Low rank approximation techniques and reduced order modeling applied to some fluid dynamics problems

Lucas Lestandi

▶ **To cite this version:**

Lucas Lestandi. Low rank approximation techniques and reduced order modeling applied to some fluid dynamics problems. Mechanics [physics]. Université de Bordeaux, 2018. English. NNT : 2018BORD0186 . tel-01947210

HAL Id: tel-01947210
https://theses.hal.science/tel-01947210

Submitted on 6 Dec 2018

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

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

Thèse présentée pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE DES SCIENCES PHYSIQUES ET DE L'INGÉNIEUR

SPÉCIALITÉ : MÉCANIQUE

Par Lucas Lestandi

Approximations de rang faible et modèles d'ordre réduit appliqués à quelques problèmes de la mécanique des fluides

Low rank approximation techniques and Reduced Order Modeling applied to some fluid dynamics problems

Sous la direction de: Pr. Mejdi Azaiez
Co-directeur : Pr Tomás Chacón Rebollo

Soutenue le 16 Octobre 2018

Devant la commission d'examen formée de :

M.	Luc Mieussens	Professeur, Bordeaux INP, IMB	Président du jury
M.	Gianluigi Rozza	Professeur, SISSA Trieste	Rapporteur
M.	Elias Cueto	Professeur, Université de Saragosse	Rapporteur
M.	Marianne Beringhier	Maître de conférences, ENSMA Poitiers	Examinatrice
M.	Samuele Rubino	Chercheur, Université de Séville	Examinateur
M.	Tapan K. Sengupta	Professeur, IIT Kanpur	Examinateur
M.	Mejdi Azaïez	Professeur, Bordeaux INP	Directeur
M.	Tomás Chacón Rebollo	Professeur, Université de Séville	Co-directeur

7 novembre 2018

Remerciements

En tant que thésard, on aime se raconter une histoire dans laquelle on est le seul responsable de l'issue de ces trois années, qu'elle soit bonne ou mauvaise. Mais, il est bien plus raisonnable de reconnaître que c'est tout un environnement qui nous conduit au terme de cette aventure. Pour ma part, ces trois années ont été incroyablement enrichissantes et viennent conclure ces très nombreuses années d'étude.

Je souhaite tout d'abord remercier mes parents, mon frère et toute ma famille de m'avoir permis de grandir dans les meilleures conditions. Ils m'ont encouragé avec succès à apprendre toujours et ont su être à mes côtés.

Je souhaite remercier les membres du jury pour le temps qu'ils m'ont accordé et tout particulièrement les rapporteurs pour le travail méticuleux qu'ils ont effectué.

Je souhaite bien sûr remercier Mejdi, mon directeur de thèse, de m'avoir accordé sa confiance et d'avoir vu en moi la capacité à mener à terme ce projet, d'avoir été un directeur attentionné, d'avoir su écouter ou je voulais emmener la thèse tout en prodiguant des conseils avisés. Merci aussi d'avoir rendu possible les très nombreux déplacements qui ont rythmés ces trois années.

Merci à Tomás, mon co-directeur, de m'avoir invité à plusieurs reprises à Séville et malgré ses si nombreuses responsabilités de m'avoir accordé du temps à chaque fois que c'était possible.

Merci à Samuele de m'avoir inclus dans ce projet de modèle réduit, j'ai énormément apprécié notre collaboration et évidemment j'y ai beaucoup appris. J'espère que celle-ci continuera avec tout le succès qu'elle mérite. Merci aussi pour la qualité de ton accueil à Séville, sans quoi mes séjours n'auraient pas eu la même valeur.

Je souhaite aussi témoigner de l'immense gratitude que j'éprouve pour le Pr. Tapan Sengupta. Merci de m'avoir accueilli avec autant de gentillesse et d'attention à Kanpur. Notre collaboration a été très fructueuse d'un point de vue scientifique bien sûr mais aussi personnel, j'ai énormément appris durant nos nombreuses discussions, merci beaucoup et à bientôt. Merci aussi à tous les membres du laboratoire HPC de l'IIT Kanpur pour leur accueil et en particulier Krishna, Pushpender et Emayavarman.[1]

Ces remerciements seraient incomplets si j'oubliais les membres du laboratoire I2M,

[1]I'd like to acknowledge my immense gratitude toward Pr. Tapan Sengupta. Thank you for welcoming me with so much kindness and attention in Kanpur, it really felt like home. Our collaboration was very fruitful, from the scientific point of view of course, but also on a personal level. I have learned so much by your side, these numerous discussions were very enriching. Thank you again for everything, à bientôt ! I would also like to thank the members of IIT Kanpur HPC lab and particularly Krishna, Pushpender and Emayavarman.

TREFLE. Merci à tous pour votre bonne humeur au quotidien, les nombreux conseils et toutes ces discussions informelles dans lesquelles on apprend tant, en particulier sur le monde tourmenté de l'enseignement supérieur et de la recherche. Merci aux anciens Antoine, Arnaud, Cédric, Pierre, Stéphane, Valérie,... aux doctorants passés et présents Fabien, Florian, Julien et Kevin. Et une mention spéciale pour Mathieu qui m'aura supporté dans son bureau pour la dernière année ! Si je garde un bon souvenir de ma thèse, c'est en grande partie grâce à vous.

Mais c'est aussi grâce à mes amis ! Pendant trois ans, je n'ai pas fait que travailler et heureusement. Merci à Jean pour son coup de main bienvenu en programmation. Merci aux footeux, Brendan et Thomas, de m'avoir permis de me défouler. Merci à Clem de m'avoir accompagné un peu partout ! Et à tous les autres (Tube, Francis, Gentien, Matmau, et beaucoup d'autres enserbiens !) pour ces bons moments ! Merci aux colocs d'avoir rendu ces derniers mois si doux malgré le travail !

Enfin, merci Marie d'avoir été à mes côtés pendant toutes ces années, quelle équipe de choc !

Abstract

In the last decades, numerical simulation has experienced tremendous improvements driven by massive growth of computing power. Exascale computing has been achieved this year and will allow solving ever more complex problems. But such large systems produce colossal amounts of data which leads to its own difficulties. Moreover, many engineering problems such as multiphysics or optimisation and control, require far more power that any computer architecture could achieve within the current scientific computing paradigm. In this thesis, we propose to shift the paradigm in order to break the curse of dimensionality by introducing decomposition and building reduced order models (ROM) for complex fluid flows.

This manuscript is organized into two parts. The first one proposes an extended review of data reduction techniques and intends to bridge between applied mathematics community and the computational mechanics one. Thus, founding bivariate separation is studied, including discussions on the equivalence of proper orthogonal decomposition (POD, continuous framework) and singular value decomposition (SVD, discrete matrices). Then a wide review of tensor formats and their approximation is proposed. Such work has already been provided in the literature but either on separate papers or into a purely applied mathematics framework. Here, we offer to the data enthusiast scientist a comparison of Canonical, Tucker, Hierarchical and Tensor train formats including their approximation algorithms. Their relative benefits are studied both theoretically and numerically thanks to the python library `pydecomp` that was developed during this thesis. A careful analysis of the link between continuous and discrete methods is performed. Finally, we conclude that for most applications ST-HOSVD is best when the number of dimensions d lower than four and TT-SVD (or their POD equivalent) when d grows larger.

The second part is centered on a complex fluid dynamics flow, in particular the singular lid driven cavity at high Reynolds number. This flow exhibits a series of Hopf bifurcation which are known to be hard to capture accurately which is why a detailed analysis was performed both with classical tools and POD. Once this flow has been characterized, *time-scaling*, a new "physics based" interpolation ROM is presented on internal and external flows. This methods gives encouraging results while excluding recent advanced developments in the area such as EIM or Grassmann manifold interpolation.

Key words: Data reduction, Model Reduction, MOR, POD, lid driven cavity, Low rank approximation, tensors, HOSVD, Tensor train, tensor formats, tensor approximation, physics interpolation, time-scaling.

Résumé

Les dernières décennies ont donné lieu à d'énormes progrès dans la simulation numérique des phénomènes physiques. D'une part grâce au raffinement des méthodes de discrétisation des équations aux dérivées partielles. Et d'autre part grâce à l'explosion de la puissance de calcul disponible. Pourtant, de nombreux problèmes soulevés en ingénierie tels que les simulations multi-physiques, les problèmes d'optimisation et de contrôle restent souvent hors de portée. Le dénominateur commun de ces problèmes est le fléau des dimensions. En effet,un simple problème tridimensionnel requiert des centaines de millions de points de discrétisation auxquels il faut souvent ajouter des milliers de pas de temps pour capturer des dynamiques complexes. L'avènement des supercalculateurs permet de générer des simulations de plus en plus fines au prix de données gigantesques qui sont régulièrement de l'ordre du pétaoctet. Malgré tout, cela n'autorise pas une résolution "exacte" des problèmes requérant l'utilisation de plusieurs paramètres. L'une des voies envisagées pour résoudre ces difficultés est de proposer des représentations ne souffrant plus du fléau de la dimension. Ces représentations que l'on appelle séparées constituent en fait un changement de paradigme. Elles vont convertir des objets tensoriels dont la croissance est exponentielle n^d en fonction du nombre de dimensions d en une représentation approchée dont la taille est linéaire en d. Pour le traitement des données tensorielles, une vaste littérature a émergé ces dernières années dans le domaine des mathématiques appliquées.

Afin de faciliter leurs utilisations dans la communauté des mécaniciens et en particulier pour la simulation en mécanique des fluides, ce manuscrit présente dans un vocabulaire rigoureux mais accessible, les formats de représentation des tenseurs et propose une étude détaillée des algorithmes de décomposition de données qui y sont associées. L'accent est porté sur l'utilisation de ces méthodes, aussi la bibliothèque de calcul `pydecomp` a été développée et utilisée pour comparer l'efficacité de ces méthodes sur un ensemble de cas qui se veut représentatif. La seconde partie de ce manuscrit met en avant l'étude de l'écoulement dans une cavité entraînée à haut nombre de Reynolds. Cet écoulement propose une physique très riche (séquence de bifurcation de Hopf) qui doit être étudiée en amont de la construction de modèles réduits. Cette étude est enrichie par l'utilisation de la décomposition orthogonale aux valeurs propres (POD). Enfin une approche de construction "physique", qui diffère notablement des développements récents pour les modèles d'ordre réduit, est proposée. La connaissance détaillée de l'écoulement permet de construire un modèle réduit simple basé sur la mise à l'échelle des fréquences d'oscillation (time-scaling) et des techniques d'interpolation classiques (Lagrange,..).

Mots-clés : Réduction de données, réduction de modèle, MOR, POD, Cavitée entrainée, HOSVD, Tensor train, tenseurs, formats tensoriels, approximation de tenseurs, interpolation physique, approximation de rang faible.

Résumé (long)

Contexte.

Les dernières décennies ont donné lieu à d'énormes progrès dans la simulation numérique des phénomènes physiques. Notamment grâce au raffinement des méthodes de discrétisation des équations aux dérivées partielles mais surtout à l'explosion de la puissance de calcul disponible. Pourtant, de nombreux problèmes soulevés en ingénierie tels que les simulations multiphysique, les problèmes d'optimisation et de contrôle restent hors de porté dans la plupart des cas. Le dénominateur commun de ces problèmes est le fléau des dimensions. En effet, un simple problème tridimensionnel requiert des centaines de millions de points de discrétisation auxquels il faut souvent ajouter des milliers de pas de temps pour capturer des dynamiques complexes. Pour résoudre les problèmes d'optimisation paramétrique, il faut renouveler ces calculs plusieurs centaines ou milliers de fois pour obtenir un optimum. Ces exigences dépassent largement les capacités des ordinateurs actuels et futurs (l'ordinateur quantique n'apportera qu'une réponse partielle à ces difficultés). Enfin, l'avènement des supercalculateurs permet de générer des simulations de plus en plus fines au prix de données gigantesques qui sont régulièrement de l'ordre du Po sans pour autant autoriser une résolution "exacte" des problèmes requérant l'utilisation de plusieurs paramètres. Ces problématiques sont particulièrement aiguës dans le contexte de simulation numérique pour la mécanique des fluides.

C'est pourquoi on se propose dans cette thèse de se placer dans un nouveau paradigme, celui de modèles et approximations d'ordre faible. Connues théoriquement depuis le milieu du XX^{e} siècle, ces méthodes ont connu un formidable essor depuis la fin des années 1980. Dans un premier temps les méthodes adaptées pour les problèmes à deux variables se sont popularisées, parmi les plus utilisées on trouve la décomposition aux valeurs singulières (SVD) pour les matrices et les fameuses analyses en composante principale (PCA), décomposition orthogonale aux valeurs propres (POD) [Lum67, Lum81, Sir87] aussi connues sous le nom de décomposition de Karhunen-Loève (KLE) [Loè77]. Ces techniques ont donné lieux à de nombreuses tentatives de réduction de modèles, c'est à dire des méthodes de résolution approchée de problèmes de la physique pour lesquelles on accepte une perte de précision par rapports aux modèles complets classiques (éléments finis, volumes finis,...) en échange d'un coût de calcul et de stockage plusieurs ordres moins cher. Ces méthodes de constructions de modèles d'ordre réduit ont donné des résultats encourageants par des méthodes de projection de Galerkin [Fah01, Ber04, ILD00, QR13] ou d'interpolation [AF08, MNPP09, PR07].

Cependant, ces méthodes sont, pour la plupart, basées sur des décompositions bivariées tout en traitant des problèmes paramétrés. Une autre approche consiste à considérer les paramètres comme une variable, ce qui génère des objets tensoriels qui possèdent potentiellement un grand nombre de dimensions. Les objets tensoriels, dont la croissance est exponentiel n^d en fonction du nombre de dimensions d, représente un enjeu majeur du traitement de données pour le calcul scientifique. L'explosion de la taille des données avec le nombre de dimensions est connu sous le nom de fléau de la dimension. Tucker

a été parmi les premiers en 1966 [Tuc66] à proposer une représentation de faible rang pour les tenseurs. Cette voie a pris une grande importance dans la communauté des mathématiques appliquées depuis le début des années 2000. De nombreux formats sont apparus (Canonique, Tucker, Hierarchique) et sont dotés de méthodes d'approximation. Cette grande richesse a donné lieu à la rédaction de revues de littérature par Kolda et al. en 2009 [KB09] ou plus récemment par Grasedyck et al. [GKT13] et aussi d'un ouvrage très complet sur les tenseurs par Wolfgang Hackbush [Hac14] qui entreprend une description exhaustive des tenseurs et de leur décomposition. Enfin des approches continues, mieux adaptées aux espaces fonctionnels ont été introduites (PGD [CKL13],RPOD [ABR16] ou encore la TT-fonctionelle [GKM16]). Elles apportent des facilités numériques et surtout s'intègrent très efficacement dans la construction de modèles réduits.

Enfin, pour les raisons évoquées précédemment, la construction de modèles réduits pour la mécanique est un sujet de recherche extrêmement actif comme en témoigne les nombreux ouvrages publiés ces dernières années [QMNI, QR13, HRS16, BGW15, CL14]. L'immense majorité de ces méthodes utilise des bases réduites, c'est à dire des bases fonctionnelles de faible rang pour résoudre des équations aux dérivées partielles (de façon discrète) à faible coup. Ces approches fonctionnent convenablement pour les problèmes elliptiques [FN11, DDGS15] mais souffrent d'instabilité dans le cas hyperbolique [BCIS06, ILD00, DM13] (équation de Navier-Stokes par exemple). Ainsi d'autre approches basées sur l'interpolation peuvent être pertinentes si l'échantillonnage paramétrique est suffisamment dense. C'est ainsi que les méthodes d'interpolation empirique proposée par Patera et Maday (EIM [MNPP09]/ DEIM [CS10]) ou l'interpolation sur les variétés de Grassmann par Amsallem et Farhat [AF08, AF11] sont des méthodes construites pour représenter fidèlement la courbure de l'espace d'arrivée des EPD, c'est à dire les non-linéarités.

De ce fait, on se propose, dans ce manuscrit, d'étudier dans une première partie la décomposition de données pour la simulation en mécanique des fluides en offrant un formalisme et des exemples adaptés à la communauté des mécaniciens. La seconde partie de ce document traite de la construction d'un modèle réduit par interpolation dite "physique" après avoir réalisé une étude détaillée (dont on ne peut faire l'économie) de l'écoulement complexe dans une cavité entraînée à haut nombre de Reynolds ([8600-12000]).

Réduction de données.

Dans la première partie du manuscrit la réduction de donnée sera présentée en détails de la théorie à l'implémentation avec de nombreux tests numériques. Les décompositions tensorielles sont pour beaucoup construites en s'appuyant sur des outils bidimensionnels. C'est pourquoi le premier chapitre est consacré à l'étude les décomposition bivariées en introduisant au passage un certain nombre de notions nécessaires par la suite. On s'intéresse ici principalement à la décomposition de matrice par SVD. Toute matrice $A \in \mathbb{R}^{n\times m}$ admet une décomposition de la forme $A = U\Sigma V^\intercal$ ou $U \in \mathbb{R}^{n\times n}$ et $V \in \mathbb{R}^{m\times m}$ sont des matrices orthogonales et $\Sigma \in \mathbb{R}^{n\times m}$ est nul partout sauf les termes diagonaux qui sont les valeurs singulières. Cette décomposition permet une approximation tronquée dont l'erreur est optimale (Th. d'Eckart-Young) et connue. Il sera ensuite montré que la POD constitue en fait une généralisation au cas fonctionnel de cette approche, même si l'algorithme diffère largement. La POD lorsqu'elle est tronquée offre une approximation et une base $(\{a_k\}, \{\phi_k\})_{k=1}^R$ de rang fini R de toute fonction bivariée de la forme

$$f(x,t) = \sum_{k=1}^{\infty} a_k(t)\phi_k(x) \approx \sum_{k=1}^{R} a_k(t)\phi_k(x).$$

Enfin, on verra que la célèbre PGD peut être dégradée en une simple méthode d'approximation qui pour le cas bivarié produit rigoureusement les mêmes résultats que la POD puisqu'elles sont formellement équivalentes. Toutes ces méthodes sont mises en oeuvre numériquement et de nombreux exemples numériques sont proposés.

Les chapitres suivant ont pour but d'offrir une présentation générale des méthodes de réduction de données tensorielles en vue d'une application à la mécanique. Le fléau de la dimension génère des objets tensoriels dont la croissance est exponentielle n^d en fonction du nombre de dimensions d. Ici, un changement de paradigme s'opère puisqu'on propose des représentation approchées dont la taille varie linéairement avec d. Pour le traitement des données tensorielles, une vaste littérature a émergé ces dernières années dans le domaine des mathématiques appliquées. Afin de faciliter leur utilisation dans la communauté des mécaniciens et en particulier pour la simulation en mécanique des fluides, ce manuscrit présente dans un vocabulaire rigoureux mais accessible les formats de représentation des tenseurs et propose une étude détaillée des algorithmes de décomposition de données qui y sont associées. L'accent est porté sur l'utilisation de ces méthodes, aussi une bibliothèque de calcul `pydecomp` développée est utilisée pour comparer l'efficacité de ces méthodes sur un ensemble de cas qui se veut représentatif. Finalement 4 méthodes (et leurs variations) émergent de cette étude. La liste suivante présente l'interprétation fonctionnelle de ces méthodes et leurs principales caractéristiques pour une fonction $f(x_1, ..., x_d)$ discrétisée sur une grille cartésienne régulière de dimension n^d ou r est le rang de troncature typique d'une dimension.

Format canonique $f(x_1, ..., x_d) \approx \sum_k \prod_{i=1}^{d} X_i^k(x_i)$.
Elle est obtenu par un algorithme d'enrichissement successif de l'approximation type PGD/ALS. Il offre un coût de stockage linéaire en d mais s'avère numériquement inefficace comparé aux autres formats. La convergence n'est pas assurée.

Format de Tucker $f(x_1, ..., x_d) \approx \sum_{k_1} \cdots \sum_{k_d} w_{k_1,...,k_d} \prod_{i=1}^{d} X_i^k(x_i)$.
ST-HOSVD est le meilleur algorithme d'approximation pour $d < 4$. Il offre un coût de stockage quasi linéaire en d. La convergence est assurée avec une erreur quasi-optimale.

Recursive-POD $f(x_1, ..., x_d) \approx \sum_{k_1}^{R_1} \cdots \sum_{k_{d-1}}^{R_{d-1}(r_1,...,r_{d-2})} X_1^{r_1}(x_1)...X_d^{(r_1,...,r_{d-1})}(x_d)$.
Ce n'est pas un format mais plutôt une généralisation récursive de la POD. Cette structure en arbre ne permet pas l'orthogonalité de la base mais autorise une troncature facile. Le coût de stockage est assez difficile à estimer mais la convergence est bonne pour les fonctions régulières. Numériquement, elle s'avère moins efficace que TT et ST-HOSVD.

Tensor Train $f(x_1, ..., x_d) \approx \sum_{k_1,...,k_{d-1}} G_1(x_1, k_1)G_2(k_1, x_2, k_2) \cdots G_d(k_{d-1}, x_d)$.
C'est une méthode récente [Ose11] qui permet une implémentation facile et très efficace lorsque $d \geq 5$. Ce sous cas des formats hiérarchiques possède un coût de stockage linéaire en d. Le défaut principale est l'orthogonalité partielle des tenseurs transfert (G_i).

Toutes ces méthodes produisent des décompositions qui sont analysées dans le 4ème chapitre. La partie numérique est réalisée avec la bibliothèque `pydecomp` qui a été développée pour proposer une solution de compression de donnée pour la mécanique des fluides au sein du laboratoire. Réalisée en python (langage de programmation libre), elle autorise la lecture de nombreux types de données et fichiers grâce aux nombreuses bibliothèques libres. Pour preuve, des données expérimentales ont été traitée aussi efficacement que des simulation massivement parallèle du logiciel de simulation numérique des fluides `notus` développé au laboratoire I2M,TREFLE.

Proposition d'un modèle réduit pour écoulements complexes. La seconde partie de ce manuscrit met en avant l'étude de l'écoulement dans une cavité entraînée singulière à haut nombre de Reynolds. Cet écoulement propose une physique très riche qui doit être étudiée en amont de la construction de modèle réduit. Le modèle d'ordre complet utilisé ici a été construit afin d'assurer une précision maximum (schéma compact d'ordre 6). Il a permis la mise en évidence d'une série de bifurcations de Hopf et de caractériser l'écoulement dans la gamme $Re \in [8000, 12000]$. Il apparaît au cours de cette étude que l'extrême sensibilité du problème requiert toutes les précautions pour déterminer les Re critiques. En particulier, une excitation artificielle peut être requise pour déclencher le cycle limite. Cette étude est enrichie par l'utilisation de la décomposition orthogonale aux valeurs propres (POD). On observe dans les modes des propriétés physiques en accord avec les autres types d'analyses. Les modes spatiaux en particulier mettent en évidence les structures et leurs dimensions tout en étant corrélés avec les modes temporels. Il est d'ailleurs intéressant de noter que ces derniers vont par paires qui capturent les fréquences de vibrations principales deux à deux.

Ces nombreuses observations nous permettent de construire un modèle réduit que l'on peut qualifier de "physique". Il diffère notablement des développement récents pour les modèles d'ordre réduit. Ce modèle réduit simple est basé sur la mise à l'échelle des fréquences d'oscillation (*time-scaling*) et des techniques d'interpolation classiques (Lagrange, splines,...). En effet des observations expérimentales [FKE98] et numériques ont mis en évidence un lien entre le nombre de Strouhal[2] et le nombre de Reynolds. Une loi du puissance a permis de les relier. Ainsi pour éviter le phénomène de battement que l'on observe lors d'une interpolation directe entre deux signaux oscillants, on propose une méthode d'interpolation par mise à l'échelle en temps dite "*time scaling*". Le modèle réduit ainsi construit permet d'obtenir le champ de vorticité pour un Re cible à partir de quelques Re donneurs (3 ou 4) pour tous les pas de temps. Les résultats obtenus sont très satisfaisant avec une erreur relative de l'ordre de 10^{-4}. L'application aux modes POD semble prometteuse mais quelques difficultés sur la reconstruction de la phase subsistent.

Conclusion et perspectives.

Ce manuscrit propose une revue aussi globale que possible des méthodes de décomposition de données et apporte des recommandations d'utilisation basée sur les résultats numériques obtenus grâce à la bibliothèque de calcul `pydecomp`. Un modèle réduit par interpolation "time scaling" a été construit avec succès pour des écoulements complexes grâce à une étude détaillée des bifurcations de Hopf subies par l'écoulement dans une cavité entraînée.

Des travaux d'amélioration `pydecomp` permettront l'utilisation d'architectures parallèles et aussi de techniques avec évaluations partielles (blackbox). Concernant les modèles réduits, nous avons commencé à explorer la branche des modèles réduits par projection de Galerkin. Dans ce travail nous essayons d'apporter une nouvelle méthode de stabilisation pour palier au problème bien connu d'instabilité des ROM Galerkin-POD. L'intégration des techniques d'interpolation adaptée aux EDP non linéaires (EIM, variété de Grassmann) pour la construction est aussi à l'étude.

[2]Le nombre de Strouhal caractérise la fréquence d'oscillation d'un écoulement. Il est défini par $\mathrm{St} = fD/U_\infty$ ou f est la fréquence, D une longueur et U_∞ un vitesse caractéristiques.

Contents

Introduction **1**

I Data decomposition 7

1 Bivariate decompositions **13**
1.1 Singular Value Decomposition . . . 14
1.2 Proper Orthogonal Decomposition . . . 17
1.2.1 Building the POD . . . 17
1.2.2 Discussion on the POD variations . . . 21
1.2.2.1 Standard POD . . . 22
1.2.2.2 Snapshots POD . . . 22
1.2.3 SVD $\simeq$ POD . . . 23
1.2.3.1 Numerical implementation of snapshots L^2 POD . . . 24
1.3 Proper Generalized Decomposition . . . 25
1.3.1 Constructing a bivariate *a posteriori* PGD . . . 26
1.3.1.1 An Enrichment Process . . . 26
1.3.1.2 Fixed point algorithm . . . 27
1.3.2 Equivalence of PGD with POD/SVD for bivariate decomposition . 28
1.3.2.1 Power iteration method . . . 29
1.3.2.2 Connection between the methods . . . 30
1.4 Numerical experiments . . . 32
1.4.1 Synthetic data . . . 33
1.4.2 Image compression by decomposition . . . 36
1.4.3 Physics problem data decomposition . . . 39
1.4.3.1 Data decomposition of a singular lid driven cavity flow . . 39
1.4.4 Numerical issues and proposed improvements . . . 41

2 Tensors and their approximation in the most common formats **45**
2.1 Some basic tensor features . . . 46
2.1.1 Tensor spaces . . . 46
2.1.2 Overview of tensors of $\mathbb{R}^{n_1 \times \cdots \times n_d}$ i.e. multi-way arrays . . . 47
2.2 Tensor Formats . . . 51
2.2.1 Full format . . . 52
2.2.2 Canonical format $\mathcal{C}_r$ . . . 52
2.2.3 Tucker format $\mathcal{T}_k$ . . . 53
2.2.4 Hierarchical Tucker format $\mathcal{H}_k$ . . . 54
2.2.4.1 Tensor Train format . . . 57
2.2.4.2 Extended TT format . . . 59
2.2.5 Conclusion on tensor formats . . . 60
2.3 Tensor decomposition . . . 61

2.3.1 CP decomposition . . . 61
2.3.1.1 Existence of a low rank approximation in $\mathcal{C}_r$ . . . 63
2.3.1.2 Computing the CP decomposition : the ALS algorithm . . . 63
2.3.2 Tucker decomposition . . . 64
2.3.2.1 HOSVD . . . 66
2.3.2.2 ST-HOSVD . . . 68
2.3.3 Tensor Train decomposition . . . 71
2.3.3.1 TT-SVD . . . 71
2.3.3.2 Sampling algorithms for high dimensional TT . . . 73
2.3.4 Hierarchical Tucker decomposition . . . 75

3 Multivariate problem decomposition 77
3.1 Proper Generalized Decomposition . . . 78
3.1.1 Theoretical background of the PGD . . . 78
3.1.2 A Galerkin PGD algorithm for d parameter functions . . . 80
3.1.3 PGD and CPD . . . 82
3.2 The Recursive-POD (R-POD) . . . 82
3.2.1 Introductory example : R-POD on a 3D field . . . 82
3.2.2 R-POD: general case . . . 84
3.3 Functional tensor decomposition . . . 89
3.3.1 Functional Tucker decomposition . . . 89
3.3.2 Functional-TT . . . 91

4 Numerical Experiments 95
4.1 A decomposition library . . . 95
4.2 Synthetic data comparison . . . 100
4.3 Decomposition methods on numerical data . . . 107
4.3.1 A scalar simulation : 2D lid driven cavity at high Reynolds number 107
4.3.2 Experimental data : droplets evaporation . . . 112
4.3.3 A vectorial simulation : breaking wave . . . 115
4.4 Standard interpolation techniques for reduced basis ROM . . . 119

Summary and conclusion on data decomposition 125

II Complex fluid dynamics and Reduced Order Modeling 129

5 Complex flow analysis using decomposition 133
5.1 Singular lid driven cavity: analysis of flow behavior with high accuracy direct simulation . . . 134
5.1.1 Governing equations and numerical methods . . . 135
5.1.2 Dynamics of singular LDC flow . . . 136
5.1.3 Vorticity dynamics and polygonal vortex in LDC . . . 137
5.1.4 Multiple Hopf bifurcations . . . 138
5.1.4.1 New equilibrium state via stable limit cycle . . . 138
5.1.4.2 Frequency spectrum analysis . . . 140
5.1.5 Numerical sensitivity of the problem . . . 140
5.1.5.1 Computational bifurcation analysis: Is there a universal critical Reynolds number for primary bifurcation? . . . 142
5.2 POD analysis . . . 143
5.2.1 Analysis through POD modes . . . 144

5.2.1.1 Limit cycle POD modes 144
5.2.1.2 Primary and secondary instabilities POD modes 150

6 Interpolated ROM **157**
6.1 A physics based interpolation method: Time-scaling 158
6.1.1 Need for time scaling 160
6.1.2 Formulation and modeling of ROM 163
6.1.2.1 Computing the initial time-shift (t_0) 164
6.1.3 Time-scaling ROM algorithm 165
6.1.4 Time-shifting ROM applied to the LDC flow 166
6.1.5 Time-scaling ROM applied to the flow past a cylinder 169
6.1.6 POD and time-scaling. 172

Conclusion and perspectives **177**

Bibliography **183**

Included papers **195**

List of symbols and abbreviations

Notations

Variables

$i, j, k, \alpha_i ...$	integers indices
$\mathcal{I}$	multi-index
n, m, N	interger, often number of points
$n_1, ..., n_d$	interger, often number of points according to dimensions
$r, k, \boldsymbol{r}, R$	decomposition rank
d	number of dimension/parameters
$a, b, ...$	scalars in $\mathbb{R}$
$\boldsymbol{x}$	vectors in $\mathbb{R}^n$
$\boldsymbol{X}, \boldsymbol{A}$	matrices in $\mathbb{R}^{n \times m}$
$\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{Y}}, \boldsymbol{\mathcal{T}}, ...$	tensors in $\mathbb{R}^{n_1 \times \cdots \times n_x}$
$\boldsymbol{X}_\mu$	mode μ matricization of $\boldsymbol{\mathcal{X}}$
$\widetilde{\boldsymbol{\mathcal{X}}}$	low rank approximation of $\boldsymbol{\mathcal{X}}$
$f, g, h, u, v ...$	multi-parameter functions

Spaces

$\mathbb{N}$	set of natural numbers
$\mathbb{N}^*$	set of stricly positive integers
$\mathbb{R}$	set of real numbers
Ω	function parameter space
$V, U, \mathcal{U}, \mathcal{V}$	vector spaces
$L^2(\Omega)$	Space of square integrable functions on Ω
$H^1(\Omega)$	Sobolev space
$\mathcal{C}_r$	set of canonical tensors of rank r
$\mathcal{H}_{\boldsymbol{k}}$	set of Hierarchical Tucker tensors of rank $\boldsymbol{k}$
$\mathcal{T}_{\boldsymbol{k}}$	set of Tucker tensors of rank $\boldsymbol{k}$

Decomposition

$\boldsymbol{U}, \boldsymbol{X}$	mode matrices or sequence of modes
$\phi_i, \boldsymbol{\phi_i}$	POD space mode
a_i	POD time mode
σ_i	singular values
λ_i	eigen values
δ_{ij}	Kronecker symbol

Operators

$(\cdot, \cdot)$, $\langle \cdot, \cdot \rangle$	scalar products
$\| \cdot \|$	norm associated with above scalar product
$(\cdot, \cdot)_{L^2(\Omega)}$	L^2 scalar products
$\| \cdot \|_{L^2(\Omega)}$	$L^2(\Omega)$ norm
$\| \cdot \|_F$	Frobenius norm
$\langle \cdot \rangle$	Average operator

$\circ$	outer product
$\otimes$	Kronecker product, tensor product
$*$	Hadamard product
$\odot$	Kathri-Rao product

Physics

ρ	density
μ	dynamic viscosity
ν	kinetic viscosity
ω	vorticity
Re	Reynolds number
St	Strouhal number

Abbreviations

ALS	Alternating Least Square
CFD	Computational fluid dynamics
EVD	Eigen Value Decomposition
HOSVD	Higher Order Singular Value Decomposition
HT	Hierarchical Tucker
NSE	Navier-Stokes Equations
POD	Proper Orthogonal Decomposition
PGD	Proper Generalized Decomposition
ROM	Reduced Order Model
R-POD	Recursive Proper Orthogonal Decomposition
SVD	Singular Value Decomposition
ST-HOSVD	Sequentially Truncated HOSVD
T-HOSVD	Truncated HOSVD
TT	Tensor Train
TT-SVD	Tensor Train SVD decomposition algorithm

Introduction

Context

In the last 50 years, scientific computing has become a central tool in engineering design, especially in the mechanics field. A constant improvement in simulation techniques has accompanied the rocketing computing power embedded in Moore's law[3]. This explosion of CPU power was magnified by the introduction of supercomputers and their massively parallel architectures. Although some slowdown has been observed, this trend will continue, especially with the arrival of breakthrough technologies such as the much awaited quantum computer. Still, the advent of exascale computing has only pushed forward the boundaries of computable problems slightly while raising a series of technical issues. First, supercomputers are really expensive infrastructures that require huge amounts of energy[4]. Second, they produce data so large that storing and transferring data itself has become an issue. For instance a famous 2012 simulation of the observable universe [ABR+12] exemplifies the dizzying proportions taken by numerical simulation. Approximately 5000 computing nodes used 300 TB of memory producing 50 PB of raw data in 10 million hours of computing time of which "only" 500 TB of useful data was finally kept. This kind of data is hard to manipulate and storage is usually performed on magnetic bands making it fairly slow to access. Also, any intent at handling such data, even in small slices, is vain on a personal computer, thus impairing the efficiency of analysis.

Actually, the framework of building numerical models has remained the same across the period of popularization of numerical simulation. This process has been finely tuned, improving gradually the quality and confidence in the simulations. This technology is now massively used in the industry, especially for designing new products that require precise knowledge in fields such as mechanics, thermodynamics, chemistry, electromagnetic fields, etc. In particular, computational fluid dynamics has become a central tool in designing new aircrafts, ranging from global flow around a plane to multiphysics-multiscale combustion inside the jet engine.

Building a direct model, also known as full order model (FOM), usually involves the following steps. First, one needs to select the adequate equations from basic physics laws and define carefully the limits of simulation. Depending on the problem geometry, characteristic sizes and phenomena[5], one chooses the simplest equations set that captures the physics correctly. Then these equations are discretized in time and space while numeri-

[3]Gordon Moore predicted in 1965 that the density of transistors on chips would double every year. After being slightly downgraded to doubling every 18 month, it has been verified from 1975 to 2012. Current trend shows a slowing pace. Still, this exponential growth amounts to a 20 millions factor. Naturally, it corresponds to the computing power gain.

[4]As of june 2018, the largest supercomputer is the Summit at Oak Ridge, USA, with more than 2 million cores it requires 8MW for a peak performance of 122PFlop/s

[5]A typical example in fluid dynamics is Reynolds number $Re = UL/\mu$ which characterize the relative influence of inertia (U is a typical flow velocity and L a typical length) compared with viscosity (μ the kinematic viscosity.)

cal schemes are used to solve the constructed discrete problems. Whether one uses finite differences, finite elements or finite volumes, the problem usually boils down to a linear algebra problem

$$Ax = b$$

where A is a $n \times n$ matrix, x is the unknown vector of size n and b the right hand side term of size n. Here, n is the number of discrete space points that typically range from millions for 2D to billion for high end 3D problems. Moreover, this linear problem has to be solved at each time step, often millions of times, in spite of typically costing $\mathcal{O}(n^2)$ floating point operations. More often than not, if one wants to simulate several interacting physical phenomena, they occur at different time and space scales, meaning that one needs to solve several concurrent problems of this kind. With the figures stated above, it becomes clear direct numerical simulation (DNS) is expensive. Consequently, problems involving to perform such simulations multiple times such as optimisation or control, remain out of reach.

It has spawned a vast body of literature on how to make these simulations more affordable. Among the typical solutions in fluid dynamics, Reynolds averaged Navier-Stokes methods (RANS) and Large eddy simulation (LES) have been very successful at capturing large structures and modeling (with more or less empirical terms) the smaller structures. These solutions however generate a great loss of information as it is impossible to know how the energy dissipation occurs in the small scale structures. To some, extent it prevents relevant simulations in which the interaction of small structures drive large scale behavior i.e. chaotic systems. Many models, in all areas of numerical simulation, have been proposed to reduce the computing cost with the same idea of modeling the most expensive terms of equation while retaining the same basic principles of discretization. We observe that, within this approach, the curse of dimensionality remains the main obstacle to scientific computing development. For instance, let the number of discrete points needed to capture a phenomenon on one dimension be $n = 1000$. Now, if the problem is 3D, the cube is discretized with $n^3 = 10^9$ points. If the phenomenon is actually a dynamic one, time has to be accounted for, which means an additional dimension. The discrete space time is now $n^4 = 10^{12}$ that amounts terabytes of data for double precision real numbers. Additionally, one might want to add a few parameters on which the simulation depends and both the computing time and storage cost become out of reach. Even with very small n, for instance n=2, this kind of difficulty emerges quickly. For example, with $d = 50$ (which is far below computational chemistry requirements), storage cost of $n^d = 2^{50}$ amounts to 9PB if all entries are stored. A tensor is a well suited object for such data representation, it is the discrete representation of multidimensional fields, i.e. an order d tensor of size $n_1 \times \cdots \times n_d$ is filled by sampling a field on a tensor product space $\Omega = [0, 1]^d$ at discrete grid points. The necessity of storing low rank approximate tensors instead of keeping all the entries becomes essential in this context.

Finally, Fig. 1 summarizes the dominant work-flow in scientific computing i.e. physics modeling is followed by discretization techniques that can produce reliable simulation. The introduction of a new paradigm is represented here by tensor decomposition and the following steps of ROM.

In this paragraph we have shown the necessity to move beyond the current dominating paradigm in scientific computing. In this manuscript we will explore two complementary branches that tackle this problem. The first one is data reduction as it is the preliminary steps for most approaches of the second branch, model order reduction. The next paragraph will present the state of the art in these fields.

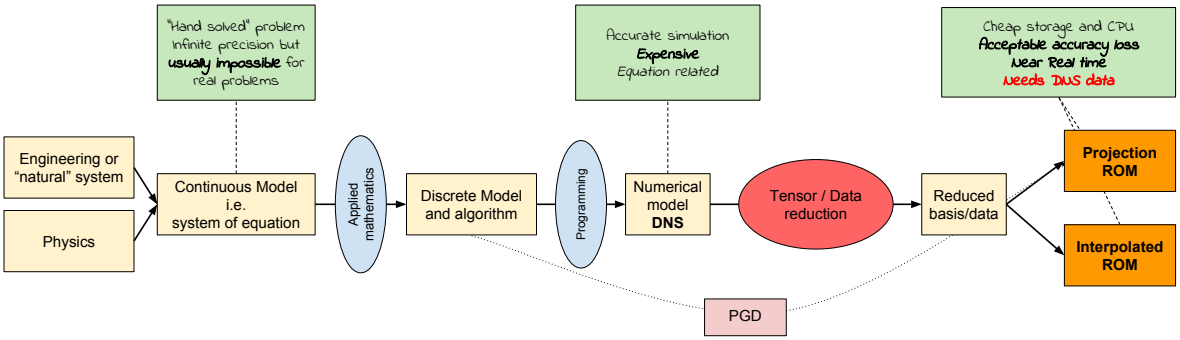


Figure 1: Scientific computing workflow enriched with tensor reduction and reduced order modeling

State of the art

The need for order reduction is as old as numerical simulation, for instance matrix analysis techniques such as eigen value decomposition or singular value decomposition (SVD) have been used in the past centuries to capture structure in complex matrices. It turns out that the bivariate decomposition methods in principle equivalent to SVD but complies with their field formalism. Actually, they have been rediscovered many times in various fields: it is known as principal component analysis (PCA) in statistics [Pea01, Hot33], Karhunen-Loève expansion (KLE) in probability theory [Loè77] or proper orthogonal decomposition (POD) in fluid dynamics [Lum81, Sir87]. These methods, by themselves, provide a decomposition that can be truncated with optimality results [EY36] and reflect the physics of the problem studied. The first wave of reduced order models (ROM) in mechanics is a consequence of POD. Indeed this decomposition provides, among the many possible bases [IR98], an orthogonal basis of the functional space in which the solution problem lives. Consequently, many attempts at building Galerkin projection ROM on these reduced bases from the 1980s onward [Sir87, DKKO91, CVVI98, Fah01, Ber04] with modest success. Indeed, in this approach, the weak form EDP is solved against test function in the selected basis. In order to decrease the size of the problem, one has to truncate the basis to a relatively small rank which means, in the case of fluid flows, that the small structures are lost[6]. Yet these structures correspond to turbulence and viscosity for high Re number which role is to dissipate energy. This is why Iollo et al. [ILD00] showed this approach is inherently unstable. Thereafter, many turbulence models and stabilization techniques have been proposed in the context of ROM [BBI09, ANR09, IW14] and continue to be an active field of research [BGW15, LCLR16, SR18]. Generally, this approach has motivated substantial amount of work that has been crystallized in various books in recent years [QR13, QMNI] under the name Reduced Bases (RB), popularized in the early 2000s by Patera, Maday and coauthors [MMPR00, PRV+02]. The efficiency of these methods can be characterized by the Kolmogorov N-width. This concept with many other tools for RB is detailed in Hesthaven, Rozza and Stamm book [HRS16].

Another approach to build ROMs is to interpolate in the parametric space of arrival of PDEs. Indeed, one can build a set of data for several parameters with FOM and later ask the database for a point that was not previously sampled and interpolate to this new location. Given the large size of the full data, brute force multidimensional interpolation through standard techniques (Lagrange, Splines,...) is not an option. Additionally,

[6] We will see in chapter 1 that the smaller structures tend to contain less energy than the larger ones.

dynamic systems, even if they are relatively similar, may produce beat phenomenon. Consequently, numerous methods were proposed to build such ROMs. Among the most successful, the empirical interpolation method (EIM) has been introduced in 2004 by Barrault et al. [BMNP04]. The idea here is to sample the parametric space by greedy algorithm with very efficient reconstruction property for non-linear problems [MNPP09]. It was later adapted into a discrete version (DEIM) using POD modes as a basis instead of samples [CS10]. This process is also extensively used for hyperreduction of models. The solution of PDEs live in a Grassmann manifold that is not flat, this is the profound reason why simple minded interpolation is doomed. Thus, Amsallem and Farhat proposed a Grassmann manifold interpolation ROM [AF08, ACCF09, AF11, AZW15] that have proved very efficient for aeroelasticity applications. It has spawned a family of methods recently reviewed by Mosquera's thesis [Mos18] in which the idea is to project the solutions from the manifold to a tangential plane to perform the interpolation (by standard means) and then return to the manifold.

All these ROM methods aim at providing quickly data on multidimensional fields, for example, to build virtual charts. The number of parameters may be large and the associated data lies on tensor (product) space of these parameters. As we have already seen, this structure in itself produces exponential amounts of data. One way to tackle this difficulty is to use separated representations. The proper generalized decomposition PGD [CLA+09, CALK11, CL14] exactly intends to solve PDEs by directly building a separated solution. It originates from Ladevèze LATIN method [CL93, LPN10] and was found to be very efficient on elliptic equations with numerous variations [LN03, CLB+17, FN11]. This method can be degraded into a data approximation technique using the simplest PDE $f = u$ which makes it a canonical tensor decomposition method [Nou10, Nou15] that is roughly equivalent to alternating least squares algorithms (ALS).

Hitchcock [Hit27] usually considered to have introduced tensor decomposition in 1927. But, it is Tucker [Tuc66] that popularized the subject in the 1960s, followed by Carroll and Chang [CC70] and Harshmann [Har70] in 1970. As for the bivariate decomposition, much of the research happened independently in several fields starting by psychometrics and chemometrics. A complete history is available in Kolda and Balder review paper [KB09]. This large overview of tensor formats includes canonical format ([CC70, Har70]) and Tucker format with the associated decomposition methods. The former has received dwindling interest due to poor numerical performance. Tucker format was at the center of attention since DeLathauwer paper in 2000 [DDV00] which proposed an efficient approximation strategy, the Higher Order SVD (HOSVD) followed by HOOI [dLdMV00]. More recently, he coauthored Vannieuwenhoven ST-HOSVD [VVM12] that improved significantly the computing time. The early 2010s have seen the introduction of formats that overcome the exponential growth of the core tensor in Tucker format. Oseledets proposed the tensor train (TT) format [OT09, Ose11, Kho11] , also known as matrix product state (MPS), together with its decomposition algorithm. The storage cost of this format is linear in d allowing tensorization of data, i.e. the method is so efficient at handling large d that a new strategy consists in increasing artificially the number of dimensions. To do so, one may need to rely on partial evaluations of the target field, `TT-DMRG-cross` performs this task [OT10, ODS18]. This approach is also known as blackbox algorithms [BGK10] in the context of hierarchical tensors (HT) developed by Grasedyck, Kessner and Tobler [Gra10, KT11]. HT actually incorporates all previously mentioned formats and approximations into a general d-linear format. These recent developments have been reviewed in [GKT13] while an extensive mathematical analysis of tensors and their approximation is given in Hackbush's book [Hac14]. A selection of publicly available libraries will be discussed in detail in chapter 4.

Finally, these formats have been extended to the continuous framework as they are often used to separate data representing functions. A functional TT was proposed by Bigoni and Gorodetsky [BEkM16, Gor16] while many approaches now consider n-way array tensors and multivariate function as a single object [Hac14, Nou15, FHN15]. Finally, a Recursive POD (RPOD) was proposed by Azaiez et al. [ABR16].

Objectives

It is clear from the scientific context that a new paradigm in scientific computing is needed. We have seen that the data produced by standard methods has become so large that a new field of data decomposition has emerged. It was shown in the literature review that most of these methods have been devised for tensors and they mostly rely on extensions of matrix decomposition and approximation techniques. It is very interesting to study these methods for CFD simulation and translate them to the continuous framework. It was highlighted that these works have been mostly conducted separately. Review articles mostly present the numerous approaches and associated work without comparing numerical results, save basic test functions. To the best of my knowledge, there exist no comprehensive comparison of the data decomposition techniques that have been developed in the last decade, especially the rare continuous versions. Additionally, the reviews of literature or books published in recent years take place in the applied mathematics field which implies few examples of actual large simulation.

Hence we define our first objectives. (a) offer a comprehensive synthesis of decomposition methods from bivariate to multivariate data including both tensors and functions frameworks. (b) provide an extensive numerical analysis and comparison of these methods. To do so, a computing library has to be programmed. Additionally, we intend to contribute to the dissemination of this approach in the CFD community which implies an adequate presentation.

Data handling is not the only difficulty incurred by modern scientific computing. We have seen that many problems are out of reach within the current paradigm. This is why many have tried to build reduced order models, with various level of success for CFD applications. Indeed, instability of PODG-ROM is a bottleneck for CFD applications since Navier-Stokes equations are hyperbolic. Interpolation on the parametric space is no trivial task and recent approaches such as Grassmann manifold interpolation involve complex numerical and mathematical setup to overcome these difficulties. Finally some problem are so sensitive that direct numerical simulation may require special care to obtain acceptable accuracy.

In this context, we propose (c) to study extensively a typical CFD test case, namely, the 2D singular lid driven cavity flow. It is known to present particularly complex features (at high Re) in spite of its simple geometry and clear boundary conditions. To do so, we will use both standard tools and decomposition tools mentioned above. In the end, the added knowledge should spur (d) new ROM concepts based on physics observation.

We can summarize objectives (a-d) that have been pursued in this thesis, as follow:

- Study tensor and multivariate function decomposition for fluid dynamics application using a formalism and numerical experiments adapted to CFD community. Build a computing library that provide the necessary tools.

- Explore ROM for complex flows that requires careful preliminary study. This analysis should lead to physics based ROM.

Manuscript layout

The manuscript is divided into 2 parts. Part I deals with data decomposition among which, Chapter 1 provides a detailed presentation of bivariate decomposition techniques and points out to the fundamental equivalence of these methods. Once these basic tools have been studied we go through tensor decomposition into format with roughly linear storage cost in chapter 2. Next, the multivariate problem decomposition is treated in chapter 3 i.e. we construct approximated functions by separated sum. Finally, chapter 4 proposes comprehensive numerical tests to compare the performance of each technique.

