q, the POD-ISAT algorithm looks for a trust region which covers this entry. If the research is successful, then we have a local ROM candidate and the approximate solution will be returned instantaneously. Otherwise, a new record will be added in the database. Progressively, the design parameter space will be filled up by all the overlapping trust regions.

DOI: 10.1002/nme

Data: Preliminary sampling of design parameter space using a DoCE procedure
Result: Preliminary adaptive Database

Compute the Finite Element solution for the preliminary sampling:
$\mathcal{S}^{N_{init}} = \left\{T^j = T(\theta^j),\ j = 1, \ldots, N_{init}\right\}$
for $i = 1$ **to** N_{init} **do**
  if $n_{records} = 0$ **then**
    Build the first local ROM at θ^i
  else
    Look for EOA which covers θ^i by the retrieve stage of ISAT.
    if *The retrieve stage is successful* **then**
      Define scenario='retrieve';
      We stock only (T^{θ^i}, θ^i) in the Database
    else
      Define scenario='addition';
      Build the new local ROM together with EOA at θ^i.
      Add this new information record to the Database.
    end
  end
end

Algorithm 1: Preliminary stage of POD-ISAT

Input: New query parameter θ^q
Output: Approximate solution $\tilde{T}^{\theta^q}$

Look for EOA containing θ^i which covers θ^q by the retrieve stage of ISAT.
if *The retrieve stage is successful* **then**
  Define scenario='retrieve';
  The solution is approached by: $\tilde{T}^{\theta^q}_{(i)}(x) = T^{lift,\theta^q}_{(i)}(x) + \sum_{k=1}^{K} a^k_{(i)}(\theta^q)\, \mathbf{\Psi}^k_{(i)}(x)$.
else
  Define scenario='addition';
  Call the fine simulation for θ^q.
  Build the new local ROM together with EOA at θ^q.
  Add this new information record to the Database.
  The fine solution is returned: $\tilde{T}^{\theta^q}(x) = T^{\theta^q}(x)$
end

Algorithm 2: Online building stage of POD-ISAT

6. NUMERICAL EXPERIMENTS AND RESULTS

The POD-ISAT method was tested on the use case described in section 3.

6.1. Example of aircraft air control system solution with two parameters

In this case, two design parameters are considered, the first is the temperature of air injected by the main ventilation T_{in} and the second is the fuselage thermal conductivity κ_f. These two parameters are supposed to vary within reasonable admissible intervals: $T_{in} \in [T_{in}^{min} = 21, \cdots, T_{in}^{max} = 28](^{\circ}\mathrm{C})$; $\kappa_f \in [\kappa_f^{min} = 10^{-4}, \cdots, \kappa_f^{max} = 10^{-3}](W \cdot m^{-1} \cdot K^{-1})$. The normalized parameters

DOI: 10.1002/nme

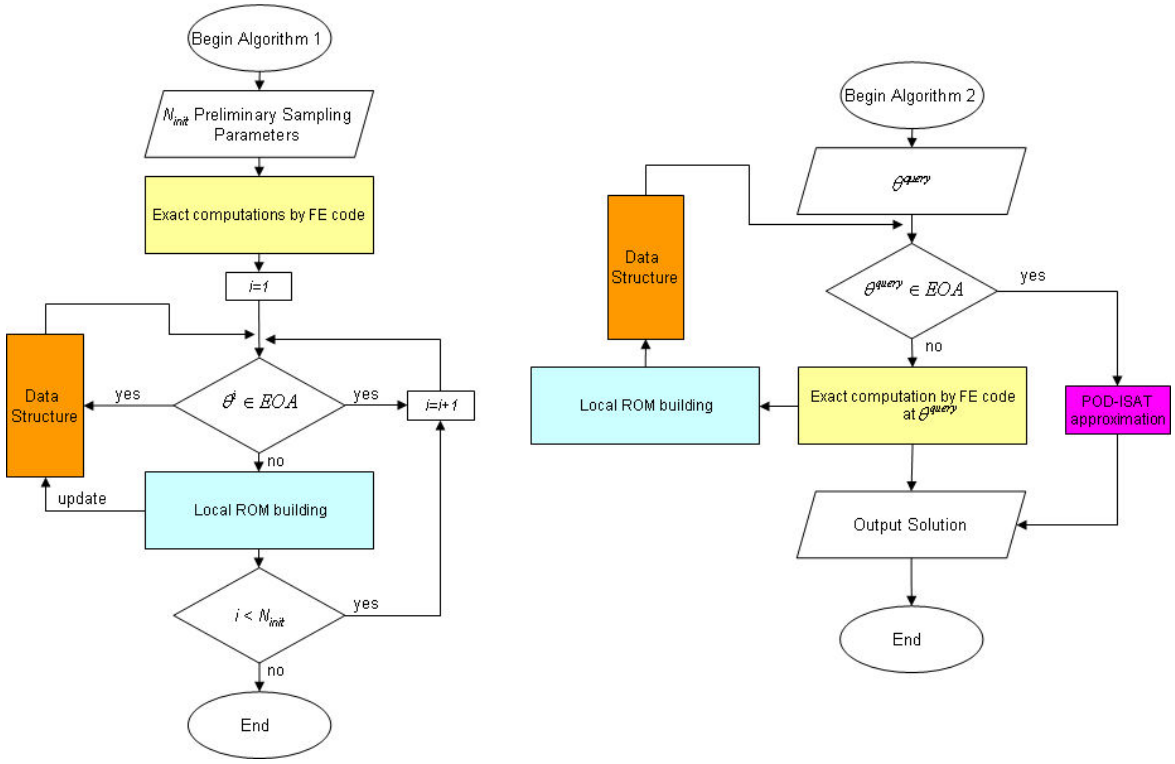

Figure 4. Diagrams presenting algorithm 1(left) and algorithm 2 (right).

$\theta = [\theta_1 \quad \theta_2]^t$ are defined as follows:

$$\theta_1 = \frac{T_{in} - \frac{1}{2}(T_{in}^{min} + T_{in}^{max})}{\frac{1}{2}(T_{in}^{max} - T_{in}^{min})},$$

$$\theta_2 = \frac{\kappa_f - \frac{1}{2}(\kappa_f^{min} + \kappa_f^{max})}{\frac{1}{2}(\kappa_f^{max} - \kappa_f^{min})},$$

The air is initially at rest and at uniform temperature in the cabin ($T_0 = 20°\mathrm{C}$). The hot air is injected at the fuselage's bottom and at the top of the cabin. We give for example four solutions (temperature fields) for four parameter vectors, chosen randomly in the admissible space (see the figure 5).

6.1.1. POD ROM sensitivity – In the perspective of evaluating the POD ROM sensitivity, a "small" DoCE (LHS) of nine reference solutions was computed on the whole design space parameter. The first 4 modes are presented in figure 6. Then, the parameter T_{in} was fixed ($T_{in} = 25.3°\mathrm{C}$) and the parameter κ_f was varied. To assess the sensitivity of the local ROM model with respect to the number of POD modes, we tested three different values of n_{POD}. For each value of the integer n_{POD}, 24 points uniformly distributed over the range of variation of the design parameter κ_f were considered. Then, for each of these points, the residual (6) of the POD ROM was minimized to obtain the POD coefficients (21). The results are shown in figure 7, where the optimal residual is normalized by a reference residual $R_0 = 10^{-5}$. Then, each curve appearing on figure 7 is composed of 24 points whose abscissa are the values of κ_f and ordinates the relative value of optimal residual . As expected, it is observed that the relative value of optimal residual ($\|R^{opt}\|/\|R_0\|$) is smaller when n_{POD} is bigger. If we take into account more POD modes, we obtain a more accurate approach. For this particular case, we found that we can only consider 4 POD modes to build a reasonably accurate reduced model. Furthermore, we assume that if the relative residual is less than 2, the corresponding POD ROM can be considered as accurate. For areas around $\theta_2 = -1$ or around $\theta_2 = -0.5$, the POD ROM is not reliable (see figure 7). It is probably due to a lack of "good" information, i.e. a lack of sample snapshots at these areas. However, in the region of $\theta_2 = 0, \cdots, 1$, the reduced

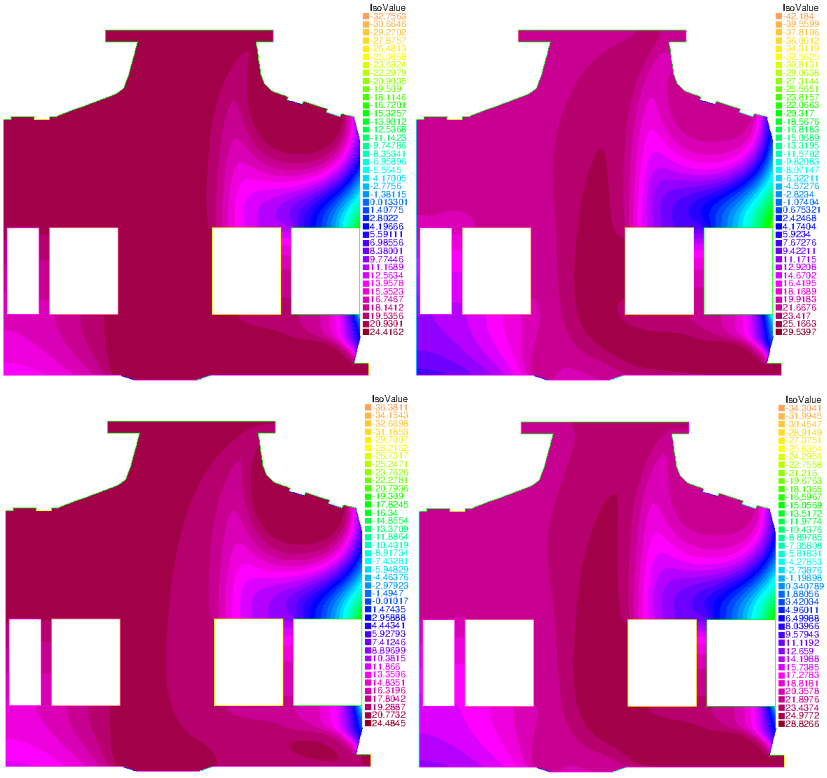

Figure 5. Reference Finite Element temperature solutions for four random parameter samples.

model is accurate. Therefore, there are areas in the design parameters space where the POD ROM is accurate and others where it is not. A way to improve the accuracy of the POD ROM model is to add snapshots in the areas where the model is not accurate. The POD-ISAT model adds new snapshots in these regions and builds up trust regions to control the model's accuracy, it is then expected to be more reliable and accurate , when the truncation rank is given.

6.1.2. POD-ISAT ROM building – To start building the POD-ISAT database, we need a preliminary snapshot set. For that we applied the algorithm 1-2. The choice of the open parameters of POD-ISAT ROM is summarized on the table I. Next, we give more details on the different POD-ISAT stages.

Table I. Properties of POD-ISAT reduced model

N_{init}	10	Initial DoCE Size
ε^2_{tol}	2	Threshold of relative residual
N_{local}	9	Number of nearest neighbours used to build local POD
K	4	Number of POD modes used
p	2	Number of parameters

Preliminary stage: At the beginning of this stage 10 reference FE solutions are computed relying on LHS sampling. At the end of this stage, $n_{records} = 3$, i.e there are 3 local ROMs which are built for three different parameter points. The other sampling points belong to these ROM's EOA. So