In the second part of this manuscript, the generation of Reduced Order Models (ROM), based on the previous decomposition techniques and constructed bases, is addressed. Through chapter 5, a comprehensive study of a complex flow is conducted both with standard analysis tool (Fourier transform, linear regressions,...) and through decomposition (POD). This analysis highlights the complexity of building ROM and leads us to propose a "physics" based interpolation ROM through the so called *time-scaling* in chapter 6.

Part I

Data decomposition

Data analysis has become a pressing issue in recent years. Indeed with the advent exascale computing and big data technologies, we are under a constant pressure to analysis this data. This is particularly true in the field of computational mechanics. Simulations are growing ever larger, routinely producing petabytes of data through thousand hours process. Then the storage and post-processing of these results has become problematic. Consequently a vast number of techniques to address these problem has been proposed. A common way to ease data storage and post-processing is to use tensors. Tensors are mathematical objects that can be visualized as d-way arrays of typical size n^d. The goal of this first part is to propose numerical algorithm and data layout to d-linear storage cost.

Hitchcock [Hit27] usually considered have introduced tensor decomposition in 1929. But, it is Tucker [Tuc66] that popularized the subject in the 1960s followed by Carroll and Chang [CC70] and Harshmann [Har70] in 1970. As for the bivariate decomposition, much of the research happened independently in several fields starting by psychometrics and chemometrics, a complete history is available in Kolda and Balder review paper [KB09]. Tucker format has received a lot of attention since DeLathauwer paper in 2000 [DDV00] which proposed an efficient approximation strategy Higher Order SVD (HOSVD) followed by HOOI [dLdMV00]. HOOI is known to be the method that provides the lowest approximation error but is rather slow to converge. More recently Vannieuwenhoven proposed a new truncation strategy for HOSVD called ST-HOSVD [VVM12] that improved significantly the computing time. However efficient these methods are, the main drawback is the core tensor that stores the weight of each modes combination. The early 2010s have seen the introduction of formats that overcome the exponential growth of the core tensor in Tucker format. Oseledets introduced the tensor train (TT) format [OT09, Ose11, Kho11] , also known as matrix product state (MPS) [VC06],together with its decomposition algorithm. The storage cost of this format is linear in d allowing tensorization of the data. For instance the method is so efficient a handling large d that a new strategy consists in increasing artificially the number of dimensions. To do so, one may need to rely on partial evaluations of the target field, `TT-DMRG-cross` performs this task [OT10, ODS18, BEkM16]. This approach is also known as blackbox algorithms [BGK10] in the context of hierarchical tensors (HT) developed by Grasedyck, Kessner and Tobler [Gra10,KT11]. HT actually incorporates all previously mentioned formats and approximations into a general d-linear format. These recent development have been reviewed in [GKT13] while an extensive mathematical analysis of tensors and their approximation is given in Hackbush's book [Hac14]. The many computing libraries are publicly available, to name only a few the Tensor Toolbox by Bader et al. [BKO17] provides a rather general API, Kressner et al. propose a HT format library `htucker` [KT13] and Oseledets offers a TT library [ODS18].

We have so far only discussed d-way array tensors decomposition, but functions of several variables defined on product spaces e.g. $\Omega = [0,1]^d$ can be viewed as tensors. This is particularly explicit for discrete representation of functions that usually equate to evaluation on a discrete grid. Still considering that the data originates from functions is interesting. Indeed, applied mathematics provides many properties and most importantly the sampling does not have to be regular. That means that a discretization strategies that fit the complexity of the studied problem can reduce drastically the size of the data while being processed correctly by the decomposition algorithms. Thus these formats have been extended to the continuous framework. A functional TT was proposed by Bigoni and Gorodetsky [BEkM16,Gor16] while many approach now consider n-way array tensors and multivariate function as a single object [Hac14, Nou15, FHN15]. This approach is natural for CFD specialists as we manipulate discretized operators and field constantly. Before

attempting to separate multivariate problems, a vast body of work adressed the bivariate decomposition. In the context of fluid dynamics, this approach is known as proper orthogonal decomposition (POD). It is usually attributed to Kosambi [Kos43] and later popularized in fluid dynamics by Lumley [Lum81] who proposed detailed analysis of flows through the standard POD while Sirovich later proposed the "snapshots" method [Sir87] which is much more adopted to numerical simulations outputs. This method was actually extended to problems of several variables recently by Azaiez et al. [Aza] who proposed a recursive POD algorithm. POD is actually equivalent to the proper generalized decomposition (PGD) [CL14, CKL13] for bivariate approximation problem. This last techniques actually provides a general iterative algorithm to compute separated multivariate solutions to PDEs that can be degraded into a decomposition algorithm.

Separating data, a problem overview

In this short section, a representative example of the problem we are studying in this section is given in order to clarify the objective.

It is assumed that the following field is known (at least in a discrete representation),

$$\begin{array}{rccc} f: & \Omega \subset \mathbb{R}^d & \to & E \subset \mathbb{R} \\ & \mathbf{x} & \longmapsto & f(\mathbf{x}) \end{array}$$

where $\Omega = \prod_{i=1}^{D} [a_i, b_i]$.

We are seeking a separated representation[7] of f so that

$$f(x_1, ..., x_d) = \sum_{m=1}^{\infty} w_m \prod_{i=1}^{d} X_i^m(x_i) \tag{0.0.1}$$

Where w_m is the weight associated with the m-th member of the sum and X_i^m is the m-th i-mode function.

It is called a separated representation since the function is represented by a combination of univariate functions. Each of this function is normalized and orthogonal to the others i.e. $\forall 1 \leq i \leq D, m \in \mathbb{N}$

$$\begin{array}{rcl} \|X_i^m\|_{L^2} & = & 1 \\ < X_i^m, X_i^n >_{L_2} & = & \delta_{mn} \end{array}$$

Knowing that as m increases, the weight w_m tends toward 0. Then we choose a criterion and stop adding terms after R terms. The finite sum approximation of f with a rank R reads

$$f(x_1, ..., x_d) \approx \tilde{f}(x_1, ..., x_D) = \sum_{m=1}^{R} w_m \prod_{i=1}^{d} X_i^m(x_i) \tag{0.0.2}$$

One should note that this process is highly compatible with discrete representation. In this context the multivariate function is replaced by an order D tensor $\mathfrak{T} \in \mathbb{R}^{n_1 \times \cdots \times n_D}$. If a few modes containing the most relevant information is conserved, then an approximation is obtained. Fig 2 shows a visual of this decomposition.

The goal of this first part of the manuscript is to provide the tools to build this kind of decomposition and more sophisticated ones. Once the theoretical aspect is complete, a

[7] We will see that this representation is known as canonical format.

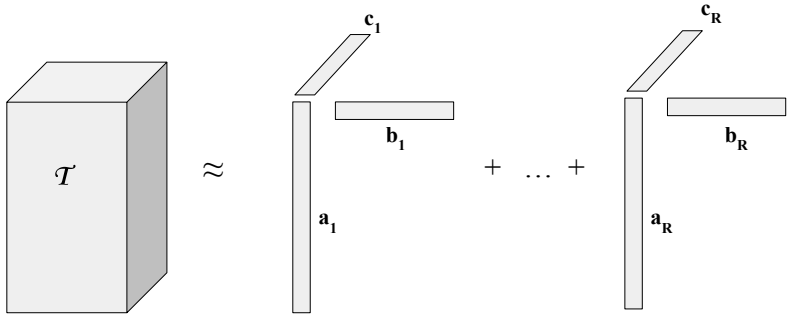


Figure 2: Separated approximation of an order 3 tensor.

comprehensive discussion is proposed with numerous numerical test to guide decomposition methods used. Overall, it is intended to bridge the gap between applied mathematics community and computational fluid dynamics in this specific area. This first part is organized as follow. First the bivariate decomposition problem is studied in chapter 1 as it is the building block of many higher order methods. Then tensor decomposition are presented in chapter 2 followed by the multivariate function decomposition problem in chapter 3. Finally, a comprehensive numerical study is presented in chapter 4 with discussions on the comparative advantages of some methods and an analysis of the modes meaning whenever possible. The python library pydecomp that was developed during this thesis has been used for this purpose

Chapter 1

Bivariate decompositions

Contents

1.1 Singular Value Decomposition 14
1.2 Proper Orthogonal Decomposition 17
1.2.1 Building the POD 17
1.2.2 Discussion on the POD variations 21
1.2.3 SVD $\simeq$ POD 23
1.3 Proper Generalized Decomposition 25
1.3.1 Constructing a bivariate *a posteriori* PGD 26
1.3.2 Equivalence of PGD with POD/SVD for bivariate decomposition 28
1.4 Numerical experiments 32
1.4.1 Synthetic data 33
1.4.2 Image compression by decomposition 36
1.4.3 Physics problem data decomposition 39
1.4.4 Numerical issues and proposed improvements 41

In order to give the full picture of data reduction technique, it is crucial to begin with bivariate problems. Indeed almost all multivariate techniques result from these 2D versions. Bivariate decomposition techniques were mainly theoretical at the time they were proposed in the first half of the 20$^{\text{th}}$ century [Pea01, Hot33], manual computations limited the size of the studied problems. But the numerical analyis and properties have been studied in details with emerging spectral theory [EY36, Kos43]. Actual implementations were carried on later in the second half of the 20$^{\text{th}}$ for fluid dynamics systems [Lum67, BHL93, Sir87]. 2D data reduction techniques are well understood and have been applied to the widest variety of problems in the last 20 years either to compress data or build reduced order model [Fah01, NAM+03, AF08].

In order to offer a broader view of the possible uses of bivariate decomposition, Fig. 1.1 proposes a schematic view of bivariate problem Reduced order modeling methods. The decomposition techniques presented in this section form the base material of many ROMs. They are organized as follow. The dashed black line shows the dichotomy between the continuous approach[1] and the discrete one. Then the orange dashed line separates

[1] These approaches are conceptually continuous but their implementations requires discrete description of the continuous space including grids, discrete operators,...

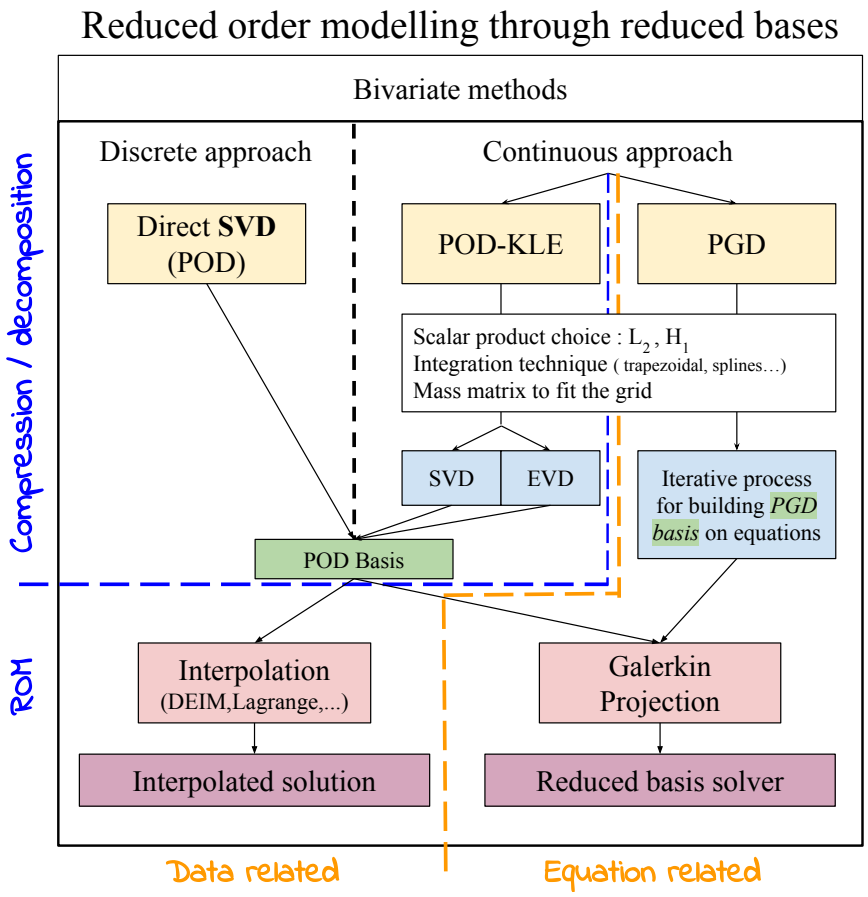


Figure 1.1: Synthetic view of the procedures described in chapter 1 for model order reduction of bivariate PDEs. The vertical arrows describe the work flow of these techniques and the dotted lines highlight the conceptual differences between them.

the techniques that only apply to data –namely SVD and POD– from the PGD which is usually used on the equation itself but can be degraded into a data decomposition method. Finally, the blue dashed line emphases the data compression nature of the POD and SVD while noting the possibility to obtain a ROM through the obtained basis as shown in the lower part of the diagram.

This chapter is organized as follow. First, section 1.1 describes the singular value decomposition (SVD) as it is both the simplest and the most used methods in this manuscript. Then, in section 1.2, proper orthogonal decomposition (POD) is built and analyzed, its discrete version is presented as well. Section 1.3 provides a 2D construction of the proper generalized decomposition (PGD), and how degrading it turns the method into a decomposition that is in fact an iterative algorithm to compute POD. Finally, section 1.4 exhibits a series of numerical tests and application of these methods as well as insight on their relative advantages. All along these sections, a particular focus is given on the numerics associated with the studied methods.

1.1 Singular Value Decomposition

The Singular Value Decomposition (referred as **SVD**) is a generalization of the eigenvalues decomposition for rectangular matrices. Among its many applications it can be seen as a discrete version of the POD.

Theorem 1.1.1 (Singular Value Decomposition [PS14]). *For any matrix $A \in \mathbb{R}^{m\times n}$, there are orthogonal matrices $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ so that*

$$A = U\Sigma V^\intercal$$

where Σ is a diagonal matrix of size $n \times m$ with diagonal elements $\sigma_{ii} \geq 0$.

Hereafter, it is assumed that the singular values are ordered decreasingly i.e. if $i < j$ then $\sigma_{ii} \geq \sigma_{jj}$. The SVD is not unique since the signs of U and V may vary.

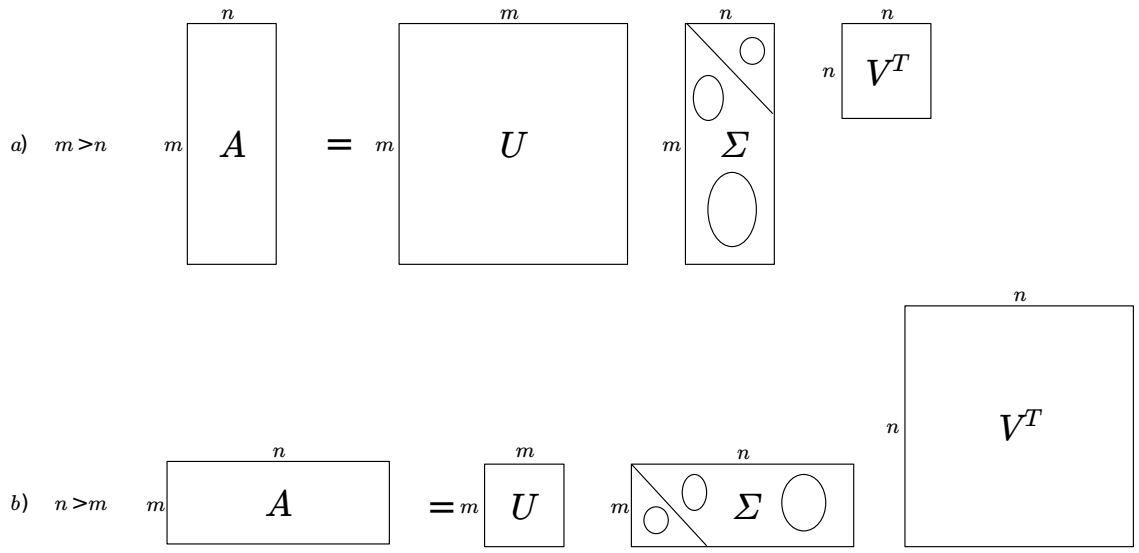


Figure 1.2: Singular Value Decomposition two configurations

One should note from figure 1.2 that a part of U in case a) and V in case b) only serves a dimension match without entering calculation of A, then the SVD reads for case a)

$$A = [U_1, U_2][\Sigma_1, 0]^\intercal V^\intercal = U_1\Sigma_1 V^\intercal$$

Let $\text{rank}(A) = r$ then for $k > r$, $\sigma_k = 0$. The SVD of A can be written as sum

$$A = \sum_{i=1}^{r} \sigma_i U_i V_i^\intercal$$

where σ_i are the diagonal entries of Σ and U_i and V_i refer to the columns of U and V respectively. Then $\|A\|_2 = \sqrt{\sum_{i=1}^{r} \sigma_i^2}$ leads to the optimality theorem proven by Eckart and Young in 1936 [EY36].

Theorem 1.1.2 (Eckart-Young). *Let $k < r$ and $A_k = \sum_{i=1}^{k} \sigma_i U_i V_i^\intercal$ where the singular values are ordered decreasingly then*

$$\min_{\text{rank}(B)=k} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1} \tag{1.1.1}$$

Remark (Link with the eigenvalue decomposition). Singular and eigenvalues are closely linked. Let $A \in \mathbb{R}^{m\times n}$ with $m > n$. $A^\intercal A = V\Sigma^\intercal\Sigma V^\intercal \longrightarrow AA^\intercal = V\Sigma_1^2 V^\intercal$. Then the eigenvalue problem of $A^\intercal A$ is equivalent to the right singular value problem of A with $\lambda_i = \sigma_i^2$ and the eigenvectors of $A^\intercal A$ are collinear to A's right singular vectors $\boldsymbol{v}_i$. The same applies to $\boldsymbol{u}_i$ and the eigenvectors of $A^\intercal A$.

Remark (Solving least square minimization problem with the SVD). The classical least square minimization problem i.e. find x^n of minimum Euclidean norm that reaches the minimum of $\|b - Ax\|_2$ for $A \in \mathbb{R}^{m\times n}$, is solved by the SVD and the Monroe-Penrose pseudo inverse of A (see [PS14]).

The main information contained in the Eckart-Young theorem is that the truncated SVD (see Fig. 1.3) i.e. only keeping the k dominant modes gives an optimal approximation of rank-k of the matrix A which rank is $r \geq k$. It means that the k first singular vectors form the optimal projection basis of size k that reads as follow,

$$A \approx A_k = \sum_{i=1}^{k} \sigma_i \boldsymbol{u}_i \otimes \boldsymbol{v}_i \tag{1.1.2}$$

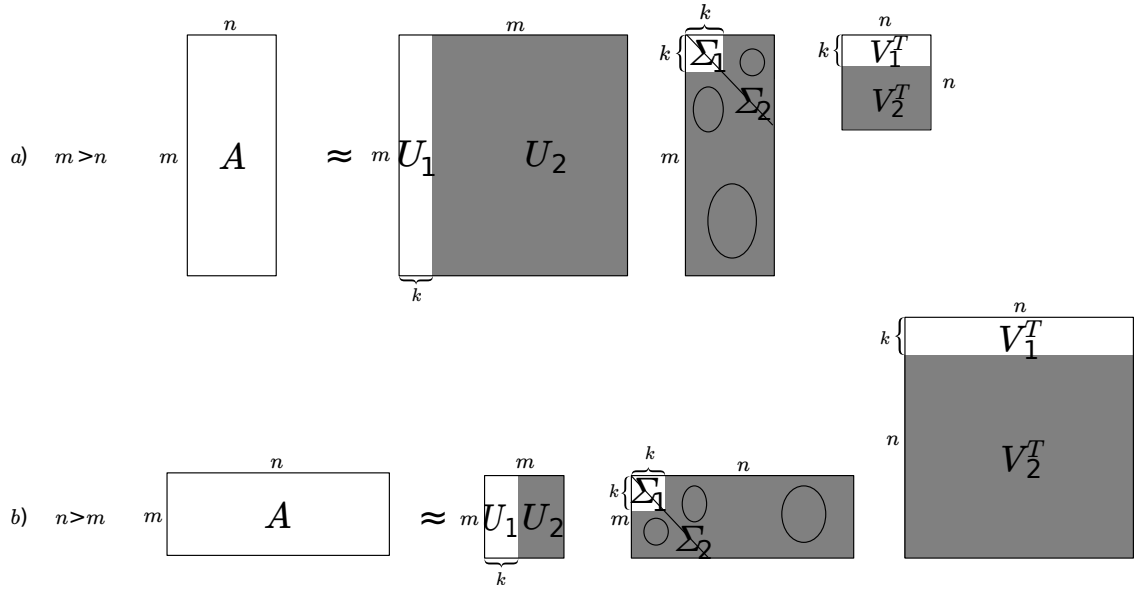


Figure 1.3: Rank k truncated-SVD for both configurations, the shadowed part is dropped upon truncation. $k \leq n,\ k \leq m$.

Numerics As for the eigenvalue decomposition, there are many algorithm to compute the SVD, among them, the QR algorithm is particularly well suited to slim matrices. In subsequent numerical experiment the LAPACK library is used either as direct SVD solver `dgesdd` or through eigenvalue decomposition `dsyev` if the matrix is slim (this strategy is also well suited for discrete POD as discussed in the next section). `dgesdd` relies on a divide and conquer approach which is one of the most efficient way to handle matrices of large size.

Other algorithm provide direct truncated SVD mainly based on iterative algorithm such as *Arnoldi procedure* based library `ARPACK`. However it is mostly suited for sparse matrices and the PGD fixed point procedure (presented in section 1.3) provides us with a way to obtain of truncated basis. It should be noted that iterative algorithm are very efficient at finding eigen/singular values at both end of the spectrum but face accuracy issues in other regions, especially for ill-conditioned matrices. This results in non orthonormal bases which may impair decomposition or ROM accuracy.

1.2 Proper Orthogonal Decomposition

The POD was discovered many times in many different fields, however it is often attributed to Kosambi [Kos43] who introduced it in 1943. Also, the POD comes under many names depending on the field in which it is used or devised. For instance, it is rigorously equivalent to the Karhunen-Loève expansion [Loè77] or Principal Component Analysis (PCA) usually attributed to [Pea01]. It is an elegant way to approximate a high dimensional[2] system into a low dimensional one. To do so, a linear procedure is devised to compute a basis of orthogonal proper modes that represent the energy repartition of the system. They are obtained by solving Fredholm's equation for data (usually) obtained through numerical simulations. Additionally the POD offers an optimal representation of the energy in term of L^2 norm.

It has been applied to extract dominant patterns and properties in wide variety of fields such as signal, data compression, neural activity, mechanics or fluid dynamics to name only a few. An enlightening description of the use of POD is given by Bergmann [Ber04]": *The POD defines uniquely and without ambiguity coherent structures*[3]*, as the realization of largest projection on the mean realization contained in the database*".

Problem formulation (scalar case). Find the best approximation, in the sense of a given inner product $(\cdot,\cdot)$ and average operator $\langle\cdot,\cdot\rangle$, of $f : \mathcal{D} = \Omega_x \times \Omega_t \longrightarrow \mathbb{R}$ as a finite sum in the form

$$\tilde{f}_r(\boldsymbol{x}, t) = \sum_{k=1}^{r} a_k(t)\phi_k(\boldsymbol{x}) \tag{1.2.1}$$

where $(\phi_k)_k$ are orthogonal for the chosen inner product. a_k is given by $a_k(t) = (f(\cdot, t), \phi_k(\cdot))$ then a_k only depends on ϕ_k.

Discrete POD problem is often found in the literature as follow. Let $\{f_1, ..., f_{n_t}\}$ the snapshots of f i.e. the representation of f at discrete time $\{t_j\}_{j=1}^{n_t}$. It is assumed that $\mathcal{F} = \text{span}\{f_1, f_2, ..., f_{n_t}\}$.

POD generates an orthonormal basis of dimension $r \leq n_t$, which minimizes the error from approximating the snapshots space $\mathcal{F}$. The POD basis verifies the optimum of the following:

$$\min_{\{\phi\}_{k=1}^{r}} \sum_{j=1}^{n_t} \|f_j - \tilde{f}_{r,j}\|^2, \text{ s.t. } (\phi_k, \phi_j) = \delta_{kj} \tag{1.2.2}$$

where $\tilde{f}_{r,j} = \sum_{k=1}^{r} (f_j, \phi_k)\, \phi_k$ and δ_{kj} is the Kronecker symbol. One may observe that $\sum_{k=1}^{r} \cdot$ is the first order approximation of the time mean operator $\langle\cdot\rangle$. This problem can be solved with discrete Eigen Value Decomposition (EVD). Although it is the most common formulation of discrete POD in mechanics literature, it can be misleading regarding the construction and properties of the POD. This is why a much more detailed presentation is given in this thesis.

1.2.1 Building the POD

This subsection aims at providing a rigorous, however mechanics oriented presentation of the POD. The present approach is based on Bergmann thesis manuscript [Ber04] and

[2]Here, high dimensionality is to be understood as rich phenomenon that require many degrees of freedom to be described properly as opposed to simpler system which are described by few degrees of freedom e.g. simple pendulum.

[3]The notion of coherent structures, introduced by Lumley (1967) [Lum67,Lum81] is central in the use of POD for mechanics.

lecture notes at Von Karman Institute together with Cordier [CB03a, CB03b] as well as some of the vast corpus available including [Ale15, Cha00, Fah01]. Since POD is the cornerstone of several multivariate data reduction techniques, it is crucial to provide the mathematics underlying this method. Without loss of generality, the usual framework for POD where the two variables are space (possibly a position vector) and times. It makes mental representation easier for the reader and most of the POD jargon was introduced with time-space POD.

Let $\boldsymbol{X} = (\boldsymbol{x}, t) \in \mathcal{D} = \Omega_x \times \Omega_t$ and $\boldsymbol{u} : \mathcal{D} \longrightarrow \mathbb{R}^d$ a vector valued function. Additionally we assume[4] that a scalar product $(\cdot, \cdot)$ is defined on $\mathcal{D}$ and $||\cdot||$ its associated norm while an average operator $\langle\cdot\rangle$ is defined on $\mathcal{D}$[5]. We also need the following u to be of finite norm. The dominant modes of a set of realization $\{\boldsymbol{u}(\boldsymbol{X})\}$ are sought, i.e. the function $\boldsymbol{\phi}$ whith the largest projection on realizations $\{\boldsymbol{u}(\boldsymbol{X})\}$ in the least square sense. In other words, we seek $\boldsymbol{\phi}$ that maximizes $|(\boldsymbol{u}, \boldsymbol{\phi})|$ where $\boldsymbol{\phi}$ is normalized. Then the maximum of this expression is sought

$$\frac{\langle|(\boldsymbol{u}, \boldsymbol{\phi})|^2\rangle}{\|\boldsymbol{\phi}\|^2} \tag{1.2.3}$$

This leads to the following constrained maximization problem

$$\max_{\boldsymbol{\psi} \in L^2(\mathcal{D})} \frac{\langle|(\boldsymbol{u}, \boldsymbol{\psi})|^2\rangle}{\|\boldsymbol{\psi}\|^2} = \frac{\langle|(\boldsymbol{u}, \boldsymbol{\phi})|^2\rangle}{\|\boldsymbol{\phi}\|^2} \tag{1.2.4}$$

with

$$(\boldsymbol{\phi}, \boldsymbol{\phi}) = 1$$

In order to rewrite problem (1.2.4), a linear operator $\mathcal{R} : L^2(\mathcal{D}) \longrightarrow L^2(\mathcal{D})$ is introduced, it is defined as

$$\mathcal{R}\boldsymbol{\phi}(\boldsymbol{X}) = \int_{\mathcal{D}} R(\boldsymbol{X}, \boldsymbol{X'})\boldsymbol{\phi}(\boldsymbol{X'})d\boldsymbol{X'} \tag{1.2.5}$$

where $R(\boldsymbol{X}, \boldsymbol{X'}) = \langle \boldsymbol{u}(\boldsymbol{X}) \otimes \boldsymbol{u}(\boldsymbol{X'})\rangle$ is the tensor of spatio-temporal correlations. Now suppose that $\langle\cdot\rangle$ and $\int$ can be permuted then the following holds

$$\begin{aligned} (\mathcal{R}\boldsymbol{\phi}, \boldsymbol{\phi}) &= \langle|(\boldsymbol{u}, \boldsymbol{\phi})|^2\rangle \quad \geq 0 \\ (\mathcal{R}\boldsymbol{\phi}, \boldsymbol{\psi}) &= (\boldsymbol{\phi}, \mathcal{R}\boldsymbol{\psi}) \qquad \forall(\boldsymbol{\phi}, \boldsymbol{\psi}) \in [L^2(\mathcal{D})]^2 \end{aligned}$$

Since $\mathcal{R}$ is a positive self-adjoint operator, the spectral theory applies and the solution of problem (1.2.4) is given by the largest eigen value of this new problem

$$\mathcal{R}\boldsymbol{\phi} = \lambda\boldsymbol{\phi} \tag{1.2.6}$$

It can be written as a Fredholm integral equation:

$$\sum_{j=1}^{d} \int_{\mathcal{D}} R_{ij}(\boldsymbol{X}, \boldsymbol{X'})\phi^j(\boldsymbol{X'})d\boldsymbol{X'} = \lambda\phi^i(\boldsymbol{X}) \quad \forall i \tag{1.2.7}$$

[4] We will see in 1.2.2 that x and t play symmetric roles as long as the operators are well defined.

[5] The natural choice for fluid dynamics applications $L^2(\Omega_x)$ scalar product and a time average. The choice of the average operator $\langle\cdot\rangle$ kind (temporal, spatial,...) determines which kind of POD is used.

Some fundamental properties of the POD.

1. For $\mathcal{D}$ bounded, Hilbert-Schmidt theory applies and ensures the existence of countably infinitely many solutions to equation (1.2.7)

$$\sum_{j=1}^{d} \int_{\mathcal{D}} R_{ij}(\boldsymbol{X}, \boldsymbol{X'}) \phi_r^j(\boldsymbol{X'}) d\boldsymbol{X'} = \lambda_r \phi_r^i(\boldsymbol{X}) \tag{1.2.8}$$

where $\lambda_r, \boldsymbol{\phi}_r$ are respectively the POD eigenvalues and eigen functions of order $r = 1, 2, ..., +\infty$. Each new eigen function is defined as the solution of problem (1.2.6) adding a new constraint: orthogonality with the already known eigen functions.

$$\sum_{i=1}^{d} \int_{\mathcal{D}} \phi_r^i(\boldsymbol{X}) \phi_p^i(\boldsymbol{X}) d\boldsymbol{X} = \delta_{rp} \tag{1.2.9}$$

2. $\mathcal{R}$ is positive self-adjoint then $\lambda_i \geq 0$. Additionally, they are taken decreasing and they form a converging series i.e.

$$\sum_{r=1}^{\infty} \lambda_i \leq +\infty$$

.

3. The POD eigen functions form a complete basis, any realization $u(\boldsymbol{X})$ can be represented in that basis.

$$u^i(\boldsymbol{X}) = \sum_{r=1}^{\infty} a_r \phi_r^i(\boldsymbol{X}) \tag{1.2.10}$$

4. a_r is obtained by projecting $\boldsymbol{u}$ on $\boldsymbol{\phi}_r$

$$a_r = (\boldsymbol{u}, \boldsymbol{\phi}_r) = \sum_{i=1}^{d} \int_{\mathcal{D}} u_i(\boldsymbol{X}) \phi_r^i(\boldsymbol{X}) d\boldsymbol{X} \tag{1.2.11}$$

5. **Mercer's Theorem.** The spatio-temporal correlation matrix at two points R_{ij} is kernel based on $\mathcal{R}$ then Mercer's theorem provides a series representation,

$$R_{ij}(\boldsymbol{X}, \boldsymbol{X'}) = \sum_{r=1}^{\infty} \lambda_r \phi_r^i(\boldsymbol{X}) \phi_r^j(\boldsymbol{X'}) \tag{1.2.12}$$

6. Thanks to the previous property, it can be shown [CB03a] that the coefficients a_r are uncorrelated and their quadratic average is equal to the POD eigenvalues

$$\langle a_r, a_p \rangle = \delta_{rp} \lambda_r \tag{1.2.13}$$

7. Using Mercer's theorem and the orthogonality of POD eigen functions, the following expression emerges

$$\sum_{i=1}^{d} \int_{\mathcal{D}} R_{ij}(\boldsymbol{X}, \boldsymbol{X}) d\boldsymbol{X} = \sum_{r=1}^{\infty} \lambda_r = E \tag{1.2.14}$$

Where E coincides with kinetics energy if $\boldsymbol{u}$ is the velocity field of a fluid for example. Then λ_r indicates the weight of each modes in terms of energy.

Remark. These properties ensure the uniqueness of the proper orthogonal decomposition (given that $||\boldsymbol{\Phi}|| = 1$).

Optimality of the POD basis. Let $\boldsymbol{u} : \mathcal{D} \longrightarrow \mathcal{E} \subset \mathbb{R}^d$ with $\boldsymbol{u} \in L^2(\mathcal{D})$ and $\bar{\boldsymbol{u}}$ an approximation of $\boldsymbol{u}$. On a any basis $(\boldsymbol{\psi}_r(\boldsymbol{X}))_{r=1}^{\infty}$ one can write

$$\bar{u}_i(\boldsymbol{X}) = \sum_{r=1}^{\infty} b_r \psi_r^i(\boldsymbol{X}) \tag{1.2.15}$$

Let $\{\boldsymbol{\phi}(\boldsymbol{X})\}_{r=1}^{\infty}$ a set of orthogonal POD eigen functions and $\{\lambda_r\}_{r=1}^{\infty}$ their associated eigenvalues. Then, $\boldsymbol{u}^{POD}$ the POD approximation of u is considered

$$u_i^{POD}(\boldsymbol{X}) = \sum_{r=1}^{\infty} a_r \phi_r^i(\boldsymbol{X}) \tag{1.2.16}$$

Properties 6 and 7 state that if $(\boldsymbol{\psi}_r(\boldsymbol{X}))_{r=1}^{\infty}$ are non dimensional, $\langle b_r, b_r \rangle$ represents the energy of mode n. Cordier and Bergmann [CB03b] proved the optimality of the POD basis through the following lemma.

Lemma 1.2.1. *Optimality of POD basis For any rank $R \in \mathbb{N}^*$ the following inequality holds*

$$\sum_{r=1}^{R} \langle a_r, a_r \rangle = \sum_{r=1}^{R} \lambda_r \geq \sum_{r=1}^{R} \langle b_r, b_r \rangle \tag{1.2.17}$$

In other words, among all linear decomposition, POD is the most efficient, i.e. for a given number of POD modes R, the projection on the subset produced by the first R POD eigen-functions is the one that contains on average the most (kinetic) energy possible.

Operation Count In order to evaluate the number of operations required to compute the POD decomposition of simulation data we consider the simple case where both variables have the same number of samples N, i.e. $\Omega = \Omega_x \times \Omega_t$ is discretized in an $N \times N$ matrix. For $f : \Omega \to \mathbb{R}$, first, the correlation matrix is computed

$$\boldsymbol{R}(x, x') = \int_{\Omega_t} f(x, t) f(x', t) dt \tag{1.2.18}$$

Obviously, the cost depends on the integration technique. For this evaluation we choose a second order method: trapezoidal integration rule which cost is in $\mathcal{O}(N)$. Then this operation has to be performed for each discrete combination of x and x' which results in N^2 evaluations. The global cost to evaluate $\boldsymbol{R}(x, x')$ on the discrete grid is $\mathcal{O}(N^3)$ double precision operations. Then the first $R \ll N$ eigen values of problem (1.2.5) are sought. This problem can be solved using a Lanczos algorithm which requires very few iteration to compute the first eigenvalues, it requires $\mathcal{O}(MN^2)$. Then an estimate of the operation count to compute the M mode POD of f with a Lanczos algorithm is

$$\mathcal{O}(N^3 + MN^2) = \mathcal{O}(N^3) \tag{1.2.19}$$

As shown in the next sections choosing to apply the POD to the dimension with the lowest degrees of freedom (DoF) number will lead to much lower number of operation. Especially if one dimension DoF number is much lower than the other.

A POD algorithm. One of the many possible implementations of the POD is proposed in this section. Although it might not be the most computationally efficient version, it preserves all the functional approach framework. Indeed the user is free to implement any integration method so that the projector also apply to L^2, not to any matrix space. This

statement is also true the linear operator $\boldsymbol{\mathcal{R}}\phi^m$ which eigenvalue problem is solved by a iterative orthonormal power method.

Algorithm 1: POD (*Standard, Deflation Power Method*)

input : f, target error ϵ
output: $\tilde{f}_r = \sum_{k=1}^{r} \sigma_k X_k Y_k$

m=0
$\boldsymbol{R}(x,x') = \int_{\Omega_t} f(x,t)f(x',t)dt$;
while $\frac{\sigma_m}{\|f\|_{L^2}} \geq \epsilon$ **do**
  $k = k+1$
  $(\lambda_k, \phi_k) =$ `Orthonormal_Power_method` $[(\boldsymbol{\mathcal{R}} - \tilde{\boldsymbol{f}}_{\boldsymbol{k-1}})\phi_k = \lambda_k\phi_k]$
  $\sigma_k = \sqrt{\lambda_k}$
  $a_k = \int_{\Omega_x} f(x,t)\phi_k(x)dx/\ \sigma_k$
  $\tilde{f}^k = \tilde{f}_{k-1} + \sigma_k\phi_k a_k$
return f_k

1.2.2 Discussion on the POD variations

Choosing the right realizations Since all information used in POD comes from the chosen realizations[6] $\boldsymbol{X}$, one might wonder on which criteria to choose them. It is a complex question and in some cases it has been shown that major flow properties are not preserved (e.g. Noack [NAM+03]).

Choosing the inner product One very interesting property of POD is that the inner product can be chosen depending on the studied problem. For instance, if a problem requires to preserve certain properties such as incompressibility, one can choose a inner product that is more suitable for this task. In fluid dynamics applications, mainly the following two possibilities emerge.

Inner product on L^2. $L^2(\Omega)$ is the Hilbert space of square integrable functions is usually well suited for fluid dynamics applications. The inner product on $L^2(\Omega)$ for two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is defined by

$$(\boldsymbol{u}, \boldsymbol{v}) = \int_{\Omega} \left(\sum_i u_i v_i \right) d\boldsymbol{x} \tag{1.2.20}$$

where $\|\boldsymbol{u}\|^2 = (\boldsymbol{u}, \boldsymbol{u})$ is the associated norm.

A fluid kinetic energy is proportional to $\|\boldsymbol{u}\|^2$ in fluid dynamics applications. Then, it seems reasonable to use this inner product for general fluid dynamics problems.

Inner product on H^1. Iollo et al. [ILD00] were among the first to advocate the use of Sobolev spaces for improved quality in POD based reduced models. Indeed, L^2 norm was found to be unstable. H^1 norm has been continuously used since then, for example for parabolized Navier-Stokes equation [DNS+12]. $H^1(\Omega)$ is the Sobolev space containing $L^2(\Omega)$ functions which first derivative are also part of $L^2(\Omega)$. The inner product on H^1 for two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ is defined as

$$(\boldsymbol{u}, \boldsymbol{v})_\epsilon = \int_{\Omega} \boldsymbol{u} \cdot \boldsymbol{v}\ d\boldsymbol{x} + \epsilon \int_{\Omega} (\boldsymbol{\nabla u} \cdot \boldsymbol{\nabla v})\ d\boldsymbol{x} \tag{1.2.21}$$

where ϵ is a numerical parameter accounting for different measures.

[6] Also known as *snapshots* in the fluid dynamics community.

The POD was described in a general framework in the previous section. However in practical applications, the choice of the actual first and second variable has a great influence on the numerical computing of POD bases. Choosing time or space as the first variable will affect both speed and accuracy of POD algorithms. The standard POD was introduced by Lumley [Lum81] while Sirovich [Sir87] proposed the snapshots version.

1.2.2.1 Standard POD

Lumley's approach [Lum81] for POD relies on choosing $\langle\cdot\rangle$ as a temporal average of the realizations i.e.

$$\langle \cdot \rangle = \frac{1}{T}\int_{\mathcal{T}} \cdot \, dt \tag{1.2.22}$$

where $\mathcal{T} = [0, T]$. It is assumed that $\mathcal{T}$ is a period in which all realizations are known and is long enough to represent the flow. The space variable, $\boldsymbol{x}$, lives in $\Omega \in \mathbb{R}^d$. For a 3D fluid dynamics simulation, $d = 3$, we can focus on the velocity fields $\boldsymbol{u} : \Omega \times \mathcal{T} \longrightarrow \mathbb{R}^3$ with the usual $L^2(\Omega)$ scalar product:

$$(u, v)_{L^2(\Omega)} = \sum_{i=1}^{d}\sum_{j=1}^{d}\int_{\Omega} u_i v_j d\boldsymbol{x}, \quad \forall u, v \in L^2(\Omega) \tag{1.2.23}$$

Then Fredholm's equation (1.2.7) now reads

$$\sum_{j=1}^{d}\int_{\Omega} R_{ij}(\boldsymbol{x}, \boldsymbol{x'})\phi^j(\boldsymbol{x'})d\boldsymbol{x'} = \lambda\phi^i(\boldsymbol{x}), \quad \forall 1 \leq j \leq d \tag{1.2.24}$$

Where R_{ij} is the spatial correlation tensor now reads

$$R_{ij}(\boldsymbol{x}, \boldsymbol{x'}) = \frac{1}{T}\int_{\mathcal{T}} u_i(\boldsymbol{x}, t)u_j(\boldsymbol{x'}, t)\, dt \tag{1.2.25}$$

It should be noted that in this case, $\boldsymbol{\phi}$ is purely spatial.

Size of the eigenvalue problem, discrete case. Let n_x be the number of spatial grid points. Usually, $n_x \in [10^5, 10^9]$ for DNS simulations. d is the number of elements of $\boldsymbol{u}$, e.g. $d = 3$ for 3D cases. Then this approach becomes intractable as soon as 3D problems are studied, even using adapted algorithm/softwares.

It should be used only when data is spatially sparse, for example within particle tracking problems. An efficient way to overcome this difficulty for DNS is to use the snapshots method.

1.2.2.2 Snapshots POD

The snapshots method was originally introduced by Sirovich [Sir87] in 1987. It is the counterpart of the standard POD where the role of $\boldsymbol{x}$ and t are inverted. It is well suited for data where there is a large number of spatial grid points while relatively low number of time frames e.g. DNS output data. The average operator is a spatial average that reads

$$\langle \cdot \rangle = \int_{\Omega} \cdot \, d\boldsymbol{x} \tag{1.2.26}$$

i.e. for two fields, the $L^2()$ scalar product defined in 1.2.23. Then, the POD scalar product is the time integral on $\mathcal{T}$. The eigen problem now reads

$$\int_{\mathcal{T}} C(t, t')a(t')\, dt' = \lambda a(t) \tag{1.2.27}$$

where C is the temporal correlation matrix that does not account for cross correlation. In order to preserve consistency with the previous definition a $1/T$ is imposed before the integral in C definition.

$$C(t,t') = \int_\Omega \sum_{i,j=1}^{d} u_i(\boldsymbol{x},t)u_j(\boldsymbol{x},t')\, d\boldsymbol{x} \tag{1.2.28}$$

The eigen functions are functions of time only. Then the eigen problem of this snapshot POD is of reasonable size, $r = n_t$ the number of time frames/snapshots. This is particularly well suited if $n_t \ll n_x$ which is the usual case for DNS output data. One can recover the spatial POD modes $\boldsymbol{\phi}_n(\boldsymbol{x})$ by projecting the snapshots on the function with or without normalization as per the user preference i.e.

$$\boldsymbol{\phi}_k(\boldsymbol{x}) = \int_{\Omega_t} \boldsymbol{u}(\boldsymbol{x},t)a_k(t)dt, \quad \forall 1 \leq k \leq r \tag{1.2.29}$$

Both these methods share a set of properties.

- Any realization $u_i(\boldsymbol{x},t)$ can be represented exactly on the full POD basis which is orthonormal.
$$u_i(\boldsymbol{x},t) = \sum_{n=1}^{N_{POD}} \sigma_n a_n(t)\phi_n^i(\boldsymbol{x})$$
where $N_{POD} \leq \infty$. Potentially, an infinite number of modes may be required.

- Temporal modes (a_n) form an orthogonal family (that can be normalized) while spatial modes $(\boldsymbol{\Phi}_n)$ form an orthonormal family.

- Any property that can be written as a linear combination of the realizations is directly passed to the spatial eigen-functions. Incompressibility or Dirichlet boundary conditions are two examples of properties that are passed to the basis $\boldsymbol{\phi}_n$, $\forall n < N_{POD}$ if u has these properties.

1.2.3 SVD $\simeq$ POD

From the previous sections, it clearly appears that POD and SVD share many of their properties. One can adopt two different angles to explore the link between POD and SVD.

a. Use the the optimality of the SVD to solve the discrete POD minimization problem. This is a straightforward application of the fact that eigenvectors can be computed either from eigenvalue decomposition or SVD. This approach has been described in detail by Bergmann and Fahl's work [Ber04,Fah01]. Compared to the SVD technique described in section 1.1, it adds a mass matrix but does not require any linear algebra analysis to be linked with POD. Details are given in section 1.2.3.1.

b. The other way of looking at this link, was proposed among many others by Chaterjee [Cha00]. It is a simpler presentation of the POD, only valid in the discrete framework. It relies on the fact that the SVD solves optimally a matrix problem that may be seen as the discrete equivalent of the infinite dimensional problem (1.2.2) using the Euclidian norm for vectors. As will be shown in the general framework of tensor spaces in chapter 2, this approach is justified by the applicability of the same algorithm to tensor spaces of different nature, either continuous or discrete.

It shall be noted that these two interpretations leads to different algorithms which may not display the same properties of accuracy or efficiency especially when the basis is used for reduced order modeling as its orthogonality is a very important feature. The very illustration is the possibility to chose a problem adapted inner product in the POD algorithm while SVD is blind to data and will be performed in the same fashion for any problem, sometimes without preserving physical properties.

1.2.3.1 Numerical implementation of snapshots L^2 POD

In order to implement efficiently the snapshot L^2 POD one needs to carefully compute all the scalar products according to a given integration scheme and accuracy. From that point there are two possibilities either one chose to program weighted sum and solve Fredholm's equation (1.2.7) using a suitable algorithm e.g. power iteration/deflation method. Or they chose to use linear algebra tools (SVD, EVD solver) and the integration becomes a matrix vector product. Additionally a change of basis on the snapshot data is required for the discrete operator to preserve L^2 properties. The second option has been selected in this work after observing that it is much more computationally efficient in the fortran implementation. Moreover, it allows one to use optimally programmed solver such as `LAPACK` or `ARPACK` libraries as well as `BLAS` operations. A brief overview of this method is given next.

A reasonably general setup is chosen for this typical implementation of discrete POD. Let $\mathcal{D} = \Omega \times T \subset \mathbb{R}^d \times \mathbb{R}$ a spatio-temporal domain that is discretized in $n_x \times n_t$ elements and scalar function $f \in L^2(\mathcal{D})$. Let $\boldsymbol{F} \in \mathbb{R}^{n_x \times n_t}$ the matrix representation of f which columns are $\boldsymbol{f}_i = (f(x_1, t_i), f(x_2, t_i), \cdots, f(x_{n_x}, t_i))^\intercal$. The goal is to find the discrete representation $\{\Phi, A\}$ of the POD basis $\{\phi_i, a_i\}$.

- **Inner product and associated norm:**
  Let u and v two scalar functions of $L^2(\mathcal{X})$, $\mathcal{X}$ is either Ω or T which discretization are $\boldsymbol{u}$ and $\boldsymbol{v} \in \mathbb{R}^n$. Then $(u, v)_{L^2(\mathcal{X})}$ the inner product of these functions in $L^2(\mathcal{X})$ is discretized as $(\boldsymbol{u}, \boldsymbol{v})_{\boldsymbol{M}}$ where $\boldsymbol{M}$ is the interpolation matrix. For instance, $\boldsymbol{M}$ is diagonal for a trapezoidal rule with the weights associated to this integration scheme. The discrete integration operator reads

$$(u, v)_{L^2(\mathcal{X})} \xrightarrow{disc} (\boldsymbol{u}, \boldsymbol{v})_{\boldsymbol{M}} = \boldsymbol{u}^\intercal \boldsymbol{M} \boldsymbol{v}$$

  The norm associated to these inner product behave identically $||u||_{L^2} \xrightarrow{disc} \|\boldsymbol{u}\|_{\boldsymbol{M}}$. This procedure is applied on both time and space variables. $M^{1/2}$ is the Choleski decomposition left matrix of $\boldsymbol{M}$ i.e. $\boldsymbol{M} = \boldsymbol{M}^{1/2}\boldsymbol{M}^{1/2\intercal}$. As stated earlier for usual integration techniques (trapozoidal, Simpson's, etc.), the matrix is diagonal and $m_{ij}^{1/2} = \sqrt{m_{ij}}$. Let $\boldsymbol{M}_t$ be the time integration matrix and $\boldsymbol{M}_x$ the space integration matrix. For instance a trapezoidal rule on a uniform grid yields the following time integration matrix,

$$(M_t)_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ \delta t & \text{if } 1 < i < N_t \\ \delta t/2 & else. \end{cases}$$

  We assume that the space integration is also diagonal with $(M_x)_{ii} = w_i$ where w_i is the weight corresponding to the integration formula e.g. P0, P1, etc. for finite elements, trapezoidal for finite differences, etc.