DOI: 10.1002/nme

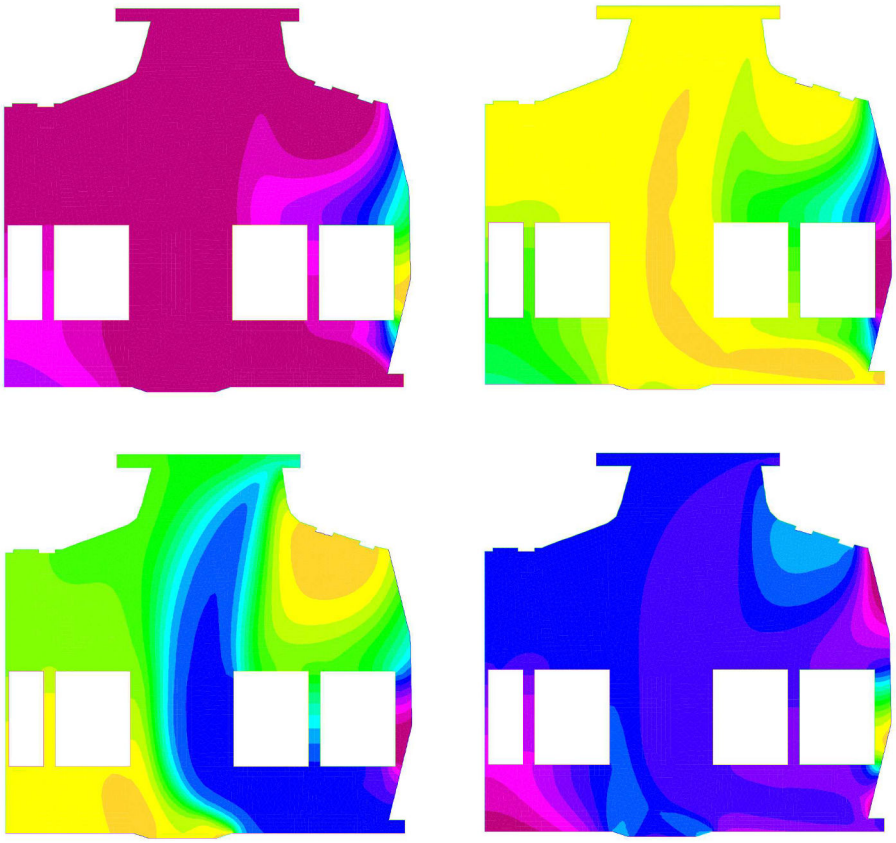

Figure 6. Isocontours of the four first POD eigenmodes: Ψ^1(top-left), Ψ^2 (top-right), Ψ^3 (bottom-left), Ψ^4 (bottom-right).

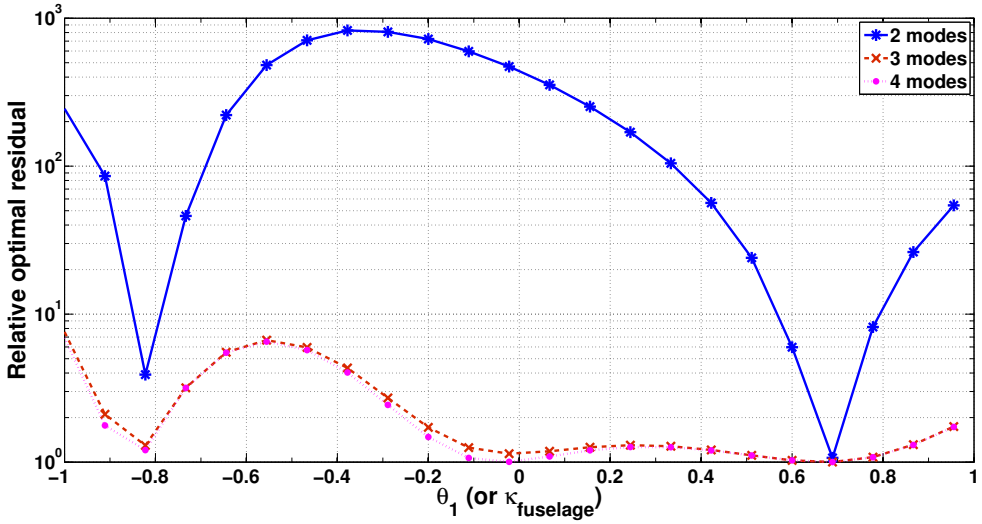

Figure 7. Relative optimal residual values according to the value of $\kappa_{fuselage}$ and the number of modes taken in account n_{POD}

we do not need to build local ROM at these points. However we need to save their computed exact solutions for use in the construction of local POD bases later.

Adaptive enrichment building stage: A number of 190 randomly generated query points are used by the algorithm 2. When POD-ISAT algorithm finishes (i.e.the whole design parameter is covered with EOAs), we have $n_{records} = 9$, i.e there are 9 local reduced models in the database. The final

DOI: 10.1002/nme

EOAs are showed in the normalized parameter space ($\theta \in [-1;1]^2$) (see figure 8). The EOAs cover almost the entire design parameter space. It is worth noticing that the POD-ISAT algorithm has examined 200 query points in total (preliminary stage and adaptive enrichment stage). These points are used to check the accuracy of the POD-ISAT approach. We computed all the fine FE solutions at these query points. Then, the POD-ISAT ROM solutions are compared with the corresponding FE solutions using the following error criterion (relative error):

$$\text{relative error} = \frac{\|\tilde{\boldsymbol{u}}(\theta) - \boldsymbol{u}(\theta)\|_{\mathcal{L}^2}}{\|\boldsymbol{u}(\theta)\|_{\mathcal{L}^2}}. \tag{22}$$

The distribution of the relative error values over the design parameter space is shown in figure 9, where one can find that the maximum relative error is 1.2×10^{-3}. This maximal error occurs close to a boundary of the design space parameter domain in the area of low inlet temperatures T_{in} and high fuselage conductivities κ_f.