- **Discrete autocorrelation function for snapshot POD:**
  The autocorrelation function

$$C(t,t') = \int_\Omega f(\boldsymbol{x},t) f(\boldsymbol{x},t')\, d\boldsymbol{x}$$

  is discretized into a matrix $\boldsymbol{C} \in \mathbb{R}^{n_t \times n_t}$ whose elements are defined as

$$C(t_i,t_j) \approx C_{ij} = \sqrt{(M_t)_{ii}(M_t)_{jj}}\, \boldsymbol{f}_i^\intercal \boldsymbol{M}_x \boldsymbol{f}_j \tag{1.2.30}$$

  One can write $\boldsymbol{C}$ as a matrix product that reads $\boldsymbol{C} = \boldsymbol{M}^{1/2}\boldsymbol{M}^{1/2\intercal}\boldsymbol{F}^\intercal \boldsymbol{M}_x \boldsymbol{F}$, this operation can be seen as a change of basis for the autocorrelation matrix. Then one can apply the SVD (thin SVD or any EVD algorithm) on $\boldsymbol{C}$,

$$\boldsymbol{C} = \tilde{\boldsymbol{U}}\boldsymbol{\Sigma}\,\tilde{\boldsymbol{V}}^\intercal \tag{1.2.31}$$

  Here, it is chosen to keep the full discrete basis. In practice, a truncated SVD will be used as discussed in sections 1.1 and 1.2.

  In order to recover the actual $\{\phi_i, a_i\}$ POD basis one need to apply the change of basis back to the discrete representation of the basis function and project f on the temporal basis,

$$\{a_i\}_i \approx \boldsymbol{A} = \boldsymbol{M}_t^{-1/2\intercal}\tilde{\boldsymbol{U}} \tag{1.2.32}$$

$$\{\phi_i\}_i \approx \boldsymbol{\Phi} = (\boldsymbol{F}^\intercal, \boldsymbol{A})_{\boldsymbol{M}_t} = \boldsymbol{F}\boldsymbol{M}_t\boldsymbol{A} \tag{1.2.33}$$

where $\boldsymbol{\Phi}_i$ is to be normalized for actual basis.

Proof. Proving that these matrix-vector products are indeed equivalent to the discrete L^2 POD is fairly straightforward discretization of operators. One can wonder why the POD is often presented as in eq. (1.2.2), this is because, in standard POD, only the spatial basis is kept to build a ROM, thus this mixed $L^2\Omega/l^1(T)$ is sufficient. □

Remark (Versatility of such implementation). Implementing this discrete POD enables very easy change in the integration method. The special case of $M = I$ is equivalent to the usual algebraic solver (with a slight overhead). This means that numerically, any method relying on SVD is interchangeable with the equivalent discrete POD. Thus the user can very easily switch from l^2-norm to L^2-norm and virtually use any integration scheme for POD. In order to prevent misunderstanding, l^2 POD algorithm will be referred as `SVD_by_EVD`.

1.3 Proper Generalized Decomposition

In order to provide a general overview of the bivariate methods, we now focus on a method that is more recent than POD and SVD. The Proper Generalized Decomposition (PGD) has been developed by a relatively small cluster of researchers during the 2000's including Chinesta, Cueto, Ladevèze, Ammar among others. The method is a variation of the popular LATIN method [CL93] developed in the 1980's by Ladevèze et al. It was first developed in the context of mechanics [CLA+09, CALK11, ACDH10] and later extended to general systems of PDEs. Nouy and Falco have extended it to a more general framework [Nou10, FN12] and provide additional numerical analysis.

This presentation of the PGD is very restrictive as compared to the full capabilities of this method. On the one hand, only the bivariate case is shown here, a multivariate

version is given in section 3.1. On the other hand, the PGD is a general algorithm that can be applied to partial differential equation (PDE), the separated approximation problem can be written as PDE $u = f$ where u is sought as a separate sum, this problem can be refered as *a posteriori* PGD. This section is restricted to describing how the PGD is build in the case of bivariate data post-processing.

1.3.1 Constructing a bivariate *a posteriori* PGD

Let $f : \Omega = \Omega_x \times \Omega_y \to \mathbb{R}$ a square integrable function, i.e. $f \in L^2(\Omega)$. As in the POD, the goal of the PGD is to provide a separated approximation of f that reads

$$f(x, y) \approx u_r(x, y) = \sum_{i=1}^{r} X_i(x) Y_i(y) \tag{1.3.1}$$

where $X_i \in L^2(\Omega_x)$, $Y_i \in L^2(\Omega_y)$, $\forall i \leq r$ that form an orthogonal basis of rank r of $L^2(\Omega_x)$ and $L^2(\Omega_y)$. The sequence (u_r) converges toward f, i.e. $u_r \xrightarrow[r\to\infty]{} u = f$

1.3.1.1 An Enrichment Process

In order to determine each element of the sequence an enrichment process has been proposed by Chinesta et al. [CKL13]. This algorithm enriches the decomposition basis recursively, each time adding a new pair of basis vectors $\{X_i, Y_i\}$.

X_r and Y_r are computed by a fixed point algorithm alternating directions. The weak formulation of the *a posteriori* PGD problem reads

$$\forall u^\star \in H^1(\Omega), \quad \int_\Omega u^\star (u - f) = 0 \tag{1.3.2}$$

At the begining of each step it is assumed that $u_{r-1}(x, y) = \sum_{i=1}^{r-1} X_i(x) Y_i(y)$ is known thus u_r is sought under the form

$$u_r(x, y) = u_{r-1}(x, y) + X_r(x) Y_r(y) \tag{1.3.3}$$

The process of adding terms to the sum, i.e. computing the sequence $(u_p)_{p=1}^r$ is called the *enrichment process*. This process ends when a stopping criterion is fulfilled. Since in the general case, one does not know the exact solution, it is chosen to stop the process when the weight of the last term compared to the rest of the series becomes negligible. This reads

$$\mathcal{E}(r) = \frac{||X_r Y_r||_{L^2(\Omega)}}{||X_1 Y_1||_{L^2(\Omega)}} = \frac{||Y_r||_{L^2(\Omega)}}{||Y_1||_{L^2(\Omega)}} \leq \varepsilon_{enrichment} \tag{1.3.4}$$

Indeed the terms are of decreasing norm, then there is no need to compare the whole series, the first term is sufficient. In addition to that, we define $\{X_i\}$ such that $\forall i < N, \quad ||X_i||_{L^2(\Omega)} = 1$ all the information about the norm is transfered to $\{Y_i\}$.

Remark (Choice of u_0). It is not trivial to chose u_0 as it influences the convergence speed of the fixed point algorithm and may even prevent it from converging. However in most cases it does not have much influence as the first mode contains a lot of energy that leads to quick convergence of the fixed point algorithm.

1.3.1.2 Fixed point algorithm

In this section, an iterative algorithm called Fixed Point Algorithm (FPA) is described, it enables the computing of a new term (X_r, Y_r) to the basis. This is the version described by Chinesta [CKL13]. In practice, either it converges in a few iterations or it does not converge at all. The key feature of this algorithm is its alternated direction nature, i.e. each direction is computed one at a time.

It is assumed that $\tilde{X}^k$ and $\tilde{Y}^k$ are known after step k of the FPA . Thus $\tilde{u}(x,y) = u_{r-1}(x,y) + \tilde{X}^k(x)\tilde{Y}^k(y)$. Moreover, the computing of $\tilde{X}^{k+1}$ relies on $\tilde{Y}^k$ while $\tilde{X}^{k+1}$is used to compute $\tilde{Y}^{k+1}$. $u^\star$ is set to

$$u^\star(x,y) = \begin{cases} X^\star(x)\tilde{Y}^k(y) & \text{searching } X^{k+1} \\ \tilde{X}^{k+1}(x)Y^\star(y) & \text{searching } Y^{k+1} \end{cases} \tag{1.3.5}$$

For simplicity reasons, the subsequent development will only address the computations needed to evaluate $\tilde{X}^{k+1}$ since the same process is at work for $\tilde{Y}^{k+1}$. Given the previous equations, the following weak formulation holds

$$\int_\Omega \left[X^\star(x)\tilde{Y}^k(y) \left(u_{r-1}(x,y) + \tilde{X}^{k+1}(x)\tilde{Y}^k(y) - f(x,y) \right) \right] dxdy = 0 \tag{1.3.6}$$

This equation can be written as follow

$$\alpha^x \int_{\Omega_x} X^\star(x)\tilde{X}^{k+1}(x)dx = -\int_{\Omega_x} X^\star(x) \sum_{j=1}^{r-1} \left(\beta_j^x X_j(x)\right) dx + \int_{\Omega_x} X^\star(x)\gamma^x(x)dx \tag{1.3.7}$$

where

$$\alpha^x = \int_{\Omega_y} (\tilde{Y}_i^k)^2 \tag{1.3.8}$$

$$\beta_j^x = \int_{\Omega_y} \tilde{Y}^k Y_j \quad \forall j < p \tag{1.3.9}$$

$$\gamma^x(x) = \int_{\Omega_y} \tilde{Y}^k f \tag{1.3.10}$$

Finally the strong formulation stands

$$\begin{cases} \tilde{X}^{k+1}(x) = \frac{-\sum_{j=1}^{r-1}\left(\beta_j^x X_j(x)\right) + \gamma^x(x)}{\alpha} & ,\forall x \in \Omega_x \\ \tilde{Y}^{k+1}(y) = \frac{-\sum_{j=1}^{r-1}\left(\beta_j^y Y_j(y)\right) + \gamma^y(y)}{\alpha} & ,\forall y \in \Omega_y \end{cases} \tag{1.3.11}$$

$\tilde{X}^{k+1}$ is normalized i.e. $||\tilde{X}^{k+1}||_{L^2(\Omega_x)} = 1$, so that all the information relative to the norm is transfered to $\tilde{Y}^{k+1}$. This algorithm is performed alternatively along x and y direction, every time $\tilde{Y}^{k+1}$ is computed the subsequent stopping criterion is checked.

$$\mathcal{E}_{fixed\,point}(k) = \frac{||\tilde{Y}^{k+1} - \tilde{Y}^k||_{L^2(\Omega_y)}}{||\tilde{Y}^k||_{L^2(\Omega_y)}} < \varepsilon_{fixed\,point} \tag{1.3.12}$$

Algorithm. A synthetic view of the algorithm is given to ease the implementation of the PGD.

Algorithm 2: PGD (*a posteriori*)

input : f
output: $u_r = \sum_{i=1}^{r} X_i Y_i$

$u_0 = 0$, $n = 0$, $(X_i, Y_i)_{i=1}^{r} = \{\}$;
while $\mathcal{E}(r) \geq \mathcal{E}_{enrich}$ **do**
  $r = r + 1$
  $(X_r, Y_r) =$ `fixed_point` $((X_i, Y_i)_{i=1}^{r-1}, f, n)$
  $u_r = u_r + X_r Y_r$
  $\mathcal{E}(r) = \frac{||Y_r||_{L^2(\Omega_y)}}{||Y_1||_{L^2(\Omega_y)}}$
return $u_r = \sum_{i=1}^{r} X_i Y_i$

Algorithm 3: `Fixed_point`

input : $(X_i, Y_i)_{i=1}^{r-1}$, f, n
output: $X_r Y_r$

$u_{r-1} = \sum_{i=1}^{r-1} X_i Y_i$, $k = 0$, $(\tilde{X}^k, \tilde{Y}^k) = (0, 0)$;
while $\varepsilon \geq \mathcal{E}_{fixed_point}$ **do**
  $k = k + 1$
  compute α^x, $\beta^x(j)\ \forall j < n$, γ^x
  $\tilde{X}^{k+1} = \frac{-\sum_{j=1}^{r-1}\left(\beta_j^x X^j(x)\right) + \gamma^x(x)}{\alpha}$, $\forall x \in \Omega_x$
  $\tilde{X}^{k+1} = \tilde{X}^{k+1} / \|\tilde{X}^{k+1}\|_{L^2(\Omega_x)}$
  compute α^y, $\beta^y(j)\ \forall j < n$, γ^y
  $\tilde{Y}^{k+1}(Y) = \frac{-\sum_{j=1}^{r-1}\left(\beta_j^y Y^j(y)\right) + \gamma^y(y)}{\alpha}$, $\forall y \in \Omega_y$
  $\varepsilon = ||\tilde{Y}^{k+1}||_{L^2(\Omega_y)} / ||\tilde{Y}^{k}||_{L^2(\Omega_y)}$
return $(X_r = \tilde{X}^k, Y_r = \tilde{Y}^k)$

1.3.2 Equivalence of PGD with POD/SVD for bivariate decomposition

In section 1.2, algorithm 1 was given to compute the POD of a function, it relies on a deflated power iteration method. In order to show the connection between PGD and POD/SVD, this method is detailed here.

POD reminder and notations. Assume that $X \subset \mathbb{R}^d$ and $Y \subset \mathbb{R}^s$ are two bounded domains, d and s are integers ≥ 1. Let f be a given function in the Lebesgue space $L^2(X \times Y)$. For practical reasons, the integral operator $\mathcal{B}$ with kernel f is introduced

$$\varphi \mapsto \mathcal{B}\,\varphi, \qquad (\mathcal{B}\,\varphi)(y) = \int_X f(x,y)\varphi(x)\,dx. \tag{1.3.13}$$

The operator $\mathcal{B}$ maps $L^2(X)$ into $L^2(Y)$, is bounded and has an adjoint operator $\mathcal{B}^*$ defined from $L^2(Y)$ into $L^2(X)$ as

$$v \mapsto \mathcal{B}^*\,v, \qquad (\mathcal{B}^*\,v)(x) = \int_Y f(x,y)v(y)dy. \tag{1.3.14}$$

The self-adjoint operator $\mathcal{R} = \mathcal{B}^*\mathcal{B}$ is also an integral operator whose kernel $R \in L^2(X \times X)$ is the autocorrelation function defined in section 1.2.1,

$$R(x,x') = \int_Y f(x,y)\,f(x',y)\,dy. \tag{1.3.15}$$

Thus we recover Fredholm's equation ((1.2.7)) with the eigenvalues λ_n, such as

$$\mathcal{R}\,\varphi_n = \lambda_n\,\varphi_n, \qquad \forall n \geq 0. \tag{1.3.16}$$

A straightforward effect of the diagonalization of the operator $\mathcal{R}$ is the following singular value decomposition of the operator $\mathcal{B}$.

Lemma 1.3.1. *There exists a system $(\varphi_n, v_n, \sigma_n)_{n\geq 0}$ such that $(\varphi_n)_{n\geq 0}$ is an orthonormal basis in $L^2(X)$, $(v_n)_{n\geq 0}$ an orthonormal system in $L^2(Y)$ and $(\sigma_n)_{n\geq 0}$ a sequence of nonnegative real numbers such that*

$$\mathcal{B}\,\varphi_n = \sigma_n\, v_n, \qquad \mathcal{B}^*\, v_n = \sigma_n\, \varphi_n. \tag{1.3.17}$$

The sequence $(\sigma_n)_{n\geq 1}$ is ordered decreasingly and decays toward zero.

As discussed previously, we have $\sigma_n = \sqrt{\lambda_n}$, $\forall n \geq 1$, additionally the singular vectors $(\varphi_n)_{n\geq 1}$ are the same as the eigenvectors of $\mathcal{R}$.

1.3.2.1 Power iteration method

The power iteration ranges among the simplest computational eigenvalue methods. We apply this iterative method to approximate the dominant eigenvalues of $\mathcal{R}$ or equivalently the dominant singular values of $\mathcal{B}$. The convergence results of the power iteration will addressed of briefly. A lot of work has been done in this issue. For instance, one can refer to [GL96] and references there in.

The basic form of the iterate power method applied to $\mathcal{R} = \mathcal{B}^*\mathcal{B}$ aims to construct the largest eigenvalue $\lambda_1 = (\sigma_1)^2$ and the related eigenvector φ_1. Thus, σ_1 is the largest singular value of B. The scaled version of it is recommended to avoid underflow/overflow. It can be presented as follows:

Algorithm 4: Power iteration (eigen value problem)

input : $\mathcal{R}$
output: Largest eigenvalue and vector, $\{\lambda_1, \varphi_1\}$

Choose $\varphi^{(0)} \in L^2(X)$ with $\|\varphi^{(0)}\|_{L^2(X)} = 1$
repeat
  $\chi^{(k)} = \mathcal{R}\varphi^{(k-1)}$
  $\varphi^{(k)} = \frac{\chi^{(k)}}{\|\chi^{(k)}\|_{L^2(X)}}$
until *convergence*;
return $\{\lambda_1, \varphi_1\} = \{\|\chi^{(k)}\|_{L^2(X)}, \varphi^{(k)}\}$

The limit of the sequence $(\varphi^{(k)})_{k\geq 0}$ is the eigen-function φ_1 and the sequence $(\|\chi^{(k)}\|_{L^2(X)})_{k\geq 0}$ converges toward the dominant eigenvalue λ_0. Also, for practical reasons, we set $\lambda^{(k)} = \|\chi^{(k)}\|_{L^2(X)}$.
Rewritten in terms of the singular value approximation the algorithm reads

Algorithm 5: Power iteration (singular value ploblem)

input : $\mathcal{B}, \mathcal{B}*$
output: Largest singular value and vectors, $\{\sigma_1, \varphi_1, v_1\}$

Choose $\varphi^{(0)} \in L^2(X)$ with $\|\varphi^{(0)}\|_{L^2(X)} = 1$
repeat
  $w^{(k)} = \mathcal{B}\varphi^{(k-1)}$
  $\chi^{(k)} = \mathcal{B}^* w^{(k)}$
  $\varphi^{(k)} = \frac{\chi^{(k)}}{\|\chi^{(k)}\|_{L^2(X)}}$
until *convergence*;
return $\{\sigma_1, \varphi_1, v_1\} = \{\|w^{(k)}\|_{L^2(X)}, \varphi^{(k)}, w^{(k)}/\|w^{(k)}\|_{L^2(Y)}\}$

If the convergence is ensured then the sequence $(w^{(k)}))_{n\geq 0}$ tends toward w_0. We introduce also the notation $\sigma^{(k)} = \|w^{(k)}\|_{L^2(Y)}$. Passing to the limit, it is easily checked out that $\lim_{k\to\infty} \sigma^{(k)} = \sigma_0$.

1.3.2.2 Connection between the methods

Let us try first to express in a variational form the collection of problems involved in the second version of the power iteration algorithm (5). Line 3 variational form reads

$$\int_Y w^{(k)}(y)w^*(y)\,dy = \int_Y (f(\cdot,y),\varphi^{(k-1)})_{L^2(X)}\,w^*(y)\,dy, \qquad \forall w^* \in L^2(Y).$$

On account of the normalization $\|\varphi^{(n-1)}\|_{L^2(X)} = 1$ we obtain that

$$\int_{X\times Y} \left(f - \varphi^{(k-1)} \otimes w^{(k)}\right)(\varphi^{(k-1)} \otimes w^*)\,dxdy = 0, \qquad \forall w^* \in L^2(Y).$$

We turn now to the evaluation of $\varphi^{(k)}$. A variational form of line 4 $\chi^{(k)} = \mathcal{B}^* w^{(k)}$ reads

$$\int_X \chi^{(k)}(x)\varphi^*(x)\,dx = \int_X (f(x,\cdot),w^{(k)})_{L^2(Y)}\,\varphi^*(x)\,dx, \qquad \forall \varphi^* \in L^2(X).$$

Since $\|w^{(k)}\|_{L^2(Y)} = \sigma^{(k)}$ we derive

$$\int_{X\times Y} \left(f - \frac{1}{(\sigma^{(k)})^2}\chi^{(r)} \otimes w^{(k)}\right)(\varphi^* \otimes w^{(k)})\,dxdy = 0, \qquad \forall \varphi^* \in L^2(X).$$

Remark. In the last equation, the function $\frac{\chi^{(k)}}{(\sigma^{(k)})^2}$ converges toward φ_0. As a result, the following limit holds

$$\lim_{k\to\infty} \frac{\lambda^{(k)}}{(\sigma^{(k)})^2} = 1.$$

Once the dominant singular value σ_1 with its corresponding modes (φ_1, w_1) are approximated, one has to evaluate the following ones. The *deflation-based power iteration process* succeeds in doing so. Assume the first modes $(\sigma_n, \varphi_n, w_n)$ with $n < N$ are known, then compute the next mode $(\sigma_N, \varphi_N, w_N)$. The deflation mechanism is described first for the computation of (σ_N, φ_N) as the eigen(vector, value) of the operator $\mathcal{R}$. Let

$$\tilde{\mathcal{R}} = \mathcal{R} - \mathcal{R}_{N-1} = \mathcal{R} - \sum_{1\le n<N} \lambda_n\varphi_n \otimes \varphi_n. \tag{1.3.18}$$

Then, the deflated iterate power method is given in algorithm 6

Algorithm 6: Deflated Power Iteration method (eigen value problem)

input : $\mathcal{R}$, required number of modes N
output: Eigenvalues and vectors, $\{\lambda_n, \varphi_n\}_{n>0}$

1 n=0
2 **while** $n < N$ **do**
3 Choose $\varphi^{(0)} \in L^2(X)$ with $\|\varphi^{(0)}\|_{L^2(X)} = 1$
4 **repeat**
5 $\chi^{(k)} = \tilde{\mathcal{R}}\varphi^{(k-1)}$
6 $\varphi^{(k)} = \frac{\chi^{(k)}}{\|\chi^{(k)}\|_{L^2(X)}}$
7 **until** *convergence*;
8 $\{\lambda_n, \varphi_n\} = \{\|\chi^{(k)}\|_{L^2(X)}, \varphi^{(k)}\}$
9 **return** $\{\lambda_n, \varphi_n\}_{n\le N}$

In order to operate the deflated algorithm on the operator $\mathcal{B}$, the following result on the kernels is necessary

Proposition 1.3.2. *This equality holds*

$$(R - R_{N-1})(x, x') = \int_Y (f - f_{N-1})(x, y)(f - f_{N-1})(x', y)\, dy, \qquad \forall (x, \xi) \in X \times X.$$

Proof. Computations starts as follows

$$\begin{aligned}\int_Y (f - f_{N-1})(x, y)(f - f_{N-1})(x', y)\, dy = &\int_Y f(x, y) f(x', y)\, dy \\ &- \sum_{0 \le n < N} \varphi_n(x') \int_Y f(x, y) w_n(y)\, dy - \sum_{0 \le n < N} \varphi_n(x) \int_Y w_n(y) f(x', y)\, dy \\ &+ \sum_{0 \le n < N} \sum_{0 \le k < N} \varphi_n(x) \varphi_k(x') \int_Y w_n(y) w_k(y)\, dy\end{aligned}$$

Various orthogonalities yield that

$$\begin{aligned}\int_Y (f - f_{N-1})(x, y)(f - f_{N-1})(x', y)\, dy = &\int_Y f(x, y) f(x', y)\, dy \\ &+ \sum_{0 \le n < N} \lambda_n \varphi_n(x) \varphi_n(x') - 2 \sum_{0 \le n < N} \lambda_n \varphi_n(x) \varphi_n(x') \\ = &\int_Y f(x, y) f(x', y)\, dy - \sum_{0 \le n < N} \lambda_n \varphi_n(x) \varphi_n(x') = (\mathcal{R} - \mathcal{R}_{N-1})(x, x').\end{aligned}$$

The proof is complete. □

Now, we introduce the deflated operators $\tilde{\mathcal{B}}$ defined by

$$\tilde{\mathcal{B}} = \tilde{\mathcal{B}} - \tilde{\mathcal{B}}_{M-1} = B - \sum_{1 \le k < M} \lambda_k \varphi_k \otimes w_k.$$

Corollary 1.3.2.1. $\tilde{\mathcal{R}} = \tilde{\mathcal{B}}^* \tilde{\mathcal{B}}$ *holds.*

Proof. After observing that that kernels of $\tilde{\mathcal{R}}$ and $\tilde{\mathcal{B}}$ are $(\mathcal{R} - \mathcal{R}_{N-1})(x, x')$ and $(f - f_{N-1})$ respectively, it is a direct consequence Proposition 1.3.2. □

The power iterations on $\mathcal{R}$ can be written as in algo. 4 for the approximation of λ_N. Then, each iteration can be split into two steps as in algo. 5 where the operator $\tilde{\mathcal{B}}$ plays the central role. Both versions are necessarily equivalent. Detailing in the same way as for the dominant singular value, one obtains

$$\int_Y w^{(r)}(y) w^*(y)\, dy = \int_Y \left((f - f_{N-1})(\cdot, y), \varphi^{(k-1)} \right)_{L^2(X)} w^*(y)\, dy, \qquad \forall w^* \in L^2(Y),$$

that also reads

$$\int_{X \times Y} \left(f - f_{N-1} - \varphi^{(k-1)} \otimes w^{(r)} \right) \left(\varphi^{(k-1)} \otimes w^* \right) dx dy = 0, \qquad \forall w^* \in L^2(Y). \tag{1.3.19}$$

One can recognize (1.3.6). The next line of the algorithm is the calculation of $\varphi^{(k)}$, it is conducted as follows

$$\int_X \chi^{(k)}(x) \varphi^*(x)\, dx = \int_X \left((f - f_{N-1})(x, \cdot), w^{(k)} \right)_{L^2(Y)} \varphi^*(x)\, dx, \qquad \forall \varphi^* \in L^2(X).$$

or equivalently

$$\int_{X\times Y}\left(f - f_{N-1} - \frac{1}{(\sigma^{(k)})^2}\chi^{(k)}\otimes w^{(k)}\right)(\varphi^*\otimes w^{(k)})\,dxdy = 0, \qquad \forall\varphi^*\in L^2(X). \quad (1.3.20)$$

The $\varphi^{(k)}$ is then defined by the normalization of $\chi^{(k)}$.

Finally equations 1.3.19 and 1.3.20 are identical to the weak *a posteriori* PGD algorithm presented in section 1.3.1. This construction of the deflated power iteration (DPI) is based on a POD and equivalent SVD formulation. Thus one can conclude that in the bivariate framework the *a posteriori* PGD is equivalent to a POD solved through DPI algorithm. It should also be noted that this section has also offered and equivalence between the usual POD and a singular alternative that one may identify to a continuous SVD (algorihtm 4 versus 5). Thus three algorithm families emerge to separate bivariate data : SVD , POD/EV and PGD/DPI solvers.

Additionally the interpretation of the PGD as a DPI algorithm open the field of improving PGD algorithms through the vast knowledge on deflation algorithm, first in the *a posteriori* framework, then in the more complex *a posteriori* framework. One among the attractive features common to numerous iterative solvers (Richardson, Gradient, Arnoldi, GMRES, etc.) of linear (and even nonlinear!) systems resides in their capacity to come up with the solution without accessing the related full matrix at once. Being able to operate that matrix on vectors is sufficient to start and end up those iterative solvers.

Such improvements of the PGD have been investigated in the literature, some interesting papers [TLN14, ACL15, CKL13] among many others.

1.4 Numerical experiments

In this section a few numerical tests are conducted on all three methods. Although it has been shown that they are mathematically equivalent, the difference between these algorithm will inevitably produce different behavior, especially for ill-conditioned problems/matrices. This first numerical section provides a suggested technique over the others depending on the problems studied. First some synthetic data is used i.e. analytical functions, then an image is compressed with various levels of accuracy. Finally, data from numerical simulations is separated.

Here we briefly recall the problem and the measure of success to solve it, i.e., the approximation error.

Find the best approximation of $f(x,y)$ such as $f_r(x,y) = \sum_{i=1}^{r} X_i(x)Y_i(y)$

with the error measured as $||f - f_r||_{L^2}$ or $||f - f_r||_{\mathrm{F}}$ depending on the nature of the method[7].

[7] Actually the choice of the norm has little influence on the numerical results. This is especially true for trapezoidal rule on a Cartesian grid. The main purpose of this distinction is consistency and to some extent application to ROM in part II.

1.4.1 Synthetic data

Let $\Omega = [0,1]\times[0,1]$ be the studied domain and four square integrable functions $f_1, f_2, f_3, f_4 : \Omega \to \mathbb{R}$ defined by

$$f_1(x,y) = xy \tag{1.4.1}$$
$$f_2(x,y) = \frac{1}{1+xy} \tag{1.4.2}$$
$$f_3(x,y) = \sin(\sqrt{x^2+y^2}) \tag{1.4.3}$$
$$f_4(x,y) = \sqrt{1-xy} \tag{1.4.4}$$
$$f_5(x,y) = \frac{1}{(1+xe^y)} \tag{1.4.5}$$

These functions range from already separated (f_1) to weakly separable, also known as *singular* functions in the literature. Thus these two expressions will be used indifferently in this manuscript. They are chosen to be easily extended to multiple variables.

The four methods PGD, POD ($L^2(\Omega)$) SVD and `SVD_by_EVD` are applied on these functions for a 32×32 regular Cartesian grid on a single processor *Fortran* or *python* implementation. Integrals are computed with trapezoidal rule.

Modes shape. It is not a trivial task to interpret the modes computed by these methods especially for (x, y) as shown in Fig. 1.4. First we focus on Fig. 1.4a and 1.4b which shows the first modes in for both variables yielded by all three methods. As expected, since l^2 scalar product is used, all methods yield the same normalized modes (Fig. 1.4a and 1.4b) for x and y since f_3 is symmetric. In Fig. 1.4c, one can see that the amplitudes of the PGD modes is plummetting with i, this is simply caused by the definition of the PGD sequence (see eq. (1.3.1)) which transfers the "relative weight" of a couple of mode to the last coordinate i.e. Y_i for the bivariate problem. The decay here is very fast since this function is easily separable. Finally, Fig. 1.4d displays the same modes obtained through POD, thus of norm 1. The apparent lack of smoothness is due to the coarseness of the mesh but it affects very weakly the accuracy[8]

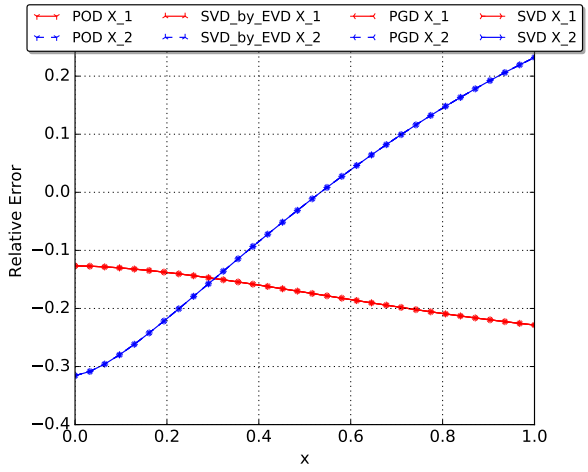
Typical Decrease in approximation error and singular values. The bivariate functions can be sorted in two groups with respect to these decomposition techniques.

Definition 1.4.1 (Exponentially Separable function)**.** *A function is called* exponentially separable *if the decrease in the singular values, thus in the approximation error, is exponential. In other words, a* semi-log *plot of the error is a straight line, regardless of its slope.*

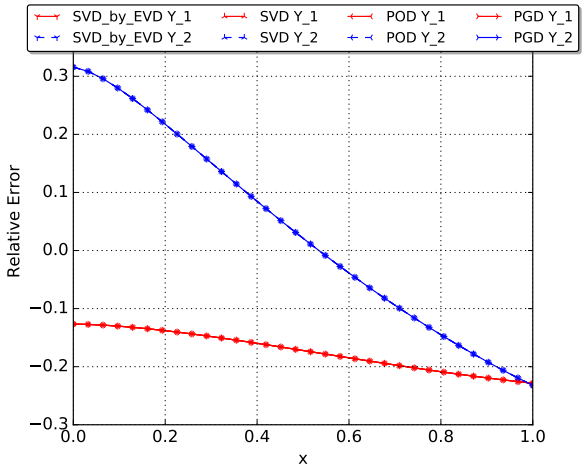
Definition 1.4.2 (Linearly separable function)**.** *A function is called* linearly separable *or weakly separable if the decrease in the singular values, thus in the approximation error, is linear. In other words, a* log-log *plot of the error is a straight line, regardless of its slope.*

These definition will be extended directly to multiparameter functions. Typically, weakly separable function are produced by highly non-linear processes or functions that

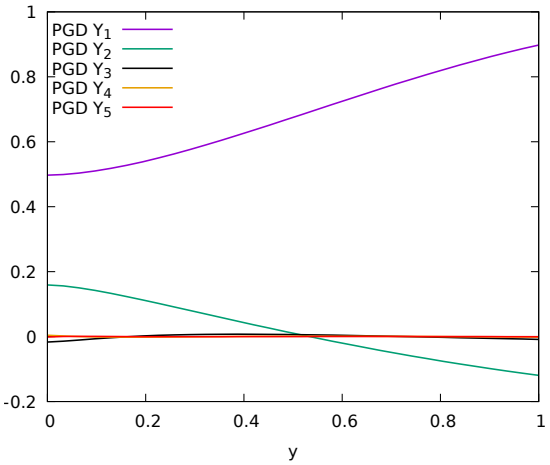
[8] For instance using several grids from 16×16 to 1024×1024 has shown no improvement in the approximation error for 5 modes and a slight decrease in the accuracy with finer grid when using 10 modes. Thus as long as the sampling is fine enough to capture the features of the field $f(x,t)$ refining the grid does not improve the accuracy of the representation at the sampling points.



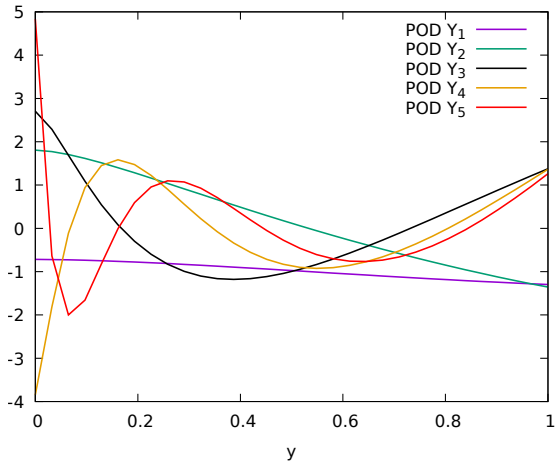
(a) Normalized modes X_1, X_2 with each method.



(b) Normalized modes Y_1, Y_2 with each method.



(c) Modes $(Y_i)_{i=1}^5$ obtained with PGD.



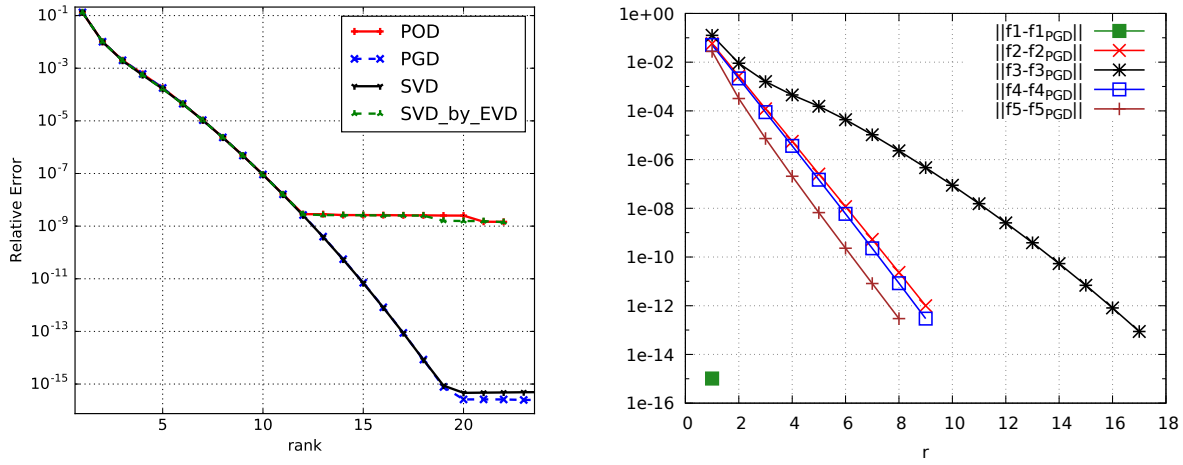
(d) Modes $(Y_i)_{i=1}^5$ obtained with POD.

Figure 1.4: Decomposition modes of f_3.

display a sharp singularity. Thus singular function is often used to replace weakly separable in the literature as well as in this manuscript. Additionally, various levels of separability may be observed depending on the nature of the function. A moderate slope will often be referred to as less separable and an almost linear decay declared weakly separable. Finally, some peculiar function may show two different regions (relative to r) with distinct behavior such as first a sharp exponential decay followed by a milder linear one. This generally fits the properties of the function such as length scale or turbulent behavior in fluid dynamics.

As mentioned in the theoretical paragraphs, a very efficient way to measure the separability of a field is to observe the decay of the singular values. It is also a reliable way to estimate the error decay. Fig. 1.5a presents a comparative view of the decay of the approximation error for f_3 which is a very common function for testing this property. The singular values are not displayed as their behavior is very similar to the error. All four methods are equivalent up to $r \simeq 12$ which is in agreement with the mathematical equivalence shown in the theoretical presentation. However for $r \geq 12$ it seems that the error is stuck in the 10^{-8} regions. This is explained by the ill-conditioning[9] of matrix C

[9] As the conditionning is defined by the ratio of the largest and smallest singular values, it is obvious that the conditionning is very poor since the singular values range from $\mathcal{O}(1)$ and $\mathcal{O}(10^{-16})$.



(a) 2D decompositions error of f_3.

(b) PGD applied on a pool of functions.

Figure 1.5: Approximation error (L^2 or Frobenius norm) for bivariate methods

in eq. (1.2.30) that causes a loss or the orthonormality property of the POD/SVD basis. Most importantly, this is due to the limited computer precision for solving the intermediate eigen value problem for `SVD_by_EVD` and POD. Since $\lambda_{\min} \approx 10^{-16}$ and $\sigma_i = \sqrt{\lambda_i}$, the smallest singular value is $\sigma_{\min} \approx 10^{-8}$ which is also approximately the approximation error.

Table 1.1: Numerical orthonormality of the snapshot POD basis for f3

i	$\|\|X_i\|\|_{L^2}$	$\|\|Y_i\|\|_{L^2}$	(X_i, X_{i-1})	(Y_i, Y_{i-1})
1-8	1.0000	1.000	$\mathcal{O}(10^{-16})$	$\leq 10^{-8}$
9	0.999999	0.9999	4.02E-16	-2.93E-7
10	0.999999	1.000	-1.52E-16	5.16E-6
11	0.999999	1.000	1.68E-16	**-6.77E-6**
12	0.999999	1.000	1.66E-16	**-3.19E-3**
13	1.00000	0.9999	6.24E-17	**0.558**
14	0.999999	0.9999	-8.76E-17	0.934
15-32	0.999999	0.9999	$\mathcal{O}(10^{-16})$	$\mathcal{O}(1)$

Table 1.1 presents a test of the orthogonality of the basis obtained through POD. One can see that the `dsyev` routine preserves the orthonormality of the $\{X_i\}$ basis (the one it is directly computing) but that $\{Y_i\}$ gradually looses orthonormality as i grows. The transition from suitable orthogonality to none takes place on a very limited number of modes, here from 11 to 13 this property is lost with the consequence that the accuracy of the representation reaches a threshold. An efficient solution to overcome this limit is to use a method that ensures this property. One can choose a reorthogonalization technique such as Gram-Schmidt orthogonalization process to the POD basis or alternatively as proposed here rely of recursive algorithm to compute both bases, namely the PGD.

As one can see in Fig. 1.4d, in this case as long as the process converges[10] the approximation error decreases (exponentially here) as the number of enrichment grows. Then we

[10] Convergence of the PGD fixed point alternating direction method is not ensured (especially for weakly separable functions) and may be improved with gradient research for instance. However stopping it at a reasonable number of iteration e.g. 10 or 20, has proven efficient in the many numerical experiments

Table 1.2: Comparison of POD, PGD and SVD for a target error of $\epsilon = 10^{-6}$.

	SVD			PGD			POD		
	n	σ^{n+1}	error	n	σ^{n+1}	error	n	σ^{n+1}	error
f_1	1	1.27E-15	6.68E-16	1	≈ 0	1.02E-15	1	9.22E-17	3.45E-16
f_2	5	7.91E-6	2.96E-7	5	2.12E-7	2.55E-7	5	2.12E-7	2.55E-7
f_3	9	1.04E-5	4.73E-7	9	3.20E-7	4.67E-7	9	4.67E-7	3.20E-7
f_4	4	1.03E-5	2.45E-7	4	2.74E-7	2.07E-7	4	2.74E-7	2.07E-7
f_5	5	3.32E-6	1.67E-7	5	9.20E-8	1.49E-7	5	9.20E-8	1.49E-7

can conclude that this function is separable. Up to $r = 12$ one may choose any of the three presented method as the result are extremely similar. However for the next experiment, shown in Fig. 1.4c, functions f_1 to f_5 have been separated using only the PGD. One can see that all these functions are separable although two functions stand off. $f_1 = xy$ is already separated and the PGD only requires 1 mode to represent it to the machine error. f_3 seems to be less separable than the others as its slope is lower. Nonetheless it clearly displays an exponential decay as the section from $r = 12$ to 17 is straight.

Comparison of the methods. Last the mathematical equivalence of the four methods is tested on the least separable of our synthetic function. Fig. 1.4 and 1.5 let us think that they are also equivalent numerically, at least as long as the POD/SVD is properly solved. Table 1.2 further confirms that statement. Indeed, one can see that the number of modes to reach the target error of 10^{-6} is always the same for all three methods and the observed error is also very close. Meanwhile the σ^{n+1} depend on the scalar product used in these methods which is why POD and PGD versions are close while SVD singular values are 1 or 2 orders of magnitude bigger.

Consequently, one may choose whichever of these three method to separate a bivariate function. However, this must be done knowing the relative limitations of these functions. Having both available in code is highly advisable as their advantages are situation related.

1.4.2 Image compression by decomposition

As stated in the SVD section, these techniques can be used on any kind of data. An interesting example while presenting the data compression aspect of these methods is to apply it to images. Indeed it is efficient to compress large images. Indeed numerical images are stored in many formats but it always boil down to an array of integers representing colors. Let us consider the simpler case of grayscale images, usually stored in 1 byte per pixel. That is to say, the original 4000×3000 pixels grayscale image `"singapore.tiff"` used in Fig. 1.6 is a matrix of the same size which coefficients are integers in $[\![0, 255]\!]$ which means 12×10^6bytes $\approx$ 12Mb without compression. Table 1.3 gives the compression rate for different number of SVD modes retained as displayed in Fig. 1.6. One can see easily that preserving very few modes yields high levels of compression but the image features are not preserved. Indeed, on can see in Fig. 1.6 top two lines[11] that keeping

that I have ran during this research. The "remaining part" of the basis is "caught" by the next enrichment step in an adjustment pattern.

[11]The reader is advised to follow this description in the PDF version as it allows zooming of the row of small pictures.

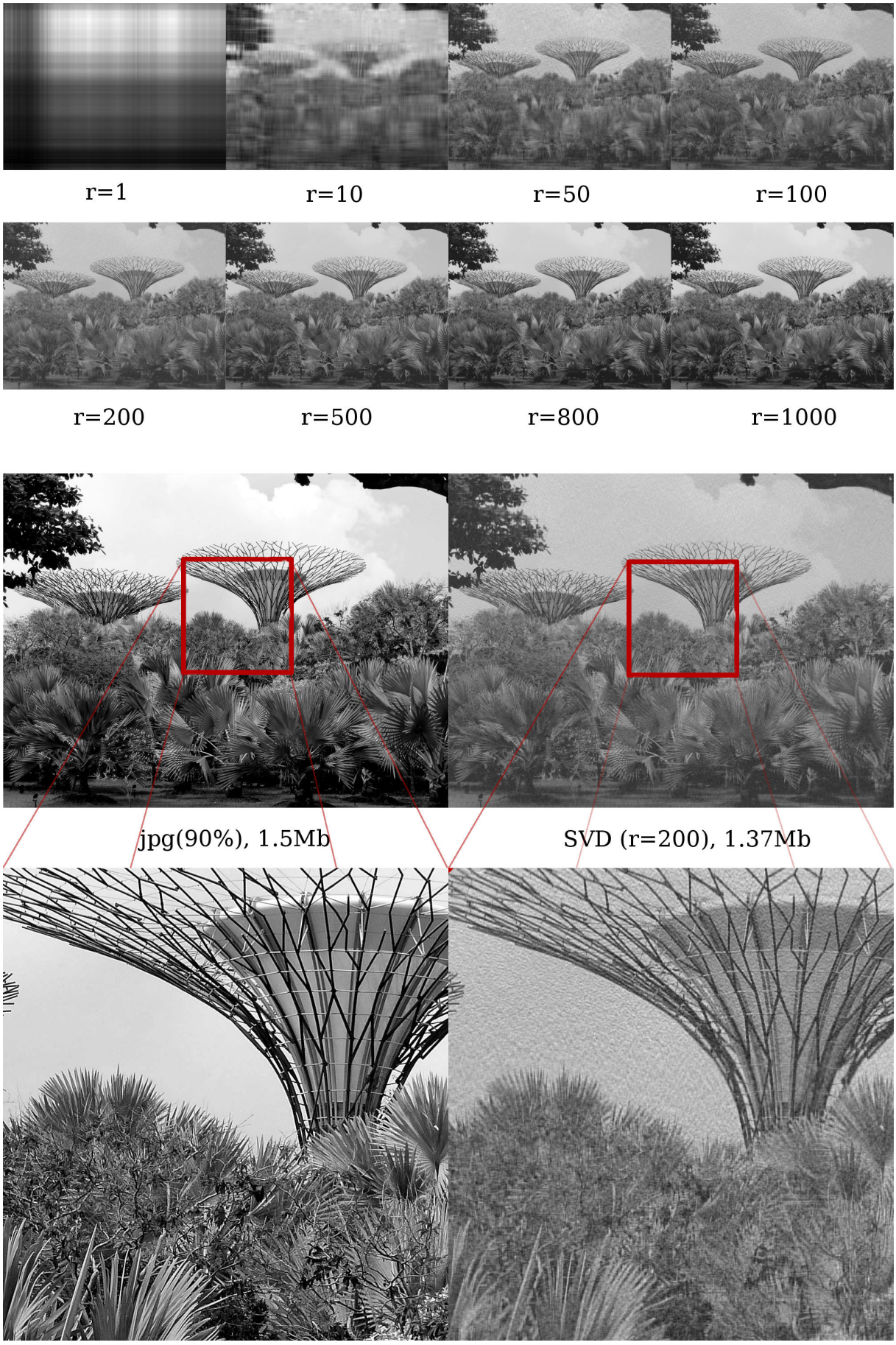


Figure 1.6: A 4000×3000 pixels picture of Singapore Gardens by the Bay compressed through SVD as compared with JPEG compression.

only one mode gives a unrecognizable image. Increasing number of retained modes r leads to gradually better representation, 10 modes is sufficient to perceive the big structures of the image. The big leaves and sharp metallic structures are captured with 50 modes while 100 modes is enough to distinguish palm leaves. This behavior continues up to a few hundreds where all human-eye relevant structures are captured by the SVD compressed image. However at $r = 200$, the image is grainy (especially visible in the sky part) which is striking in the larger SVD image and close-up in the lower part of Fig. 1.6. Adding more and more modes reduces the noise of the image, at $r = 1000$ it is hard to tell that the image has been compressed without any reference point, while the size of the image is still halved as compared with the original uncompressed file. The only difference lies in the contrast level as one can see that the very dark and very bright regions of the image are not as deep as in the original image.

r	SVD size (Mb)	CR (%)	Err. (%)
1	0,01	99.9	41.5
10	0,07	99	31.2
50	0,33	97	25.7
100	0,67	94	22.2
200	1,34	89	17.2
500	3,34	72	9.4
800	5,34	55	5.2
1000	6,68	44	3.2

Table 1.3: Compression rate using SVD on 4000×3000 pixels grayscale image. Where CR is the compression rate and the error is computed with Frobenius norm.

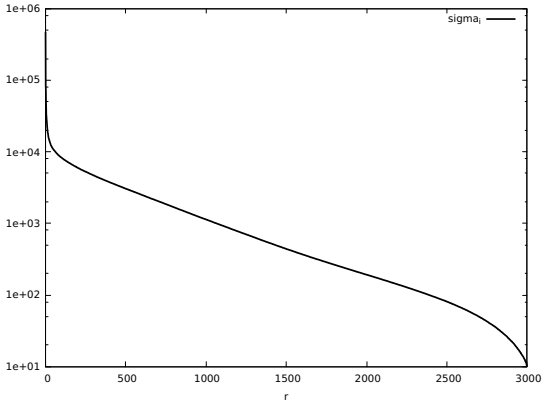


Figure 1.7: Singular values of `"singapore.tiff"`

A very interesting feature of this data lies in the very slow decay of the singular values, shown in Fig. 1.7. Indeed is was chosen on purpose so that no clear directional pattern appeared in the image and all length scales were present. Consequently, the first 50 singular values plummet then the slope become a lot milder with a decay of one order of magnitude per thousand modes. One can assert that the first exponential decay, associated with the large structures of the image, is followed by a linear one due to the profusion of small scales. This is the first example of this behavior shown in this thesis. It will appear again in complex flows and physics problem, either in 2D or 3D. As usual, if all modes are kept the image is exactly recovered. However, there is overhead in the storage space as U is of the same size as the original data and one still needs to store V and Σ.

To conclude on the image compression abilities of SVD, it is fairly efficient for large images as the ratio r/n_{pix} is very small but the method is not well suited for human-eye use. The Frobenius error presented in table 1.3 does not fit with the human experience of the image produced by SVD comparison. Indeed, SVD compares poorly with well established formats such as JPEG which was specifically designed to retain eye sensitivity such as contrast, color depth, etc.