Using the same test case and the same database size, we built a non-intrusive POD ROM using the

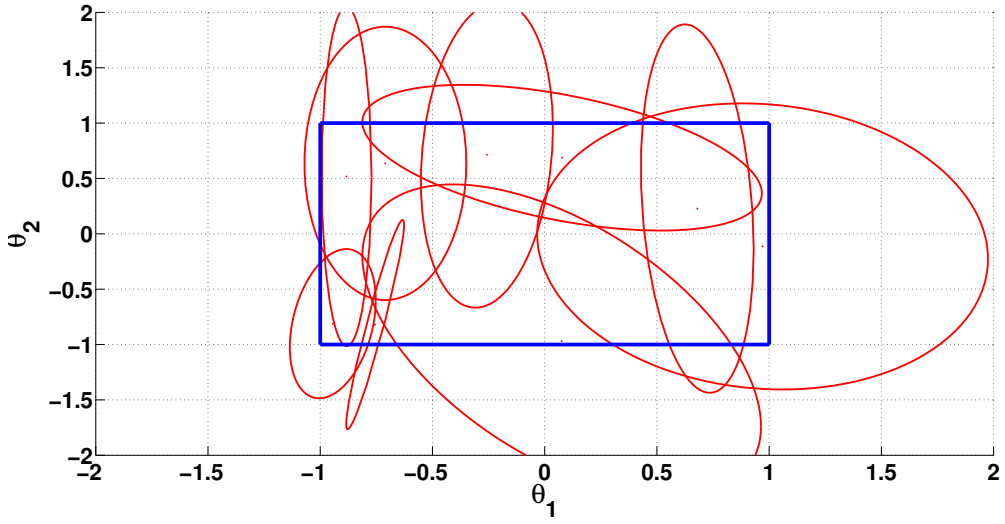

Figure 8. Representation of the EOAs covering the design space parameter $[-1;1]^2$ (rectangle)

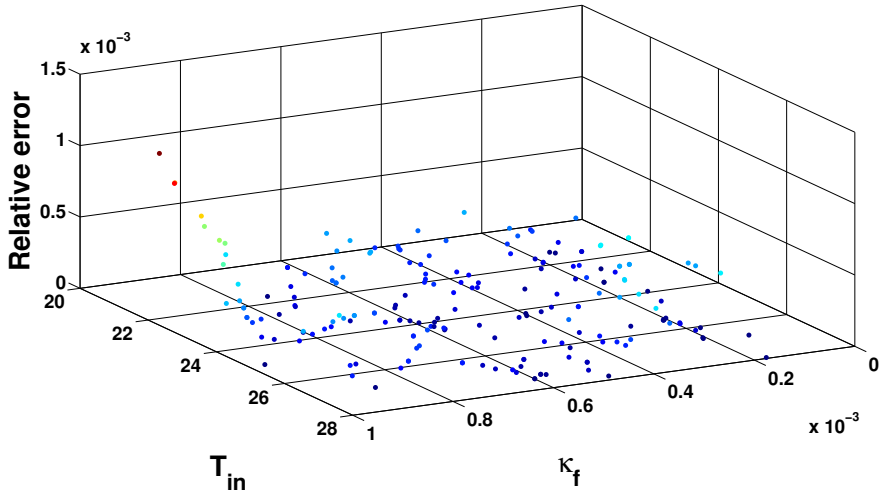

Figure 9. Relative errors for 200 queries computed by POD-ISAT ROM in design parameter space with 2 parameters

formula (17). We used the same number of high-fidelity simulations as the POD-ISAT method. To

DOI: 10.1002/nme

compute the POD modes, we formed a snapshot matrix of 16 points, 10 of them are the same as the 10 initial reference FE solutions quoted above and the remaining 6 others are generated by a LHS procedure. The POD coefficients are computed using a standard kriging method.
Then, the POD ROM was evaluated on the same 200 query points as outlined above and the relative

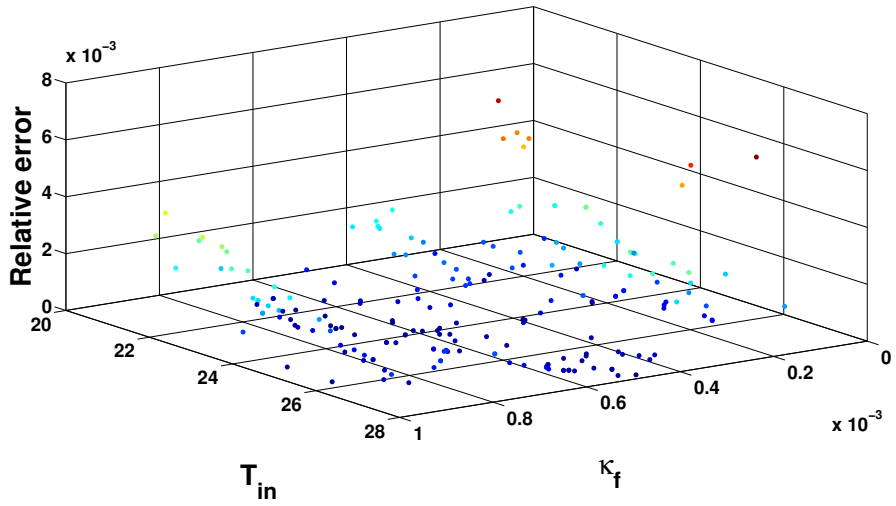

Figure 10. Relative errors for 200 queries computed by POD ROM in design parameter space with 2 parameters

error with FE solutions was computed. The distribution of the POD ROM relative error over the design parameter space is shown on figure 10, where one can see that, like for POD-ISAT (see figure 9), the maximal error occurs close to a boundary of the design space parameter domain in the area of low inlet temperatures T_{in} and high fuselage conductivities κ_f. But, unlike POD-ISAT, the POD ROM is inaccurate also in the area of low inlet temperatures T_{in} and low fuselage conductivities κ_f. The maximal relative error have been computed over 200 queries as can be seen on figure 11. It can be noticed that the POD-ISAT method is far more efficient than the standard POD method.