1.4.3 Physics problem data decomposition

We have shown in the previous examples a series of properties of the bivariate decomposition. Now, we focus on data from physics, in particular data obtained by numerically solving partial differential equations (PDEs). A single example is presented as most data from fluid dynamics problem share similar decomposition pattern.

1.4.3.1 Data decomposition of a singular lid driven cavity flow

In this subsection, a brief analysis of the separation properties of the POD on the classical instability problem of a singular lid driven cavity (LDC) at high Reynolds number is given. Further detailed on this very complex flow are given in chapter 5.

The 2D LDC problem is defined on a square domain $\Omega = [0,1] \times [0,1]$ on a time domain $\mathcal{T} = [0,T]$. The upper side of the domain is moving rightward at constant speed U while all other walls are immobile as shown in Fig. 1.8. We chose a "high" Reynolds number, here $Re = 9000$, which means above the first Hopf bifurcation (see 5 for more details) i.e. the flow is unstable. The fluid is at rest at t=0. The data is obtained through a Cartesian grid high accuracy CFD code developed by T.K Sengupta's team (more details available in section 5.1 and [LBA+18]. The vorticity formulation of the Navier-Stokes equation is used, i.e. $\omega = \nabla \times u$, ultimately the non-dimensionalized problem reads

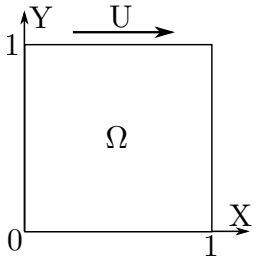


Figure 1.8: Schematic view of the LDC

$$\begin{cases} \nabla^2 \psi = -\omega \\ \frac{\partial \omega}{\partial t} + (\vec{V} \cdot \nabla)\omega = \frac{1}{Re} \nabla^2 \omega \end{cases} \tag{1.4.6}$$

The nature of this flow is very complex and is detailed later in section 5.1. Nevertheless a sample of the vorticity field at time $t = 1900.2$ for $Re = 9800$ is given in Fig. 1.9. The reader can see that the flow is mainly composed of three zones.

Drive This is the dark blue and red region at the top of the cavity. It is characterized by high amplitude vorticity and high shear in the flow, especially near the top right angle.

Core This is the green region of the flow displays very little variation as the exponential contour line levels highlight. Indeed the green color is limited to ± 0.3 around the value of the center of the cavity. This part of the flow has been shown by T.K. Sengupta et al. [SLV09] to display triangular vortices using high accuracy NCCD scheme. These triangular vortices have also been observed for real fluids [CK94, BvH98]. This is the region that presents the most interest for POD analysis as it is complex and very sensitive to numerical error.

Edges/Corners This regions is composed of the three remaining edges and "corner" zones. As for the drive region, they display high levels of vorticity but shear is usually lower since the fluctuations in vorticity are less dramatic. The usual streamlines plot for the LDC would show (nested) recirculation at both lower corners.

In order to have the best accuracy in the POD decomposition, a centering of the vorticity is performed, then all results presented subsequently are produced from the fluctuation of the vorticity field $\omega' = \omega - \bar{\omega}$ where $\bar{\omega}$ the time average of ω is given by

$$\bar{\omega} = \frac{1}{T} \int_{\mathcal{T}} \omega$$

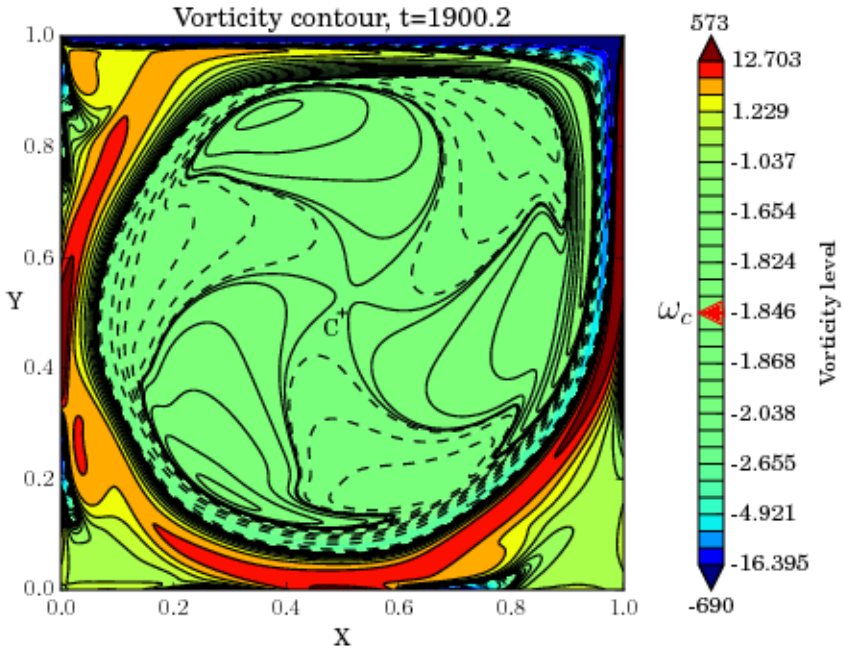


Figure 1.9: Vorticity contour of LDC DNS at $Re = 9800$, $t = 1900.2$. The vorticity contour lines are "centered" to $\omega_c = \omega(0.5, 0.5) \approx -1.84$ to emphasize the orbiting triangular structures in the middle,

Here the POD is applied on the bivariate scalar field $\omega'(\boldsymbol{x}, t)$ in a snapshot fashion, the scalar product used is a measure of enstrophy $||\omega'||^2_L(\Omega)$ instead of the historic approach of kinetic energy [HLB96, NAM+03, Sir87]. In vortex dominated inhomogeneous flows, rotational energy is a better descriptor of POD over translational kinetic energy, as highlighted by Sengupta et al. work [SDS03, Sen12]. Then the correlation matrix is

$$C(t, t') = \int_\Omega \omega'(\boldsymbol{x}, t)\omega'(\boldsymbol{x}, t')\, d\boldsymbol{x} \tag{1.4.7}$$

which is numerically treated by a 2D trapezoidal rule.

Finally, to ease the presentation the POD approximation of the vorticity of rank r reads

$$\omega_r^{\text{POD}}(\boldsymbol{x}, t) = \sum_{i=1}^{r} \phi_i(x) a_i(t)$$

where ϕ_i is normalized ($||\phi_r||_{L^2} = 1$) and the weight of the mode is carried by a_i. In this section the time interval for the POD of the vorticity field is taken in the stable limit cycle range (see 5.1) $T = [1900, 1940]$ with 200 equally spaced snapshots.

Fig. 1.10 shows the decay of the POD approximation error with the number of modes, up to $r \approx 14$ (region A), the decay is exponential while it is linear right to the dashed line (region B). Looking closely, one can see that the decay is actually occurring by pairs of modes in the exponential part. This is due to the nature of flow as the representation of the different physical frequencies yields two POD modes. Indeed one can see that the time modes of Fig. 1.11 are almost identical functions, only separated by a quarter period phase shift. Their amplitude i.e. the measure of their relative contribution to enstrophy is also very close within pairs but changes of magnitude between pairs. Finally the frequency is doubled for every added pair. This behavior is also followed by the spatial modes shown in Fig. 1.12. The number of structures in the core is doubled for every pair and one can

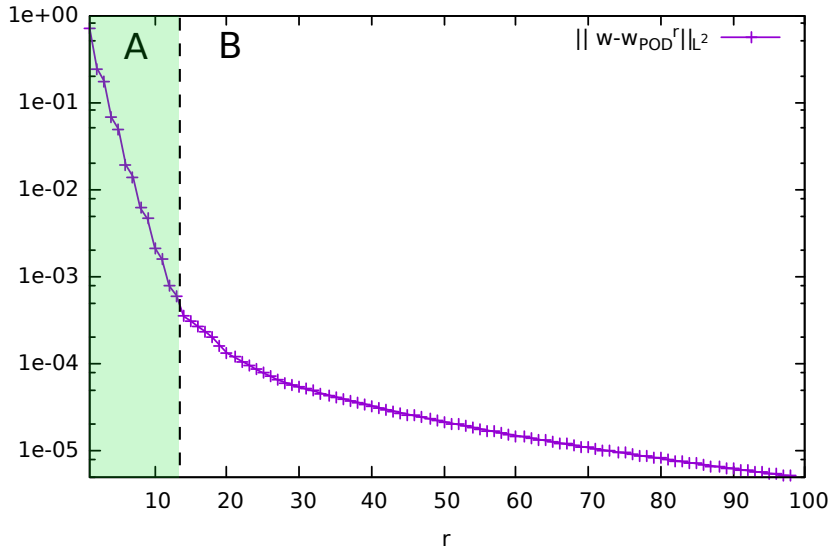


Figure 1.10: POD approximation error decay with the number of modes.

see that the structures are slightly shifted between two members of a pair. The linear decay is associated to numerical noise of the simulation, one can infer that variations below the threshold $r = 14$ or vorticity variation smaller than 10^3 are not representative of the physics of this flow.

This unstable flow example will be studied thoroughly using POD in section 5.

1.4.4 Numerical issues and proposed improvements

The bivariate techniques presented in this section are the foundation of all multivariate methods described in this manuscript. Then it is crucial to target the numerical shortcomings that have been observed. Mainly, one can sort them as accuracy issues (the main focus of this work) and computing efficiency issues. Some solution solve both these problems but may interfere with programming ease. The main difficulty lies in solving the eigenvalue problem (1.2.7) in its various formulations.

Accuracy issues For applications such as building ROMs from separated fields, the accuracy of the basis must be ensured since properties such as orthogonality of the basis are fundamental in many cases (Galerkin projection, etc.). To a lesser extend, loss of orthonormality may prevent the decomposition to converge. Moreover, many of the mathematical properties of these bases rely on the orhtonormality of the bases.

It was observed that the orthogonality is not always preserved for low energy modes. This is generally explained by the poor conditioning of the correlation matrix, typical values of the condition number $\mathcal{K}(A) = ||A^{-1}|| \cdot ||A|| = \frac{\sigma_{\max}}{\sigma_{\min}}$ have been evaluated to 10^{16}. This result is actually expected since the decomposition yields singular values ranging from $\mathcal{O}(1)$ or more to $\mathcal{O}(10^{-15})$. Usual preconditioning techniques such multiplying by the diagonal prove inefficient to reduce the condition number as well as the orthogonality of the basis.

As preconditioning is hopeless due to the nature of the studied matrices, one should carefully choose the eigenproblem solver. For instance, `LAPACK`'s `DGEEV` does not preserve the orthogonality of the eigen vectors beyond the sixth or seventh mode, on the other hand, `LAPACK`'s `DSYEV`, that uses symmetric matrices properties, ensure orthogonality to machine precision. However, the secondary basis (the one obtained by projection of the data on

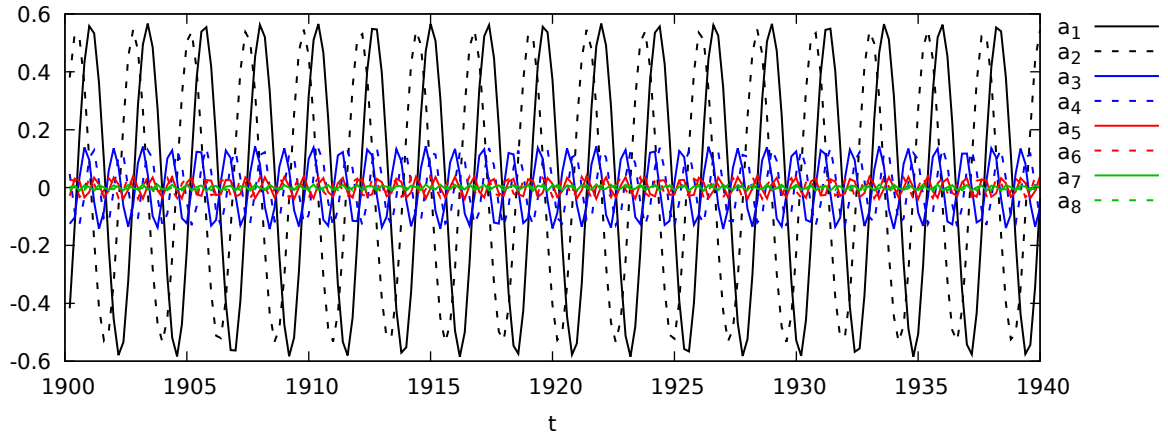


Figure 1.11: The first 8 time modes obtained through POD of the vorticity disturbance field ω' of the LDC for $t \in [1900 : 1940]$. The norm of the couples $\{ \phi_i, a_i \}$ is stored in a_i which is why they tend toward zero.

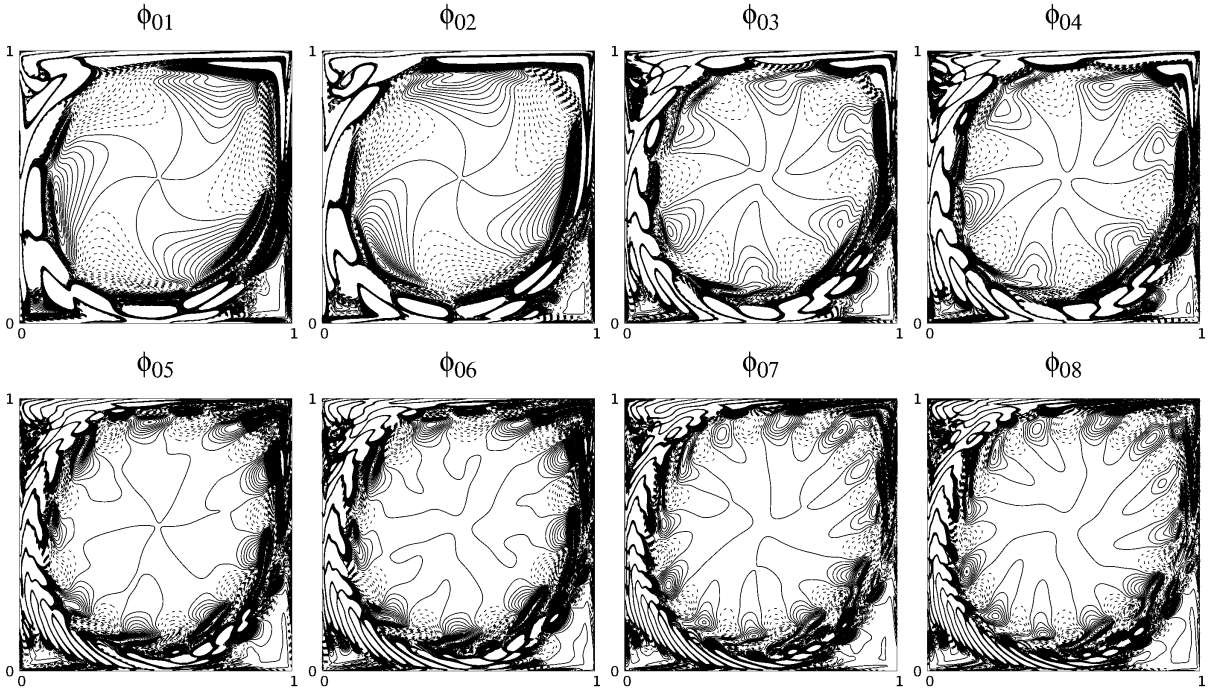


Figure 1.12: The first 8 spatial modes contour obtained through POD of the vorticity disturbance field ω' of the LDC for $t \in [1900 : 1940]$.

the primary basis) does not ensures orthogonality as the modes number grows as shown in Tab. 1.1. Synthetic data exacerbate this phenomenon as numerical simulation output do not provide data with sufficient accuracy to preserve lower modes. Other algorithm were tested performs variably well, `DSYEV` being the upper bound for accuracy. Consequently one must always check the accuracy of the orthogonality if this property is used for further development. For compression purpose this assumption is secondary and the only measure of efficiency is the compression rate.

Efficiency issues. It clearly appears that solving the full eigen problem is generally inefficient regarding the operation count. Indeed, the nature of these decomposition strategies is to truncate the separated representation to the smallest number of modes possible. Then, computing all modes to throw away the vast majority of them, as one would do us-

ing `LAPACK` routines, is inappropriate. However, to the best of my knowledge, most partial eigenproblem solvers rely on iterative processes that are very efficient for sparse matrices or when the user requires very few ($\mathcal{O}(10)$) eigenvectors. In our framework, it is usually interesting (sometimes compulsory) to compute tens of modes to represent accurately complex datasets. So far using full direct `LAPACK` solvers on correlation matrices have proven quick. Regarding our PGD based iterative solver, it turns out that it's efficiency for numerical simulation output of reasonable size ($\mathcal{O}(100MB)$) is poor. Computing time for 10 modes is orders of magnitudes longer than its `LAPACK` counterpart to obtain the full basis. Scaling to bigger problem is simply out of range with the proposed implementation as the number of numerical integration is too big. One way to circumvent these issues is to make sure one solves the eigen problem on the smallest of the dimension, usually using a snapshot method for CFD outputs.

Memory overload. The main limitation to the current version of the library, is actually memory use as some datasets don't fit in `RAM` typically $\mathcal{O}(100GB)$ is overloading memory even on large memory computing nodes.

Proposed Improvement. The following points are possible directions for solving these issues and improve the library.

- Implement a version of the library that does not require loading the full dataset as to compute the decomposition, rather loading periodically chunks of the data to compute parts of the eigen-problem. Iterative methods are the most promising candidates. One could use a block power iterate (that would also allow *parallel computing*) or some improvement of it such as Arnoldi although the implementation for this kind of application might be more difficult. However, the implementation of such techniques must be thought carefully as the alternate direction scheme (PGD) has proved inefficient for large problems.

- In case of overload of `dsyev` processing capacity, one can rearrange data in a multivariate fashion and use tensor decomposition technique. For example instead of separating time and space, one could also separate x and y thus making a trivariate decomposition.

- Implement a parallel version of these algorithms to overcome memory limitation as well as reducing computing time.

Conclusion

In this section, three bivariate data decomposition approaches were presented: SVD, POD and PGD. It was shown that they are different algorithm that produce mathematically equivalent results in the sense that the modes obtained are the same, with the same decay of singular values. This is confirmed by numerical experiments (Fig. 1.4). But they require different computing time, PGD is thus discarded as soon as one requires several modes for large problems. The main difference between these methods lies in the choice of the inner product determines the set in which the orthonormality of the decomposition is ensured. On the one hand, one can choose SVD which is well suited for simple data compression as it does not require any knowledge on the data properties or any grid/integration scheme. Thus it allows to compress any kind of data from images Fig. 1.6 to CFD data. On the other hand, one can prefer POD if interested in inner products that are suited to physical

properties of the data. A typical example is the analysis of CFD data with an energetic norm ($L^2(\Omega)$) or a norm adapted to the properties of the underlying equation (e.g. $H^1(\Omega)$ for NSE) for further use in a ROM by Galerkin projection on the reduced basis.

The presented method have been implemented and thoroughly tested. Consequently they form the basic unit of the multivariate decomposition methods that presented in the next sections.

Chapter 2

Tensors and their approximation in the most common formats

Contents

2.1 Some basic tensor features 46

2.1.1 Tensor spaces 46

2.1.2 Overview of tensors of $\mathbb{R}^{n_1 \times \cdots \times n_d}$ i.e. multi-way arrays 47

2.2 Tensor Formats 51

2.2.1 Full format 52

2.2.2 Canonical format $\mathcal{C}_r$ 52

2.2.3 Tucker format $\mathcal{T}_k$ 53

2.2.4 Hierarchical Tucker format $\mathcal{H}_k$ 54

2.2.5 Conclusion on tensor formats 60

2.3 Tensor decomposition 61

2.3.1 CP decomposition 61

2.3.2 Tucker decomposition 64

2.3.3 Tensor Train decomposition 71

2.3.4 Hierarchical Tucker decomposition 75

Tensors can be viewed as generalization of matrix to higher dimension i.e. an order d tensor is a d-way array or a function of d arguments. Such object rapidly become intractable, indeed for large $d > 3$, data size n^d is out of reach even for the most advanced computers and will remain that way for direct handling. A simple example of the *curse of dimensionality* is to take $n = 2$ and $d = 50$, although it appears to be of reasonable size, $n^d \approx 10^{15}$. This is of course far below the requirement of many scientific areas such as chemometrics, Boltzmann equation, multiparameter PDEs etc. This has led to the introduction of reduction techniques to overcome the *curse of dimensionality* starting with Hitchcock in 1927 [Hit27]. Many work has been separately performed in separate fields such as psychometrics (Tucker [Tuc66] and Caroll and Chang [CC70]) in the 1960s and 1980s or chemometrics from 1981 onwards ([AD81]). Since 2000, tensor decomposition has gained a lot of interest in many fields including solution of stochastic PDEs [DI09, KT11], solution of high dimensional Schrödinger equation, Boltzmann equation, computational finance, etc. Many more references are available in literature

surveys by Kolda and Bader [KB09] and Grasedyck et al. [GKT13]. Actually, these surveys together with W. Hackbush 2014 book "Tensor spaces and numerical Tensor calculus" [Hac14] demonstrate the growing interest for decomposition among the applied mathematics community. CFD has also seen numerous applications of such techniques due to its large production of data. Also, as we will see in part II of this manuscript, tensor decomposition techniques can be seen as the first stage of building multiparameter ROM.

This chapter is organized as follow. First, general concepts and definitions required for tensor reduction are given. The second section presents four of the most common tensor formats : canonical, Tucker, hierarchical and tensor train. These formats are not to be confused with their associated decomposition techniques and approximation that are described in the third section.

2.1 Some basic tensor features

This section will first address the issue of building a general framework that works equally well for continuous and discrete multidimensional problems. The concept of a tensor space structure and its main properties are described. In the second subsection the main features of tensors are presented on the particular case of multi-way arrays but are expendable to other kind of tensors. This dichotomy provides a general framework that will be needed in further development and eases the understanding of complex definition with the n-way array.

2.1.1 Tensor spaces

In order to build the approximation presented in the subsequent sections, a general framework is introduced. The mathematical framework we use in this thesis is based on W. Hackbush's book *"Tensor Spaces and Numerical Tensor Calculus"* [Hac14] with addition from other authors. Further details can be found in the original manuscript while we only cover the necessary notions for tensor decomposition.

Definition 2.1.1 (Tensor Space)**.** *Let V and W be vector spaces. The* ***algebraic tensor space*** *$\mathcal{V}$ is defined by*

$$\mathcal{V} = V \otimes_a W = \text{span}\{v \otimes w \ : \ v \in V, \ w \in W\} \tag{2.1.1}$$

Where $\otimes_a$ connects vectors spaces and $v \otimes w$ is an element of $\mathcal{V}$.

If a topological norm is given, the completion with respect to the given norm $\|\cdot\|$ yields the topological tensor space

$$V \otimes_{\|\cdot\|} W := \overline{V \otimes_a W} \tag{2.1.2}$$

This is then a Banach tensor space $(\mathcal{V}, \|\cdot\|)$.

Obviously, a tensor space is still a vector space however given a special structure.

Proposition 2.1.1. *Let V and W be vector spaces with respective bases B_V and B_W such that T be a tensor space over the field $\mathbb{R}$. A product $\otimes : V \times W \to T$ is a tensor product and T a tensor space, i.e., it is isomorphic to $V \otimes_a W$, if the following properties hold:*

i) span property : $T = \text{span}\{v \otimes w \ : \ V \in V, w \in W\}$

ii) bilinearity

iii) linearly independent vectors $\{v_i \ : \ i \in B_V\} \subset V$ *and* $\{w_i \ : \ i \in B_W\} \subset W$ *lead to independent vectors* $\{v_i \otimes w_j \ : \ i \in B_V, j \in B_W\}$ *in* T

Note that the tensor product is associative and *universal*, i.e.

Proposition 2.1.2 (Universality of the tensor product)**.** *For any multilinear map* $\varphi : V_1 \times \cdots \times V_d \to V$*, there is a unique linear mapping* $\Phi : \otimes_{j=1}^{d} V_j \to V$ *so that* $\varphi(v_1, ..., v_d) = \Phi(v_1 \otimes \cdots \otimes v_d)$.

It is possible to give an algebraic structure to a tensor space. Let the multiplication $\circ : A_j \times A_j \to A_j$ define a (non-commutative) algebra with a unit element 1. Then it is possible to define on $A =_a \otimes_{j=1}^{d} A_j$ an operation $\circ : A \times A \to A$ by means of

$$\left(\bigotimes_{j=1}^{d} a_j\right) \circ \left(\bigotimes_{j=1}^{d} b_j\right) = \bigotimes_{j=1}^{d} (a_j \circ b_j) \tag{2.1.3}$$

It is considered that the reader is familiar with the properties of Banach spaces as well as Hilbert spaces. Consequently, only results that are of particular interest for low rank tensor approximation are presented in this document.

Theorem 2.1.3. *Let* $(X, \|\cdot\|)$ *be a reflexive Banach space with a weakly closed subset* $\emptyset \neq M \subset X$*. Then the following minimisation problem has a solution:*

$$\forall x \in X, \text{ find } v \in M \text{ so that } \|x - v\| = \inf_{w \in M}(\|x - w\|) \tag{2.1.4}$$

This theorem is very useful to show that the low rank approximation of a tensor possesses a solution.

2.1.2 Overview of tensors of $\mathbb{R}^{n_1 \times \cdots \times n_d}$ i.e. multi-way arrays

In this section, a series of definitions and properties of the multi-way array tensors is provided. It should be noted that most of these definitions extend to other tensor spaces but most, if not all the work presented in this thesis manuscript uses discrete tensors. Reference article written by T.G. Kolda and B.W. Bader *"Tensor Decomposition and Applications"* in 2009 [KB09] set most of the terms, notations and definitions used in this section. The properties and definitions presented here are limited to the one necessary for approximation of tensors, many more work on tensors has been proposed in the literature [dSL08, Hac14].

First we introduce some notations. Let $d \in \mathbb{N}$ be the number of dimensions and $n_1, ..., n_d \in \mathbb{N}$ the number of entries along each of these dimensions. Let $D = \{1, ..., d\}$ be a tuple and $\mathcal{I} = \mathcal{I}_1 \times \cdots \times \mathcal{I}_d$ be a d-fold product index set with $\mathcal{I}_\mu = \{1, ..., n_\mu\}$

Definition 2.1.2 (Tensor)**.** *A* tensor *is a multidimensional array i.e. a d-way or* d^{th}*-order tensor is an element of the tensor product of d vector spaces, each of which has its own coordinate system.*

In terms of tensor space, here we have $\boldsymbol{\mathcal{X}} \in \mathbf{V} = \bigotimes_{a,\, i=1}^{d} V_i$ where $V_i = \mathbb{R}^{n_i}$. This notion of tensor is different from the many physical tensors which generally refer to a third order tensor that is defined in every points of the space. This forms a tensor field. Bold Euler script letters refer to order d tensors e.g. $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{\mathcal{I}}$.

Definition 2.1.3 (Order of a tensor)**.** *The order of a tensor is defined as the number of dimensions, also known as ways or modes.* $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ *where* $\mathcal{I} = \mathcal{I}_1 \times \cdots \times \mathcal{I}_d$*, is an order* d *tensor*[1]*.*

Remark. A first-order tensor is a vector, a second-order tensor is a matrix and a third order tensor or more is called a higher order tensor. A visual representation of a third order tensors is proposed in figure 2.1.

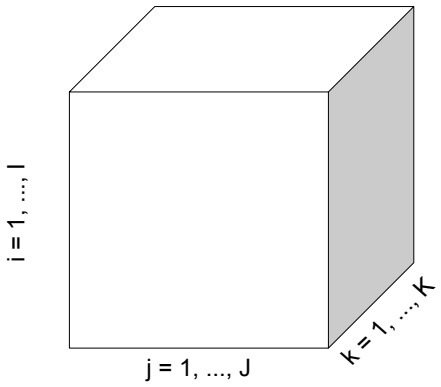


Figure 2.1: A third order tensor with $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$.

The entries of a tensor are denoted in the same fashion as for vectors or matrices i.e.

- entry i of vector $\mathbf{a}$ is a_i
- entry (i, j) of matrix $\mathbf{A}$ is a_{ij}
- entry $(i_1, i_2, \cdots, i_d)$ of order d tensor $\mathcal{A}$ is $a_{i_1 i_2 \cdots i_d}$

A subarray is formed when a subset of a tensor is taken e.g. subarrays of matrices are columns and rows. A colon is used to state that every element of a dimension is taken.

Definition 2.1.4 (Fibres)**.** *Fibres are the higher order analogue of matrix rows and columns. A fibre is defined by fixing every indices but one. Mode-1 fibre of a matrix is a column mode-2 fibres are rows and mode-3 fibres are tube fibres as shown in figure 2.2.*

Remark. Slices are two dimensional sections of a tensor defined by fixing every indices but two.

Definition 2.1.5 (Inner product and norm)**.** *Given two same-sized tensors* $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{\mathcal{I}}$*, the* Inner Product *is defined as follow*

$$\langle \mathcal{X}, \mathcal{Y} \rangle_F = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} x_{i_1 \cdots i_d} y_{i_1 \cdots i_d} \tag{2.1.5}$$

When there is no ambiguity on the nature of the inner product, the Frobenius inner product is simply noted $\langle \mathcal{X}, \mathcal{Y} \rangle$*.*

The norm associated with this inner product is the Frobenius norm defined by $\|\mathcal{X}\|_F = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle}$ *also*

$$\|\mathcal{X}\|_F = \sqrt{\sum_{i_1=1}^{n_1} \cdots \sum_{i_D=1}^{n_d} x_{i_1 \cdots i_d}^2} \tag{2.1.6}$$

[1]The order of a tensor is not to be confused with the rank of a tensor.

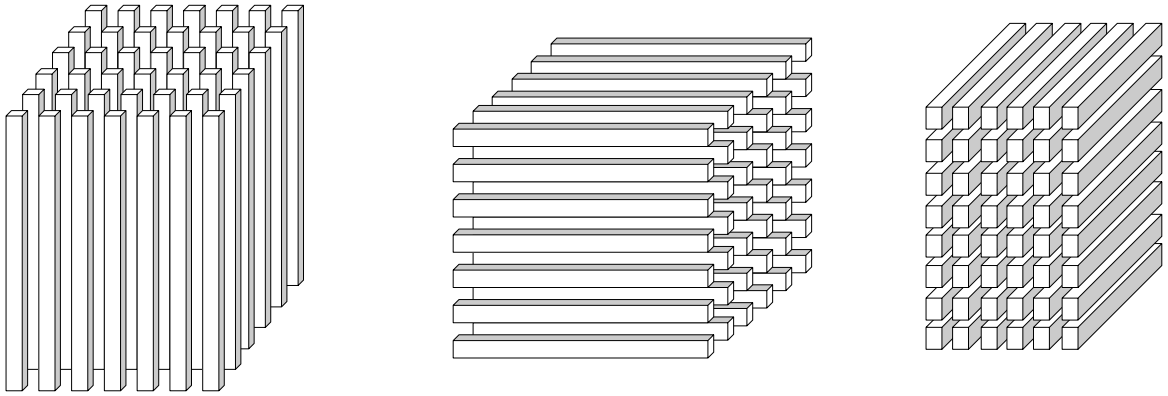


Figure 2.2: The fibres of a third order tensor.

Definition 2.1.6 (Rank-One tensor)**.** *An N-way tensor* $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ *is* rank-one *if it can be written as the outer product of* d *vectors* $(\mathbf{a}^{(j)})_{j=1}^{d}$*, i.e.*

$$\mathcal{X} = \mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(d)} \Leftrightarrow \forall\ 1 \leq i_j \leq n_j,\ x_{i_1 \cdots i_d} = \prod_{j=1}^{d} a_{i_j}^{(j)}$$

Definition 2.1.7 (Rank of a tensor)**.** *The rank of a tensor, denoted* $\text{rank}(\mathcal{X})$*, is the minimum number of rank-one tensor that generate* $\mathcal{X}$ *as their sum. In other words, this is the smallest number of components in an exact CP decomposition (see the definition 2.3.2). Further details are available in [KB09] concerning the link with the matrix rank.*

Remark. There is no straightforward way to determine the rank of a higher order tensor even for small sizes (the problem is NP-hard).

Definition 2.1.8 (μ-rank or multilinear rank of a tensor)**.** *The* μ*-rank of tensor* $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$*, denoted* $\text{rank}_\mu(\mathcal{X})$ *is the rank of* $\boldsymbol{X}_{(\mu)}$*. If we let* $r_\mu = \text{rank}_\mu(\mathcal{X})$ *for* $\mu = 1, ..., d$ *then we can say that* $\mathcal{X}$ *is rank-*$(r_1, \cdots, r_d)$ *tensor. Beware not to confuse the* μ*-rank with the previous notion of rank of a tensor.*

Remark. The notion of n-rank was popularised by De Lathauwer [DDV00].

Definition 2.1.9. ***matricization*** *or* unfolding
Matricization *is the process of ordering the elements of a tensor into a matrix. The mode-*n *matricization of a tensor* $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ *is denoted by* $\mathbf{X}_{(\mu)}$ *and arranges the mode-*μ *fibres to be the columns of the resulting matrix. We define the index set* $\mathcal{I}^{(\mu)} = \mathcal{I}_1 \times \cdots \times \mathcal{I}_{\mu-1} \times \mathcal{I}_{\mu+1} \times \cdots \times \mathcal{I}_d$*. The formal notation is more complex than the concept of unfolding, indeed the map from the tensor entries* $(i_1, i_2, \cdots, i_d) \in \mathcal{I}$ *to the matrix entries* $(i_\mu, j) \in \mathcal{I}_\mu \times \mathcal{I}^{(\mu)}$ *is*

$$j = 1 + \sum_{\substack{k=1 \\ k \neq \mu}}^{d} (i_k - 1) J_k \quad \text{with} \quad J_k = \prod_{\substack{m=1 \\ m \neq \mu}}^{k-1} I_m \tag{2.1.7}$$

Only the special case of mode-n matricization is considered here, further details are available in [Kol06].

Remark. The ordering in which the matricization does not matter as long as it is consistent through the computation.
One can also vectorize a tensor, the same goes concerning ordering

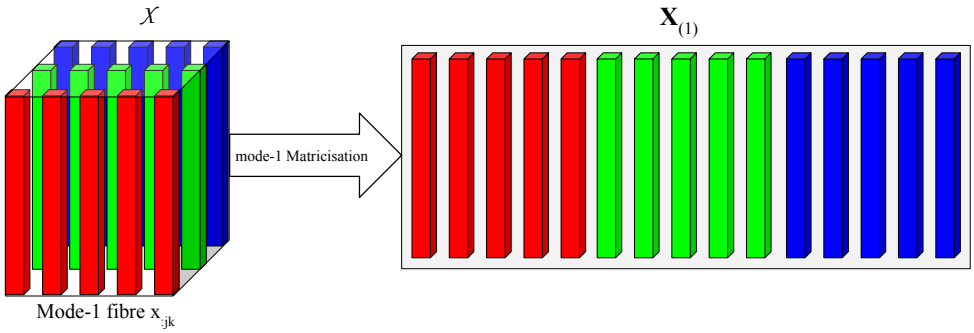


Figure 2.3: Mode one matricization of third order tensor with $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$.

Tensor multiplication It is possible to define product between tensors in a varieties of ways. It does require more complex notations and symbols that for matrices. We restrict ourselves to the ones which are actively used to describe tensor reduction. Information about other tensor products is widely available in the literature [KB09, Hac14].

Definition 2.1.10 (Tensor product)**.** *The tensor product is a special case of the outer product that allow multiplication between tensors is denoted by* $\otimes$ *or* $\circ$ *if a confusion with the Kronecker product is possible. Let* $\mathcal{I} = \mathcal{I}_1 \times \cdots \mathcal{I}_p$ *and* $\mathcal{J} = \mathcal{J}_1 \times \cdots \mathcal{J}_q$ *be multi index series. The the tensor product is defined by*

$$\begin{aligned} \otimes : \quad \mathbb{R}^{\mathcal{I}} \times \mathbb{R}^{\mathcal{J}} &\to \quad \mathbb{R}^{\mathcal{I} \times \mathcal{J}} \\ (\mathcal{X}, \mathcal{Y}) &\mapsto \quad \mathcal{X} \otimes \mathcal{Y} \end{aligned}$$

Enrty-wise $\mathcal{T} = \mathcal{X} \otimes \mathcal{Y}$ *writes*

$$T_{\boldsymbol{ij}} = x_{\boldsymbol{i}} y_{\boldsymbol{j}}$$

where $\boldsymbol{i} = \{i_1, ..., i_p\}$ *and* $\boldsymbol{j} = \{j_1, ..., j_q\}$.

Definition 2.1.11 (Kronecker product)**.** *Kronecker product of matrices* $\mathbf{A} \in \mathbb{R}^{I \times J}$ *and* $\mathbf{B} \in \mathbb{R}^{K \times L}$ *is denoted by* $\mathbf{A} \otimes \mathbf{B}$*. The result is a matrix of size* $(IK) \times (JL)$ *and defined by*

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \cdots & a_{IJ}\mathbf{B} \end{bmatrix}$$

Remark. It should be noted the outer product of vectors is a special case of the Kronecker product.

Definition 2.1.12 (Kathri-Rao product)**.** *of matrices* $\mathbf{A} \in \mathbb{R}^{I \times K}$ *and* $\mathbf{B} \in \mathbb{R}^{J \times K}$ *is denoted by* $\mathbf{A} \odot \mathbf{B}$*. The result is a matrix of size* $(IJ) \times (K)$ *and defined by*

$$\mathbf{A} \odot \mathbf{B} = [\boldsymbol{a}_1 \otimes \boldsymbol{b}_1 \quad \boldsymbol{a}_2 \otimes \boldsymbol{b}_2 \quad \cdots \boldsymbol{a}_K \otimes \boldsymbol{b}_K]$$

If $\boldsymbol{a}$ *and* $\boldsymbol{b}$ *are vectors, then the Kathri-Rao product and Kronecker product are identical.*

Definition 2.1.13 (Hadamard product)**.** *It is the elementwise matrix product. Let* $\boldsymbol{A}$*and* $\boldsymbol{B} \in \mathbb{R}^{I\times J}$*, their Hadamard product is denoted by* $\boldsymbol{A} * \boldsymbol{B}$ *and it is also of size* $I \times J$*.*

$$\boldsymbol{A} * \boldsymbol{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \cdots & a_{1J}b_{1J} \\ a_{21}b_{21} & a_{22}b_{22} & \cdots & a_{2J}b_{2J} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}b_{I1} & a_{I2}b_{I2} & \cdots & a_{IJ}b_{IJ} \end{bmatrix} \tag{2.1.8}$$

These products have many properties [KB09] that are relied upon to devise decomposition algorithms.

Definition 2.1.14 (μ-mode product)**.** *The* μ-mode (matrix) product*, for* $1 \leq \mu \leq d$ *of tensor* $\mathcal{X} \in \mathbb{R}^{n_1\times\cdots\times n_d}$ *with matrix* $\mathbf{A} \in \mathbb{R}^{m\times n_\mu}$ *is denoted by* $\mathcal{X} \times_d \mathbf{A}$ *and is of size* $n_1 \times \cdots \times n_{\mu-1} \times m \times n_{\mu+1} \times \cdots n_d$*. Element-wise, we have*

$$(\mathcal{X} \times_\mu \mathbf{A})_{i_1\cdots i_{\mu-1} j i_{\mu+1}\cdots i_d} = \sum_{i_\mu=1}^{n_\mu} x_{i_1 i_2\cdots i_d} u_{j i_\mu}$$

It is equivalent to say that each mode-μ *fiber is multiplied by the matrix A, i.e.* $\mathcal{Y} = \mathcal{X} \times_\mu \mathbf{A} \Leftrightarrow \mathbf{Y}_{(\mu)} = \mathbf{A}\mathbf{X}_{(\mu)}$*.*

Definition 2.1.15 (multilinear multiplication [VVM12])**.** *Multilinear multiplication in one mode is equivalent to n-mode multiplication but is usefull to introduce a new notation*

$$[(\boldsymbol{I}, ...\boldsymbol{I}, \boldsymbol{M}, \boldsymbol{I}, ..., \boldsymbol{I}) \cdot \mathcal{X}]^{(n)} = \boldsymbol{M}\boldsymbol{X}^{(n)} \tag{2.1.9}$$

Then in general, the unfolding of a multilinear multiplication is given by

$$[(\boldsymbol{M}_1, \boldsymbol{M}_2, ..., \boldsymbol{M}_d) \cdot \mathcal{X}]^{(n)} = M_n \boldsymbol{X}^{(n)} (\boldsymbol{M}_1 \otimes \cdots \otimes \boldsymbol{M}_{n-1} \otimes \boldsymbol{M}_{n+1} \otimes \cdots \otimes \boldsymbol{M}_d)^\intercal, \tag{2.1.10}$$

Two multilinear multiplications can be transformed into one, as follow

$$(\boldsymbol{L}_1, \boldsymbol{L}_2, ..., \boldsymbol{L}_d)\,[(\boldsymbol{M}_1, \boldsymbol{M}_2, ..., \boldsymbol{M}_d) \cdot \mathcal{X}] = (\boldsymbol{L}_1\boldsymbol{M}_1, \boldsymbol{L}_2\boldsymbol{M}_2, ..., \boldsymbol{L}_d\boldsymbol{M}_d) \cdot \mathcal{X} \tag{2.1.11}$$

2.2 Tensor Formats

In this section, some of the most common tensor formats or representations are described. Indeed, in applications one needs to represent the properties of a tensor using a finite numbers of parameters. Not all tensors belong to spaces of finite dimension (e.g. tensor Hilbert spaces), then the question of finite approximation arises. The decomposition or approximation of a tensor in a certain format is addressed in the next section 2.3.

Before entering these descriptions, one should note the difference between *representation* and *decomposition* that are complementary notions. On the one hand, the *representation* of a tensor is any way used to describe a tensor using a set of parameters $(p_1, ..., p_n)$ e.g. representation of tensor $\mathcal{X}$ on a computer using full real array format : $(p_1, ..., p_n) \to \mathcal{X}$. On the other hand the *decomposition* does the opposite way by analyzing a tensor to determine a set of properties : $\mathcal{X} \to (p_1, ..., p_n)$.

These operation can be used alternately, for example the CP decomposition of a tensor yields a representation of it with a given accuracy. This leads to the following statement by Hackbush : *"'tensor decomposition' is applied, when features of a concrete object should be characterized by parameters of a tensor-valued data about this object".*

For the sake of simplicity, the following presentation uses d-way array formats but they are equivalent version for arbitrary tensor spaces so long as a finite basis exists.

2.2.1 Full format

Let $\mathcal{I} = I_1 \times \cdots \times I_d$ a d-fold product index and a tensor $\mathfrak{X} \in \mathbb{R}^{\mathcal{I}}$. Then the full format consists in storing the values taken by $\mathfrak{X}$ for all $(i_1, ..., i_d) \in \mathcal{I}$ with the standard basis $\boldsymbol{e}_{\mu,i_\mu} \in \mathbb{R}^{\mathcal{I}^\mu}$ is defined by $(\boldsymbol{e}_{\mu,i_\mu})_{j_m u} = \delta_{i_\mu,j_\mu}$. We have

$$\mathfrak{X} = \sum_{\boldsymbol{i}\in\mathcal{I}} x_{\boldsymbol{i}}\ \boldsymbol{e}_{1,i_1} \otimes \cdots \otimes \boldsymbol{e}_{d,i_d} \tag{2.2.1}$$

Storage. Since the basis is trivial, it is not needed to store the basis and the storage cost is $\prod_{\mu=1}^{d} n_\mu$. Let $n = \max_{\mu\in D} n_\mu$ then the storage cost is in $\mathcal{O}(n^d)$ which is intractable for large d. A more general definition of full format for tensors is given by J. Ballani in his thesis dissertation [Bal12].

Evaluation Cost. The evaluation cost in full format is nil since one just need to recover the value at a given index in the computer memory.

2.2.2 Canonical format or r-term format $\mathcal{C}_r$

Definition 2.2.1 (Canonical Format)**.** *In this format, any tensor $\mathfrak{X} \in V = \bigotimes_{\mu=1}^{d} V_\mu$ a tensor space, is written as the finite sum of rank-1 tensors. $\mathfrak{X} \in \mathcal{C}_r(\mathbb{R}^{\mathcal{I}})$ is said to be represented in the canonical format and it reads,*

$$\mathfrak{X} = \sum_{i=1}^{r} \bigotimes_{\mu=1}^{d} \boldsymbol{u}_{\mu,i} \quad \text{where } \boldsymbol{u}_{\mu,i} \in V_\mu = \mathbb{R}^{\mathcal{I}_\mu} \tag{2.2.2}$$

where $\boldsymbol{U}_\mu = [\boldsymbol{u}_{\mu,1}\ \boldsymbol{u}_{\mu,2}\ \cdots\ \boldsymbol{u}_{\mu,r}]$ for $\mu \in D$. The μ-matricization of $\mathfrak{X}$ can be computed by

$$\boldsymbol{X}_{(\mu)} = \boldsymbol{U}_\mu(\boldsymbol{U}_1 \odot \cdots \odot \boldsymbol{U}_{\mu-1} \odot \boldsymbol{U}_{\mu-1} \odot \cdots \odot \boldsymbol{U}_d)^\intercal \tag{2.2.3}$$

Remark. a. r, the length of the sum, is the tensor rank of $\mathfrak{X}$ as stated in definition 2.1.7. However, the reader is reminded that computing the rank of an arbitrary tensor is a NP-complex problem.

b. $\mathcal{C}_r$ is not a linear space since the sum of $\mathfrak{X}, \mathfrak{Y} \in \mathcal{C}_r$ belongs to $\mathcal{C}_{2r}$ and $\mathfrak{X} + \mathfrak{Y} \notin \mathcal{C}_r$ in general.

Storage. Accordingly to the previous remark, it is assumed that r is known since the tensor is already in $\mathcal{C}_r$. Then each parameter vector $(\boldsymbol{u}_{\mu,i})$ storage complexity is in $\mathcal{O}(\#\mathcal{I}_\mu)$ which leads to the following tensor storage complexity in $\mathcal{C}_r$ with $n = \max_{\mu\in D}(n_\mu)$.

$$N_{storage}(\mathcal{C}_r) = r \sum_{\mu=1}^{d} \mathcal{I}_\mu = \mathcal{O}(drn) \tag{2.2.4}$$

If r remains small then the storage complexity remains moderate even for a large number of dimensions.

Evaluation. The evaluation of a single entry $x_{\boldsymbol{i}}$, $\boldsymbol{i} = (i_1, ..., i_d) \in \mathcal{I}$ of $\mathcal{X} \in \mathcal{C}_r$ requires the multiplication of the values $(u_{\mu,i})_{i_\mu}$ for $\mu \in D$. Indeed $x_{\boldsymbol{i}} = \sum_{j=1}^{r} \prod_{\mu=1}^{d} (u_{\mu,j})_{i_\mu}$ which means the complexity to evaluate a single entry is $N_{entry}(\mathcal{C}_r) = dr$ leading to the following complexity to evaluate the whole tensor

$$N_{full\ eval}(\mathcal{C}_r) = \mathcal{O}(n^d dr) \tag{2.2.5}$$

This cost is optimal in the sense of linear complexity, however the non-linearity of the space raises the question of truncation or approximation which is treated in section 2.3.1.

As for the full format, $\mathcal{C}_r$ is fully compatible with other underlying vector spaces. Further information is available in [Bal12, Hac14].

2.2.3 Tucker format $\mathcal{T}_k$

This section focuses on the crucial Tucker format which consists for $\mathcal{X} \in V = \mathbb{R}^{\mathcal{I}}$ in finding smaller subspaces $U_\mu \subset V_\mu$ such that $\mathcal{X} \in \bigotimes_{\mu=1}^{d} U_\mu$. Indeed if $k_\mu = \dim(U_\mu) < \dim(V_\mu)$ then $\mathcal{X}$ can be represented more efficiently than in full representation. This leads to the following definition.

Definition 2.2.2 (Tucker format $\mathcal{T}_k$)**.** *Let $\boldsymbol{k} = (k_1, ..., k_d) \in \mathbb{N}^d$ and a family of linearly independent vectors $(\boldsymbol{u}_{\mu,i})_{\mu,1\leq i\leq k_\mu}$ for $\mu \in D$ such that $(\boldsymbol{u}_{\mu,i})_{\mu,1\leq i\leq k_\mu}$ is a basis of U_μ. Then the tucker representation of $\mathcal{X} \in U$ is*

$$\mathcal{X} = \sum_{i_1=1}^{k_1} \cdots \sum_{i_d=1}^{k_d} w_{i_1,...,i_d} \boldsymbol{u}_{1,i_1} \otimes \cdots \otimes \boldsymbol{u}_{d,i_d} \tag{2.2.6}$$

with the weights $w_{i_1,...,i_d} \in \mathbb{R}$. They form the core tensor $\mathcal{W} \in \mathbb{R}^{k_1 \times \cdots \times k_d}$.
$\boldsymbol{k}$ is the representation rank (or Tucker rank*) of $\mathcal{X}$ in the tucker format $\mathcal{T}_{\boldsymbol{k}}$. One can also write $\mathcal{X}$ as a product of $\mathcal{W}$ and matrices $\boldsymbol{U}_\mu = [(\boldsymbol{u}_{\mu,i})]_{i=1}^{k_\mu}$ which reads*

$$\mathcal{X} = \mathcal{W} \times_1 \boldsymbol{U}_1 \times_2 \boldsymbol{U}_2 \cdots \times_d \boldsymbol{U}_d. \tag{2.2.7}$$

Its μ-matricized version reads

$$\boldsymbol{X}_{(n)} = \boldsymbol{U}_\mu \boldsymbol{W}_{(\mu)} (\boldsymbol{U}_1 \otimes \cdots \otimes \boldsymbol{U}_{\mu-1} \otimes \boldsymbol{U}_{\mu+1} \otimes \cdots \otimes \boldsymbol{U}_d)^{\intercal}. \tag{2.2.8}$$

Remark. a. As stated by Ballani, for general tensors, $\mathcal{T}_{\boldsymbol{k}}$ the set of tensors which Tucker representation rank is lower than $\boldsymbol{k}$ is not a linear space.

b. The tuple formed of all the μ-ranks is the lowest $\boldsymbol{k}$ for which $\mathcal{X} \in \mathcal{T}_{\boldsymbol{k}}$.

Storage complexity. In order to represent a tensor in $\mathcal{T}_{\boldsymbol{k}}$ format, one only need to store the core tensor of size $\mathcal{O}(\prod_{\mu=1}^{d} k_\mu)$ and the basis vectors stored in matrices for each dimension of size $\mathcal{O}(k_\mu n_\mu$. This yields a total storage complexity of

$$N_{storage}(\mathcal{T}_{\boldsymbol{k}}) = \prod_{\mu=1}^{d} k_\mu + \sum_{\mu=1}^{d} k_\mu n_\mu = \mathcal{O}(k^d + dkn) \tag{2.2.9}$$

One can clearly see that the term $\mathcal{O}(k^d)$ is very interesting if d is small since overhead cost compared with $\mathcal{C}_r$ is limited. However if d grows above 5, it will become impossible to use this format even if k remains small.

Evaluation complexity. In order to evaluate a single entry of a tensor in tucker format, one needs to compute the sum 2.2.6. Each term of the sum requires $(d+1)$ operations which leads to the entry evaluation complexity complexity of

$$N_{entry\ eval}(\mathcal{T}_{\boldsymbol{k}}) = (d+1)\prod_{\mu=1}^{d} k_\mu \tag{2.2.10}$$

Then the overall complexity to evaluate the full tensor is in $\mathcal{O}((d+1)k^d n^d)$ which is very costly. However this representation remains interesting since the evaluation of the tucker rank only requires standard linear algebra tools and approximations of lower rank are easily accessible through HOSVD. See sections 2.3.2 and 2.3.2.1.

2.2.4 Hierarchical Tucker format $\mathcal{H}_k$

When dimension d gets above 5 to 10 the HT format becomes an efficient alternative to the Tucker decomposition. It is based on the idea of recursively splitting the modes of the tensor. The process results in a binary tree $\mathcal{T}_D$ containing a subset $t \subset D := \{1, ..., d\}$ at each node e.g. figure 2.4.

The HT format is much more complex than the previous ones, then we need to introduce some definitions proposed by Lars Grasedyck in [Gra10] and Jonas Ballani [Bal12].

Definition 2.2.3 (Dimension partition Tree)**.** *The tree T_D is called a* dimension partition tree *of D if*

a. all vertices $\alpha \in T_D$ are non-empty subset of D,

b. D is the root of T_D,

c. every vertex $\alpha \in T_D$ with $\#\alpha \geq 2$ has two sons $\alpha_1, \alpha_2 \in T_D$ such that

$$\alpha = \alpha_1 \cup \alpha_2, \quad \alpha_1 \cap \alpha_2 \neq \emptyset \tag{2.2.11}$$

The set of sons of alpha is denoted by $S(\alpha)$. If $S(\alpha) = \emptyset$, α is called a leaf. The set of leaves is denoted by $\mathcal{L}(T_D)$. The level number of a vertex is defined recursively as.

$$level(D) = 0, \quad \sigma \in S(\alpha) \Rightarrow level(\sigma) = level(\alpha) + 1 \tag{2.2.12}$$

The set of all the vertices of a given level l is denoted T_D^l.
The depth of the tree *is defined as*

$$L = depth(T_D) = \max_{\alpha \in T_D}\{level(\alpha)\} \tag{2.2.13}$$

Figure 2.4 displays the binary dimension partition tree of $D = \{1, ..., 5\}$ which depth is 3. A binary tree means that all non-leaf vertex has two sons.

Remark. As for the other formats, a description for general tensor spaces is available in Hackbush's book [Hac14] with similar properties that is useful when working in functional spaces like H^1. However for the sake of simplicity, we restrain ourselves to tensors of $\mathbb{R}^{\mathcal{I}}$ and follow the original description of $\mathcal{H}_{\boldsymbol{r}}$ format by Grasedyck [Gra10].

Definition 2.2.4 (Generalization of the Matricization [Gra10, Definition 3.3])**.** *For a mode cluster t in a dimension tree $T_{\mathcal{I}}$ we define the complementary cluster $t' = D/t$ as*

$$\mathcal{I}_t = \times_{\mu \in t} \mathcal{I}_\mu, \quad \mathcal{I}_{t'} = \times_{\mu \in t'} \mathcal{I}_\mu \tag{2.2.14}$$

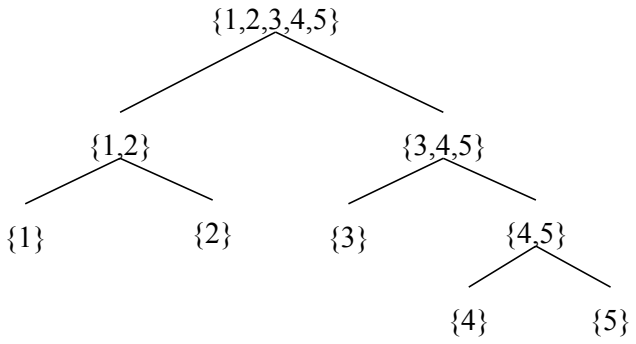


Figure 2.4: Binary dimension partition tree of $D = \{1, ..., 5\}$

and the corresponding t-matricization *as*

$$\mathcal{M}_t : \mathbb{R}^{\mathcal{I}} \longrightarrow \mathcal{I}_t \times \mathcal{I}_{t'}, \quad (\mathcal{M}_t(\mathcal{X}))_{(i_\mu)_{\mu\in t},(i_\mu)_{\mu\in t'}} = \mathcal{X}_{(i_1,\dots,i_d)} \tag{2.2.15}$$

where the special case is $\mathcal{M}_\emptyset(\mathcal{X}) = \mathcal{M}_{\{1,\dots,d\}}(\mathcal{X}) = \mathbf{x}$. *Thereafter, we use the short notation* $\boldsymbol{X}^{(t)} = \mathcal{M}_t(\mathcal{X})$.

Definition 2.2.5 (Hierarchical rank, [Gra10, Definition 3.4])**.** *Let* $T_\mathcal{I}$ *be a dimension tree. The* hierarchical rank $(k_t)_{t\in T_\mathcal{I}}$ *of a tensor* $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ *is defined by*

$$\forall t \in T_\mathcal{I},\ k_t = \operatorname{rank}(\boldsymbol{X}^{(t)}) \tag{2.2.16}$$

The set of all tensors of hierarchical rank (nodewise) at most $(k_t)_{t\in T_\mathcal{I}}$ *is denoted by*

$$\mathrm{HT}(k_t) = \mathcal{H}\text{−Tucker}((k_t)_{t\in T_\mathcal{I}}) = \{\mathcal{X} \in \mathbb{R}^{\mathcal{I}} \,|\, \forall t \in T_\mathcal{I} :\ \operatorname{rank}(\boldsymbol{X}^{(t)}) \le k_t\} \tag{2.2.17}$$

Remark. From this point, it is possible to define a SVD at each node with respect to $\boldsymbol{X}^{(t)}$. However it is not clear yet why it makes sense to use such techniques. The next definitions will reveal the connection with the nested representation.

Definition 2.2.6 (frame tree; t-frame, transfer tensor, [Gra10, Definition 3.5])**.** *Let* $t \in T_\mathcal{I}$ *be a mode cluster and* $(k_t)_{t\in T_\mathcal{I}}$ *a family of nonnegative integers. We call a matrix* $\boldsymbol{U}_t \in \mathbb{R}^{\mathcal{I}_t\times k_t}$ *a* t*-frame and a tuple* $(\boldsymbol{U}_s)_{s\in T_\mathcal{I}}$ *of frames a frame tree. A frame is called orthogonal if the columns are orthonormal and a frame tree is called orthogonal if each frame* except the root frame *is orthogonal. A frame tree is nested if for each interior node cluster* t *with successors* $S(t) = \{t_1, t_2\}$ *the following relation holds:*

$$\operatorname{span}\{(\boldsymbol{U}_t)_i \,|\, 1 \le i \le k_t\} \ \subset\ \operatorname{span}\{(\boldsymbol{U}_{t_1})_i \otimes (\boldsymbol{U}_{t_2})_j \,|\, 1 \le i \le k_{t_1}, 1 \le j \le k_{t_2}\} \tag{2.2.18}$$

The corresponding tensor $\mathcal{B}_t \in \mathbb{R}^{k_t\times k_{t_1}\times k_{t_2}}$ *of coefficients for the representation of the columns* $(\boldsymbol{U}_t)_i$ *of* $\boldsymbol{U}_t$ *by the columns of* $\boldsymbol{U}_{t_1}, \boldsymbol{U}_{t_2}$

$$(\boldsymbol{U}_t)_i = \sum_{j=1}^{k_{t_1}} \sum_{l=1}^{k_{t_2}} (\mathcal{B}_t)_{i,j,l} (\boldsymbol{U}_{t_1})_j \otimes (\boldsymbol{U}_{t_2})_l, \tag{2.2.19}$$

is called the transfer tensor.

For a nested frame tree it is sufficient to provide the transfer tensor $(\mathcal{B}_t)$ of all interior node i.e. $t \in \mathcal{I}(T_\mathcal{I})$ and the t-frames $(\boldsymbol{U}_t)$ for only for the leaf nodes $t \in \mathcal{L}(T_\mathcal{I})$. So far, no orthogonality condition has been imposed. One should not that the t-frames represent the full tensor space of their mode cluster even if they are matrices i.e. they are t-matricization of a $\#t$ mode tensor. For example, let $t = \{1, 2, 3\}$ then $\boldsymbol{U}_{\{1,2,3\}}$ has three "degrees of freedom" and the root represent the full tensor in a t-frame which is equivalent to a tensor. For a better understanding of this subtlety, the reader is advised to refer to Ballani [Bal12] or Hackbush for a more detailed description [Hac14, Chapter 11]. Figure 2.5 shows the structure of the HT format that is defined next.

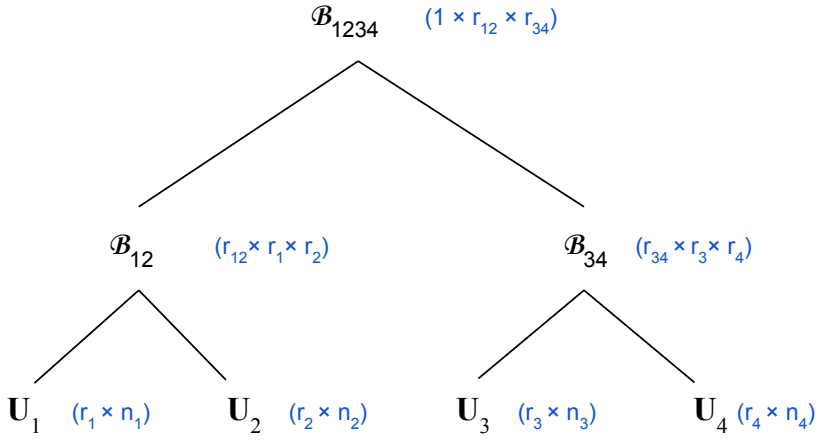


Figure 2.5: Tree representation of the HT format of $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$. The size of the matrices and tensors are inside blue braces.

Definition 2.2.7 (Hierarchical Tucker format $\mathcal{H}_{\boldsymbol{k}}$, [Gra10, Definition 3.6])**.** *Let $T_\mathcal{I}$ be a dimension tree, $(k_t)_{t \in T_\mathcal{I}}$ a family of non negative integers, and $\mathcal{X} \in \mathcal{H}-\text{Tucker}((k_t)_{t \in T_\mathcal{I}}) = \mathcal{H}_{\boldsymbol{k}}$. Let $(\boldsymbol{U}_t)_{t \in T_\mathcal{I}}$ be a nested frame tree with transfer tensors $(\mathcal{B}_t)_{t \in \mathcal{I}(T_\mathcal{I})}$ and*

$$\forall t \in T_\mathcal{I}, \text{image}(\boldsymbol{X}^{(t)}) = \text{image}(\boldsymbol{U}_t), \quad \mathcal{X} = \mathcal{U}_{\{1,\dots,d\}}. \tag{2.2.20}$$

Then the representation $((\mathcal{B}_t)_{t \in \mathcal{I}(T_\mathcal{I})}, (\boldsymbol{U}_t)_{t \in T_\mathcal{I}})$ is a hierarchical Tucker representation *of $\mathcal{X}$. The family $(k_t)_{t \in T_\mathcal{I}}$ is the hierarchical representation rank. Note that the columns of $\boldsymbol{U}_t$ need not be linear independent.*

Remark. The HT representation of $\mathcal{X} \in \mathcal{H}_{\boldsymbol{k}}$ with orthogonal frames and a minimal k_t is unique up to orthogonal transformation of the t-frames.

Storage Complexity. The storage cost for $\mathcal{X} \in \mathcal{H}_{\boldsymbol{k}}$ is determined by the sum of all leaves matrices of size $(\boldsymbol{U}_t)_{t \in \mathcal{I}(T_\mathcal{I})} \in \mathbb{R}^{\mathcal{I}_t \times k_t}$ with all the transfer tensors $(\mathcal{B}_t)_{t \in \mathcal{I}(T_\mathcal{I})} \in \mathbb{R}^{k_t \times k_{t_1} \times k_{t_2}}$. This leads to the following number of entries bound

$$N_{storage}((\boldsymbol{U}_t)_{t \in \mathcal{I}(T_\mathcal{I})}, (\mathcal{B}_t)_{t \in \mathcal{I}(T_\mathcal{I})}) \leq k \sum_{\mu=1}^{d} n_\mu + (d-1)k^3 \leq kdn + (d-1)k^3, \tag{2.2.21}$$

where $k = \max_{t \in T_\mathcal{I}} k_t$ and $n = \max_{i \in D} n_i$. This means that the storage cost is linear in the dimension d provided that k is uniformly bounded and $k \ll d$. This allows applications with large number of dimensions d.

Evaluation Complexity. The computation of a single entry $x_{\boldsymbol{i}}$, $\boldsymbol{i} = \{i_1, ..., i_d\}$ requires to compute the recursive sum which is equation 2.2.19 entry-wise,

$$\forall 1 \leq i \leq k_t, \quad u^t_{m,i} = \sum_{j=1}^{k_{t_1}} \sum_{l=1}^{k_{t_2}} b^t_{i,j,l}\, u^{t_1}_{mj}\, u^{t_2}_{ml} \tag{2.2.22}$$

which means for each cluster node in the interior of the tree $k_t k_{t_1} k_{t_2}$ and no cost on the leafs. Then the evaluation complexity of a tensor entry in HT format is bound by

$$N_{entry\ eval}(HT) \leq (d-1)k^3 \tag{2.2.23}$$

Then it is only linear in d and the cubic factor on k is not problematic as long as k remains small. Moreover, as described in section 2.3.4, truncation techniques with error bounds are available as well as hierarchical rank linear algebra routines. All things considered, HT format and the associated decomposition is well suited to high dimensional tensor reduction.

2.2.4.1 Tensor Train format

The tensor train format (TT) is a special case of hierarchical tensor formats which displays some advantages. It was popularized by Oseledets et al. [OT09] followed by a substantial series of paper that is condensed in [SO11]. This format was first presented as a product of matrices that describes each element of the tensor which is why it is also known as matrix product state (MPS) in the literature. Entry-wise, $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{\mathcal{N}}$ is given by the following product of matrices

$$x_{i_1,...,i_d} = \boldsymbol{G}_1(i_1)\boldsymbol{G}_2(i_2)\cdots\boldsymbol{G}_d(i_d), \quad \boldsymbol{G}_\mu \in \mathbb{R}^{k_{\mu-1}\times k_\mu} \tag{2.2.24}$$

where $k_0 = k_d = 1$. For every mode μ and every index i_μ the coefficients $\boldsymbol{G}_\mu(i_\mu)$ are matrices. There is no specific assumption on the orthogonality of the modes $G(:)_{i,j}$, only the construction of such representation may ensure it. The following definitions comes naturally.

Definition 2.2.8 (TT-decompostion)**.** *Let $G_\mu \in \mathbb{R}^{k_{\mu-1}\times n_\mu \times k_\mu}$ for all $\mu \in [\![1, d]\!]$ a set of order 3 tensors called* TT-cores*. Then the order d tensor $\boldsymbol{\mathcal{X}} \in \mathcal{R}^{\mathcal{N}}$ with* TT-rank *$\boldsymbol{r} = \{k_i\}_{i=0}^d$ ($k_0 = k_d = 1$) has the following TT decomposition*

$$\boldsymbol{\mathcal{X}} = \sum_{\alpha_0,...,\alpha_d=1}^{\boldsymbol{k}} G_1(\alpha_0, i_1, \alpha_1)\cdots G_1(\alpha_{d-1}, i_d, \alpha_d) \tag{2.2.25}$$

Additionally, the TT format can be seen as a special case of the HT format with a linear structure. Here, all nodes have at least one son that is a leaf. One can see in Fig. 2.6 the link between HT and TT regarding the shape of the tree while Fig. 2.7 shows the dimension tree associated with TT format.

Storage Complexity. It can be easily shown [GHN11] that the storage cost is

$$\mathcal{O}(k^2 dn) \tag{2.2.26}$$

where $k = \max_{t\in T_{\mathcal{I}}} k_t$ and $n = \max_{i\in D} n_i$.

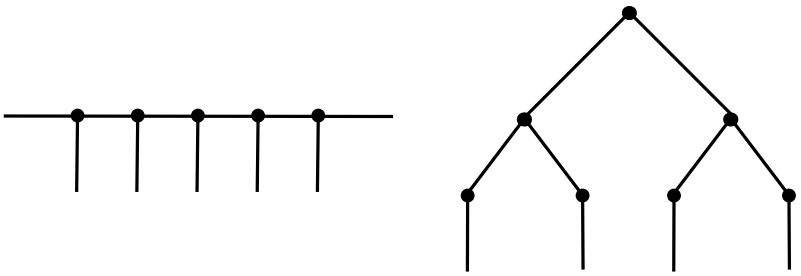


Figure 2.6: A graph representation of TT (left) and HT (right) format highlighting their similarities and differences.

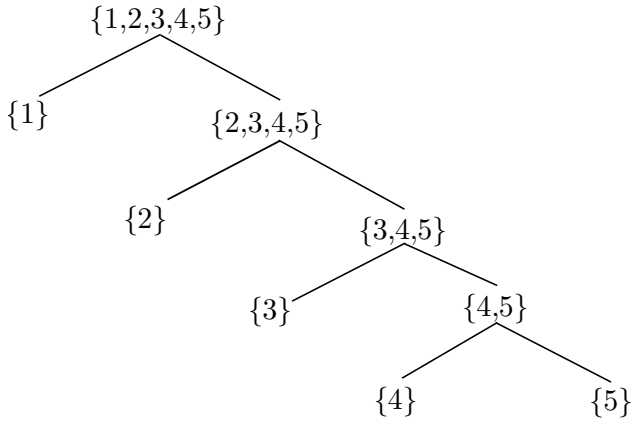


Figure 2.7: "Recursive" dimension tree associated with the extended tensor train of a 5th order tensor

Evaluation Complexity. In order to evaluate one entry of the tensor, one simply needs to apply (2.2.24), which yields with the usual assumption on the rank and dimension of $\mathcal{X}$

$$N_{\text{entry eval}}(TT) = (d-1)k^3 \tag{2.2.27}$$

Remark. By construction, it is very easy (and cheap) to evaluate a single entry of a tensor. Same goes with very efficient algorithm for numerical integration/contraction as given by Oseledets [OT10].

Remark. Linear operations are straightforward to implement in TT format, including multiplications with matrices, vectors, tensor products, Hadamard product. See [Ose11, Sec. 4] for details and algorithms as it is out of the scope of this manuscript.

TT format possesses many of the required properties for tensor reduction:

- simple structure,
- easier to handle than HT,
- any tensor can be represented exactly,
- memory complexity that scales linearly with d,
- Straightforward multilinear algebra operations.

However, the bases associated with each space do not appear explicitly. Indeed the long fibers (middle dimension) of the cores span a vector space but do not form an orthonormal basis (naturally). This is a problematic feature for physics related applications where one usually want to manipulate modes directly whether it is for analysis or processing.

Consequently, TT format needs to be improved for our applications. The reader might refer to the literature survey [GKT13] for a bibliographic overview and a theoretical presentation of TT is given in [Hac14, Chap. 12]. In the next section, we introduce the extended tensor train format as proposed by [OT09] which displays the same recursive structures as TT while leaving direct access to the modes.

2.2.4.2 Extended TT format

The extended TT format (Ext-TT) blends together the recursive structure of TT (with all associated procedures) while being written as a hierarchical tensor which leaves are mode matrices. This structure is shown in Fig. 2.8 where the tuples indicates the dimensions of each nodes. One can see that the linearity of the TT tree is preserved while the leaves strictly account for one dimension. In terms of data structure, it leads to the tree shown in Fig. 2.8. Here B_t can be seen as a frame (according to Hackbush's terminology) or in terms of representation, as a *transfer tensor* (third order) of dimension $k_t \times k_{t1} \times k_{t2}$ where k_t is the hierarchical rank of node t as defined in 2.2.5 and k_{t1}, k_{t2} the rank of the sons of t. U_i are the mode matrices of size $n_i \times r_i$ where n_i is the size of dimension k while $r_i = k_i$. This allows to reorthogonalize and/or truncate a TT decomposition that produces a basis that is orthonormal and optimal in terms on rank/storage ratio for the same amount of information. Finally one can use the hierarchical evaluation formula to build the full tensor

$$\mathcal{X} = \sum_{\substack{\alpha_i \\ (0 \ \leq i \leq d)}} \prod_{\substack{1 \leq i_j \leq r_j \\ (0 \ \leq j \leq d)}} b_i^{(j,\alpha_{j-1},\alpha_j)} \bigotimes_{j=1}^{d} \boldsymbol{u}_{i_j}^{(j)} \tag{2.2.28}$$

where $b_i^{(j,\alpha_{j-1},\alpha_j)}$ are column vectors (of size r_j) of the transfer tensors. Entry-wise, it reads

$$\mathcal{X}(i_1, ..., i_d) = \sum_{\beta_1,...,\beta_d} U(i_1, \beta_1) \cdots U(i_d, \beta_d) \sum_{\alpha_0,...,\alpha_d} \mathcal{B}(\alpha_0, \beta_1, \alpha_1) \cdots \mathcal{B}(\alpha_{d-1}, \beta_d, \alpha_d) \tag{2.2.29}$$

with $\alpha_0 = \alpha_d = 1$. It is easy to recover the TT-cores from eq. (2.2.25) by

$$G_k(\alpha_{k-1}, i_k, \alpha_k) = \sum_{\beta_k} U(i_k, \beta_k) \mathcal{B}(\alpha_{k-1}, \beta_k, \alpha_k) \tag{2.2.30}$$

Definition 2.2.9 (Extended-TT)**.** *Any tensor in TT format can be written in HT format exactly using eq. (2.2.30), this format is called extended TT.*

Storage cost A brief evaluation yields the following improved storage cost

$$\mathcal{O}(dr\rho^2 + drn) \tag{2.2.31}$$

where ρ are the ranks of the original TT and r the (dimension-wise) rank of the ExtTT, usually $r \leq \rho$.

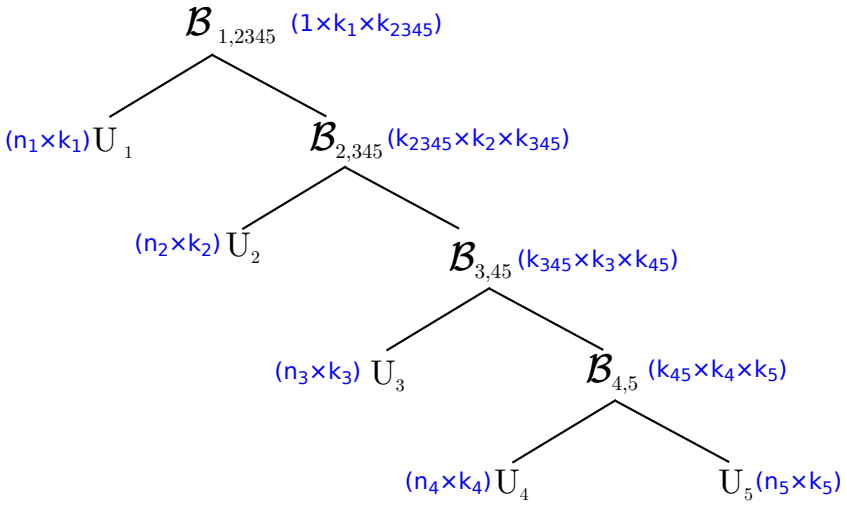


Figure 2.8: Tree representation of the arrays associated with a 5th order extended TT tensor. Dimensions are given according to HT labeling, first and last dimensions of the transfer tensor correspond to the TT ranks while the middle ones k_j is the "dimension" rank, expected to match the rank of the vector space spanned by the core tensors.

Table 2.1: Synoptic table of tensor formats

Format	Brief	Storage	Evaluation
Full	Original data format. Exponential storage cost. No evaluation cost. Only use as a temporary step toward more efficient formats.	$\mathcal{O}(n^d)$	0
Canonical	Linear storage cost. Easy writing/programming. Satisfactory for any d though not the best. The set $\mathcal{C}_r$ is not closed $\Rightarrow$approximation issues.	$\mathcal{O}(drn)$	$\mathcal{O}(dr)$
Tucker	d exponential storage cost on rank. Good for small d. The set $\mathcal{T}_{\boldsymbol{k}}$ is closed. Easy approximation with HOSVD.	$\mathcal{O}(k^d + dkn)$	$\mathcal{O}((d+1)k^d)$
Hierarchical	d linear storage cost. Efficient when d is big. The set $\mathcal{HT}_{\boldsymbol{k}}$ is closed. $\mathcal{C}_r, \mathcal{HT}_{\boldsymbol{k}} \subset \mathcal{T}_{\boldsymbol{k}}$. All previous algorithms available. Complex writing/programming.	$\mathcal{O}(kdn$ $+(d-1)k^3)$	$\mathcal{O}((d-1)k^3)$
Train	d linear storage cost. Very efficient when d is big. Subset of $\mathcal{HT}_{\boldsymbol{k}}$. All previous algorithms available + TTD. Easy to use, conversion to Ext-TT.	$\mathcal{O}(dk^2n)$	$\mathcal{O}((d-1)k^3)$

2.2.5 Conclusion on tensor formats

Five tensor formats have been investigated so far, before entering the approximation of tensor under these formats, a brief recap of their properties is proposed.

From table 2.1, it appears that data low rank reduction/approximation is compulsory for $d \geq 3$. Indeed storage cost in full format is intractable as is grows exponentially with d. In the next section, approximation in low rank for the three other formats will be investigated. From this section it is already clear that HT is the most versatile format since it can represent exactly both $\mathcal{C}_r$, $\mathcal{T}_{\boldsymbol{k}}$ and TT. However it is the most complex in term of data structure and algorithm. Then it will be restricted to high dimension where the efficiency improvement justify to invest in advanced formats, more specifically, its crossover with TT will be most suited to our applications when d grows larger than 4. Indeed the next section

will show that CP decomposition is affected by several approximation issues. The Tucker format is relatively easy to handle and the HOSVD is an efficient reduction technique. For low values of d, the Tucker format should be the basic tool for tensor reduction.

2.3 Tensor decomposition

In this section, we finally tackle the approximation of tensors to reduced rank. This allows huge storage savings as each of the presented formats separates dimensions thus breaking the curse of dimensionality as long as the rank is kept small. In this section, three decomposition methods are studied starting with canonical decomposition. Then higher order SVD is used to compute truncated Tucker representations. Finally Tensor train decomposition through SVD is described. Hierarchical decompositions are obtained by reorganizing data in the other formats through algorithms that have been omitted in this document. Indeed, it does not improve the decomposition properties, only the storage cost is reduced. Thus due to the increased complexity, it was decided not to study Hierarchical tucker decomposition, the reader is referred to [BG14, Gra10, KT13] for additional information and implementations.

In order to describe decomposition techniques which are ways to approximate a tensor into a particular format, it is necessary to first define what is a best approximation.

Definition 2.3.1 (best approximation)**.** *Let $(\mathcal{V}, \|\cdot\|)$ be a normed vector space and let $\emptyset \neq \mathcal{U} \subseteq \mathcal{V}$. An element $u_{\text{best}} \in \mathcal{U}$ is called a* best approximation *of $v \in \mathcal{V}$ (with respect to $\mathcal{U}$ if*

$$\|v - u_{\text{best}}\| \leq \|v - u\| \quad \forall u \in \mathcal{U}$$

Remark. Theorem 2.1.3 ensures the existence of u best for any *weakly closed reflexive Banach vector space.* Then one just need to verify these properties to ensure the existence of a best approximation for a given tensor space.

2.3.1 CP decomposition

The idea of decomposing a tensor as a finite sum of rank one tensors was first expressed by Hitchcock in 1927 [Hit27] which he called polyadic form. It finally became popular when reintroduced by Caroll and Chang [CC70] in the form of CANDECOMP and Harshman [Har70] as PARAFAC (parallel factors). Then the method CANDECOMP/PARAFAC is referred as *CP Decomposition* but it can be found under other names such as polyadic decomposition of Topographic components models.

The CP decomposition yields a tensor in the canonical format $\mathcal{C}_r$.

Definition 2.3.2. *The CP decomposition of a tensor $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ is to factorize it into a finite sum of rank-one tensors i.e. it is an approximation of a tensor of $\mathbb{R}^{\mathcal{I}}$ in $\mathcal{C}_r$. It means that either of these problems have to be solved*

a. Given $\varepsilon > 0$, find $\tilde{\mathcal{X}} \in \mathcal{C}_r$ with minimal $r \in \mathbb{N}^*$ such that $\|\tilde{\mathcal{X}} - \mathcal{X}\| \leq \varepsilon$.

b. Given $r \in \mathbb{N}$, find $\tilde{\mathcal{X}} \in \mathcal{C}_r$ that minimizes the error $\varepsilon = \|\tilde{\mathcal{X}} - \mathcal{X}\|$

Given that either of these problem has a solution the following identity is obtained

$$\mathcal{X} \approx \tilde{\mathcal{X}} = \sum_{i=1}^{r} \bigotimes_{\mu=1}^{d} \tilde{\boldsymbol{x}}_{\mu}^{i} \tag{2.3.1}$$

Remark. $\tilde{\mathcal{X}}$ can be seen as the optimal projection of $\mathcal{X}$ on $\mathcal{C}_r$.

Example 2.3.1 (3D case)**.** Then we want to write the CP decomposition of $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ a rank 3 tensor with $R \in \mathbb{N}_+$ terms

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r \tag{2.3.2}$$

where $\mathbf{a}_r \in \mathbb{R}^{n_1}$, $\mathbf{b}_r \in \mathbb{R}^{n_2}$ and $\mathbf{c}_r \in \mathbb{R}^{n_3}$. Alternatively, it can be written element-wise as

$$\forall (i,j,k) \in [\![1, n_1]\!] \times [\![1, n_2]\!] \times [\![1, n_3]\!], \quad x_{ijk} \approx \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr}$$

Figure 2.9 displays a visual of the CP decomposition where the rank one tensors are represented directly as a product of vectors.

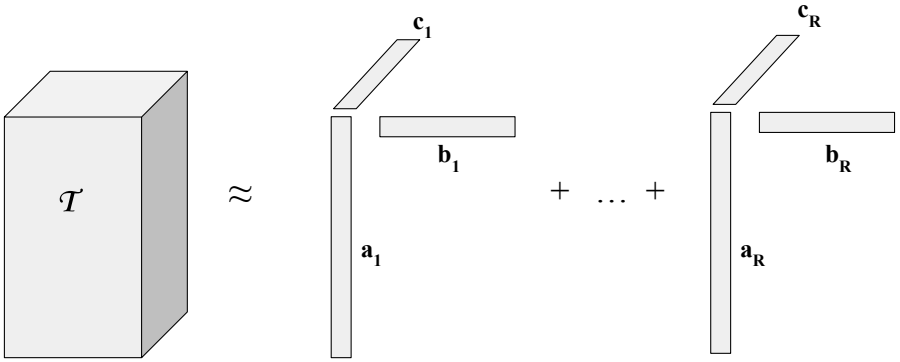


Figure 2.9: CP decomposition of third order tensor $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$.

The matrix $\mathbf{A}$ formed by the combination of vectors from the rank-one components (the *factor vectors*) i.e. $\mathbf{A} = [\mathbf{a}_1\ \mathbf{a}_2 \cdots \mathbf{a}_R]$ likewise for each dimension. They are referred as *factor matrices*. Then Kolda introduced the following concise notation for CP decomposition

$$\mathcal{X} \approx [\![\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]\!] \equiv \sum_{r=1}^{R} \boldsymbol{a}_r \circ \boldsymbol{b}_r \circ \boldsymbol{c}_r$$

It is of practical interest to assume that the factor vectors are normalized to one and their weights are stored into a vector $\boldsymbol{\lambda} \in \mathbb{R}^R$ so that

$$\mathcal{X} \approx [\![\boldsymbol{\lambda}; \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]\!] \equiv \sum_{r=1}^{R} \lambda_r\, \boldsymbol{a}_r \circ \boldsymbol{b}_r \circ \boldsymbol{c}_r \tag{2.3.3}$$

Remark. There is *no direct algorithm* to compute the optimal CP decomposition of a tensor, the problem is NP complex [HL96]. Although the uniqueness condition for rank decomposition is weaker for tensors than for matrices (permutation and scaling are allowed), it is often unique (e.g. [Har70]). Some criteria for uniqueness have been proposed in the literature.

2.3.1.1 Existence of a low rank approximation in $\mathcal{C}_r$

Lemma 2.3.1 ([Hac14, Remark 9.1] and [Bal12, Lemma 4.7]). *Problem (a) in definition 2.3.2 has a solution.*

For a matrix, the best rank-k approximation is given by the k first factors of the Singular Value Decomposition of that matrix (see 1.1). Then for $d = 2$ problem 2.3.2(b) has a solution however statement becomes false for tensors of higher order.

A tensor is called *degenerate* if several rank-k approximation give the same arbitrary approximation, in this case there is no best rank-k approximation. The best rank-k approximation may not be found *sequentially*, e.g. the best rank one approximation of $\boldsymbol{\mathcal{X}}$ may not be found in the best rank 2 approximation of $\boldsymbol{\mathcal{X}}$. Then all factors must be found *simultaneously* to ensure optimality.

Lemma 2.3.2 (Special case $\mathcal{C}_1$). *The set $\mathcal{C}_1$ is closed for all $d \in \mathcal{N}^*$.*

Indeed $\mathcal{T}_{1,\dots,1} = \mathcal{C}_1$ and $\mathcal{T}_{\boldsymbol{k}}$ is closed for any $\boldsymbol{k}$ [Bal12, Lemma 4.20]. This means that problem 2.3.2(b) has a solution in $\mathcal{C}_1$. However this is not true for higher ranks if $d \geq 3$, indeed it has been shown repeatedly [dSL08,KB09] that $\mathcal{C}_r$ is not closed in these conditions. Ballani provides a nice view of the issue [Bal12, Lemma 4.15]. The literature provides abundant examples of series of rank r tensors converging toward a rank $r+1$ tensor. This is mainly due to severe cancellation effects.

Lemma 2.3.3. *Given $r \geq 2$ and $d \geq 3$, the set $\mathcal{C}_r$ is not closed.*

It means that in the general case, problem 2.3.2(b) does not necessarily have a solution, theorem 2.1.3 hypotheses are not fulfilled. The occurrence of such tensors is not rare event, see [KB09].
The next set is introduced in order to overcome these difficulties.

Lemma 2.3.4 ([Bal12, Lemma 4.16]). *Let $r \in \mathbb{N}^*$ and $c > 0$. The set*

$$\mathcal{C}_r^c = \left\{ \sum_{j=1}^{r} \boldsymbol{\mathcal{X}}_j \ : \ \boldsymbol{\mathcal{X}}_j \in \mathcal{C}_1(\mathbb{R}^{\mathcal{I}}), \|\boldsymbol{\mathcal{X}}_j\| \leq c, \ j = 1, ..., r \right\} \subset \mathcal{C}_r(\mathbb{R}^{\mathcal{I}})$$

is closed.

Corollary 2.3.4.1. *Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{\mathcal{I}}$. The following problem has a solution : Given $r \in \mathbb{N}$ and $c > 0$, find a tensor $\tilde{\boldsymbol{\mathcal{X}}} \in \mathcal{C}_r^c$ that minimizes the error $\varepsilon = \|\tilde{\boldsymbol{\mathcal{X}}} - \boldsymbol{\mathcal{X}}\|$.*

Several algorithm ensure the boundedness of the norms of the terms $\boldsymbol{\mathcal{X}}_j$ but the drawback is the existence of local minima which are usually not a problem in practical applications. Next section introduces a classical CP decomposition algorithm.

2.3.1.2 Computing the CP decomposition : the ALS algorithm

Although there are many approaches to compute a CP decomposition, in this section we focus on the classical Alternating Least Square (**ALS**) approach. This method was introduced by Carroll and Chang [CC70] and Harshman [Har70]. If not the most efficient it is highly reliable and quite simple. To ease the presentation we stick to a third order tensor although the algorithm can be easily extended to a d-way tensor.

Let $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ a third order tensor. $\tilde{\mathcal{X}}$, the best rank-R approximation of $\mathcal{X}$ is sought i.e.

$$\min_{\tilde{\mathcal{X}}} \|\mathcal{X} - \tilde{\mathcal{X}}\| \quad \text{with} \quad \tilde{\mathcal{X}} = [\![\boldsymbol{\lambda}; \boldsymbol{A}, \boldsymbol{B}, \mathbf{C}]\!] \equiv \sum_{r=1}^{R} \lambda_r \, \boldsymbol{a}_r \circ \boldsymbol{b}_r \circ \boldsymbol{c}_r \tag{2.3.4}$$

The ALS approach is to fix $\boldsymbol{B}$ and $\boldsymbol{C}$ to solve for $\boldsymbol{A}$ then fix $\boldsymbol{A}$ and $\boldsymbol{C}$ to solve for $\boldsymbol{B}$ etc. until the procedure converges. Having fixed all but one matrices, the problem reduces to a linear least-square problem which can be solved using the usual tools. Although this algorithm is quite simple to implement and understand, it does not necessarily converges to the global minimum of the objective function. Only a local minimum is ensured. Moreover, it can take a large number of iteration to converge. Finally, its result may depend on the arbitrary initial values (see Kolda [KB09] for a detailed algorithm).

Algorithm 7: ALS

input : $\mathcal{F} \in \mathbb{R}^{I_1 \times \cdots \times I_d}$
output: $\mathcal{X} = w \bigotimes_{i=1}^{d} \boldsymbol{x}_i$

Initialize $\forall 1 \leq i \leq d, \quad \boldsymbol{x}_i$;
while *Error* $\geq \varepsilon$ **do**
  for $i = 1, d$ **do**
1 $V = \boldsymbol{X}_1{}^{\intercal}\boldsymbol{X}_1 * \cdots * \boldsymbol{X}_{i-1}{}^{\intercal}\boldsymbol{X}_{i-1}{}^{\intercal} * \boldsymbol{X}_{i+1}{}^{\intercal}\boldsymbol{X}_{i+1} * \cdots * \boldsymbol{X}_d{}^{\intercal}\boldsymbol{X}_d$; /* $V \in \boldsymbol{R}^{R \times R}$ */
2 $\boldsymbol{X}_i = \mathcal{F} \cdot (\boldsymbol{X}_d \odot \cdots \odot \boldsymbol{X}_{i+1} \odot \boldsymbol{X}_{i-1} \odot \cdots \odot \boldsymbol{X}_1)V^{\dagger}$; /* † refer to the Monroe-Penrose pseudo-inverse */
    $w_i = \|\boldsymbol{X}_i\|_2$;
    $\boldsymbol{X}_i = \frac{\boldsymbol{X}_i}{w_i}$
return $\mathcal{X} = [\![\boldsymbol{w}; \boldsymbol{X_1}, \cdots, \boldsymbol{X_d}]\!]$

This algorithm led to many development but they are generally outperformed in the production stage by several Tucker Decomposition methods such as the HOSVD (which will be discussed later, see section 2.3.2.1). Finally, a link can be observed with the PGD algorithm (see section 1.3 and 3.1) which also perform a minimization algorithm through an iterative process.

It is possible to rewrite the CP format using vector spaces of unknown nature such as infinite spaces. Still one needs to define storage on a computer the continuous bases function for example. The case of function decomposition into CP format is studied in section 3.1.

2.3.2 Tucker decomposition

Introduction to the Tucker Decomposition : A 3D example The Tucker decomposition was first introduced by Tucker during the 1960s [Tuc66] and further refined. As for the CP decomposition, the Tucker Decomposition has been "rediscovered" many times in several fields leading to several names (HOSVD [DDV00, dLdMV00], N-Modes PCA, etc.). It is an extension of the SVD to higher dimensions. A tensor is decomposed into a *core* tensor that is multiplied by a matrix along each mode.

Once again, the case of a third order tensor is proposed for introduction simplicity. But, the Tucker decomposition is well defined for dimensions higher than 3. Figure 2.10 shows a graphical interpretation of the following equation for $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$,

$$\boldsymbol{\mathcal{X}} \approx [\![\boldsymbol{\mathcal{W}}; \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]\!] = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} w_{pqr}\, \boldsymbol{a}_p \circ \boldsymbol{b}_q \circ \boldsymbol{c}_r \tag{2.3.5}$$

Where $\boldsymbol{A} \in \mathbb{R}^{I \times P}$, $\boldsymbol{B} \in \mathbb{R}^{J \times Q}$ and $\boldsymbol{C} \in \mathbb{R}^{K \times R}$ are the factor matrices. There are usually set orthonormal and can be viewed as the principal components of each modes. $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{P \times Q \times R}$ is the *core tensor*. If $I < P$, $J < Q$ and $K < R$ then it can be seen as the compression of $\boldsymbol{\mathcal{X}}$ given the basis formed by $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$.

Element wise, the tucker decomposition in 2.3.5 is $\forall (i, j, k) \in [\![1, I]\!] \times [\![1, J]\!] \times [\![1, K]\!]$,

$$x_{ijk} = \sum_{p=1}^{P} \sum_{q=1}^{Q} \sum_{r=1}^{R} w_{pqr}\, a_{ip}\, b_{jq}\, c_{kr}$$

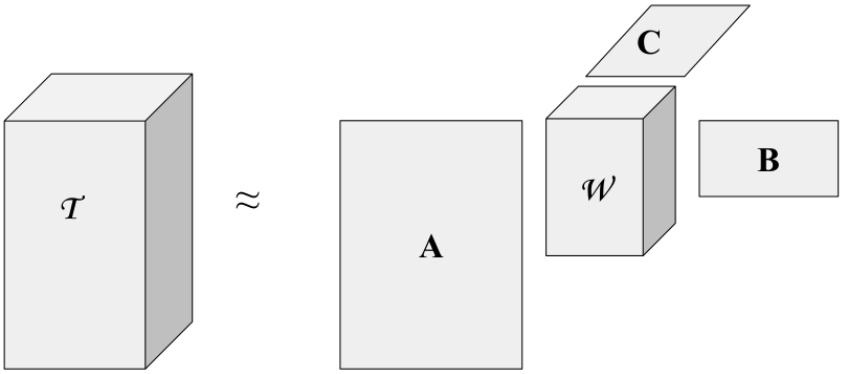


Figure 2.10: Tucker Decomposition of a third order array $\boldsymbol{\mathcal{T}}$

It is easy to find the exact decomposition of a rank-$(R_1, ..., R_D)$ tensor (see def. 2.1.8) as presented in the next subsection. However, if one wants to compute a rank-$(R_1, ..., R_D)$ Tucker decomposition of a tensor where $\exists\, n \leq D \mid R_n < \text{rank}_n(\boldsymbol{\mathcal{X}})$ then this decomposition is necessarily inexact which may raise some computational difficulties. Since such a decomposition exclude some eigen vectors, it is called a *truncated* Tucker decomposition, a visual example is shown in figure 2.11.

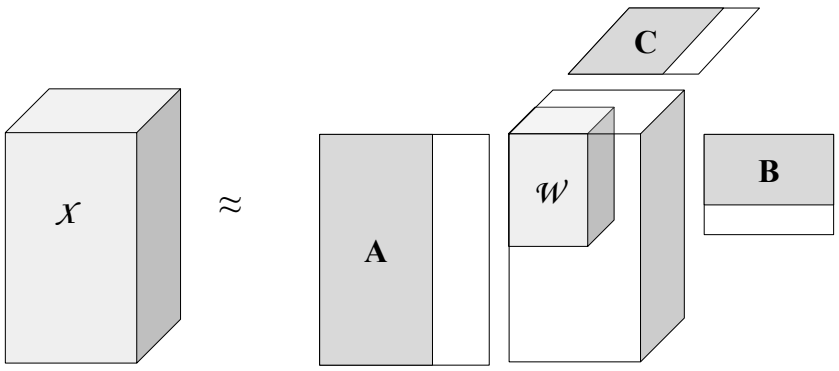


Figure 2.11: *Truncated* Tucker Decomposition of a third order array $\boldsymbol{\mathcal{X}}$

It should be noted that there are many ways to compute truncated tucker decompositions, among them various ALS based methods and the Higher Order Orthogonal Iteration (HOOI) proposed by De Lathauwer et al. [dLdMV00] which yields some optimality properties. Finally, the most common method, because it is computationally the most efficient, is the Higher Order Singular Value Decomposition (**HOSVD**) which was introduced by De Lathauwer et al in 2000 [DDV00]. It was then extensively studied and improved as in [VVM12] with the Sequentially Truncated HOSVD.

From the Tucker Format to the Tucker Decomposition (HOSVD) In this paragraph, some mathematical properties of the Tucker decomposition are reviewed. They lead to the classical tensor Tucker format reduction technique Higher Order Singular Value Decomposition (HOSVD) which is presented in two forms. The first one was proposed by De Lathauwer et al in 2000 [DDV00] and the second one is a 2012 improvement from Van Nieuwenhoven [VVM12], the Sequentially Truncated HOSVD.

Definition 2.3.3. *The Tucker decomposition of a tensor $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ is to find an approximation of a tensor of $\mathbb{R}^{\mathcal{I}}$ in $\mathcal{T}_{\boldsymbol{k}}$. It means that either of these problems have to be solved*

a. Given $\varepsilon > 0$, find $\tilde{\mathcal{X}} \in \mathcal{T}_{\boldsymbol{k}}$ with minimal $N_{\text{storage}}(\mathcal{T}_{\boldsymbol{k}})$ such that $\|\tilde{\mathcal{X}} - \mathcal{X}\| \leq \varepsilon$.

b. Given $\boldsymbol{k} \in (\mathbb{N}*)^d$, find $\tilde{\mathcal{X}} \in \mathcal{T}_{\boldsymbol{k}}$ that minimises the error $\varepsilon = \|\tilde{\mathcal{X}} - \mathcal{X}\|$.

Given that either of these problem has a solution the following identity is obtained

$$\mathcal{X} \approx \tilde{\mathcal{X}} = \sum_{j_1=1}^{k_1} \cdots \sum_{j_d=1}^{k_d} w_{\boldsymbol{j}} \bigotimes_{\mu=1}^{d} \tilde{\boldsymbol{x}}_{\mu}^{i} \tag{2.3.6}$$

Lemma 2.3.5. *Problem (a) has a solution.*

Lemma 2.3.6. *Let $\boldsymbol{k} = (k_1, ..., k_d) \in (\mathbb{N}*)^d$. The set $\mathcal{T}_{\boldsymbol{k}} \subset \mathbb{R}^{\mathcal{I}}$ is closed. Consequently Problem (b) has a solution.*

Since the Tucker format is closely related to the matricization of tensors. Then the idea of using the SVD (see 1.1) on matricizations of the investigated tensor has been used to devise algorithm to give an approximate solution to problems 2.3.3(a) and (b). For most applications, it is not necessary to find the best approximation, an *almost* best approximation is sufficient. A common tool to perform this task is the Higher Order Singular Value Decomposition (HOSVD) introduced by De Lathauwer in [DDV00].

2.3.2.1 HOSVD

Theorem 2.3.7 (HOSVD as proved in [DDV00] by De Lathauwer et al.)**.** *Every tensor $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ admits a* higher-order singular value decomposition*:*

$$\mathcal{X} = (\boldsymbol{U}_1, \boldsymbol{U}_2, ..., \boldsymbol{U}_d) \cdot \mathcal{W}, \tag{2.3.7}$$

where the factor matrix $\boldsymbol{U}_\mu$ *is an orthogonal $n_\mu \times n_\mu$ matrix, obtained from the SVD of the mode-μ matricization of $\mathcal{X}$,*

$$\boldsymbol{X}^{(\mu)} = \boldsymbol{U}_\mu \boldsymbol{\Sigma}_\mu \boldsymbol{V}_\mu^{\intercal}, \tag{2.3.8}$$

and the core tensor $\mathcal{W} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ *can be obtained from*

$$\mathcal{W} = (\boldsymbol{U}_1^{\intercal}, \boldsymbol{U}_2^{\intercal}, ..., \boldsymbol{U}_d^{\intercal}) \cdot \mathcal{X}, \tag{2.3.9}$$

Remark (Truncation). Theorem 2.3.7 refers to full HOSVD which is an exact Tucker decomposition. However it gives a lot of information about a studied tensor such as the multilinear rank, it is rarely the pursued goal. This kind of decomposition is aimed at extracting the most relevant information, possibly by reducing data size. The optimality of the SVD truncation encourages to think of truncating $(\boldsymbol{U}_\mu)$. This is what is done is the Truncated-HOSVD (T-HOSVD) which is generally referred as HOSVD. However in this section the T-HOSVD notation will be used in order to prevent confusion.