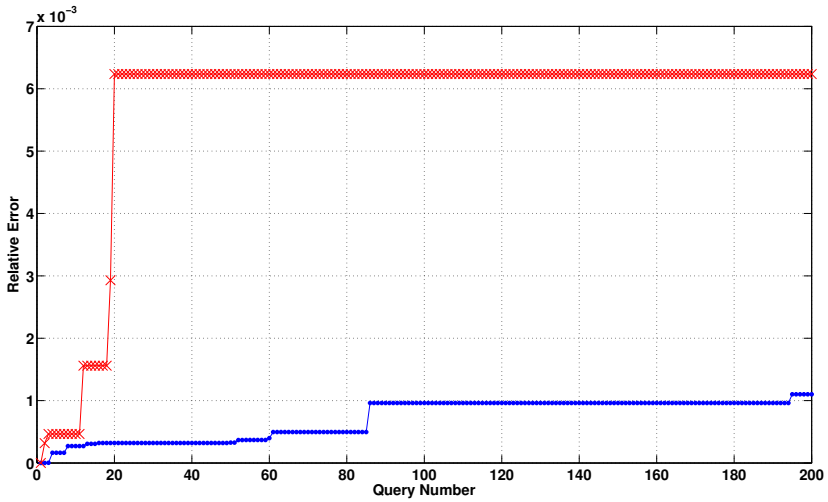

Figure 11. Maximal relative errors for POD ($\times$) and POD-ISAT ($\bullet$) for 2 parameters

6.2. *Second example: aircraft air control system with four parameters*

To show the ability of the method to deal with higher dimensional cases, we added in this test case 2 more design parameters (see figure 12). The first one is the inlet temperature $T_{in,a}$ at the aisle

DOI: 10.1002/nme

Γ_{in}^1 and the second one is the private inlet temperature $T_{in,p}$ above the passenger head Γ_{in}^2. These 2 parameters vary from $T_{in,a}^{min} = T_{in,p}^{min} = 21°\text{C}$ to $T_{in,a}^{max} = T_{in,p}^{max} = 28°\text{C}$. The dimensionless design parameter vector $\theta = [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4]^t$ is defined as follows:

$$\theta_1 = \frac{T_{in} - \frac{1}{2}(T_{in}^{min} + T_{in}^{max})}{\frac{1}{2}(T_{in}^{max} - T_{in}^{min})},$$

$$\theta_2 = \frac{\kappa_f - \frac{1}{2}(\kappa_f^{min} + \kappa_f^{max})}{\frac{1}{2}(\kappa_f^{max} - \kappa_f^{min})},$$

$$\theta_3 = \frac{T_{in,a} - \frac{1}{2}(T_{in,a}^{min} + T_{in,a}^{max})}{\frac{1}{2}(T_{in,a}^{max} - T_{in,a}^{min})},$$

$$\theta_4 = \frac{T_{in,p} - \frac{1}{2}(T_{in,p}^{min} + T_{in,p}^{max})}{\frac{1}{2}(T_{in,p}^{max} - T_{in,p}^{min})},$$

We generated $N_{init} = 20$ preliminary samples still using LHS space-filling technique. At the beginning, the algorithm 1 built up the first ROMs based on the preliminary samples. At the end of this stage $n_{records} = 9$. Then 400 queries randomly generated are called by the second stage. At the end we have $n_{records} = 28$. The relative error computed for the 420 queries are given in figure 13. The maximal relative error have been computed for 420 queries as can be seen on figure

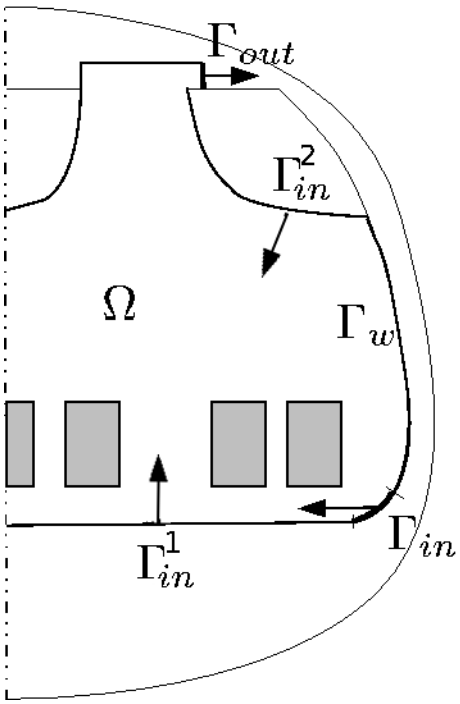

Figure 12. Two added design parameters are inlet temperature at the aisle and above the head of passenger

14. It can be noticed that the POD-ISAT method is more efficient than the standard POD method.

6.3. CPU costs evaluation

The significant CPU costs are analysed in the following. The simulation is performed on an Intel Core I7 Computer. The FE model is implemented on `FreeFem++3.10` [¶] and the POD-ISAT algorithm is implemented on `Matlab` [‖]. The total computational time t_{tot} to cover the

¶http://www.freefem.org/ff++/
‖http://www.mathworks.fr/products/matlab/

DOI: 10.1002/nme

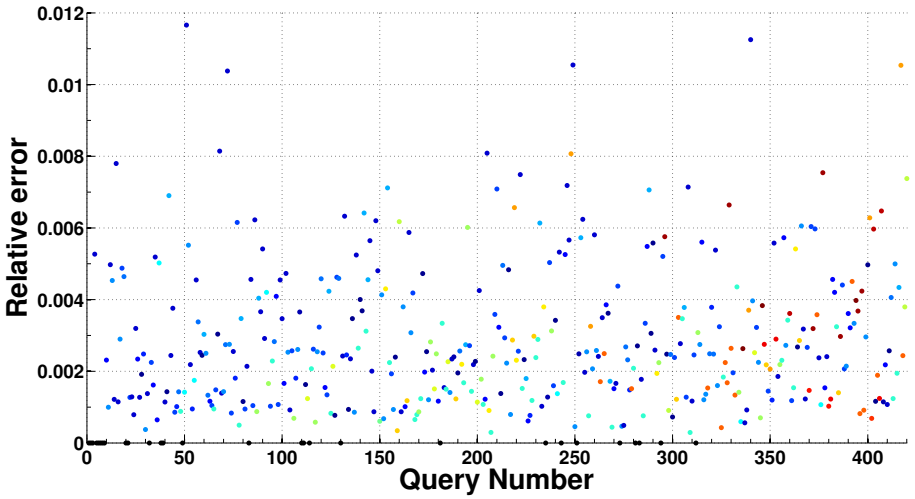

Figure 13. Relative errors for 420 queries computed by POD-ISAT for 4 parameters

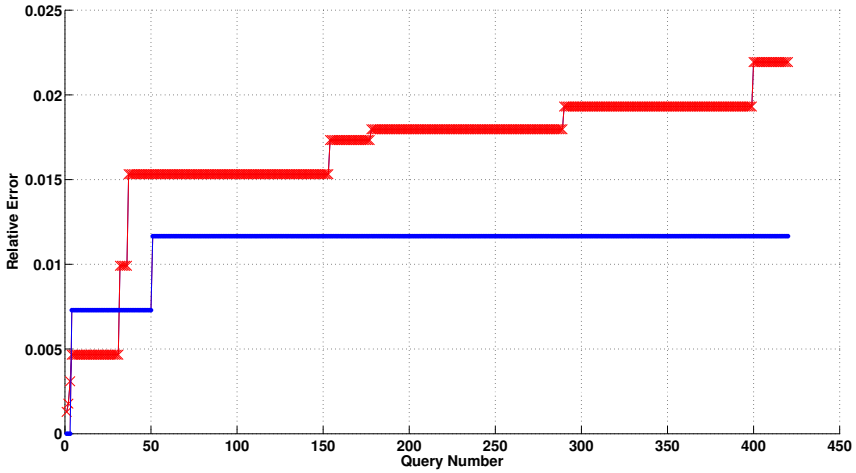

Figure 14. Maximal relative errors for POD ($\times$) and POD-ISAT ($\bullet$) for 4 parameters

whole design parameter space consists of off-line time t_{off} and adaptive enrichment time t_{enr} ($t_{tot} = t_{off} + t_{enr}$). These times have been computed as a mean over several runs. Each of these two stages of the POD-ISAT algorithm (1,2) involve elementary operations that have associated CPU times :

- the time t_{fem} for the fine evaluation of one single FE solution,
- the retrieve time t_{ret} or the time spent to search the database for the right EOA plus the time to evaluate the POD-ISAT ROM (17),
- the time t_{eoa} spent to build an EOA in section 5.5.3.
- the time t_{krg} spent to train a kriging model for local POD coefficients in section 5.5.4.
- the time spent to build a kriging model in section 5.5.2 and to modifie EOA in section 5.5.5 are negligible.

The off-line preliminary stage involves fine FE computations, EOA building, kriging model computation and retrieve attempts. The number of retrieve attempts for each stage of the POD-ISAT algorithm is stochastic. If we neglect the retrieve time, we have $t_{off} = n^{off}_{records} \times (t_{eoa} + t_{krg}) + N_{init} \times t_{fem}$ and $t_{enr} = (n^{enr}_{records} - n^{off}_{records}) \times (t_{eoa} + t_{krg} + t_{fem})$, with $n^{off}_{records}$ the number of records in the ISAT database at the end of the off-line preliminary stage and $n^{enr}_{records}$ the number of records in the ISAT database at the end of the adaptive enrichment stage. These costs are outlined in

DOI: 10.1002/nme

Table II where the times for the two cases treated above appear. For the use case with two parameters (see section 6.1), t_{tot} =5.5 hours of computation by evaluating 200 query points. For the use case with four parameters (see section 6.2), t_{tot} =15.2 hours of computation by evaluating 420 query points. Thus, building the database can be computationally intensive but it provides the coverage of the whole design parameter space with an accurate representation of the solutions compared to standard POD. The speed-up, defined as the ratio of the time to evaluate a solution with the FE code by the time needed to compute a solution with the POD-ISAT ROM once the database has been built up, is $1000/0.98$ or $1000/1.1 \sim 10^3$, which shows the computational efficiency of the POD-ISAT method once the database has been built up.

	Two parameters	Four parameters
EOA Building	316 sec	408 sec
Kriging model	95 sec	151 sec
Single FE computation	1000 sec	1000 sec
Retrieve	0.98 sec	1.1 sec
Off-line Preliminary stage	11233 sec	25031 sec
Adaptive enrichment stage	8466 sec	29621 sec

Table II. CPU costs

7. CONCLUDING REMARKS

In this paper, a general semi-intrusive adaptive reduced modeling strategy for steady state PDE discrete solutions has been presented. It combines both Proper Orthogonal Decomposition (POD) for dimensionality reduction of CFD fields and In Situ Adaptive Tabulation (ISAT) for efficient storage and retrieval of a time-consuming function by means of a database model relying on trust regions. The so called POD-ISAT algorithm has been tested and evaluated on a design problem of aircraft cabin air control system (ECS). The numerical results show that the ROM provides both high efficiency and accuracy in its on-line use. Of course it requires time-consuming evaluations of fine Finite Elements solutions during the offline preliminary stage and adaptive enrichment process, but once the database has been built up, the speed-up can reach 1000. The Trust Region (TR) strategy allows to control the accuracy of the model by means of ellipsoids of accuracy built up by minimizing the residuals of the fluid-thermal CFD equations.
The EOA building stage is an essential step of the POD-ISAT, that can be further improved in terms of accuracy and computational efficiency. Some strategies to do so have been presented in [20] to enhance the retrieve step and better the accuracy of the EOA. Moreover, the use of the residuals in the method is also a key point that can be improved by using more appropriate easy-to-compute a-posteriori error estimators for fluid-thermal CFD fields.

ACKNOWLEDGEMENT

This work is supported by the industrial Research Program “Complex System Design Lab”, Pôle de Compétitivité System@tic, Paris Région Ile-de-France, 2009-2012. We lwould like to thank Dr. John Hedengren for helping us for the implementation of the ISAT algorithm in Matlab.