Algorithm idea. The T-HOSVD algorithm relies on the simple truncation idea. First compute $(\boldsymbol{U}_\mu)$ defined in equation (2.3.8) in each direction, then truncate to a given rank/column (set prior to computing). Finally compute $\boldsymbol{\mathcal{W}}^t$, the truncated core tensor projecting $\boldsymbol{\mathcal{X}}$ on the reduced basis $(\boldsymbol{U}_\mu^t)$ as in equation (2.3.9).

Of course the truncation of the SVD does not mean that the 2D optimality is preserved. Optimality is not the goal of most applications and this algorithm is easy to use then a quasi-optimality is sufficient. The quasi-optimality with respect to the optimal rank-$\boldsymbol{k}$ approximation is given by the following theorem.

Theorem 2.3.8 (Quasi-optimality of the T-HOSVD [DDV00, Property 10])**.** *Let $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{\mathcal{I}}$ with a μ-rank $\boldsymbol{r} = (r_1, ..., r_d) \in \mathbb{N}^d$. Given $\boldsymbol{k} = (k_1, ..., k_n) \in \mathbb{N}^d$, let $\boldsymbol{\mathcal{X}}_{best}$ be the best approximation of $\boldsymbol{\mathcal{X}}$ in $\mathcal{T}_{\boldsymbol{k}}$ i.e. $\boldsymbol{\mathcal{X}}_{best} = \operatorname{argmin}_{\boldsymbol{\mathcal{Y}} \in \mathcal{T}_{\boldsymbol{k}}} \|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{Y}}\|_2$. Then the error of HOSVD projection is bounded by*

$$\|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}_{\text{hosvd}}\|_2 \leq \sqrt{\sum_{\mu=1}^{d} \sum_{j=k_\mu+1}^{r_\mu} \sigma_{\mu,j}^2} \leq \sqrt{d}\, \|\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{X}}_{best}\|_2 \tag{2.3.10}$$

where the $\sigma_{\mu,j}$ are the singular values defined in equation (2.3.8).

The approximation error of HOSVD is bounded by the middle term in equation (2.3.10), namely $\sqrt{\sum_{\mu=1}^{d} \sum_{j=k_\mu+1}^{r_\mu} \sigma_{\mu,j}^2}$. Forcing this term to be lower than a given ε leads to an adaptively truncated HOSVD for which an error bound is chosen.

Algorithm 8 presents the truncated HOSVD algorithm that computes $\boldsymbol{\mathcal{X}} \in \mathcal{T}_{\boldsymbol{k}}$ of rank $\boldsymbol{k}$ the approximation of $\boldsymbol{\mathcal{F}} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$. It is a rather compact algorithm given that one has efficient methods to compute basic tensor operations. The implementation simplicity of the algorithm is one of the main reason of its success.

Algorithm 8: T-HOSVD

input : $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, imposed rank : $\boldsymbol{k} = (k_1, ..., k_d)$
output: $\widetilde{\boldsymbol{\mathcal{X}}} = (\boldsymbol{U}_1, \cdots, \boldsymbol{U}_d) \cdot \boldsymbol{\mathcal{W}}$

for $i = 1, d$ **do**
1 $\quad \boldsymbol{X}^{(\mu)}$ =matricize$(\boldsymbol{\mathcal{F}}, \mu)$;
2 $\quad (\boldsymbol{U}_\mu, \boldsymbol{\Sigma}_\mu, \boldsymbol{V}_\mu{}^\intercal)$=SVD$(\boldsymbol{X}^{(\mu)})$;
$\boldsymbol{\mathcal{W}} = ((\boldsymbol{U}_1^{k_1})^\intercal, (\boldsymbol{U}_2^{k_2})^\intercal, ..., (\boldsymbol{U}_d^{k_d})^\intercal) \cdot \boldsymbol{\mathcal{X}}$;
`/*` $\boldsymbol{U}_i^{k_i}$ `contains the first` k_i `columns of` $\boldsymbol{U}_i$ `*/`
return $\widetilde{\boldsymbol{\mathcal{X}}} = [\![\boldsymbol{\mathcal{W}}; \boldsymbol{U}_1, \cdots, \boldsymbol{U}_d]\!]$

Remark. This algorithm is easily parallelized to the number of dimension (lines **1** and **2**), each processor computing an SVD. Additionally, it is possible to reach higher level of parallelization using parallel linear algebra routines.

Operation count In order to simplify the computations, the case of an order d cubic tensor is investigated, e.g. $\mathcal{I} = n \times \cdots \times n$. This tensor is approximated by a rank $(r, ..., r)$ Truncated HOSVD. Then in every mode of the tensor the SVD of a $n \times n^{d-1}$ matrix is computed plus computing a core tensor which means d matrix multiplications. Thus the operation count is

$$\mathcal{O}\left(dn^{d+1} + \sum_{k=1}^{d} r^k n^{d-k+1}\right) \tag{2.3.11}$$

In order to ease comparison with the following subsection, it should be noted that the T-HOSVD can be seen as a series of orthogonal projections onto the tensor basis $(\boldsymbol{U}_1^{k_1}, ..., \boldsymbol{U}_d^{k_d})$. Then we define

$$\boldsymbol{\mathcal{X}}_{\text{hosvd}} = \pi_1 \pi_2 \cdots \pi_d \boldsymbol{\mathcal{X}} = (\boldsymbol{U}_1^{k_1} \boldsymbol{U}_1^{k_1\intercal}, ..., \boldsymbol{U}_d^{k_d} \boldsymbol{U}_d^{k_d\intercal}) \boldsymbol{\mathcal{X}} \approx \boldsymbol{\mathcal{X}} \tag{2.3.12}$$

where $\pi_i = (I, ..., I, \boldsymbol{U}_i^{k_i} \boldsymbol{U}_i^{k_i\intercal}, I, ..., I)$ is the projector onto mode i.

Continuous equivalent. The T-HOSVD was presented for tensors, however it was seen in section 1 that SVD and POD are closely related and may be considered equivalent. Then it is easy to adapt this algorithm to a multivariate square integrable function. One has to replace SVD with POD and discrete scalar products by integrals one.

2.3.2.2 ST-HOSVD

The Sequentially Truncated HOSVD (ST-HOSVD) was introduced by Vanniewenhoven et al. [VVM12]. This method is a variation of the usual T-HOSVD. Basically, instead of throwing away most of the work performed by each SVD as shown in figure 2.12, it is chosen to keep that information and perform SVD sequentially -on a reduced tensor- along all dimensions. The divergence of these approaches is highlighted in figure 2.13. Since processing is sequential and the order in which the operations are performed has an influence on the approximation, the sequence order is stored in a vector $\boldsymbol{p}$. For the sake of simplicity, it is assumed that $\boldsymbol{p} = (1, 2, ..., d)$ even though many of the results depend on the permutations of $\boldsymbol{p}$.

The ST-HOSVD has been presented using successive projections. In this framework it is easy to both understand the idea of the method and to demonstrate its properties.

Definition 2.3.4 (Orthogonal multilinear projector)**.** *An* orthogonal projector *is a linear transformation P that projects a vector $\boldsymbol{x} \in \mathbb{R}^n$ onto a vector space $E \subseteq \mathbb{R}^n$ such that the residual $\boldsymbol{x} - P\boldsymbol{x}$ is orthogonal to E. Such a projector can always be represented as in matrix form $P = \boldsymbol{U}\boldsymbol{U}^\intercal$ given that the columns of $\boldsymbol{U}$ form an orthonormal basis of E.*

Then Silva and Lim [dSL08] proposed the introduction of orthogonal multilinear projectors *from tensor space $\boldsymbol{\mathcal{V}} = V_1 \otimes \cdots \otimes V_d$ onto $\boldsymbol{\mathcal{U}} = U_1 \otimes \cdots \otimes U_d \subset \boldsymbol{\mathcal{V}}$. It is given by*

$$\pi_i \boldsymbol{\mathcal{X}} := (I, ..., I, \boldsymbol{U}_i \boldsymbol{U}_i^\intercal, I, ..., I) \cdot \boldsymbol{\mathcal{X}} \quad \text{with } \boldsymbol{\mathcal{X}} \in \boldsymbol{\mathcal{V}} = \mathbb{R}^{\mathcal{I}} \tag{2.3.13}$$

Definition 2.3.5. *ST-HOSVD [VVM12, Def. 6.1.]*
A rank-$(r_1, ..., r_d)$ sequentially truncated higher-order singular value decomposition *(ST-HOSVD) of a tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{\mathcal{I}}$, corresponding to the processing order $\boldsymbol{p} = [1, 2, ..., d]$, is an approximation of the form*

$$\hat{\boldsymbol{\mathcal{X}}}_{\boldsymbol{p}} := (\hat{\boldsymbol{U}}_1, \hat{\boldsymbol{U}}_2, ..., \hat{\boldsymbol{U}}_d) \cdot \hat{\boldsymbol{\mathcal{W}}} \approx \boldsymbol{\mathcal{X}} \quad \in \mathbb{R}^{n_1 \times \cdots \times n_d} \tag{2.3.14}$$

whose truncated core tensor *is defined as*

$$\hat{\boldsymbol{\mathcal{W}}} := (\hat{\boldsymbol{U}}_1^\intercal, \hat{\boldsymbol{U}}_2^\intercal, ..., \hat{\boldsymbol{U}}_d^\intercal) \cdot \boldsymbol{\mathcal{X}} \quad \in \mathbb{R}^{r_1 \times \cdots \times r_d} \tag{2.3.15}$$

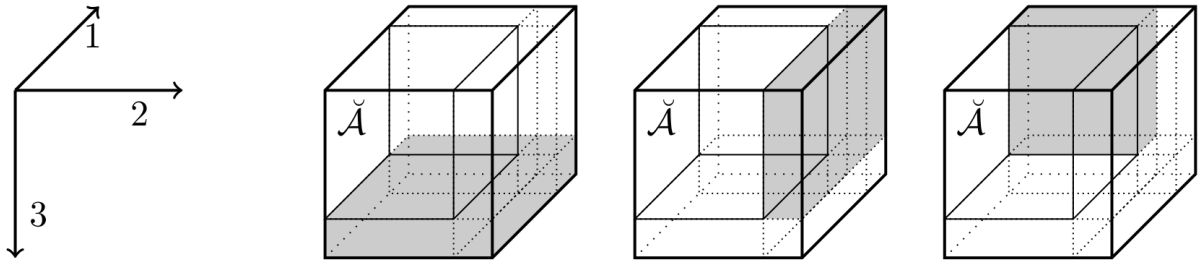


Figure 2.12: A visual of the truncated HOSVD, it shows that the approximation $\breve{\boldsymbol{\mathcal{A}}}$ is defined simultaneously for each direction through least square approximation. Figure from [VVM12].

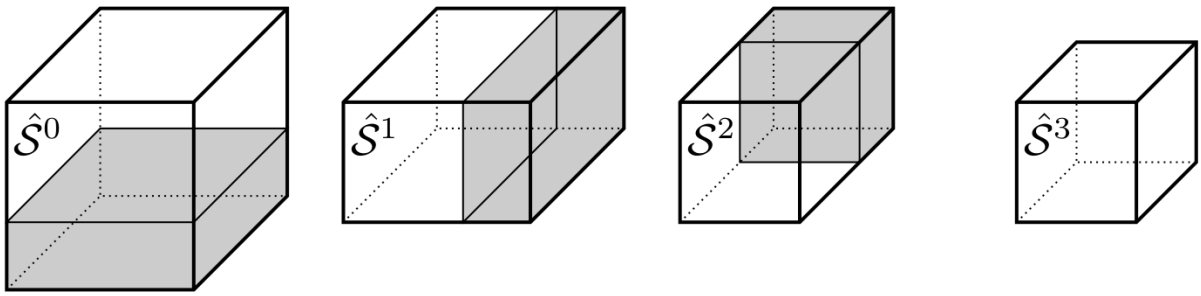


Figure 2.13: A visual of the *sequentially* truncated HOSVD with the same axis as figure 2.12. Here each approximated tensor $\hat{\boldsymbol{\mathcal{S}}}^i$ is performed on one dimension then a new tensor $\hat{\boldsymbol{\mathcal{S}}}^{i+1}$ is processed on the next direction. Processing order is (3,2,1). Figure from [VVM12].

and every factor matrix $\hat{\boldsymbol{U}}_i^\intercal \in \mathbb{R}^{n_i \times r_i}$ *has orthonormal columns. In terms of orthogonal multilinear projectors, one writes*

$$\hat{\boldsymbol{\mathcal{X}}}_{\boldsymbol{p}} := \hat{\pi}_1 \hat{\pi}_2 \cdots \hat{\pi}_d \boldsymbol{\mathcal{X}} = (\hat{\boldsymbol{U}}_1 \hat{\boldsymbol{U}}_1^\intercal, \hat{\boldsymbol{U}}_2 \hat{\boldsymbol{U}}_2^\intercal, ..., \hat{\boldsymbol{U}}_d \hat{\boldsymbol{U}}_d^\intercal) \cdot \boldsymbol{\mathcal{X}}$$

The i-th partially truncated core tensor *is defined as*

$$\hat{\boldsymbol{\mathcal{W}}}^i := (\hat{\boldsymbol{U}}_1^\intercal, \hat{\boldsymbol{U}}_2^\intercal, ..., \hat{\boldsymbol{U}}_i^\intercal, \boldsymbol{I}, ..., \boldsymbol{I}) \cdot \boldsymbol{\mathcal{X}} \quad \in \mathbb{R}^{r_1 \times \cdots \times r_i \times n_{i+1} \times \cdots \times n_d} \tag{2.3.16}$$

with $\hat{\boldsymbol{\mathcal{W}}}^0 := \boldsymbol{\mathcal{X}}$ *and* $\hat{\boldsymbol{\mathcal{W}}}_d = \hat{\boldsymbol{\mathcal{W}}}$. *The rank-*$(r_1, ..., r_i, n_{i+1}, ..., n_d)$ partial approximation to *to* $\boldsymbol{\mathcal{X}}$ *is defined as*

$$\hat{\boldsymbol{\mathcal{X}}}^i = (\hat{\boldsymbol{U}}_1, \hat{\boldsymbol{U}}_2, ..., \hat{\boldsymbol{U}}_i, \boldsymbol{I}, ..., \boldsymbol{I}) \cdot \hat{\boldsymbol{\mathcal{W}}}^i \quad \in \mathbb{R}^{n_1 \times \cdots \times n_d}$$

with $\hat{\boldsymbol{\mathcal{X}}}^0 = \boldsymbol{\mathcal{X}}$ *and* $\hat{\boldsymbol{\mathcal{X}}}^0 = \hat{\boldsymbol{\mathcal{X}}}$.

The factor matrix $\hat{\boldsymbol{U}}_i$, $1 \leq i \leq d$, *is the matrix of the* r_i *dominant left singular vectors of the mode-i vector space of* $\hat{\boldsymbol{\mathcal{W}}}^{i-1}$. *It is obtained from the rank* r_i *truncated singular value decomposition of the* $(i-1)$*th partially truncated core tensor, as follows:*

$$\hat{\boldsymbol{\mathcal{W}}}_{(i)}^{i-1} = \boldsymbol{U}_i \boldsymbol{\Sigma}_i \boldsymbol{V}_i^\intercal$$

where $\boldsymbol{U}_i = [\hat{\boldsymbol{U}}_i \; \tilde{\boldsymbol{U}}_i]$.

The hat projector $\hat{\pi}_i$ is defined recursively contrary to T-HOSVD. Indeed, the definition of the $i+1$ projector is optimal for the partially approximated tensor $\hat{\boldsymbol{\mathcal{X}}}^i$. This leads to strongly improved performance if r_i is small. However, as stated earlier, the processing order is very importante since it changes both the approximation and projectors. The ST-HOSVD algorithm is given next.

Algorithm 9: ST-HOSVD

input : $\mathcal{F} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, truncation rank $\boldsymbol{r}$, processing order $\boldsymbol{p}$
output: $\hat{\mathcal{X}} = (\hat{\boldsymbol{X}}_1, ..., \hat{\boldsymbol{X}}_d) \cdot \hat{\mathcal{W}}$

$\hat{\mathcal{W}} = \mathcal{F}$;
for $i = p_1, ..., p_d$ **do**
    /* Compute SVD of $\hat{\mathcal{W}}_{(i)}$ then truncate to r_i */
1   $(\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}^\intercal) = \text{SVD}(\hat{\mathcal{W}}_{(i)})$;
2   $(\boldsymbol{U}_{tr}, \boldsymbol{\Sigma}_{tr}, \boldsymbol{V}_{tr}^\intercal) = trunctate(\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}^\intercal, r_i)$;
3   $\hat{\boldsymbol{X}}_i = \boldsymbol{U}_{tr}$;
4   $\hat{\mathcal{W}}_{(i)} = \boldsymbol{\Sigma}_{tr} \boldsymbol{V}_{tr}^\intercal$;
return $\mathcal{X} = [\![\hat{\mathcal{W}}; \hat{\boldsymbol{X}}_1, ..., \hat{\boldsymbol{X}}_d]\!]$

It is possible to use a compact SVD which only yields the truncated SVD. This improve memory efficiency as well as computing speed, especially if the multilinear rank is small. A visual of algorithm 9 is given in figure 2.13. One can see that the approximated tensor reduces after each truncated SVD finally reaching its final shape after the last dimension has been reduced. It is interesting to note that if the gray area is large, the next tensor size can be much smaller than the original tensor (see figure 2.12). Thus the SVD will be much faster than its T-HOSVD counterpart.

Remark. The processing order has been reported to influence greatly the computing time in addition to the obvious influence on the approximation itself. [VVM12] proposed a heuristic that attempts to minimize the number of operations required to compute the dominant subspace. Then one should first process the dimension with lowest size and so on. This may even reduced the rank of the remaining terms, i.e. "*forcing more energy into fewer modes*". However choosing a processing order that minimizes the error is still an open question.

Error estimate. For a given multilinear rank, both ST and T-HOSVD approximations satisfy the same error bounds. However, usually, ST-HOSVD performs better in term of actual approximation error (see [VVM12] section 7).

Theorem 2.3.9 (error bound ST-HOSVD, [VVM12, Theorem 6.5])**.** *Let $\mathcal{X} \in \mathbb{R}^{\mathcal{I}}$ a tensor and $\hat{\mathcal{X}}$ be the rank- $(r_1, ..., r_d)$ ST-HOSVD of $\mathcal{X}$. Let the SVD of $\mathcal{X}_{(i)}$ be given as in (2.3.8). Then the bounds of the ST-HOSVD are*

$$\min_i \|\tilde{\boldsymbol{\Sigma}}_i\|_F^2 \leq \|\mathcal{X} - \hat{\mathcal{X}}\|_F^2 \leq \sum_{k=1}^{d} \|\tilde{\boldsymbol{\Sigma}}_i\|_F^2 \tag{2.3.17}$$

where $\tilde{\boldsymbol{\Sigma}}$ is the discarded part of $\boldsymbol{\Sigma}$ obtained from the SVD.

Operation count. The main goal of this method is to cut-off the operation count that was observed in the T-HOSVD. Again this estimate is restricted to cubic order-d tensors in $\mathbb{R}^{n \times \cdots \times n}$ as it is straightforward to generalize to general shapes. This tensor is approximated by a rank $(r, ..., r)$ ST-HOSVD. Assume that the SVD algorithm on a $m \times n$ matrix operation count is $\mathcal{O}(m^2 n)$. Then in every mode of the tensor the SVD of a $n \times r^{i-1} n^{d-i}$ matrix is computed plus computing a core tensor which means d matrix multiplications. Thus the operation count is

$$\mathcal{O}\left(\sum_{i=1}^{d} r^{i-1} n^{d-i+2} + \sum_{i=1}^{d} r^i n^{d-i}\right) \tag{2.3.18}$$

The second term in the sum is due to the scaling of the right singular vectors with the singular values.

In this section on computing the Tucker decomposition of a tensor, 2 methods were investigated. Both satisfy the same error bounds. On the one hand, the T-HOSVD is straightforward to implement and allows easy parallelized implementation for low number of CPU. Analysis is also relatively easy and the processing order has no influence on the approximation. On the other hand the ST-HOSVD is inherently sequential which means that processing order changes both the operation count and the approximation. This leads to analysis complexity and rises the question of an optimal processing order. However, the operation count and approximation error are overwhelmingly lower compared to T-HOSVD according to Vannieuwenhoven et al. This should be confirmed in the numerical experiments section.

As a conclusion, if the problem is large and the tensor has large differences in the directions length, the ST-HOSVD should be preferred to compute truncated Tucker decomposition. Indeed the advantages overcome by large margin the implementation increased complexity.

2.3.3 Tensor Train decomposition

Tensor Train format has been discussed in section 2.2.4.1, it is specially recommanded for larger dimensions as it scales linearly with d. Moreover, numerous theorems and algorithms have been proposed in the literature, most importantly one may rely on the following set:

- existence of the full-rank approximation (2.2.8, [Ose11, Th. 2.1]),
- existence of the low-rank best approximation (2.3.10),
- `TT-SVD` algorithm for quasi optimal TT approximation (Algorithm 10),
- sampling algorithms (`TT-cross` [OT10], `TT-DMRG-cross` [Ose13], `maxvol` [OT10],...).

In this section, we go through the decomposition properties and briefly outline the sampling algorithms.

2.3.3.1 TT-SVD

As we have seen in the previous sections, SVD is a very efficient tool to decompose tensors, it turns out that TT decomposition is well suited rely on SVD too with the help of the generalized matricization from definition 2.2.4. Using the reduced notation $\boldsymbol{X}^{(\mu*)} = \mathcal{X}(i_1...i_\mu; i_{\mu+1}...i_d)$, from [OT10] we have the following property that enables the decomposition.

Theorem 2.3.10. *For any tensor $\mathcal{X} \in \mathcal{R}^{\mathcal{I}}$ there exists a TT approximation $\mathcal{T} \in \mathcal{R}^{\mathcal{I}}$ with compression rank $r_\mu = \text{rank}(\boldsymbol{X}^{(\mu*)})$ such that*

$$||\mathcal{X} - \mathcal{T}||_F \leq \sqrt{\sum_{\mu=1}^{d-1} \varepsilon_\mu^2} \tag{2.3.19}$$

where ε_μ^2 is the distance (in Frobenius norm) from $\boldsymbol{X}^{(\mu*)}$ to its best rank-r_μ approximation:

$$\varepsilon_\mu^2 = \min_{\text{rankB} \leq r_\mu} ||\boldsymbol{X}^{(\mu*)} - \boldsymbol{B}||_F \tag{2.3.20}$$

Proof. The detailed proof is available in [OT10], here an adapted version is provided as it is constructive of the `TT-SVD` algorithm.

First, consider the case $d = 2$. The TT decomposition of $\mathcal{Z}$ reads

$$Z(i_1, i_2) = \sum_{\alpha_1=1}^{r_1} G_1(i_1, \alpha_1) G_2(\alpha_1, i_2) \tag{2.3.21}$$

and coincides with the dyadic decomposition of matrix $\boldsymbol{Z}$. As shown in section 1.1, such an expression can be obtained optimally using truncated SVD at rank r_1 which is associated with truncation error ε_1.

By induction, the same is true for $\boldsymbol{X}^{(1)}$ the 1-matricization of $\mathcal{X}$ an order d tensor.

$$\boldsymbol{X}^{(1)} = [X(i_1; i_2 ... i_d)] = U \Sigma V^\intercal \tag{2.3.22}$$

Let $\boldsymbol{Y}_1 = \boldsymbol{U}_1 \tilde{\boldsymbol{\Sigma}} \tilde{V^\intercal}$ be the (best) r_1-rank approximation of $\boldsymbol{X}^{(1)}$ by truncated SVD i.e.

$$\boldsymbol{X}^{(1)} = \boldsymbol{Y}_1 + \boldsymbol{E}_1 \tag{2.3.23}$$

where $||E_1||_F = \varepsilon_1$. Of course, $\boldsymbol{Y}_1$ can be considered as a tensor $\mathcal{Y} = [Y(i_1, ..., i_d)]$. Then the approximation problem of $\mathcal{X}$ reduces to the one for $\mathcal{Y}$. $\mathcal{Y}$ being the best r_1-rank approximation any tensor $\mathcal{T}$ with $\boldsymbol{T}^{(1)} = \boldsymbol{U}_1 \boldsymbol{W}$ has a nil projection on E_1. It implies the following equality

$$||(\mathcal{X} - \mathcal{Y}) + (\mathcal{Y} - \mathcal{T})||_F = ||\mathcal{X} - \mathcal{Y}||_F + ||\mathcal{Y} - \mathcal{T}||_F \tag{2.3.24}$$

So far the dimensionality of $\mathcal{Y}$ has not been reduced, to do so one can rewrite $\boldsymbol{Y}^{(1)}$ such that element-wise it reads

$$Y(i_1; i_2, ..., i_d) = \sum_{\alpha_1=1}^{r_1} U_1(i_1; \alpha_1) \tilde{X}(\alpha_1; i_2 ..., i_d)$$

where $\tilde{\boldsymbol{X}} = \tilde{\Sigma} V_1$. Then, the concatenation of indices α_1 and i_2 into one long index leads to the following order $(d-1)$ tensor

$$\tilde{\mathcal{X}} = [\tilde{X}(\alpha_1 i_2, i_3, ..., i_d)]$$

By induction, $\tilde{\mathcal{X}}$ admits a TT approximation $\tilde{\mathcal{T}} = [\tilde{T}(\alpha_1 i_2, i_3, ..., i_d)]$ of the form

$$\tilde{T}(\alpha_1 i_2, i_3, ..., i_d) = \sum_{\alpha_2, ..., \alpha_{d-1}} G_2(\alpha_1 i_2, \alpha_2) G_3(\alpha_2, i_3, \alpha_3) \cdots G_d(\alpha_{d-1}, i_d)$$

Such that

$$||\tilde{\mathcal{X}} - \tilde{\mathcal{T}}||_F \leq \sqrt{\sum_{k=2}^{d} \tilde{\varepsilon}_k^2}$$

with $\tilde{\varepsilon}_k^2 = \min_{\text{rank}(C) \leq r_\mu} ||\tilde{\boldsymbol{X}}^{(\mu *)} - C||_F$.

Now let us set $G_1(i_1, \alpha_1) = U_1(i_1, \alpha_1)$, separate indices α_1 and i_2 from the long index $\alpha_1 i_2$ and define $\mathcal{T}$ by the following tensor train:

$$T(i_1, i_2, ..., i_d) = \sum_{\alpha_1, ..., \alpha_{d-1}} G_1(i_1, \alpha_1) G_2(\alpha_1, i_2, \alpha_2) \cdots G_d(\alpha_{d-1}, i_d)$$

The rest of the demonstration consists in estimating $||\mathcal{X} - \mathcal{T}||_F$ through evaluations of the approximation error between $||\tilde{\mathcal{X}} - \tilde{\mathcal{T}}||_F$ which bounds the former. Details in [OT10]. □

Corollary 2.3.10.1. *2.1 [OT10] If a tensor $\mathcal{X}$ admits a canonical approximation rank R and accuracy ε, then there exists a tensor train approximation with compression ranks $r_k \leq R$ and accuracy $\sqrt{d-1}\varepsilon$.*

Corollary 2.3.10.2. *2.2 [OT10] Given a tensor $\mathcal{X}$, denote by $\varepsilon = \inf_{\mathcal{Y}} ||\mathcal{X} - \mathcal{Y}||_F$ the infimum of distances between $\mathcal{X}$ and tensor train $\mathcal{Y}$ with prescribed upper bounds r_μ on the ranks of unfoldings matrices (compression ranks), i.e. $\text{rank}\boldsymbol{Y}^{(\mu)} \leq r_\mu$. Then the optimal $\mathcal{Y}$ exists (in a fact a minimum) and the TT approximation $\mathcal{T}$ constructed in the proof of Theorem 2.3.10 is quasi-optimal in the sense that*

$$||\mathcal{X} - \mathcal{T}||_F \leq \sqrt{d-1}\varepsilon. \tag{2.3.25}$$

It is then natural to propose the `TT-SVD` [Ose11] algorithm for the approximation of a full format tensor into TT format.

Algorithm 10: `TT-SVD`

input : $\mathcal{F} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, truncation rank $\boldsymbol{r}$ or prescribed error ε
output: $\mathcal{X}(i_1, ..., i_d) = \sum_{\alpha_0,...,\alpha_d=1}^{\boldsymbol{r}} G_1(\alpha_0, i_1, \alpha_1) \cdots G_1(\alpha_{d-1}, i_d, \alpha_d)$

1 Compute the truncation parameter $\delta = \frac{\varepsilon}{\sqrt{d-1}}||\mathcal{F}||_F$;
2 Temporary tensor: $\mathcal{C} = \mathcal{A}$, $r_0 = 1$;
for $i = 1, ..., d$ **do**
  `/* reshape(`$\mathcal{C}, r_{i-1}n_i, \frac{\texttt{numel}(C)}{r_{i-1}n_i}$`)` `*/`
3 $\boldsymbol{C} = \boldsymbol{C}^{(i*)}$;
  `/* truncated SVD at given rank` r_i `*/`
4 $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^\intercal = \text{tSVD}(C, r_k, \delta)$;
5 $\mathcal{G}_i = \text{reshape}(\boldsymbol{U}, [r_{i-1}, n_i, r_i])$;
6 $\boldsymbol{C} = \boldsymbol{\Sigma}\boldsymbol{V}^\intercal$;
7 $\mathcal{G}_d = \boldsymbol{C}$;
return $\mathcal{X} = [\![\mathcal{G}_1, \mathcal{G}_2, ...\mathcal{G}_d]\!]$

Remark. In addition to the linear algebra algorithms mentioned in section 2.2.4.1, many algorihtms have been developed to convert from canonical [Ose11], Tucker or HT to TT, for instance, one can refer to [Hac14, Chap. 12 & 13]. Also, one may need to recompress an existing TT tensor (for example after summing two TT tensors), to do so Oseledets proposes the `TT-rounding` algorithm [Ose11] based on a combination of QR decompositions and SVD.

Actually this algorithm relies on the same methodology as the `ST-HOSVD` but stores the results in the cores thus leading to TT format. As stated earlier, this leads to a linear storage cost in d which is much more efficient that Tucker format. In addition to that, the weights of the entries are stored in the last mode/core $\boldsymbol{G}_d = \mathcal{G}_d$ and modes relations are store within the cores themselves without requiring a single core tensor.

2.3.3.2 Sampling algorithms for high dimensional TT

This kind of algorithm is very well suited to analyze data from existing simulations in the context of fluid dynamics. However, if the dimensions of the studied problem grows above 5 it becomes intractable to either store the data or solve the SVD problem. In order to circumvent this difficulty, one might rely on family of methods that will be referred as *sampling* algorithms. They come under many names including `maxvol` for matrices skeleton decomposition or `TT-cross, TT-DMRG-cross,...`. Obviously this

can be done in many formats, included the also well suited HT format (see `BlackBox algorithm` [BGK10]). A short overview is proposed, many more can be found in the literature, including in [Ose11, OT10, Ose13].

The idea here is quite simple, given a suitable technique,

Find an approximation of tensor $\mathcal{X}$ with as few evaluation as possible for a precision ϵ.

the tensor does not need to be known fully, only an access to its entries is needed, for instance a blackbox function.

Skeleton decomposition This is the building block of all sampling algorithms methods. One wants to obtain the following dyadic decomposition of a matrix[2].

Definition 2.3.6 (Skeleton decomposition)**.** *Let a matrix $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ and sets of indices $\mathcal{I} = (i_1, ..., i_r) \in \{1, ..., n\}$ and $\mathcal{J} = (j_1, ..., i_r) \in \{1, ..., m\}$. Then the skeleton decomposition of $\boldsymbol{A}$ reads*

$$\boldsymbol{A} \approx \boldsymbol{A}(\mathcal{I}, :)\boldsymbol{A}(\mathcal{I}, \mathcal{J})^{-1}\boldsymbol{A}(:, \mathcal{J}) \tag{2.3.26}$$

where $A(\mathcal{I}, :)$ is the matrix extracted from $\mathcal{A}$ rows which indices are described by $\mathcal{I}$, other submatrices are defined in the same fashion.

There are many ways to compute an approximation[3], including the famous cross approximation method (see [Tyr00]), one will find alternatively the row/column and iteratively the entries that yields the lowest residual. A very efficient, although not optimal, algorithm for tensor approximation using skeleton decomposition is the `maxvol` procedure [Tyr00, SO11, OT10]. This volume maximization entry selection strategy aims at maximmizing $|\det(\boldsymbol{A}(\mathcal{I}, \mathcal{J}))|$, i.e. finding the *maximum volume submatrix*. It is well suited for TT approximation algorithms and computationally inexpensive, requiring $2c(n - r)r$, where c is usually a small constant.

TT-DMRG-cross [SO11, Algorithm 1: TT-RC] To the best of my knowledge, this method is the state of the art to perform TT decomposition in large dimension. In addition to the `maxvol` and `cross` procedure, it introduces the matrix renormalization group (`DMRG`) procedure, which makes the method rank revealing[4]. The idea here is to introduce temporary cores $W_k(i_k, i_{k+1}) = G_k(i_k)G_{k+1}(i_{k+1})$ so that the optimization is performed two terms at a time, using `maxvol` for instance. Then the modes are separated through SVD.

One major drawback of TT decomposition is that by constructions the 2-fibers of $\{\mathcal{G}_i\}$ are not necessarily orthogonal, this property is only ensure by block. Several solutions are possible to overcome it, mainly based on reorthogonalization process (usually QR algorithms) but again only left or right[5] orthogonality can be ensured for a given tensor train. Thus, in section 2.3.4 we investigate the HT decomposition methods, especially the orthogonalization process and conversion of TT to ExtTT format. This allows to access the basis directly for physical analysis as well as slightly lower storage cost. Indeed the cardinality or the vector spaces spanned by $\{\mathcal{G}_i\}$ is expected to be lower than the number of fibers stored.

[2] Skeleton decomposition is also known as matrix cross approximation.

[3] The problem of finding the optimal submatrix is NP-hard [ÇMi09]

[4] On the other hand, `TT-cross` algorithm needs a predetermined rank which may lead to a poor approximation or overestimated rank

[5] See [SO11] for details, basically with respect to the 1 and 3-matricization of the core tensors. "Note the cores can not be both left and right orthogonal." [SO11]

2.3.4 Hierarchical Tucker decomposition

Hierarchical Tucker decomposition is a growing topic in the tensor decomposition community [GKT13, BGK10, BG14, KT13]. It has been shown to be very efficient to tackle large datasets [Gra10, HK09, BGK10] since it can be viewed as a "specialization of Tucker format" for large number of dimensions. As seen in the tensor format section (2.2), efficient strategies have been developed to convert other formats into $\mathcal{H}T$ (see [Hac14]) as well as truncation (leaf to root and root to leaf) and orthonormalization strategies proposed by Grasedyck [Gra10]. These algorithms have already been implemented in publicly available libraries including D. Kressner and C. Tobler `htucker` MATLAB library [KT13]. It has also been shown that $\mathcal{H}T$ decomposition is very well suited for sampling algorithms, one such example is the Black Box algorithm proposed by Ballany, Grasedyck and Kluge in [BGK10]. We will see in the numerics chapter (4) that it was chosen not to use MATLAB for implementing these methods. As a consequence of that choice and the elements exposed above, it was chosen not to study further hierarchical decomposition in this manuscript, for additional information, the reader can refer the above references.

Conclusion

In this chapter, the mathematical background of tensors and their decomposition was presented. Specifically, general notion on tensor spaces and tensor calculus have been extracted from the literature in order to introduce tensor formats. For the sake of simplicity, this chapter is limited to the case of d-way array but most of the results are applicable in general (see [Hac14]). These formats (canonical, Tucker, tensor train and hierarchical tensor) present a "nested" structure with specific properties and interest regarding storage as summarized in table 2.1 at the end of section 2.2. Tucker format contains the canonical format while proposing a better structure for approximation. Indeed, CP decomposition computes sequentially suboptimal approximation while tucker decompositions (T-HOSVD and ST-HOSVD) provide quasi-optimal decompositions. Finally, section 2.3 proposes a simple algorithm for large number of dimensions with the TT decomposition method. Each of these approximation technique was given with detailed algorithm that will be tested in chapter 4. In the next chapter, in order to integrate properly data from CFD simulation, the decomposition of multivariate functions (possibly vectorial) is studied.

Chapter 3

Multivariate problem decomposition

Contents

3.1 Proper Generalized Decomposition 78

3.1.1 Theoretical background of the PGD 78

3.1.2 A Galerkin PGD algorithm for d parameter functions 80

3.1.3 PGD and CPD 82

3.2 The Recursive-POD (R-POD) 82

3.2.1 Introductory example : R-POD on a 3D field 82

3.2.2 R-POD: general case 84

3.3 Functional tensor decomposition 89

3.3.1 Functional Tucker decomposition 89

3.3.2 Functional-TT 91

In this chapter, the multivariate data decomposition problem is addressed i.e. the focus is given on methods that apply in the continuous/functional framework. It is actually the continuation of previous chapter. Indeed, as already d-way array decomposition can easily be used on data corresponding to a function evaluated at grid points. With some manipulation such as flattening, vectorial functions can also be treated. However such techniques do not offer correlation among different elements of the output field. For instance, there is no control on how two elements of the velocity field in CFD applications are correlated. This is why the introduction of scalar product more subtle than vector dot product is desirable. Continuous methods allow such control on the scalar product as discussed for POD and PGD in chapter 1. Additionally, it allows to take advantage of non-uniform meshing (few points in regular regions of the studied domain) while attributing proportionate weight to each evaluation.

Consequently in this chapter we will focus on two families of multivariate decomposition methods. The first one has been specifically developed in the context of multivariate functions and then discretized while the second family is a mere conversion to the continuous framework of d-way array tensors. The first section will present the multiparameter version of PGD which was already discussed in the bivariate chapter 1. Then the Recursive POD (R-POD) [ABR16], an extension of the POD to multivariate functions is described. In the third section, the continuous equivalent of the decomposition methods presented in chapter 2 are studied.

3.1 Proper Generalized Decomposition

In this section we present the PGD for d parameter functions both for *a priori* and *a posteriori* model reduction. The first section present the theoretical justification of this class of methods. The second section focuses on the algorithm proposed by Chinesta which is the only PGD implemented so far. Finally, in subsection 3.1.3, a brief overview of the link with the CP decomposition is proposed and some conclusion about this kind of methods are drawn.

3.1.1 Theoretical background of the PGD

The PGD seen as an error minimization algorithm. The general setting of weak formulation in an Hilbert space is used in this demonstration as it is in most presentation of the PGD.

On V an Hilbert space, we define the following abstract formulation

$$u \in V,\ \mathcal{A}(u,v) = \mathcal{L}(v) \quad \forall v \in V \tag{3.1.1}$$

Where $\mathcal{A}$ is a bilinear form on V and $\mathcal{L}$ is a linear form on V. $V = V_1 \otimes \cdots \otimes V_d$ is a tensor product of Hilbert spaces provided with a scalar product and its associated norm.

$\mathcal{S}_1$ the set of rank-one tensors is introduced

$$\mathcal{S}_1 = \{z = w^1 \otimes \cdots \otimes w^d\ ;\ w^k \in V_k,\ k \in \{1, ..., d\}\} \tag{3.1.2}$$

as well as $\mathcal{S}_m$ the set of rank-m tensors

$$\mathcal{S}_m = \{v = \sum_{i=1}^{m} z_i\ ;\ z_i \in \mathcal{S}_1,\ i \in \{1, ..., m\}\} \tag{3.1.3}$$

The naive problem of finding an optimal representation $u_m \in \mathcal{S}_m$ of a given element $u \in V$ is not trivial and has been extensively studied. As stated in section 2.3.1, the problem is even ill posed for $d \geq 3$. Then one must add suitable constraints like orthogonality or boundedness to define a suitable optimization problem on $\mathcal{S}_m$. In the context of PGD, the orthogonality is chosen in addition to normalizing all modes save one dimension.

For *a posteriori* processing, we have

$$\mathcal{A}(u,v) = \int_\Omega uv\ d\mu \tag{3.1.4}$$

$$\mathcal{L}(v) = \int_\Omega fv\ d\mu \tag{3.1.5}$$

Introducing these notations might seem cumbersome, however it ease a lot the use of more complex functionals as long as they verify the same properties. Now a short version of the rigorous analysis of the progressive PGD proposed by Falcó in [FN12, FHMM13]. In the following all the assumed properties are easily verified for $\mathcal{A}$ the scalar product operator and $\mathcal{L}$ a scalar product against f as defined in equations (3.1.4) and (3.1.5).

It is assumed that $\mathcal{A}$ is bounded and coercive. Then equation 3.1.1 is project on V_N an N-dimensional subspace of V which the classical way of Galerkin methods.

$$u \in V_N,\ \mathcal{A}(u,v) = \mathcal{L}(v) \quad \forall v \in V_N \tag{3.1.6}$$

Thanks to Riesz representation theorem, $A : V \longrightarrow V$ the operator associated to $\mathcal{A}$ is introduced

$$\mathcal{A}(u, v) = \langle Au, v\rangle \tag{3.1.7}$$

and $f \in V$ associated with $\mathcal{L}$

$$\mathcal{L}(v) = \langle f, v\rangle \tag{3.1.8}$$

Then the problem (3.1.6) can be rewritten in an operator form

$$Au = f \tag{3.1.9}$$

It is further assumed that $\forall v \in V, \exists c > 0$ such that $\|Av\| \geq C\|v\|$. From the properties of A and its adjoint $A^\star$, $AA^\star$ is a self adjoint continuous and V-elliptic operator. Consequently it defines an inner product on V denoted $\langle \cdot, \cdot\rangle_{AA^\star} = \langle A\cdot, A\cdot\rangle$ whose associated norm is equivalent to the $\|\cdot\|$ norm. Then formulation 3.1.6 is equivalent to the following minimal residual formulation

$$u_n = \arg\min_{v\in V_n} \|f - Av\| = \arg\min_{v\in V_n} \|A^{-1}f - v\| \tag{3.1.10}$$

If one chooses $V_N = \mathcal{S}_N$ then for $A = I$ the PGD solves the same problem as the truncated CP decomposition provided by an ALS algorithm. Now, a convergent PGD algorithm is provided. Moreover it coincides with the PGD definition given by Chinesta et al. [CKL13].

Remark. The Galerkin problem can be solved on several basis which means that PGD is available on Hilbert tensor spaces in format that mimic any tensor reduction technique. For example Falcó demonstrates the convergence of PGD on a basis similar to the HOSVD in [FHMM13]. Thus one can conclude that PGD for *a posteriori* processing is the continuous version of well established tensor low rank approximation. Even though a wide variety of integration technique and PGD algorithms are available, it seems that the vast literature investigating tensor reduction proves to be much more efficient at post-processing.
Additionally, the inverse observation can be made for solving PDEs on reduced basis using reduced tensor representation. These algorithms might benefit for the preexisting knowledge in *a priori* PGD.

On the convergence of the progressive PGD. In order to show the converge of this algorithm, a generalization of the Eckart-Young theorem has been provided by Falcó and Nouy in [FN12]. Since the general problem of a rank-k separated representation is ill posed [dSL08], they proposed a progressive algorithm that converges. It is based on successive rank-1 approximations which are known to be optimal thus the link with singular values.

Lemma 3.1.1. *Given that $\mathcal{S}_1$ is weakly closed for $\|\cdot\|$ then for each $z \in V, \exists v^* \in \mathcal{S}_1$ such that*

$$\|z - v^*\|^2 = \min_{v\in\mathcal{S}_1} \|z - v\|^2$$

Finding v^* in the previous equation is a map defined by

$$\begin{aligned} \Pi : \quad & z \in V \longrightarrow \Pi(z) \in \mathcal{S}_1 \\ & z \longmapsto \arg\min_{v\in\mathcal{S}_1} \|z - v\|^2 \end{aligned}$$

Definition 3.1.1 (Progressive separated representation of an element in V)**.** *For a Given $z \in V$, the sequence $\{z_n\}_{n\geq 0}$ with $z_n \in \mathcal{S}_n$ is defined as follow: $z_0 = 0$ and for $n \geq 1$,*

$$z_n = \sum_{i=1}^{n} z^{(i)} = \sum_{i=1}^{n} \sigma_i w^{(i)}, \quad z^{(i)} \in \Pi(z - z_{i-1}) \tag{3.1.11}$$

z_n is the rank-n progressive separated representation of z with respect to the norm $\|\cdot\|$.

Theorem 3.1.2 (Generalized Eckart-Young theorem according to Falcó and Nouy)**.** *For $z \in V$, the sequence $\{z_n\}_{n\geq 0}$ from definition 3.1.1 verifies*

$$z = \lim_{n\to\infty} z_n = \sum_{i=1}^{\infty} \sigma_i w^{(i)}$$

This proves the convergence of the PGD algorithm which is a succession of optimal progressive separated representation as defined in (3.1.1) with the projector associated to A.

Remark. As stated by Falcó and Nouy, this is the simplest definition of PGD, other definitions where provided in the literature which may display better convergence properties. One of them is the direct equivalent of the ALS algorithm [FHMM13].

3.1.2 A Galerkin PGD algorithm for d parameter functions according to Chinesta

In order to determine each element of the sequence an enrichment process is devised. Let $\Omega = \Omega_1 \times \cdots \times \Omega_d$ where each $\Omega_i \in \mathbb{R}$ and $f \in L^2(\Omega)$[1]. Then, the goal is to compute univariate basis functions $(X_i^k)_{k=1}^r$, $\forall\, 1 \leq i \leq d$ using a fixed point algorithm in alternating directions. The weak formulation of our problem reads

$$\forall u^\star \in H^1(\Omega), \quad \int_\Omega u^\star(u - f) = 0 \tag{3.1.12}$$

It is assumed that $u^{r-1} = \sum_{k=1}^{r-1} \prod_{i=1}^{D} X_i^k(x_i)$ is known thus u^r is sought under the form

$$u^r = u^{r-1} + \prod_{i=1}^{d} X_i^r(x_i) \tag{3.1.13}$$

The process of adding terms to the sum, i.e. computing the sequence (u^r) is called the *enrichment process*. This process ends when a stopping criterion is fulfilled. Since in the general case, one does not knows the exact solution, it is chosen to stop the process when the weight of the last term compared to the rest of the series becomes negligible. This reads

$$\mathcal{E}(r) = \frac{||\prod_{i=1}^{d} X_i^r||_{L^2(\Omega)}}{||\prod_{i=1}^{d} X_i^1||_{L^2(\Omega)}} = \frac{||X_d^r||_{L^2(\Omega)}}{||X_d^1||_{L^2(\Omega)}} \leq \varepsilon_{enrichment} \tag{3.1.14}$$

Indeed the terms are of decreasing norm, then there is no need to compare the whole series, the first term is sufficient. In addition to that, we define X_i such as $\forall i < D, \quad ||X_i||_{L^2(\Omega)} = 1$ all the information about the norm is enclosed in X_D.

[1] Here we assume without loss of generality that Ω_i is a subset of $\mathbb{R}$ but it could be any domain on which an integral can be defined. e.g. 2D or 3D domains.

Fixed point algorithm. This is an iterative algorithm that, in practice, usually converges in a few iterations. It is an alternated direction algorithm, i.e. each direction is computed one at a time.

Remark. From now on, r in X_i^r is omitted to simplify the writing at enrichment step r.

It is assumed that the fixed point series $\{\widehat{X}_i^k\}_k, \forall i < d$ is known after step k. Thus $u = u^{r-1} + \prod_{i=1}^{D} \widehat{X}_i^k$. Moreover, it is assumed that direction s is to be updated which means $\widehat{X}_i^{k+1}$ is already known $\forall i < s$.

The test function $u^\star$ is set to

$$u^\star = \prod_{i=1}^{s-1} \widehat{X}_i^{k+1}(x_i) X^\star(x_s) \prod_{i=s+1}^{d} \widehat{X}_i^k(x_i) \tag{3.1.15}$$

Given all previous equation, the following weak formulation stands

$$\int_\Omega \left[\prod_{i=1}^{s-1} \widehat{X}_i^{k+1} X^\star \prod_{i=s+1}^{d} \widehat{X}_i^k \left(u^{r-1} + \prod_{i=1}^{s} \widehat{X}_i^{k+1} \prod_{i=s+1}^{d} \widehat{X}_i^k - f \right) \right] = 0 \tag{3.1.16}$$

This equation can be written as follow

$$\alpha^s \int_{\Omega_s} X^\star(x_s) \widehat{X}_s^{k+1}(x_s) dx_s = - \int_{\Omega_s} X^\star(x_s) \sum_{j=1}^{p-1} \left(\beta^s(j) \widehat{X}_s^j \right) dx_s + \int_{\Omega_s} X^\star(x_s) \gamma^s(x_s) dx_s \tag{3.1.17}$$

where

$$\alpha^s = \int_{\Omega/\Omega_s} \prod_{i=1}^{s-1} (\widehat{X}_i^{k+1})^2 \prod_{i=s+1}^{d} (\widehat{X}_i^k)^2 = \prod_{i=1}^{s-1} \int_{\Omega_i} (\widehat{X}_i^{k+1})^2 \prod_{i=s+1}^{d} \int_{\Omega_i} (\widehat{X}_i^k)^2 \tag{3.1.18}$$

$$\beta^s(j) = \int_{\Omega/\Omega_s} \prod_{i=1}^{s-1} (\widehat{X}_i^{k+1} X_i^j) \prod_{i=s+1}^{d} (\widehat{X}_i^k X_i^j) = \prod_{i=1}^{s-1} \int_{\Omega_i} \widehat{X}_i^{k+1} X_i^j \prod_{i=s+1}^{d} \int_{\Omega_i} \widehat{X}_i^k X_i^j \tag{3.1.19}$$

$$\gamma^s(x_s) = \int_{\Omega/\Omega_s} \prod_{i=1}^{s-1} \widehat{X}_i^{k+1} \prod_{i=s+1}^{D} \widehat{X}_i^k f \tag{3.1.20}$$

Remark 1 The evaluation of α^s and $\beta^s(j)$ are relatively cheap in term of computing cost since they consist in a product of 1D integrals.