REFERENCES

DOI: 10.1002/nme

1. Geradin, Michel, Rixen, and Daniel. *Mechanical Vibrations. Theory and Applications to Structural Dynamics*. Wiley, 1994. 03-53.
2. W.C. Hurty. Dynamic analysis of structural systems using component modes. *AIAA Journal*, 3:678–685, April 1965.
3. M. C. C. Bampton and R. R. Craig, Jr. Coupling of substructures for dynamic analyses. *AIAA Journal*, 6:1313–1319, July 1968.
4. Ladevèze Pierre. *Nonlinear Computational Structural Mechanics. New Approaches and Non-Incremental Methods of Calculation*. Mech. Eng. Series. Springer-verlag, 2009.
5. D. Ryckelynck. A priori hyperreduction method: an adaptive approach. *Journal of Computational Physics*, 202(1):346–366, 2005.
6. David Ryckelynck, Florence Vincent, and Sabine Cantournet. Multidimensional a priori hyper-reduction of mechanical models involving internal variables. *Computer Methods in Applied Mechanics and Engineering*, 225-228(0):28 – 43, 2012.
7. L. Sirovich. Turbulence and the dynamics of coherent structures. I - Coherent structures. II - Symmetries and transformations. III - Dynamics and scaling. *Quarterly of Applied Mathematics*, 45:561–571, October 1987.
8. G Berkooz, P Holmes, and JL Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25:539–575, 1993.
9. Yvon Maday and Einar M. Ronquist. A reduced-basis element method. *Comptes Rendus Mathématique*, 335(2):195 – 200, 2002.
10. G. Rozza, D. Huynh, and A. Patera. Reduced basis approximation and a posteriori error estimation for affinely parametrized elliptic coercive partial differential equations. *Archives of Computational Methods in Engineering*, 15:229–275, 2007. 10.1007/s11831-008-9019-9.
11. Kevin Carlberg and Charbel Farhat. A low-cost, goal-oriented compact proper orthogonal decomposition basis for model reduction of static systems. *International Journal for Numerical Methods in Engineering*, 86(3):381–402, 2011.
12. T. Bui-Thanh, K. Willcox, O. Ghattas, and B. van Bloemen Waanders. Goal-oriented, model-constrained optimization for reduction of large-scale systems. *J. Comput. Phys.*, 224(2):880–896, June 2007.
13. A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings. A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids. *Journal of Non-Newtonian Fluid Mechanics*, 139(3):153 – 176, 2006.
14. D. Amsallem and C. Farhat. Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity. *AIAA Journal*, 46:1803–1813, July 2008.
15. D. Amsallem and C. Farhat. An online method for interpolating linear parametric reduced-order models. *SIAM Journal on Scientific Computing*, 33(5):2169–2198, 2011.
16. David Amsallem, Matthew J. Zahr, and Charbel Farhat. Nonlinear model order reduction based on local reduced-order bases. *International Journal for Numerical Methods in Engineering*, pages n/a–n/a, 2012.
17. M. Bergmann and L. Cordier. Optimal control of the cylinder wake in the laminar regime by trust-region methods and pod reduced-order models. *Journal of Computational Physics*, 227(16):7813 – 7840, 2008.
18. Chenjie Gu and Jaijeet Roychowdhury. Model reduction via projection onto nonlinear manifolds, with applications to analog circuits and biochemical systems. In *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*, ICCAD '08, pages 85–92, Piscataway, NJ, USA, 2008. IEEE Press.
19. S.B. Pope. Computationally efficient implementation of combustion chemistry using in situ adaptive tabulation. *Combustion Theory and Modelling*, 1(1):41–63, 1997.
20. Liuyan Lu and Stephen B. Pope. An improved algorithm for in situ adaptive tabulation. *Journal of Computational Physics*, 228(2):361 – 386, 2009.
21. Rogers DJ. Tatem AJ, Hay SI. Global traffic and disease vector dispersal. *Proc Natl Acad Sci U S A*, 103(16):6242–6247, 2006.
22. Pavia AT. Germs on a plane: aircraft, international travel, and the global spread of disease. *J Infect Dis*, 195(1):621–622, 2007.
23. Sonja J. Olsen, Hsiao-Ling Chang, Terence Yung-Yan Cheung, Antony Fai-Yu Tang, Tamara L. Fisk, Steven Peng-Lim Ooi, Hung-Wei Kuo, Donald Dah-Shyong Jiang, Kow-Tong Chen, Jim Lando, Kwo-Hsiung Hsu, Tzay-Jinn Chen, and Scott F. Dowell. Transmission of the severe acute respiratory syndrome on aircraft. *New England Journal of Medicine*, 349(25):2416–2422, 2003.
24. P.O. Fanger. *Thermal comfort - Analysis and applications in environmental engineering*. PhD thesis, TU Denmark, 1970.
25. Sjoerd S. Burgers Henry Dol Stefan P. Spekreijse Johan C. Kok, Jaap van Muijden. Enhancement of aircraft cabin comfort studies by coupling of models for human thermoregulation, internal radiation and turbulent flows. In E. Onate P. Wesseling and J. Periaux, editors, *Proceedings ECCOMAS CFD 2006*. ECCOMAS CFD, 2006.
26. Shinichi Tanabe, Kozo Kobayashi, Junta Nakano, Yoshiichi Ozeki, and Masaaki Konishi. Evaluation of thermal comfort using combined multi-node thermoregulation (65mn) and radiation models and computational fluid dynamics (cfd). *Energy and Buildings*, 34(6):637 – 646, 2002. Special Issue on Thermal Comfort Standards.
27. Wei Liu, Sagnik Mazumdar, Zhao Zhang, Stephane B. Poussou, Junjie Liu, Chao-Hsin Lin, and Qingyan Chen. State-of-the-art methods for studying air distributions in commercial airliner cabins. *Building and Environment*, 47(0):5 – 12, 2012. International Workshop on Ventilation, Comfort, and Health in Transport Vehicles.
28. Louis C. Burmeister. *Convective heat transfer*. New York : Wiley, c1993, United-States of America, $2^n d$ edition, 1993. "A Wiley-Interscience publication.".
29. F. De Vuyst. *Multidisciplinary Design Optimization in Computational Mechanics*, chapter PDE Metamodeling using Principal Component Analysis. Wiley ISTE, April 2010.
30. Kunisch, Karl and Volkwein, Stefan. Optimal snapshot location for computing pod basis functions. *ESAIM: M2AN*, 44(3):509–529, 2010.

31. K. Veroy and A. T. Patera. Certified real-time solution of the parametrized steady incompressible navier-stokes equations: rigorous reduced-basis a posteriori error bounds. *International Journal for Numerical Methods in Fluids*, 47(8-9):773–788, 2005.
32. Gianluigi Rozza and Karen Veroy. On the stability of the reduced basis method for stokes equations in parametrized domains. *CMAME*, 196(7):1244 – 1260, 2007.
33. Simone Deparis. Reduced basis error bound computation of parameter-dependent Navier-Stokes equations by the natural norm approach. *SIAM J. Num. A.*, 46(4):2039–2067, 2008.
34. A. Dumon, C. Allery, and A. Ammar. Proper general decomposition (pgd) for the resolution of navier-stokes equations. *Journal of Computational Physics*, 230(4):1387 – 1407, 2011.
35. F. Chinesta, A. Ammar, A. Leygue, and R. Keunings. An overview of the proper generalized decomposition with applications in computational rheology. *Journal of Non-Newtonian Fluid Mechanics*, In Press, Corrected Proof:–, 2011.
36. G. Bonithon, P. Joyot, F. Chinesta, and P. Villon. Non-incremental boundary element discretization of parabolic models based on the use of the proper generalized decompositions. *Engineering Analysis with Boundary Elements*, 35(1):2–17, 2011.
37. A. Leygue and E. Verron. A first step towards the use of proper general decomposition method for structural optimization. *ACME*, 17:465–472, 2010. 10.1007/s11831-010-9052-3.
38. M. Beringhier, M. Gueguen, and J. Grandidier. Solution of strongly coupled multiphysics problems using space-time separated representations. application to thermoviscoelasticity. *Archives of Computational Methods in Engineering*, 17:393–401, 2010. 10.1007/s11831-010-9050-5.
39. Anthony Nouy. A priori model reduction through proper generalized decomposition for solving time-dependent partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 199(23-24):1603 – 1626, 2010.
40. P.B. Nair. Physics-based surrogate modeling of parameterized pdes for optimization and uncertainty analysis. In *Proceedings of 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA, 2002.
41. C. Audouze, F. De Vuyst, and P. B. Nair. Reduced-order modeling of parameterized pdes using time-space-parameter PCA. *IJNME*, 80(8):1025–1057, 2009.
42. K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics. *SIAM J. on Numerical Analysis*, 40(2):492–515, 2002.
43. B. Sarbandi, S. Cartel, J. Besson, and D. Ryckelynck. Truncated integration for simultaneous simulation of sintering using a separated representation. *Archives of Computational Methods in Engineering*, 17:455–463, 2010. 10.1007/s11831-010-9055-0.
44. S.B. Pope. Algorithms for ellipsoids. Technical report, Cornell University FDA, 2008.
45. John D. Hedengren and Thomas F. Edgar. Approximate nonlinear model predictive control with in situ adaptive tabulation. *Computers & Chemical Engineering*, 32(4-5):706 – 714, 2008. Festschrift devoted to Rex Reklaitis on his 65th Birthday.
46. A. Varshney and A. Armaou. An efficient optimization approach for computationally expensive timesteppers using tabulation. In AlexanderN. Gorban, IoannisG. Kevrekidis, Constantinos Theodoropoulos, NikolaosK. Kazantzis, and HansChristian ttinger, editors, *Model Reduction and Coarse-Graining Approaches for Multiscale Phenomena*, pages 515–533. Springer Berlin Heidelberg, 2006.
47. Boxin Tang. Orthogonal Array-Based Latin Hypercubes. *Journal of the American Statistical Association*, 88(424), 1993.
48. Kai-Tai Fang, Dennis K.J. Lin, Peter Winker, and Yong Zhang. Uniform design: Theory and application. *Technometrics*, 42(3):237–248, 2000.
49. P. Lancaster and K. Salkauskas. Surfaces Generated by Moving Least Squares Methods. *Mathematics of Computation*, 37(155):141–158, 1981.
50. Piotr Breitkopf, Hakim Naceur, Alain Rassineux, and Pierre Villon. Moving least squares response surface approximation: Formulation and metal forming applications. *Computers & Structures*, 83(17-18):1411 – 1428, 2005. Advances in Meshfree Methods.
51. Dreyfus Gérard. *Neural networks: methodology and applications*. Editions Eyrolles, 2005.
52. Noel Cressie. The origins of kriging. *Mathematical Geology*, 22:239–252, 1990. 10.1007/BF00889887.
53. Patrick A. LeGresley. *Application of Proper Orthogonal Decomposition (POD) to Design Decompo- sition Methods*. PhD thesis, Standford University, October 2005.
54. T. Bui-Thanh, K. Willcox, and O. Ghattas. Model reduction for large-scale systems with high-dimensional parametric input space. *SIAM Journal on Scientific Computing*, 30(6):3270–3288, 2008.
55. T. Bui-Thanh, K. Willcox, and O. Ghattas. Parametric reduced-order models for probabilistic analysis of unsteady aerodynamic applications. *AIAA Journal*, 46(10):2520–2529, 2008.
56. Kevin Carlberg, Charbel Bou-Mosleh, and Charbel Farhat. Efficient non-linear model reduction via a least-squares petrov-galerkin projection and compressive tensor approximations. *International Journal for Numerical Methods in Engineering*, 86(2):155–181, 2011.