Remark 2 The evaluation of γ^s is much more costly since it cannot be reduced to 1D integrals but requires $n_s \times (d-1)$ dimension integrals (where n_s is the number of discrete points along direction s). In order to reduce the weight of computing γ^s, one should not use classical integration tools such as composite trapezoidal rules but seek in the direction of Monte-Carlo methods, Quasi Monte-Carlo methods or Sparse grid methods.

Finally the strong formulation stands

$$\widehat{X}_s^{k+1}(x_s) = \frac{-\sum_{j=1}^{p-1} \left(\beta^s(j) X_s^j(x_s) \right) + \gamma^s(x_s)}{\alpha^s} \quad , \forall x_s \in \Omega_s \tag{3.1.21}$$

All the $\widehat{X}_i$ are normalized i.e. $||\widehat{X}_i^{k+1}||_{L^2(\Omega_i)} = 1$ so that all the information relative to the norm is transfered to the last element X_d. This algorithm is performed for $s = 1, d$ and each time a family $(\widehat{X}_i^{k+1})_{1 \leq i \leq d}$ is complete the convergence stopping criterion is tested. It reads

$$\mathcal{E}_{fixed\,point}(k) = \frac{||X_d^{k+1} - X_d^k||_{L^2(\Omega_i)}}{||X_d^k||_{L^2(\Omega_i)}} < \varepsilon_{fixed\,point} \tag{3.1.22}$$

3.1.3 PGD and CPD

It clearly appears that the PGD falls in the domain of canonical representation format $\mathcal{C}_r$ for functional spaces. The same statement can be made for CP decomposition where the underlying space is $\mathbb{R}^d$. Consequently, for *a posteriori* processing of tensor data, these two decomposition techniques are freely interchangeable. Then any favorable property of one is applicable to the other. Unfortunately, this remains true for the downsides like the ill-posedness of a general best rank-r approximation. This approach has been shown to be poorly efficient compared to Tucker format methods but may represent a first step in an attempt to compute low rank approximations of tensors.

However, the main strength of the recursive techniques is that they are mostly cheap[2], easy to program and produces *a priori* reduced order bases. Indeed in many situations where high precision is not a goal or simply unrealistic but many parameters are used, PGD (or CP alternatives) in some of its formulation is a very interesting process that enables calculations that are simply out of reach for direct simulations.

Remark. There is a vast literature concerning PGD algorithm applied to (mainly elliptic) problems [CL14, FN11]. It turns out that different kind of PGD algorithm [FN12] work best on different kind of problems (Galerkin PGD, minimum residual PGD, Krylov PGD, Greedy Completely Orthogonal PGD ,etc.) . Then, there is no general PGD algorihtm however the one that was presented in the previous section seems to be robust though may require many iteration to converge.

3.2 The Recursive-POD (R-POD)

The Recursive POD [ABR16] is an extension of the usual bivariate POD, it fulfills quasi-optimality in higher dimension. The essence of this method is to perform successive (recursively) POD on the field that is to be tensorized. A field function $f : \mathcal{D} \subset \mathbb{R}^d \longrightarrow \mathbb{R}^q$ i.e. a function of d variables is first processed as a $(1, d-1)$ field that can be separated thanks to POD. Once the first POD has been performed, one obtains the POD modes $\boldsymbol{X}_1^r : \Omega_1 \longrightarrow \mathbb{R}^q$ basis functions of Ω_1 and $\boldsymbol{\phi}_1^r : \mathcal{D}/\Omega_1 \longrightarrow \mathbb{R}^q$, a set of functions of $d-1$ variables. The the same POD process is performed again on each POD mode recursively until the POD modes are univariate functions.

Remark. It should be noted that the RPOD is one the many extensions to multiple variables that overcome the bivariate nature of POD. Consequently every conclusion concerning POD remain true except optimality properties. For short, it means that any algorithm available to compute a POD i.e. POD, PGD and to some extent direct SVD may be used to compute the recursive POD. All algorithmic properties are preserved and method choices should align with 2D experiment conclusion.

3.2.1 Introductory example : R-POD on a 3D field

Let $f : \mathcal{D} = \Omega_1 \times \Omega_2 \times \Omega_3 \subset \mathbb{R}^3 \longrightarrow \mathbb{R}$ a Lebesgue square integrable function and $w = (y, z) \in \mathbb{R}^2$. Since $L^2(\mathcal{D})$ and $L^2(\Omega_1, L^2(\mathcal{D}/\Omega_1))$ are isometric, the POD of $f(x, w)$ is well defined and reads

$$f(x, y, z) = f(x, w) \approx f_{POD}^M(x, w) = \sum_{m=1}^{M} X_m(x)\phi_m(w) \tag{3.2.1}$$

[2] As long a one only requires a small number of modes as compared to the full representation, PGD can be efficient since it computes only the required information.

As for the 2D POD it is handy to normalize all modes and store their relative weight into $(\boldsymbol{\sigma}) \in \mathbb{R}^M$. Then the POD of f reads

$$f(x,y,z) = f(x,w) \approx f^M_{POD}(x,w) = \sum_{m=1}^{M} \sigma_m X_m(x)\phi_m(w) \tag{3.2.2}$$

with $X_m = X_m/\|X_m\|$, $\phi_m = \phi_m/\|\phi_m\|$ and $\sigma_m = \langle f, X_m\phi_m\rangle$.

It is now necessary to separate each 2D field ϕ_m obtained during the first step i.e.

$$\forall 1 \leq m \leq M, \quad \phi_m(w) = \phi_m(y,z) \approx \phi_{m,K(m)}(y,z) = \sum_{k=1}^{K(m)} \tilde{\sigma}^m_k \; Y^m_k(y) Z^m_k(z) \tag{3.2.3}$$

Then, these two results are combined into one tensorisation of field f,

$$f(x,y,z) \approx f_M(x,y,z) = \sum_{m=1}^{M} \sum_{k=1}^{K(m)} \sigma_m \tilde{\sigma}^m_k X_m(x) Y^m_k(y) Z^m_k(z) \tag{3.2.4}$$

Remark ($K(m)$). As each POD on level $\phi_m(y,z)$ is performed independently, if the number of dominant POD mode is dependent of an error estimator, then $K(m)$ may change with m. Then a *R-POD rank* is defined as the number of modes at each recursion level. An illustration of the spread of $K(m)$ is provided by figure 3.1. It can be seen that in this matrix representation that some $\tilde{\sigma}^m_k$ are missing. They correspond to the unneeded/uncomputed modes. For practical reason, this representation of $\tilde{\Sigma} = (\tilde{\sigma}^m_k)_{km}$ may be useful to compare RPOD with ST-HOSVD and also facilitates the implementation setting discarded modes as constant functions with a nil weight.

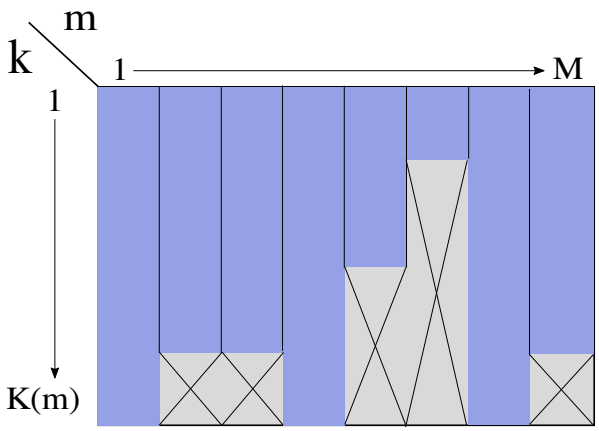


Figure 3.1: Visual RPOD rank for 3 parameter function. Blue columns correspond to the coordinates (m,k) where $\tilde{\sigma}^m_k$ is defined while gray crossed areas correspond to coordinates where $\tilde{\sigma}^m_k$ is not defined (not computed).

Obviously, this sum of sum can be reordered and written as one single sum. In this work, the following bijective numbering function is used

$$\begin{aligned} h: \quad \mathbb{N}^2 &\longmapsto \mathbb{N} \\ (m,k) &\longmapsto l = k + \sum_{i=1}^{m-1} K(i) \end{aligned}$$

Then a new weight list is defined as $\sigma_{l=h(m,k)} = \sigma_m \tilde{\sigma}_k^m$. Finally the R-POD approximation of f reads

$$f_L(x,y,z) = \sum_{l=1}^{L} \sigma_l \; X_l(x) Y_l(y) Z_l(z) \tag{3.2.5}$$

Generalization of this example is straightforward, however, notations quickly become cumbersome for higher dimension.

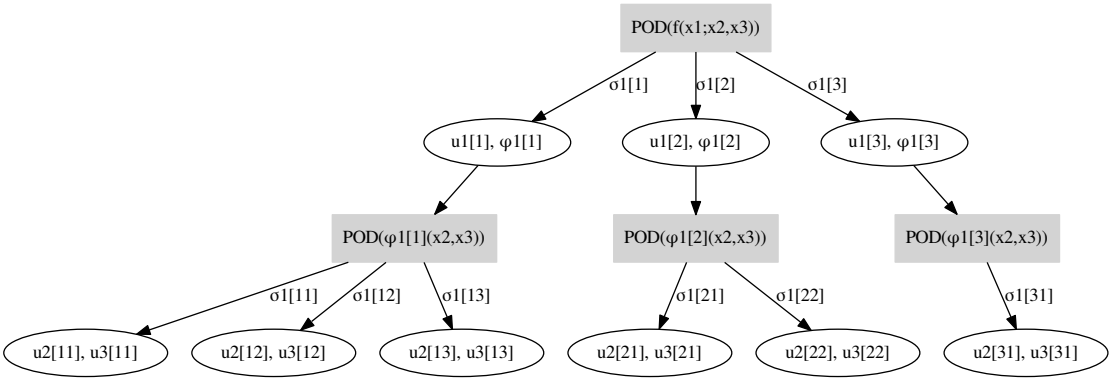


Figure 3.2: Example of *a Recursive POD graph* of $f(x_1, x_2, x_3)$

Another approach is to represent the *recursion graph* or *decomposition graph* as shown in Fig. 3.2. In this case, there is no need to introduce a renumbering, all the information is contained in the graph. Notations and programming remain simple as each decomposition (as well as reconstruction) is performed independently, each node of the tree only "knows" its children. This approach is very natural from a mathematical point of view however it is uncommon in computational mechanics.

3.2.2 R-POD: general case

The R-POD is now presented for a field of d variables. Since this algorithm is recursive, the processing order $\boldsymbol{p} = (p_1, ..., p_d)$ of the variables may influence the accuracy and the number of modes. However, $\boldsymbol{p} = [1, ...d]$ is used in most of this presentation to lighten notations. The orthonormal modes version of the POD is the only one used in this section even though it is not necessary to impose orthonormality.

Prior to the definition of RPOD, as for the other data reduction method, a rank definition for RPOD approximation is needed. In fact, the different forms that can take the RPOD representation leads to the introduction of three RPOD ranks.

Definition 3.2.1 (RPOD ranks)**. Scalar-RPOD-rank** *R is the RPOD scalar rank, it is defined as the total terms in sum from equation 3.2.12. If R is given alone, it gives no information concerning how modes are distributed among dimensions. Then it is not unique. However it is sufficient if reordered sum is the goal.*

$$R = \sum_{r_1}^{R_1} \cdots \sum_{r_{d-1}}^{R_{d-1}(r_1,...,r_{d-2})} R_{d-1}(r_1, ..., r_{d-2}) \tag{3.2.6}$$

Exact-RPOD-rank $\mathcal{R}$ *is the exact RPOD rank, it describe exactly how the recursive distribution of modes is laid out. It is a recursively defined vector of sequences that reads*

$$\mathcal{R} = \left(R_1, (R_2(r_1))_{r_1=1}^{R_1}, \cdots, \left(\left((R_{d-1}(r_1, r_2, ..., r_{d-2}))_{r_{d-2}=1}^{R_{d-2}(r_1,...,r_d-2)} \cdots \right)_{r_1=1}^{R_2(r_1)} \right)_{r_1=1}^{R_1} \right) \tag{3.2.7}$$

$\mathcal{R}$ *may be written as a* $(d-1) \times \left(\prod_{r_1,...,r_{d-2}} R_{d-1}(m_1, ..., r_{d-2}) \right)$ *matrix* S *that stores the allowed tuples* $\boldsymbol{r}$ *with* $r_{d-1} \leq R_{d-1}(r_1, ..., r_{d-2})$..

Multilinear-rank *Let* $\boldsymbol{R} \in \mathbb{N}^{d-1}$ *a* "cubic RPOD rank"*. It is a vector that stores the largest value of* $R_i(r_1, ..., r_{i-1})$ *for all* $1 \leq i \leq d-1$*. Then any* $\sigma_{\boldsymbol{r}}$ *can be stored in a tensor of multilinear rank that is equal to the cubic-RPOD-rank. Since it aligns with the ST-HOSVD multilinear approximation rank –provided that the last index is duplicated to account for the last dimension– it is called the same. It is then defined as follow*

$$\boldsymbol{R} = (R_1, R_2, \cdots, R_{d-1}) \tag{3.2.8}$$

where $R_i = \max_{\bar{\boldsymbol{r}} \in \mathcal{R}(1:i-1)} R_i(\bar{\boldsymbol{r}})$*. It should be noted that this definition provides somewhat redundant information since* $R_{d-1} = R_d$*, however it is interesting to stick to the shape of the multilinear rank. The* i partial multilinear rank *is defined as* $\boldsymbol{R_i} = (R_1, R_2, \cdots, R_i)$*.*

From definition 3.2.1 one can build a tensor representation of the rank of the exact-RPOD-rank. It enables easy mapping/representation of the defined truncated orthogonal vectors and their associated weights. Then it is called sigma map tensor and defined as follows.

Definition 3.2.2 (Sigma map tensor)**.** *Let* f *a function of* d *parameters and* $f_{\mathcal{M}}$*, its RPOD approximation of exact rank* $\mathcal{M}$*. Let* $\boldsymbol{M} = (M_1, M_2, \cdots, M_d)$ *be the associated multilinear rank defined in 3.2.1. The* sigma map tensor $\boldsymbol{S} \in \mathbb{R}^{M_1 \times \cdots \times M_d - 1}$ *stores the values of defined elements of the RPOD approximation and zeros elsewhere which read element wise*

$$s_{i_1,...,i_{d-1}} = \begin{cases} \sigma_{i_1} \sigma_{i_1,i_2} ... \sigma_{i_1,i_2,...,i_{d-1}} & \text{if } (i_1, i_2, ..., i_{d-1}, i_{d-1}) \in \mathcal{M} \\ 0 & \text{else} \end{cases} \tag{3.2.9}$$

The last dimension describes both $d-1$ *and* d *parameter since they share the same singular values.*

Figure 3.3 display the order 3 sigma map tensor associated with the RPOD of a 4 parameter field. It can be seen that the third axis represent both parameter 3 and 4 associated singular values. One can see that many σ_I are left empty, it corresponds to the non defined vectors of the basis. Indeed, the RPOD allows to define a different truncation rank for each sub POD leading to optimal size of the approximation for a given target approximation error. However, using a tensor full representation is handy for programming and analysis purpose. On this figure, the exact RPOD rank can be seen a the series storing the maximum coordinate of each blue bar. The scalar rank is the sum of the blue bars length and the multilinear rank is the hypercube/tensor size.

Definition 3.2.3 (R-POD)**.** *Let* $f : \mathcal{D} = \Omega_1 \times \cdots \times \Omega_d \subset \mathbb{R}^d \longrightarrow \mathbb{R}^q \in L^2(\mathcal{D})$ *where* q *and* d *are positive integers. Let the tuple* $\boldsymbol{x} = (x_1, ..., x_d) \in \mathcal{D}$ *be the general notation for the variables of* f*. Let* $\boldsymbol{p} = [1, ...d]$ *be the processing order.*

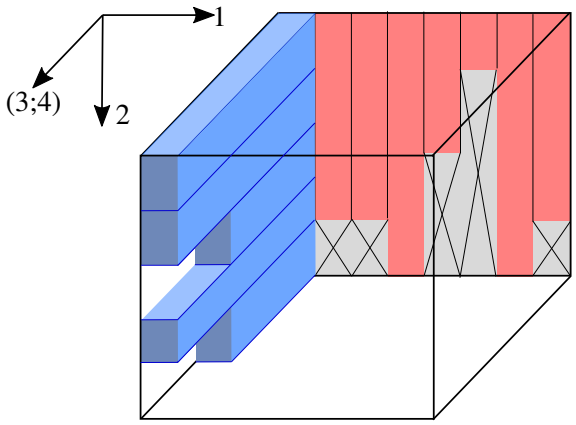


Figure 3.3: Visual RPOD rank for 4 parameter function. Red columns correspond to the coordinates (m_1, m_2) where σ_{m_1,m_2} are defined while gray crossed areas correspond to coordinates where they are not defined (not computed) i.e. arbitrarily set to 0. The third dimension (Blue bars) represent defined $\sigma_{m_1=1,m_2,m_3}$, $m_1 \geq 2$ was omitted to simplify the illustration however the same could have been drawn.

Since $L^2(\mathcal{D})$ and $L^2(\Omega_1, L^2(\mathcal{D}/\Omega_1))$ are isometric, the POD of $f(x_1, w)$, $w = (x_2, ..., x_d)$ is well defined. It yields a series of univariate functions $X_1^{r_1}(x_1)$, singular values $\sigma_1^r \in \mathbb{R}$, and multivariate functions $\phi^{r_1}(x_2, ..., x_d)$.

$$\text{POD}(f(x_1; x_2, ..., x_d)) \longrightarrow \tilde{f}(x_1; x_2, ..., x_d)) = \sum_{r_1=1}^{R_1} \sigma_{r_1} X_1^{r_1}(x_1)\phi^{r_1}(x_2, ..., x_d) \tag{3.2.10}$$

The same process is applied recursively on each ϕ^{r_1}

$$\forall r_1 \leq R_1, \quad \text{POD}(\phi^{r_1}(x_2; x_3, ..., x_d)) \longrightarrow \tilde{\phi}^{r_1}(x_2; x_3, ..., x_d)) = \sum_{r_2=1}^{R_2(r_1)} \sigma_{r_1} X_2^{r_1 r_2}(x_2)\phi^{r_1 r_2}(x_3, ..., x_d) \tag{3.2.11}$$

And so on recursively until all POD modes up to $\left(X_{d-1}^{(r_1,...,r_{d-2})}\right)$ and $\left(X_d^{(r_1,...,r_{d-2})}\right)$ have been computed. Each univariate POD mode $X_i^{(r_1,...,r_{i-1})}$ is normalized with respect to the scalar product associated norm.

Then, $f_{\mathcal{R}}$ the R-POD approximation of f of exact RPOD rank $\mathcal{R}$ reads

$$\begin{aligned} f(x_1, ..., x_d) &\approx f_{\mathcal{R}}(x_1, ..., x_d) \\ &= \sum_{r_1}^{R_1} \cdots \sum_{r_i}^{R_i(r_1,...,r_{i-1})} \cdots \sum_{r_{d-1}}^{R_{d-1}(r_1,...,r_{d-2})} \sigma_{r_1...r_{d-1}} X_1^{r_1} \cdots X_i^{(r_1,...,r_i)} \cdots X_d^{(r_1,...,r_{d-1})} \\ &= \sum_{r_1}^{R_1} \cdots \sum_{r_{d-1}}^{R_{d-1}(r_1,...,r_{d-2})} \sigma_{r_1...r_{d-1}} \prod_{i=1}^{d} X_i^{(r_1,...,r_i)}(x_i) \end{aligned} \tag{3.2.12}$$

L a global index is defined to write the RPOD as a single sum. Let a vector of coordinates $\boldsymbol{r} = (r_1, ..., r_{d-1}) \in \mathbb{N}^d$ that corresponds to the indices of a defined $\sigma_{\boldsymbol{r}}$ of our problem.

It is given by the linear map h.

$$r = h(\boldsymbol{r}) = r_{d-1} + \sum_{\substack{i_1 \le r_1 \\ i_2 \le r_2 \\ \vdots \\ i_{d-2} \le r_{d-2}}} R_d(r_1, ..., r_{d-2}) \tag{3.2.13}$$

Now the RPOD written as a single sum reads

$$f(x_1, ..., x_d) \approx f_R(x_1, ..., x_d) = \sum_{r}^{R} \sigma_r \prod_{i=1}^{d} X_i^r(x_i) \tag{3.2.14}$$

Let $\boldsymbol{\mathcal{S}} \in \boldsymbol{R}$ the sigma map tensor associated with this RPOD approximation. Then the cubic representation of the RPOD is defined by

$$f(x_1, ..., x_d) \approx f_{\boldsymbol{R}}(x_1, ..., x_d) = \sum_{r_1}^{R_1} \cdots \sum_{r_{d-1}}^{R_{d-1}} s_{r_1,...,r_{d-1}} X_1^{(r_1,...,r_d)}(x_1) \otimes \cdots \otimes X_d^{(r_1,...,r_d)}(x_d) \tag{3.2.15}$$

where $X_i^{(r_1,...,r_d)} = X_i^{(r_1,...,r_{i-1})}$ if $s_{i_1,...,i_{d-1}} \neq 0$ else $X_i^{(r_1,...,r_d)}$ is a non nil constant valued function.

Lemma 3.2.1 (RPOD expansion [ACM+15, Lemma2.2])**.** *The RPOD expansion defined in 3.2.3 converges toward f when $R_i \to \infty$. for $i \le d$.*

As for POD, no precision was given concerning which scalar product is used to define RPOD. The same comment concerning the choice of POD scalar product and ordering of the dimensions remain true. Instead of snapshot versus classical method, one has to choose a processing order $\boldsymbol{p}$ that minimizes the matrix size a each step. As was shown for ST-HOSVD, this will also improve speed at each recursion.

Quasi-optimality As for ST-HOSVD and POD, RPOD displays a quasi-optimality property.

RPOD Algorithm. An algorithm is devised from the previous properties. In order to reduce operation count and the number of mode, an error estimate truncation is chosen. Although, sequential this algorithm can be written easily in parallel version since many independent problem arise at each deeper level of recursion. Only the first POD is inevitably sequential, every subsequent POD of modes can be performed on a different CPU. Then choosing the smallest dimension to be separated first is crucial. Assuming the data/variable have been ordered according to p before entering the RPOD algorithm, a recursive version of the algorithm 11 is proposed. It yields the required singular values and RPOD modes to build the compute the first version of RPOD (def. 3.2.3). Then it can be evaluated directly.

RPOD tree structure

The recursive nature of the algorithm is well fitted for implementation in tree structures. As introduced in section 2.2.4, trees are simple mathematical structures that describe links between nodes linked by edges. A tree starts by a root and ends with leaves. As it is simply tool for describing RPOD, only a brief description of the structure that I implemented is given. I chose to use a top down tree which means that a node knows its children but not its parent. Indeed the process of RPOD is top-down as well as the reconstruction and all required operation can be obtain by going through the tree. Also a node doesn't know its place in the tree.

Root Top level approximation i.e. $\tilde{f}(x_1; x_2, ..., x_d)$

Node Stores X_k^i and a POD has been applied on the multivariate function ϕ_k. Each pair of these POD mode is (X_k^{i+1}/ϕ_k) is stored in a different child node.

Leaf When a node is a bivariate function its children are leaves that contain the pair of modes (X_k^{d-1}, X_k^d).

For practical reasons it may be interesting to store the singular values as well as the the modes as shown in Fig. 3.2. This leads to the notion of branch weight which is computed as $\sigma_{\text{loc}}/\sum \sigma_i^1$. This is a very efficient tool for truncation of the RPOD tree as branches can be cut off from the top without exploring the whole tree. A typical tree for an separable function of 4 parameter is shown in Fig. 3.4 as an example.

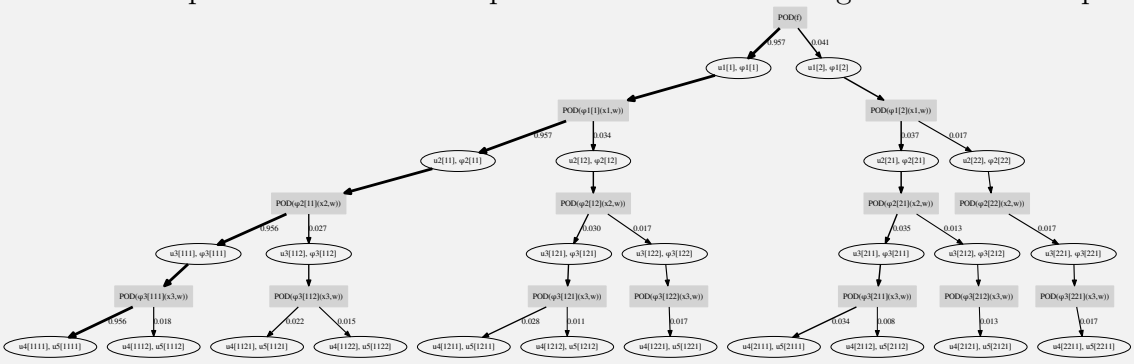


Figure 3.4: RPOD graph obtained for $f(x_1, x_2, x_3, x_4) = \sin(\sqrt{\sum_i x_i^2})$ with a POD cutoff value of 5×10^{-3}. The width of the edges represents their weight. This figure is Zoomable on PDF version.

Operation count In order to ease comparison with its most similar method, we suppose that each local POD is solved through the same truncate SVD algorithm that is used in the ST-HOSVD though it might not be the best choice for accuracy of computing efficiency. Then the SVD of a $n \times m$ matrix operation count is $\mathcal{O}(m^2 n)$.

The sum of the last column of table yields the following estimate if the samples number is identical for all variables as sigma map tensor is a full hypercube.

$$\mathcal{O}\left(\sum_{i=1}^{d-1} R^{i-1} n^{d-i+2}\right) \tag{3.2.16}$$

One can see that this is exactly the same term as the first term in ST-HOSVD operation count evaluation. The second one is not necessary since the RPOD algorithm does not

Algorithm 11: RPOD

input : $f \in L^2(\mathcal{D})$, computing domain $\mathcal{D}$, target error ε
output: rpod_tree=$[[\mathcal{R}, \mathcal{S}, \mathcal{X}]]$

$\mathcal{R} = [\,]$; /* List containing the exact RPOD rank */
$\mathcal{S} = [\,]$; /* List containing the local singular values */
$\mathcal{X} = [\,]$; /* List containing the local eigen functions */
$\phi(x, \boldsymbol{w}) = f(x_1, (x_2, ..., x_d))$;
$[R, \boldsymbol{\sigma}_R, \boldsymbol{U}_R(x), \boldsymbol{V}_R(\boldsymbol{w})] = \text{trunc_POD}(\phi, \varepsilon)$;
$\mathcal{R}$.append(R) ;
$\mathcal{S}$.append$(\boldsymbol{\sigma}_R)$;
$\mathcal{X}$.append$(\boldsymbol{U}_R)$, ;
if $dim(w) > 2$ **then**
  for $m \leq R$ **do**
    $\phi(x, \boldsymbol{s}) = V_r(\boldsymbol{w})$;
    $(\mathcal{R}_{loc}, \mathcal{S}_{loc}, \mathcal{X}_{loc})$.append(RPOD$(\phi, \mathcal{D}/\Omega_1, \varepsilon)$) ;
  $(\mathcal{R}, \mathcal{S}, \mathcal{X})$.append$(\mathcal{R}_{loc}, \mathcal{S}_{loc}, \mathcal{X}_{loc})$;
else
  $\mathcal{X}$.append$(\boldsymbol{V}_R)$; /* Last dimension, then keep $\boldsymbol{V}_R$ as RPOD modes */
return $f_{\mathcal{R}} = [[\mathcal{R}, \mathcal{S}, \mathcal{X}]]$

Level	Operations	Count	Hypercube Cost
1	$1\times$ POD$[n_1 \times (n_2...n_d)]$	$\mathcal{O}(n_1^2(n_2...n_d))$	$\mathcal{O}(n^d + 1)$
2	$M_1\times$ POD$[(n_2 \times (n_3...n_d)]$	$M_1\mathcal{O}(n_2^2(n_3...n_d))$	$\mathcal{O}(Mn^d)$
3	$\sum_{m_1 \leq M_1} M_2(m_1)\times$ POD$[(n_3 \times (n_4...n_d)]$	$M_1M_2\mathcal{O}(n_3^2(n_4...n_d))$	$\mathcal{O}(M^2n^{d-1})$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
d-1	$\sum \cdots \sum M_{d-2}(m_1, ..., m_d)\times$ POD$[(n_{d-1} \times (n_d)]$	$M_1 \cdots M_d - 2\mathcal{O}(n_{d-1}^2(n_d))$	$\mathcal{O}(M^{d-2}n^3)$

Table 3.1: Operation count at each step of the RPOD algorithm.

requires to compute an intermediate function/tensor. Additionally, this results does not account sum length number of modes variation within each dimension. This was shown with the blue bars length in figure 3.3.

3.3 Functional tensor decomposition

In section 3.1.3, is was shown that the PGD produces a canonical format approximation for functions that is equivalent to CP. Here, a brief review of the application of the other tensor decomposition methods presented in chapter 2 to functions is proposed.

Let $(\cdot, \cdot)$ be a scalar product and it associated norm $||\cdot||$. We are interested in separating function $f : \Omega \in \mathbb{R}^d \longrightarrow \mathbb{R}^p$ with $||f|| < \infty$.

3.3.1 Functional Tucker decomposition

The goal here is to obtain a separated approximation of f in the continuous version of the Tucker format i.e.

$$f(x_1, ...x_d) \approx \tilde{f}(x_1, ..., x_d) = \sum_{i_1}^{r_1} \cdots \sum_{i_d}^{r_d} w_{r_1,...,r_{d-1}} X_1^{i_1}(x_1) \otimes \cdots \otimes X_d^{i_d}(x_d) \tag{3.3.1}$$

where $\mathcal{W} \in \mathbb{R}^{r_1 \times \cdots \times r_d}$ which elements are defined as the projection of f onto the orthonormal basis $\{X_1^{i_1} \cdots, X_d^{i_d}\}_{i_1,\ldots,i_d}$ i.e. $w_{i_1 \ldots i_d} = \left(f, X_1^{i_1} \cdots X_d^{i_d}\right)$.

T-HOPOD. Then a direct adaptation of T-HOSVD (see section 2.3.2.1) is done by switching[3] from SVD to POD method to obtain each decomposition. Then the algorithm is to perform POD for each variable on f e.g. for $d = 3$ the T-HOPOD reads,

- POD(f,1) $\longrightarrow f(x_1; x_2, x_3) \approx \sum_{i=1}^{r_1} X_1^i(x_1)\phi_1^i(x_2, x_3)$
- POD(f,2) $\longrightarrow f(x_2; x_1, x_3) \approx \sum_{i=1}^{r_2} X_2^i(x_2)\phi_2^i(x_1, x_3)$
- POD(f,3) $\longrightarrow f(x_3; x_1, x_1) \approx \sum_{i=1}^{r_3} X_3^i(x_3)\phi_3^i(x_1, x_2)$
- $\forall i, j, k < r_1, r_2, r_3$ compute weights $w_{ijk} = \left(f, X_1^i X_2^j X_3^k\right)$

ST-HOPOD is more complex. The natural option is to go back to the sequential truncation idea of the ST-HOSVD. In this case one would reduce the actual rank of the function by truncating the POD decomposition and working on the approximation for the next decomposition.

- POD(f,1) $\longrightarrow f(x_1; x_2, x_3) \approx \tilde{f}_1(x_1; x_2, x_3) = \sum_{i=1}^{r_1} X_1^i(x_1)\phi_1^i(x_2, x_3)$
- POD($\tilde{f}_1$,2) $\longrightarrow \tilde{f}_1(x_2; x_1, x_3) \approx \tilde{f}_2(x_1, x_2, x_3) = \sum_{i=1}^{r_2} X_2^i(x_2)\phi_2^i(x_1, x_3)$
- POD($\tilde{f}_2$,3) $\longrightarrow \tilde{f}_2(x_3; x_1, x_1) \approx \tilde{f}_3(x_1, x_2, x_3) = \sum_{i=1}^{r_3} X_3^i(x_3)\phi_3^i(x_1, x_2)$
- $\forall i, j, k < r_1, r_2, r_3$ compute weights $w_{ijk} = \left(f, X_1^i X_2^j X_3^k\right)$

This process is formally equivalent to the ST-HOSVD for continuous functions however it is as inefficient as T-HOPOD while reducing the accuracy. In spite of our effort, there is no natural way to replace the integer index presented in the next paragraph. All of the other rewrittings attempted of this algorithm (during my thesis) that would preserve the speed and spirits of ST-HOSVD leads to a different problem with very poor convergence properties.

Introducing integer parameters. In order to provide a decent ST-HOPOD algorithm, one needs to tackle the phase of reshaping the tensor into a "partially truncated core tensor" of eq. (2.3.16) or at line 4 of 9. This is not at all a natural operation in the continuous framework. Indeed, if one tries to convert this operation into the continuous framework after the first dimension was separated, they obtain the following mixed function:

$$\begin{aligned} \mathcal{W}_{(1)} = &\quad \boldsymbol{\Sigma}_{tr} \boldsymbol{V}_{tr}^{\mathsf{T}} \\ w(\alpha_1, x_2, ..., x_d) = &\quad \sigma(\alpha_1)\phi_1(\alpha_1, x_2, ..., x_d), \quad \forall \alpha_1 < r_1 \end{aligned} \tag{3.3.2}$$

where α_1 is an integer parameter.

It is then fairly easy to adapt algorithm 9 to a "mixed POD" solver. To do so, one needs to introduce a discrete scalar product for α_1 e.g. instead of using a L^2 scalar product, one can replace it by a sum operator which is equivalent to setting the integration matrix to the identity. This process is presented in algorithm 12 for which we consider that the

[3]The equivalence and switching process has been discussed in section 1.2.3. Consequently all previous remarks on these methods apply for the continous T-HOPOD as well as for T-HOSVD.

scalar product $(\cdot,\cdot)$ is discretized for each dimension by $(u,v) \xrightarrow{disc} (\boldsymbol{u};\boldsymbol{v})_{\boldsymbol{M}_i} = \boldsymbol{u}^\intercal \boldsymbol{M}_i \boldsymbol{v}$ where $\boldsymbol{M}_i$ is a symmetric integration matrix. Thus, $\boldsymbol{M}_1, ..., \boldsymbol{M}_d$ is a set of matrices that enables discrete evaluation of the scalar product and consequently POD as described in section 1.2.3.1. It is clear that $\boldsymbol{\mathcal{F}} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is the discretization of function f.

Algorithm 12: ST-HOPOD

input : $\boldsymbol{\mathcal{F}} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, truncation rank $\boldsymbol{r}$, processing order $\boldsymbol{p}$
output: $\hat{\mathcal{X}} = (\hat{\boldsymbol{X}}_1, ..., \hat{\boldsymbol{X}}_d) \cdot \hat{\mathcal{W}}$

$\hat{\mathcal{W}} = \boldsymbol{\mathcal{F}}$;
for $i = p_1, ..., p_d$ **do**
  /* Compute POD of $\hat{\mathcal{W}}_{(i)}$ with mas matrices $\{\boldsymbol{M_k}\}_k$ then truncate to r_i */
1 $(\boldsymbol{X}, \boldsymbol{\Sigma}, \boldsymbol{\Phi}^\intercal) = \text{POD}(\hat{\mathcal{W}}_{(i)}, \{\boldsymbol{M_k}\}_k)$;
  /* Applying eq.(3.3.2) and setting integration matrix to identity. */
2 $\hat{\mathcal{W}}_{(i)} = \boldsymbol{\Sigma}_{tr} \boldsymbol{V}_{tr}^\intercal$;
3 $M_i = Id_{r_i}$
return $\mathcal{X} = [\![\hat{\mathcal{W}}; \hat{\boldsymbol{X}}_1, ..., \hat{\boldsymbol{X}}_d]\!]$

3.3.2 Functional-TT

Here we apply the same process of adapting the SVD based algorithm to continuous functions. Actually several publications have been published on this topic in the recent years, among others Oseledets implied that his technique could be applied to functions but the most determinant work came from Bigoni et al. [BEkM16] as well as in Bigoni's PhD. thesis [Big14] and Gorodetsky et al. [GKM16]. It is quite natural to draw a parallel between the TT decompostiion, ST-HOPOD and the RPOD, in both cases, the idea is to perform recursively 2D decompostions while retaining all the information obtained at each step to speed up the process. The main point here is to use the very efficient TT-format for functions. It is reminded that the storage cost is linear in d while only one SVD/POD is performed at each step. Unlike ST-HOPOD, only part of the information is explicitly transfered to the next step, this embedded implicit representation is very efficient, thus no core tensor tensor is required. A brief overview of the properties and algorithms given in [BEkM16, Gor16] is given next. We can write a function that as TT representation as

$$f(x_1, x_2, ..., x_d) = \sum_{\alpha_1, ..., \alpha_{d-1}} G_1(x_1, \alpha_1) G_2(\alpha_1, x_2, \alpha_2) \cdots G_d(\alpha_{d-1}, x_d) \tag{3.3.3}$$

where $\{\alpha_i\}$ are indices lower that the rank $\boldsymbol{r} = (r_1, ..., r_d)$. Once again we can introduce $\alpha_0 = 1$ and $\alpha_d = 1$ to make the notations uniform among functions $G_i(\alpha_{i-1}, x_i, \alpha_i)$.

Regarding the decomposition algorithm, the process relies on the same idea of replacing the SVD with PODs as well as using identity matrix to treat the discrete variable α_i. Bigoni et al. proved that such a process gives an actual Hilbert-Schmidt kernel which can in turn be separated. The first decomposition (POD) of $f \in L^2_{\boldsymbol{\mu}}(\Omega)$[4] with $\Omega = \Omega_1 \times \cdots \times \Omega_d \subset \mathbb{R}^d$, yields the following approximation

$$f(x_1; x_2, ..., x_d) = \sum_{\alpha_1 = 1}^{r_1} \sigma(\alpha_i) \gamma_1(x_1, \alpha_1) \phi_1(\alpha_1; x_2, ..., x_d) \tag{3.3.4}$$

[4]Here we assume without loss of generality that the POD is defined with the $L^2(\Omega)$ scalar product. $\boldsymbol{\mu}$ is the measure on Ω

Now, the idea is to apply a new decomposition on $\sigma(\alpha_i)\phi_1(\alpha_1; x_2, ..., x_d)$ to do so, the scalar product on the first dimension, i.e. the integral operator, has to be redefined. Let $X = \mathbb{N} \times \Omega_2$ and $Y = \Omega_3 \times \Omega_d$ and τ be the counting measure on $\mathbb{N}$. It is shown[5] easily that if $f \in L^2_{\boldsymbol{\mu}}(\Omega)$ then $(\sigma_1(\alpha_i)\phi_1(\alpha_1; x_2, ..., x_d)) \in L^2_{\tau \times \mu_2 \times \cdots \times \mu_d}(X \times Y)$. Consequently one can again perform a decomposition in which the integral with measure τ is equivalent to a sum. The the following expression is obtained,

$$(\sigma_1(\alpha_i)\phi_1(\alpha_1; x_2, ..., x_d)) = \sum_{\alpha_2=1}^{r_2} \sigma_2(\alpha_2)\gamma_2(\alpha_1, x_1, \alpha_2)\phi_2(\alpha_2; x_3, ..., x_d) \tag{3.3.5}$$

It is injected in eq.(3.3.4) which now reads

$$f(x_1; x_2, ..., x_d) = \sum_{\alpha_1=1}^{r_1} \sum_{\alpha_2=1}^{r_2} \gamma_1(x_1, \alpha_1)\gamma_2(\alpha_1, x_1, \alpha_2)\phi_2(\alpha_2; x_3, ..., x_d) \tag{3.3.6}$$

The process is repeated until all dimensions are separated and the decomposition from eq.(3.3.3) is obtained. Then $\gamma_i(\alpha_{i-1}, x_i, \alpha_i) \in L^2_{\mu_i}(\Omega_i)$ and $(\gamma_k(i, \cdot, m), \gamma_k(i, \cdot, m))_{L^2_{\mu_i}} = \delta_{mn}$.

Remark. In the last expression, the orhtonormality is preserved for the right index, as for the discrete TT decomposition it means that the counterpart for the left index is false. This is due to the order in which the decompositions are processed. Consequently, not all modes are orthonormal within a cores of the approximation $\{\gamma_i\}_i = 1^d$.

As for the other methods in this section, the properties of the discrete decomposition are preserved however some properties are given by Bigoni et al. [BEkM16]. An error estimate is given:

Proposition 3.3.1 (4.3, Bigoni et al. [BEkM16])**.** *Let the functional tensor train decomposition be truncated retaining the largest singular values* $\{\{\ \sigma_i(\alpha_i)\}_{\alpha_i=1}^{r_i}\}_{i=1}^{d}$. *$\tilde{f}$ the approximation of f fulfills the condition:*

$$||f - \tilde{f}||_{L^2_\mu} = \min_{\substack{g \in L^2_\mu \\ \mathrm{TT-rank}(g)=\boldsymbol{r}}} ||f - g||_{L^2_\mu} \leq \sum_{i=1}^{d-1} \sum_{\alpha_i=r_i+1}^{\infty} \sigma_i(\alpha_i)^2 \tag{3.3.7}$$

Some convergence properties correlated with smoothness of function f are also available in the same paper.

Algorithm. Implementing the functional TT simply means to replace SVD operations in 10 by POD and make sure operations are consistent.

Conclusion

In this chapter we have seen several functional decomposition methods. First multiparameter *a priori* PGD was presented. It was noted that the algorithm presented by Chinesta

[5] The demonstration relies on Hilbert-Schmidt kernel decomposition theory i.e. POD, it lies in the following equation

$$\begin{aligned} \int_{X \times Y} |\sigma_1(\alpha_i)\phi_1(\alpha_1; x_2, ..., x_d))|^2 d\tau(\alpha_i) d\mu_2(x_2) \cdots d\mu_d(x_d) &= \\ \sum_{\alpha_1} \sigma(\alpha_1)^2 \int_{\Omega_2 \times \cdots \times \Omega_d} |\phi_1(\alpha_1; x_2, ..., x_d))|^2 d\mu_2(x_2) \cdots d\mu_d(x_d) &= \sum_{\alpha_1} \sigma(\alpha_1)^2 < \infty \end{aligned}$$

et al. [CKL13] corresponds to a functional ALS algorithm and results as for every PGD algorithm into a CP format function decomposition. Such presentation of the PGD to a certain extent the main advantages of PGD which is a method that enables solving directly EDPs (usually elliptic ones). Then, RPOD, a natural extention of POD was presented. It consists in applying recursively POD to an bivariate interpretation of a multivariate function. Several data structures were studied and it was shown that it is possible to represent RPOD expansion as canonical or tucker format. But these are not memory efficient as they generate a lot of redundant information. Consequently a recursive tree structure was adopted for implementation. Finally, it was shown that d-way array tensor decomposition methods can be translated into functional decomposition methods with corresponding format and properties.

In the next chapter, the multivariate function and tensor decomposition methods are implemented and their approximation properties are tested. The python library `pydecomp` that was developed during this thesis has been used for this purpose.

Chapter 4

Numerical Experiments

Contents

4.1 A decomposition library 95

4.2 Synthetic data comparison 100

4.3 Decomposition methods on numerical data 107

4.3.1 A scalar simulation : 2D lid driven cavity at high Reynolds number 107

4.3.2 Experimental data : droplets evaporation 112

4.3.3 A vectorial simulation : breaking wave 115

4.4 Standard interpolation techniques for reduced basis ROM . . 119

In this chapter, we propose a comprehensive numerical study of the decomposition methods that have presented in chapter 2 and 3. Indeed, very limited comparison between these methods is available in the literature, the goal, here, is to provide a general view of decomposition methods at work and draw conclusion on their use in the context of scientific computing and in particular as a first stage to develop ROM.

This chapter is formatted as follow. In order to conduct the necessary tests, two codes were developed during my thesis and are presented in the first section. Then, in the second section, a comprehensive comparison of the decomposition methods applied to synthetic data is proposed. Discussions on the numerous options available for the user are proposed and supported with data. In the final section, the most efficient methods are applied to actual data, both from numerical simulation of fluids and actual experiment conducted at I2M in the hope of providing an extensive sample of data available in mechanics laboratories.

4.1 A decomposition library

One of the goals of my thesis was to program a library for data decomposition, its purpose being both experimentation and "industrial" use (either in the lab or outside). With Pr. Azaiez we defined a short list of necessary features that was later enriched with our findings.

Data decomposition library specifications

a. The library should allow an accurate comparison of the most promising methods proposed in the literature. In other words we should rely on this library to propose a complete analysis of these methods as it is proposed in this chapter. To do so, a number of bricks are required:

   - Efficient Bivariate solvers : SVD, POD and PGD. Relying as much as possible on existing libraries such as `LAPACK`. (Implementation of chapter 1).
   - Numerical integration methods such as trapezoidal rule.
   - Tensor formats : Full, canonical, Tucker, TT (see section 2.2) and tensor algebra (see section: 2.1)
   - Associated decomposition both in tensor and functional case (see chapter 3 and section 2.3).
   - Analysis tools such as error estimator, orthonormality tests, test functions, etc.

b. The code is meant for use in scientific computing thus is must be able to read data in the most common formats and make the reduced data as easy to use as possible. This translates in the following features:

   - Read common formats : VTK, MATLAB, Fortran typical output, Adios `.bp` (used in homemade CFD code Notus)
   - Store decomposition efficiently (binary files) for later use.
   - Provide reliability estimates

c. A complete documentation is required for easier diffusion inside and outside the lab. Also high level routines must make the program as easy to use as possible.

d. The code must be modular so that adding new features (such as formats or integration schemes) is easy even if computing efficiency is reduced. Also parallelism must be kept in mind even if it is not intended for V1.

e. Can be used as tool for ROM building.

Existing implementation The first step for anyone intending to fulfill these specification (as well as any software project) is to review existing solutions. Actually, a lot has already been done by research teams, only to cite the most prominent ones (a vast list of implimentation is given by Grasedyck et al. in [GKT13]).

- T.G. Kolda and B. Bader [KB09] propose complete implementation in their Tensor Toolbox [BKO17] both in MATLAB and C++. It proposes mainly CP and Tucker decomposition methods. Many other softwares are listed in [KB09] on the same topic.
- TT format is covered by Oseledets team as MATLAB TT-Toolbox [Ose18] and python implementation `ttpy` [ODS18]. Other python implementation can be found such as Bigoni's `TensorToolbox` [Big14].

- Kessner et al. propose a MATLAB hierarchical format decomposition library `htucker` [KT13].
- The famous TensorFlow (`www.tensor-flow.org`) although relying on tensors and providing some decomposition tools is actually not suited at all for our purpose. It is mainly build for neural network training.

As one can see, all techniques but the RPOD (due to its relative anonymity) have already been implemented in efficient and open access softwares. So why not use them? First one can see that various languages are used but mainly MATLAB. Thus fitting them all would require interfacing many codes from different languages or at the very least MATLAB Toolboxes into one entity. Although it is possible and arguably a fast way to obtain a general decomposition tool, that would have meant having little control on its evolution. Also, from an moral point of view, using a proprietary software for diffusion of research work is not very well suited. I, just as many students, would have faced difficulties to obtain a MATLAB licence (shared token system,...). Consequently, it was chosen to code our own library while using building bricks (such as `LAPACK`) as much as possible.

Discussion on the programming language. For any programming effort starting from scratch, choosing an adapted language is crucial. Our choices were actually quite narrow as shown in this short review.

MATLAB As seen briefly in the previous paragraph, MATLAB is a proprietary software used intensively in research. Prototyping methods is very fast using its interpreted language and efficient implementations are available through C/C++ compiled code. It is a reliable solution with an extensive documentation and a large community. Many if not all data reduction techniques are already proposed as toolboxes. Its main but massive drawback is the expensive license that is required.

C/C++ No need to present the most popular language of the last decade including for scientific computing applications. It allows fast execution from C and Object Oriented programming (OOP) for complex structures. Many libraries (including data reduction) are available. It would have been a good choice if I mastered this language but its complexity made it risky as little support was available in the lab.

Fortran90 Mainly used in the scientific computing community for historical reasons, it is probably the fastest language for this purpose. It allows easy conversion from mathematical formulas to fast programs. In spite of its reputation, it has integrated many OOP concepts in its last norms (Fortran2003 and Fortran2008). I have a very good knowledge of this language and it is very well suited for CPU intesive applications such as decomposition. That makes it a very good candidate. However some intrinsic limitations remain in terms of flexibility.

Python Python is also a *en vogue* language. This interpreted languages offers quick development cycle as well as an incredible versatility. It also maintains good efficiency thanks to precompiled libraries such as `numpy/scipy` for scientific computing. If one needs a very efficient implementation that cannot be written in term of existing libraries, it is possible to interface seamlessly fortran precompiled code using *f2py*. Contraty to MATLAB it is open access and its growing community in scientific computing is gradually catching up with the aforementioned. Interfacing with popular data format is easy thanks to the numerous libraries available and serialized binaries are the natural way of storing complex object without explicitly interfacing them.

Since it is not only a scientific computing language, it allows GUI development, graphical outputs. Finally it is quite easy to imagine a mixed code with fortran whenever Python becomes too slow.

Actual code Actually two seperate libraries have been implemented during my thesis. The first one, a fortran decomposition library during the first two years. Indeed, there are numerous hindsights in the literature concerning computing time. It seemed to be a central problem and HPC techniques would be needed at some point. While letting the possibility to upgrade to parallel versions a sequential code was produced. It is thoroughly documented thanks to `doxygen`. As shown in figure 4.1, it can be explored through any browser making it very easy to navigate the routines and structures. However Fortran,

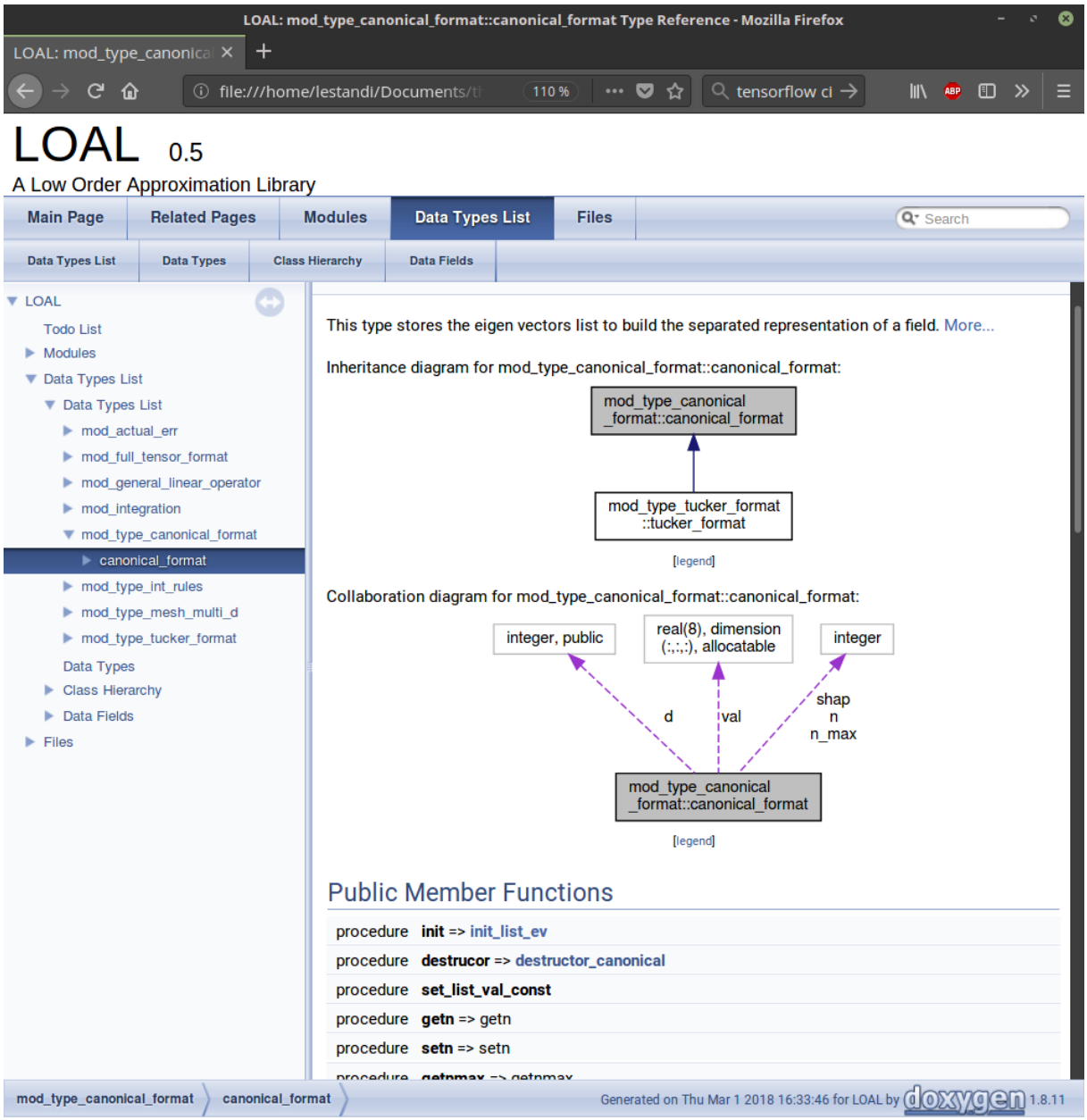


Figure 4.1: Dynamic HTML documentaion of the `Fortran low order approximation library`.

even with the newest Object Oriented (OO) features, is a bit clumsy for our goal. First, although it's possible to manipulate arrays of arbitrary dimensions, it is not native in the

language and maps from d indices to a global index have to be build with obvious loss of efficiency. Complex structures can be build but they lack flexibility. Most importantly, and this is well known issue of Fortran, IOs are not natural, every aspect must be prescribed manually except in the rare cases where libraries exist. This advocates in favor of using higher level languages for practical use. The next point in that direction is that actually for sequential decomposition using an adequate technique, CPU time are very low e.g. about a minute for a 13GB tensor (which is roughly the memory of a working station/laptop.). One can conclude that memory is the issue and an adequate use of higher level language would allow comparably fast computing with a more user friendly.

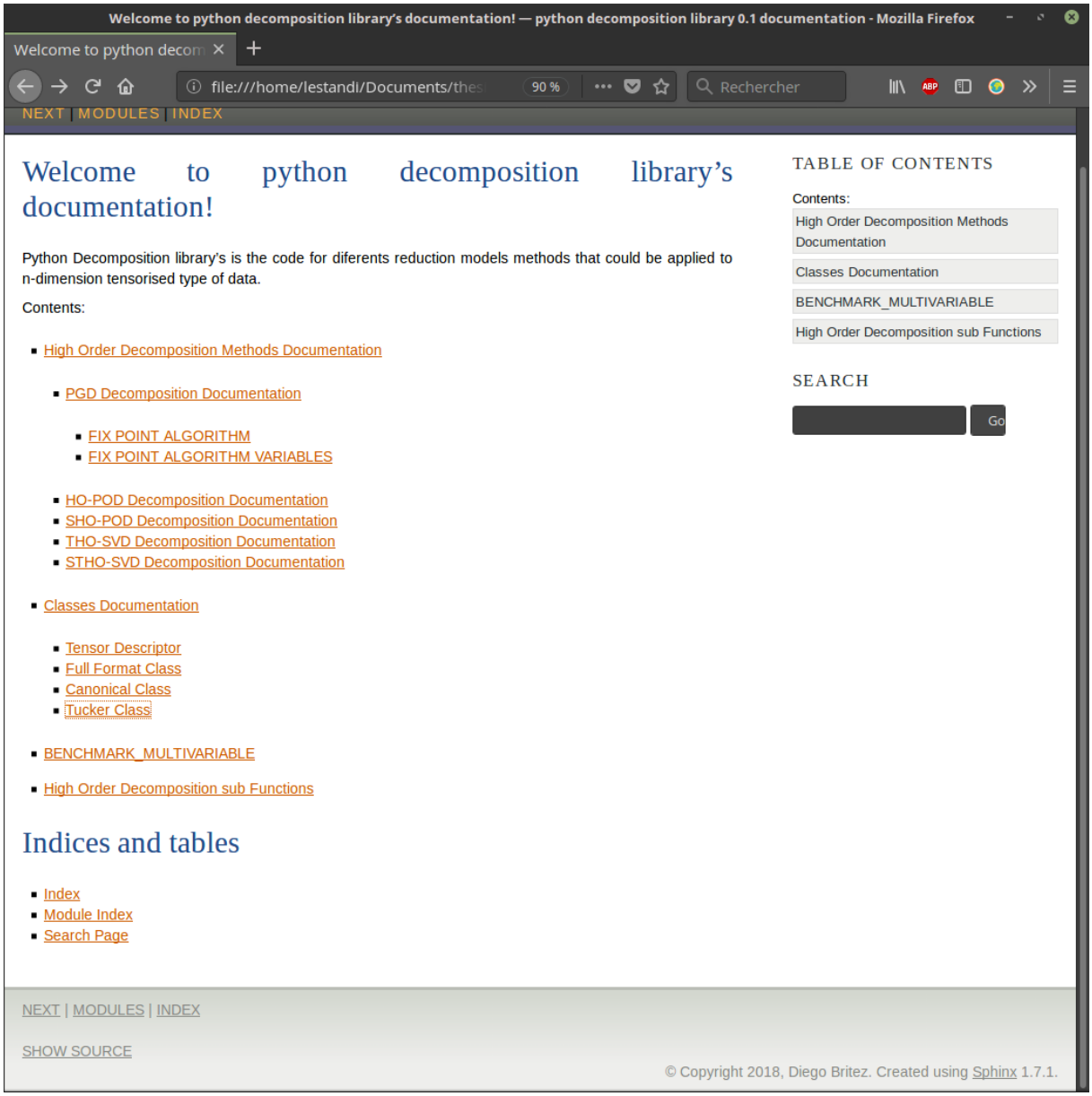


Figure 4.2: Dynamic HTML documentaion of the `pydecomp` build with Sphinx.

This is why, Diego Britez (Masters Student, 6 months internship) and I have worked to build a Python implementation of this software. It has allowed us a better hierarchy gained through my first experience and as we will see in the next section, good use of numpy solutions have been made as the code is as fast as the fortran implementation. The main difference between these two codes, is the presence of an actual RPOD algorithm and structure for any dimension as well as a TT decomposition method and structure.

This code is also thoroughly documented through Sphinx tool, an illustration is proposed in figures Fig. 4.2.

As a conclusion, the second version of the code fulfills the specifications given above, modularity in particular is ensure with classes for each format and separate decomposition methods. It was not actually used as a building brick for our ROMs for historical reasons but could do so efficiently. Interfacing with the most advanced decomposition techniques might be necessary as the algorithms are more complex e.g. Bigoni's TT-DMRG-cross [BEkM16].

4.2 Synthetic data comparison

Using synthetic data is very useful to test the methods and a variety of parameters that might influence the convergence and compression rates. Our data is generated on uniform grids[1] of $n_1 \times \cdots \times n_d$ that discretizes $\Omega = [0,1]^d$. The following real test functions are used

$$\begin{aligned} f_1(\boldsymbol{x}) &= \frac{1}{1+\sum_i x_i} \\ f_2(\boldsymbol{x}) &= \sin(||\boldsymbol{x}||_2) \\ f_3(\boldsymbol{x}) &= \sqrt{1-\prod_i x_i} \end{aligned}$$

A special function was used to reproduce singularity for $d = 5$,

$$\begin{aligned} f_s(x_1,x_2,x_3,x_4,x_5) = \; & x_1^2\{\sin[5x_2\pi + 3\log(x_1^3+x_2^2+x_4^3+x_3+\pi^2)]-1\}^2 \\ & +(x_1+x_3-1)(2x_2-x_3)(4x_5-x_4)\cos[30(x_1+x_3+x_4+x_5)] \\ & \log(6+x_1^2x_2^2+x_3^3) - 4x_1^2x_2x_5^3(-x_3+1)^{3/2} \end{aligned}$$

A typical case $d = 3$. In order to evaluate the separability of these three test functions, we chose a relatively coarse grid of $32 \times 32 \times 32$. The results are presented for all three functions in Fig. 4.3. These graphs present the relative decomposition error[2] defined by

$$\epsilon = \frac{||\mathcal{T}_{\text{exact}} - \mathcal{T}_{\text{decomp}}||}{||\mathcal{T}_{\text{exact}}||}, \tag{4.2.1}$$

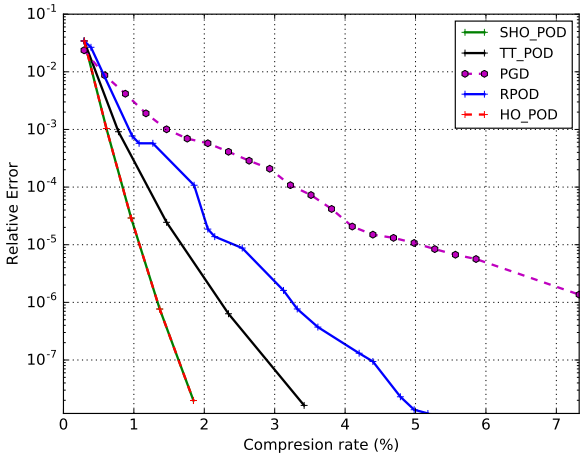
as a function of the compression rate (in %) which is the storage cost of a decomposition at a given rank divided by the storage cost of the full format tensor i.e.

$$CR = \frac{\texttt{Mem_cost}(\mathcal{T}_{\text{decomp}})}{\texttt{Mem_cost}(\mathcal{T}_{\text{exact}})} (\times 100 \text{ for } \%). \tag{4.2.2}$$

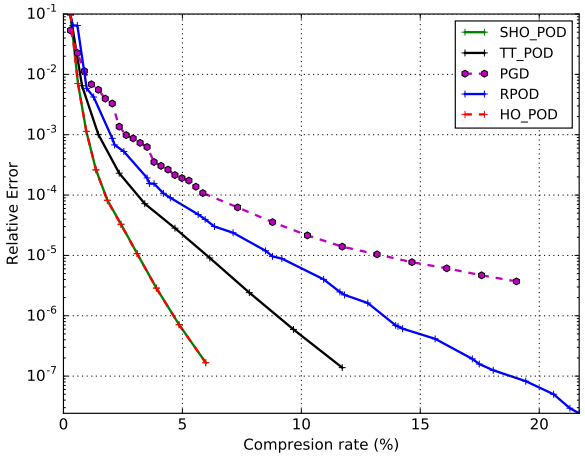
First all 5 methods are tested with L2 norm and scalar product i.e. POD is applied as a bivariate decomposition method. A distinct pattern can be observed in these 3 figures. The least efficient compression method is PGD, which was expected in terms of CPU time due to the iterative algorithm at the center of the method. However as the format is very efficient by definition one could hope that the sub-optimality of the algorithm (see sections 3.1 and 2.3.1 for PGD and ALS NP hard problem) would not impact too

[1] Using a non uniform grid would have little influence on the accuracy given that one uses accurate integration schemes. However it may help to increase the computing speed by using a sparser grid.

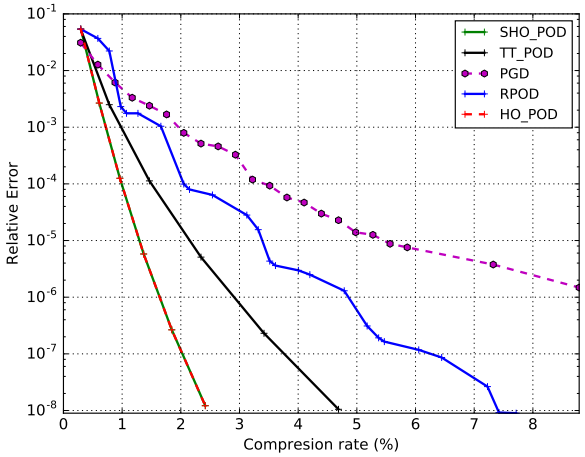
[2] The norm is not specified here as it can be either a Frobenius norm of tensors or the $L^2(\Omega)$ norm.



(a) f_1



(b) f_2



(c) f_3

Figure 4.3: Decomposition of 3 test functions with $d = 3$ on a 32^3 grid with 5 discretization methods, using L^2 integration and norm.

much the decomposition. Actually, in spite of acceptable convergence of the fixed point algorithm, the compression rate of PGD grows much quicker than any other methods. Still it should be noted that in all three cases, it provides the best rank-1 decomposition

as one should expect for a method based on successive rank-1 decompositions.Then it is clear for all 3 functions that TT-POD and ST-HOPOD are the most efficient methods, both showing exponential decay, although with a slop change for f_2 (Fig. 4.3b) as it is the least separable of all three functions. One should note that the ST-HOPOD and T-HOPOD are superposed, this behavior was already observed in [VVM12] (for SVD based decompositions) in the case of easily separable functions. As we will see in the next paragraph the main difference lies in the computing time of the methods. Additionally, one can see that TT-POD is less efficient for these small 3D problem as the core does not require much memory in Tucker format. Finally the RPOD is close to TT for the lowest truncation rank i.e. as long as they are virtually equivalent[3] but the nature of this recursive decomposition creates decomposition error jump when one enters a new branch with important weight. This phenomenon of steps is most prominent in Fig. 4.3c. As said in 1.4 it is useless to show different grid resolution as these functions are smooth and decomposition behavior is thus uncorrelated with grid density, only the compression rate would vary since it depends directly on the number of discrete points.

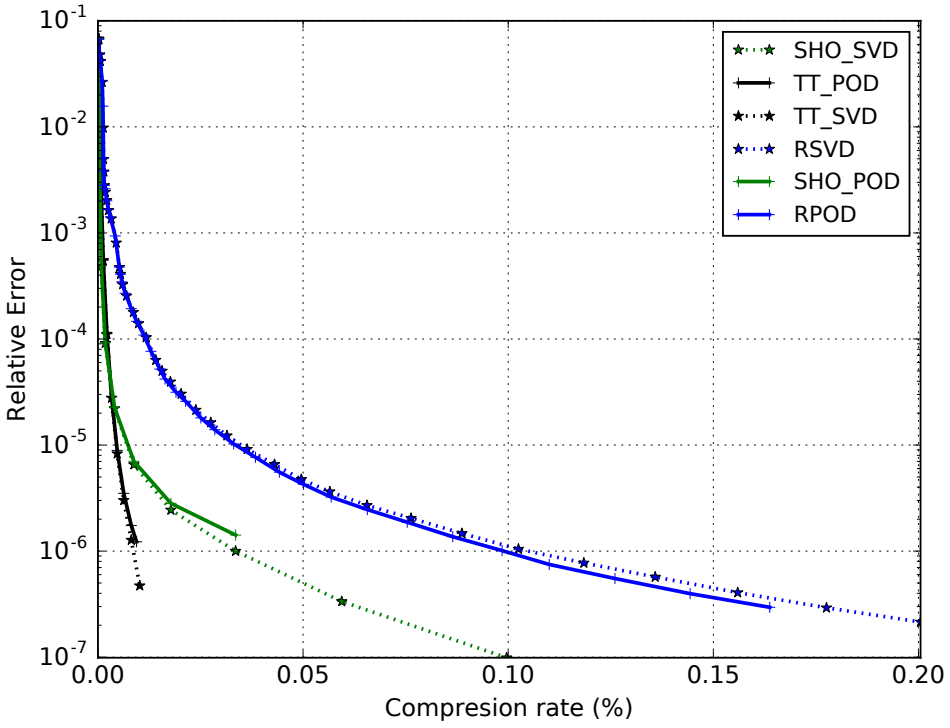


Figure 4.4: Decomposition of f_s on a 40^5 grid with L^2 and l^2 scalar products, decomposition error in their relative norm

Influence of the scalar product choice on f_s. In order to assess the influence of the scalar product for higher dimensions decomposition, f_s the least separable of the synthetic data functions is used, with $d = 5$ and 40 equispaced grid points in each dimension. Indeed,

[3] Actually, for TT rank of 1 and and RPOD rank of 1 i.e. 1 mode only for each dimension, then both algorithms are strictly equivalent, only the data structure is different. Then when the rank grows, the association of modes by explicit summation in Recursive format is less efficient than the implicit summation to the TT format. Finally the truncation strategy used in the software requires that any branch with a weight above truncation limit has at least one leaf kept in the evaluation and all other leaves below the truncation limit are ignored. This results in cumulative loss in precision which means that the rank/epsilon truncation in recursive format is less sharp than in TT format.

method	computing (s)	evaluation (s)
RPOD	9.535	31.80
RSVD	7.964	32.20
ST-HOSVD	1.096	1.23
ST-HOPOD	2.378	0.98
TT-SVD	1.205	1.19
TT-POD	2.206	1.13

Table 4.1: CPU times on f_v for $n = 40$, $d = 5$ with a tolerance of $\epsilon = 10^{-12}$

for easier decompositions on Cartesian grids, no difference can be seen in the relative error graphs. Fig. 4.4 shows recursive, TT and sequentially truncated tucker decompositions for both L^2 (POD) and l^2 (SVD) scalar products. One can see that for each method, the error and compression rate are almost the same for both scalar products. The trend being overwhelmingly driven by the method itself. Results might differ for different grid types and functions with sharp variations in which an actual integration would capture better theses phenomena. Also one should notice that in this case where $d = 5$, TT decomposition is now more efficient than ST-HOSVD when high accuracy is required ($\epsilon < 10^{-4}$) and does not show any sign of linear decay contrary the other methods. In particular RPOD clearly shows a linear decay from 10^{-3} onward in spite of being competitive for accuracy up to 1%.

In conclusion as long as one uses a Cartesian grid, using SVD or POD does not influence the compression behavior and other factors should be used to decide which one to use depending on the use of this decomposition. For ROM building, one should use a EDP adapted scalar product i.e. POD to obtain orthonormal modes. It should also be preserved for physical analysis of a problem. Another criterion is CPU time, especially if one only intends to reduce storage cost of large datasets.

Relative CPU time On the same problem, let us focus on the CPU times for each methods as well as the reconstruction time needed to obtain a full tensor from the reduced representation. Results are shown in table 4.1. PGD has been voluntarily excluded from this table as it requires several hours, T-HO**D is not shown either as it requires roughly 4 times the ST-HO**D as expected from the number of dimensions. One can see that for all these methods, SVD based decomposition is faster. This is due to the implementation of POD that requires an additional diagonal (possibly multiple diagonals) matrix multiplication each time a scalar product is needed as compared to the SVD by EVD. In the end, for TT and ST decompositions for which the cost of the bivariate decomposition is controlling CPU time, this results in doubling the time for POD. For recursive decomposition, there are numerous overheads that makes the difference much smaller. Regarding the evaluation time, it was not particularly optimized as it is not a central task to reconstruct full tensors, more likely for higher order tensors, one might need only to reconstruct a slice of the tensor. The third column of table 4.1 is the evaluation of the last data point in Fig. 4.4. First one can see that both recursive methods takes roughly the same time which is 30 times more than the other two methods. This observation definitively disqualifies recursive methods for data reduction purpose. The the Tucker and TT are in the same range of reconstruction time, the slight differences present here translate the slight variation in their number of modes due to different truncation criteria implementation.

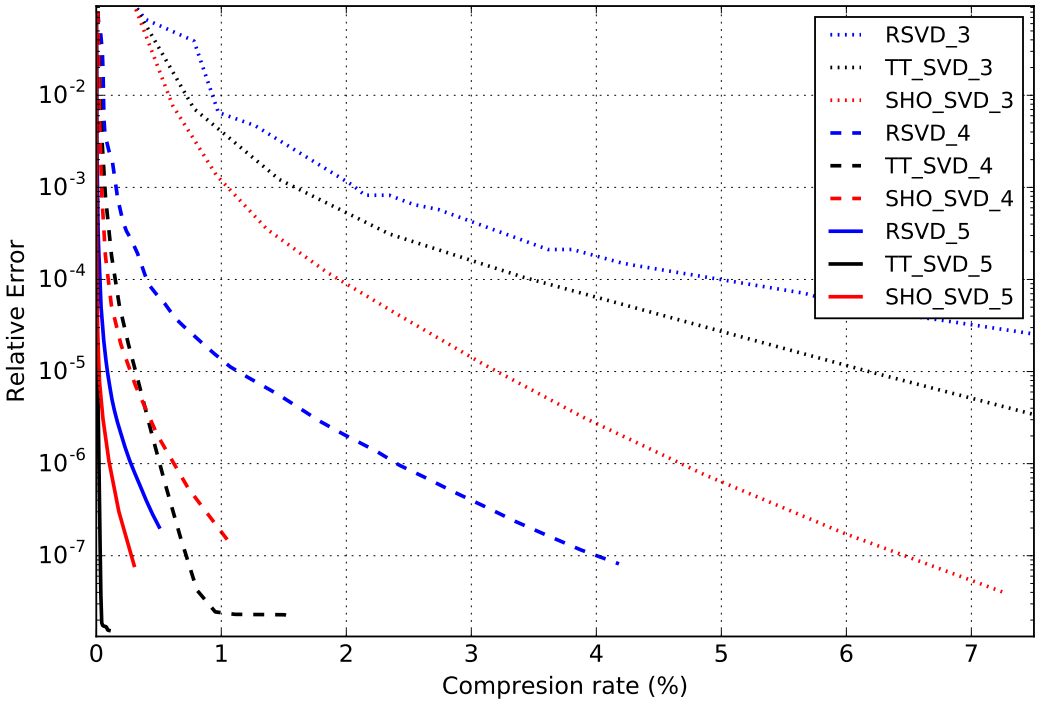
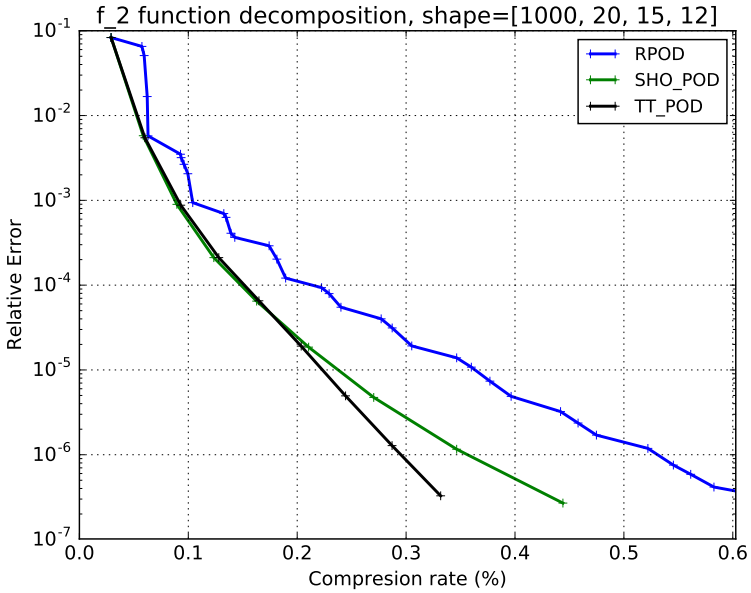


Figure 4.5: f_2 decomposition with $d = 3$ to 5 on a 32^d grid with three decomposition methods, using L^2 integration and norm.

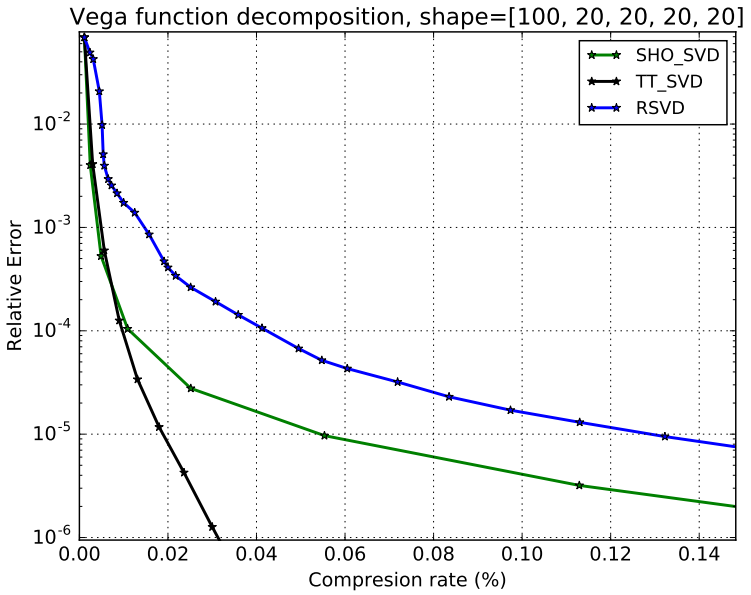
Number of dimensions and shift in the adequate methods. Now, we investigate the influence of the number of dimensions in order to decide upon which method to use. To do so, in Fig. 4.5 we compare the same 3 methods with SVD solvers and show on the same graphs relative error as a function of compression rate for $d = 3$ to 5 in the decomposition is function f_2. One can see, once again that RSVD is the worst in all cases but its distance to the other methods tends to diminish as d grows. Indeed, the recursive structures prevents the storage cost to explode with the number of dimensions d. This is also the main difference between TT and Tucker format. While the latter is more efficient for $d = 3$ and remains competitive[4] up to $d = 5$ thanks to efficient decomposition, it is outclassed for storage purpose by TT. This is particularly visible for $d = 5$ (sold lines). Thus, one can conclude that TT decomposition should be preferred as soon as $d \geq 5$ if the orhtonormality of the modes is not a criterion. For lower order problems, it is probably preferable to choose ST-HOSVD method as it ensures orhtonormality of the basis while being the most efficient method at the same time.

Unbalanced grid. Another interesting experiment is variating the grid resolution among dimensions. As mentioned by Vannieuwenhoven [VVM11], a good heuristic for CPU time is to treat the largest dimension first in ST-HOSVD, this is also true for RPOD and TT-SVD. It is also quite important for compression rate in recursive format as few modes of the first dimension will be stored. As one can see in Fig. 4.6, the large imbalance in favor of the first dimension makes recursive decompositions comparable (although less efficient) with ST-HOSVD. Fig. 4.6a shows an exponential decay of the error with respect to the compression rate, just as observed for equal grid refinements in Fig. 4.5. The main

[4] most efficient methods depends on required accuracy for $d = 4$.



(a) f_2 decomposition on a $1000 \times 20 \times 15 \times 12$ grid with POD based methods



(b) f_s decomposition on a $100 \times 20 \times 20 \times 20 \times 20$ grid with SVD based methods.

Figure 4.6: Decomposition of synthetic functions f_2 and f_s for unbalanced grid refinements.

difference lies in the comparatively higher efficiency of RPOD together with much clearer "stepping" phenomenon. In Fig. 4.6b, one can see that n_1 is 5 times bigger than the other n_i, this leads to a far greater efficiency of TT-SVD as compared with Fig. 4.4. Once again RPOD displays the same behavior as ST-HOSVD although the error is almost a decade greater. As expected, methods that treat dimensions sequentially are comparatively improving when the number of point in each dimension is imbalance. This should be taken into account when dealing with experimental data as it is in most cases largely imbalanced.

Technical limitations of the current version of the library. As already stated in the last section, the main limitation of the library (in both languages) is the memory requirement. The usual emphasis of CPU time found in the literature is mostly irrelevant as long a one uses adequate algorithm, i.e. correlation based 2D algorithm, namely POD or SVD through EVD, PGD and direct SVD using usual algorithm is to be avoided as long as the former converge. Indeed, it was shown in table 4.1 that CPU times remains within a few second range for datasets of 800MB. Consequently the main limitation is the RAM available on the working computer. Typical PC or cluster nodes offer 8GB to 64GB, which is already substantial memory. However it is far away from our target which is to compress simulation data obtained through modern computational fluid dynamics softwares. Indeed typical simulation output reach tens or hundreds of GB while exascale computing aims at generating PB of data [ABR+12].

There are mainly three directions to explore in order to solve these problems for the two most efficient procedures TT and ST:

Data segmentation The easiest way around memory limitation for storage is to split studied data into chunks that can be treated by a single computing unit. Many strategies are available (spatial, temporal) but they may compromise the physical analysis or ROM building power of these methods. It is thus easy to implement but introduces new limitation, however this kind of strategies can be turned into a tool for analyzing different ranges or behaviors in a flow as shown in section 1.4.3.1.

Parallelization The natural response to such difficulties in the context of HPC is to parallelize existing algorithm to take advantage of the computing clusters used to produce the data. This would at some point be a simple post-processing step of a large scale code. Efforts in that direction have already been made by several teams for CP decomposition (among others [KKU16, ZFXM]) as it can be expressed easily as a distributed algorithm. More work on distributed tensor computing has been performed in the recent years [Sch15, Ett15]. The most impressive of applied parallel decomposition examples lies in W. Austin et al [ABK16] in which they achieved ST-HOSVD of +500GB datasets. To do so they have proposed a *distributed memory* version of the ST-HOSVD algorithm and in particular mode-m multiplication (TTM in the paper) and autocorrellation matrix building. The eigenvalue problem is solved "sequentially" on each CPU as it is of very low dimension.

Cross approximation Another approach is to rely on partial evaluation called cross approximation of matrices and adapt it to higher order decomposition [BGK10, BEkM16, OT10], especially TT and HT formats. The idea is that given some regularity, one can evaluate a limited number of entries of the tensor to build an accurate representation (estimators are available). The memory overload problem is then bypassed since the input tensor can be viewed either as a blackbox function or a disk access. Obviously this approach can be coupled with parallelism.

Tensorization This approach does not solve *per se* the memory overload problem, but it allows hyper reduction through data layout manipulation. TT and HT methods have been shown to preform best when the number of dimension is high to very high ($d >> 1$), but simulation data usually comes with $3 < d < 7$ and largely unbalanced number of entries per dimension. In order to take advantage of this ability, one might artificially increase the number of dimension (up to $n_i = 2$). Several implementations have been proposed in the literature, among them the Quantized TT introduced by Oseledets [OT10] and Khoromskij [Kho11] has encountered some success [Cic14, Big14, SO11].

As often in scientific computing, a combination of these approaches is possible and will increase efficiency of the decomposition process.

In this section, it was shown that decomposition of multiparameter functions and high order tensors have been implemented successfully. Not all methods are equivalent in terms of compression efficiency and CPU time, two of them are clearly to be preferred: TT decomposition and sequentially truncated Tucker decomposition. HT has not been programmed as it simply provides an efficient shell for all other decomposition techniques and introduces significant complexity to the code. l^2 and L^2 norm perform equally on the regular grid and functions tested in this section, special discretization of Ω (linked with function smoothness) may reduce the evaluation cost of L^2 scalar product. It was shown that CPU time is not the main limitation as compared with memory handling. Consequently, further experiment will be limited by a single workstation/node RAM, 16GB in both cases during my thesis.

4.3 Decomposition methods on numerical data

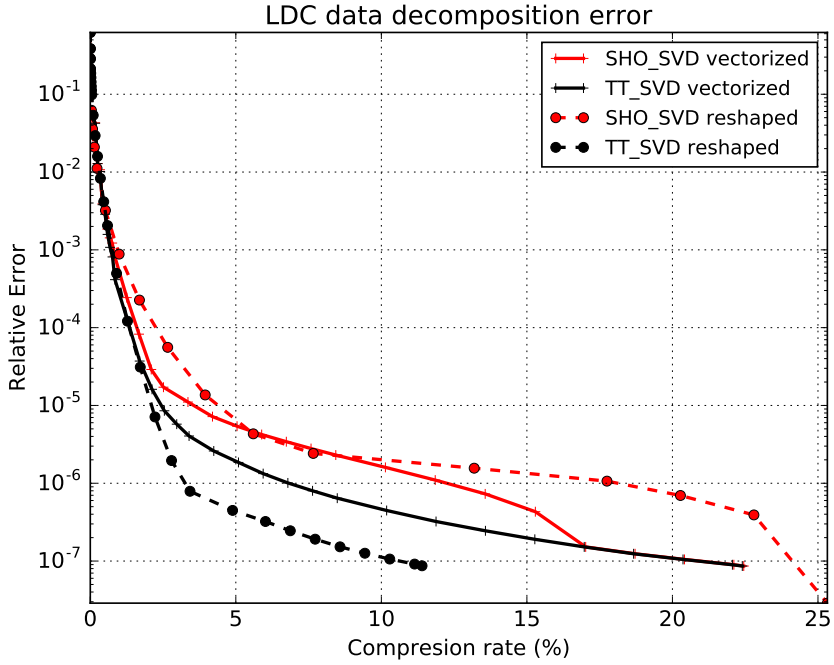
Now that the most efficient methods have been selected, TT-SVD and ST-HOSVD are applied to decompose data obtained through experiments, both numerical and physical. The goal of this section is to provide insight for efficient data reduction and qualitative analysis of their use.

This study is restricted to relatively small dataset i.e. in the order of 1GB so that decomposition as well as postprocessing fit on my laptop. Three different cases are studied. Two are scalar data of only one variable, that is to say $f : \mathbb{R}^d \longrightarrow \mathbb{R}$ as it is the simpler case, both from numerical and actual experiment. The third example intends to address the multiple variables of multivariate vectorial i.e. discretization of $f : \mathbb{R}^d \longrightarrow \mathbb{R}^p$.

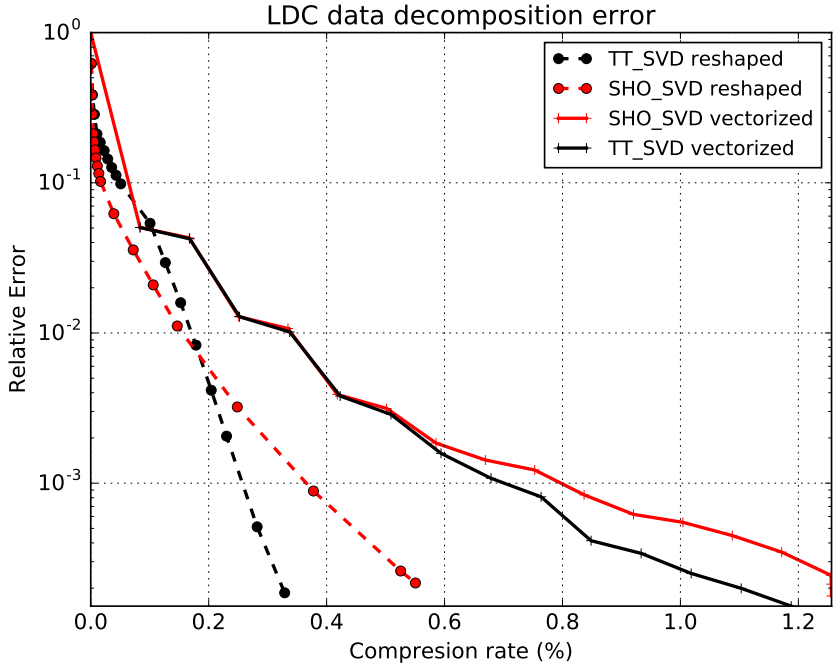
4.3.1 A scalar simulation : 2D lid driven cavity at high Reynolds number

First we investigate the simpler case of multidimensional field. Once again, the lid driven cavity simulation in stream-function vorticity formulation is used, see section 5.1 and 1.4.3.1 for further details.

The problem characteristics are briefly reminded to the reader. A DNS simulation of the lid driven cavity problem in streamfunction-vorticity formulation with high accuracy. High Reynolds numbers are studied, here we focus on range $Re \in [10000, 10100]$ with a spacing of 20. Time steps are very small, $\delta_t = 10^{-3}$ then snapshots sampling is coarser : $\delta_t = 0.2$ in order to capture longer time series and especially limit cycles. In order to capture the flow behavior from initial quiescent state to the limit cycle, simulation must run from $t = 0$ to a few thousands. Consequently, for this analysis, a narrow range of the limit cycle is sampled from $t = 1900.2$ to 1940, leading to 200 snapshots per Re. Finally, I chose a relatively coarse space grid of 257×257 for easier handling as we have shown that the number of modes is only weakly affected by grid density. In conclusion, after interfacing `pydecomp` with T.K. Sengupta LDC code (written in Fortran90), an order 3 tensor $\mathcal{T}$ of shape $66049 \times 201 \times 6$ is obtained. Space is given as a single vectorized dimension, as it is not clear if it is preferable to decompose like that or by seperating space into two dimensions leading to an order 4 tensor of shape $257 \times 257 \times 201 \times 6$, both approaches are shown in Fig. 4.7.



(a) Cutoff tolerance of $\varepsilon = 10^{-8}$



(b) Cutoff tolerance of $\varepsilon = 10^{-4}$

Figure 4.7: Lid Driven Cavity Simulation within the stable limit cycle time range, see 5, input tensor is of shape $6 \times 201 \times 66049$. $t = 1900$ to 1940 with a stepping of 0.2, space is a 257×257 grid that can be **vectorized** (solid lines) i.e. taken as a long vector of size 66049. Space treated as 2 dimension is referred as **reshaped** (dashed lines). Reynolds is a parameter dimension with $Re \in [10000, 10100]$ and a stepping of 20.

This data is strikingly separable, all four configuration offer machine precision[5] with relatively low compression rates 10% to 25%. Indeed both decomposition methods and both data layouts display exponential decay of the error as function of the compression

[5] The reader is reminded that 10^{-7} or 10^{-8} is considered machine error in the context of high order decomposition through correlation matrices.

rate. This is particularly visible when the error $E > 10^{-5}$ in the top graph Fig. 4.7a, for lower truncation one can see an abrupt change of slope which is attributed to reaching "noisy" data. Indeed, this phenomenon is observed on most actual datasets and it was already observed for 2D decomposition of the LDC data, see Fig. 1.10. Next, one can observe that all four methods display comparable accuracy for moderate accuracy, which means that the choice must be driven by the goal of the decomposition. For optimal storage, one is advised to prefer TT-SVD for both layout although vectorized layout allows the user to reduce the truncation error by almost a decade. Finally, The latter offers, by far, the best compression rate 10% for maximum accuracy as compared to the roughly 20% of concurrent methods. Regarding ST-HOSVD, the observation regarding layout is the opposite of TT-SVD as compression efficiency is (slightly) reduced with reshaping.

Remark (Handling of the space dimension). As shown for this example, the compression rate is weakly influenced by the space layout. This confirms the intuition that the amount of information contained in space does not depend on its layout. However we can see that it is not entirely true since differences appear early on, one can merely affirm that the qualitative separability of the field does not depend on the layout. In specific cases such as quasi 2D problems, the third dimension must be separated as it represents a huge gain to treat it separately. Indeed it can be seen as an identity function.

The rank however is drastically influenced by this choice as one can see in table 4.2 where the same cutoff value of $\varepsilon = 10^{-4}$ has been used with each method and the truncation error is virtually the same. It is important to notice that in spite of the sequential

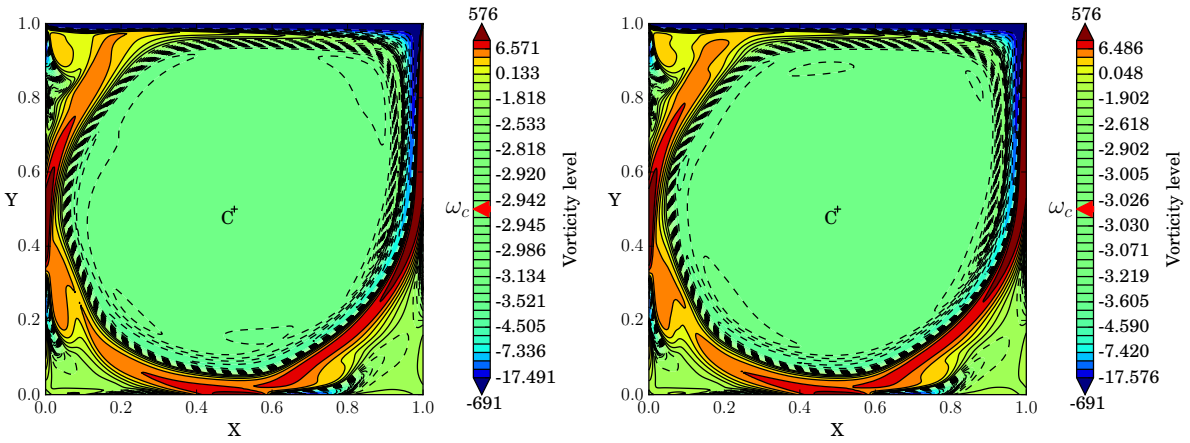
layout	vectorized	reshaped
ST-HOSVD	[15,18,6]	[59,63,18,6]
TT-SVD	[15,6]	[59,15,6,]

Table 4.2: LDC decomposition ranks with the same prescribed cutoff value $\epsilon = 10^{-4}$ (last point in Fig. 4.7b).

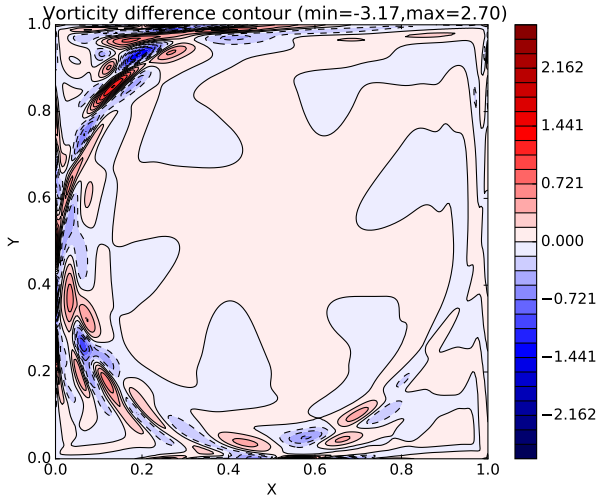
nature of these methods the ranks of time and Re are unmoved by the layout choice. Yet, spatial decomposition rank is drastically changed, being multiplied 4 times for each one in ST-HOSVD. It is interesting to notice that only the first rank in TT-SVD is big, the second one remains the same as for the vectorized layout. It can be interpreted that the space spanned by space dimensions 1 and 2 (embedded at the second stage of the algorithm) remains the same no matter the layout thus leading the same value of 15.

Now, we shift our attention to Fig. 4.7b. In this cases we are interested in the ability of these methods to compress the data with moderate accuracy. The superiority of the reshaped representation of space is blatant as it proposes a much finer range and roughly half the storage space. This phenomenon in enforced by relatively low spatial ranks as compared with n_x and n_y (see table 4.2). In terms of compression power, this largely overcomes the highly intertwined nature of both space axes i.e. the rich flow behavior lies in complex 2D structure that in spite of not being represented well in the reshaped layout is overcome by rank truncation.

Very limited physical hindsight is obtained observing "reshaped" space modes along X and Y which is why they are not shown here. the central region is mostly flat with varying mean values while the extremities of the domain show large spikes and modes Y present small scale oscillations in addition to larger structures near $y = 1$. The vectorized modes (not shown) are similar to the one given in space time decompositions in chapter 5 and section 1.4 for LDC.



(a) Original data (b) Reconstructed from ST-HOSVD, 1% error.



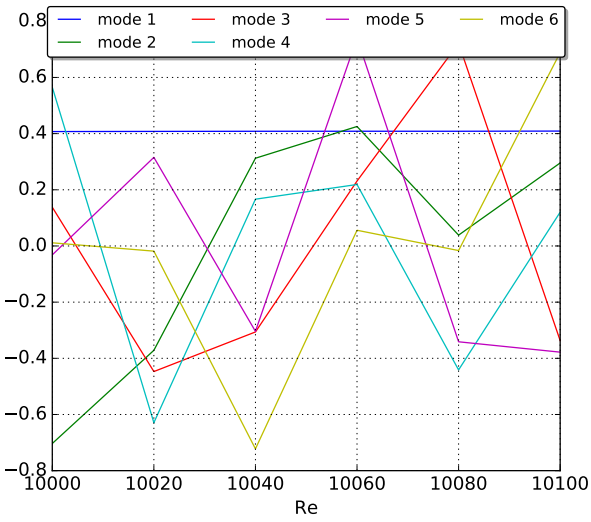
(c) Difference i.e. 4.8a- 4.8b

Figure 4.8: Vorticity field of the lid driven cavity at Re=10000, t=1900s decomposition is reconstructed compared with 1% relative error in Frobenius norm i.e. rank=(10,10,3) and compared to original dataset. Isolines are plotted as well as colormap, they are exponentially spaced from the center of the square value, solid is superior to $\omega(C)$ while dashed lines are inferior. This is to make comparison with centered data.

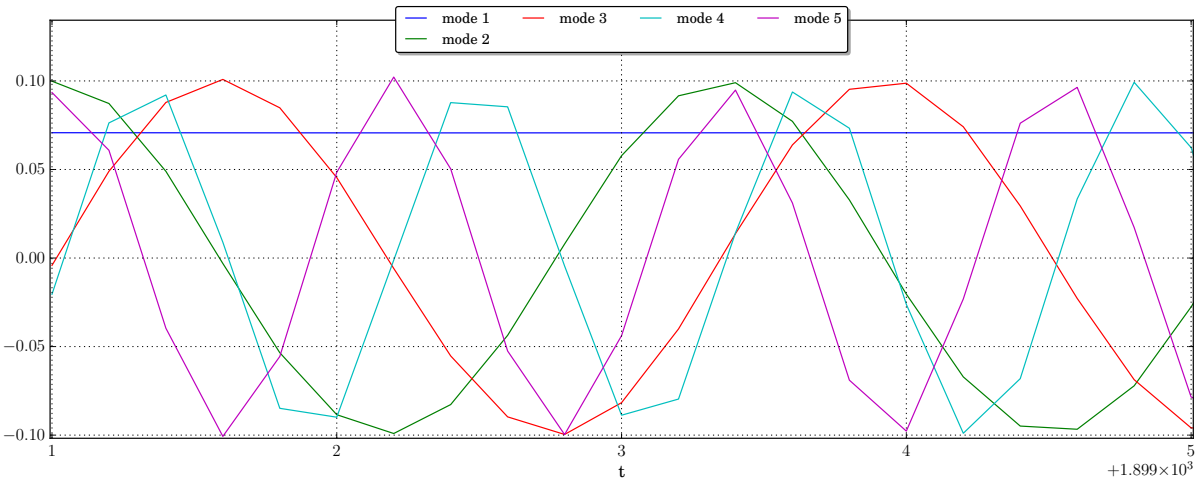
Fig. 4.8 shows that even with moderate accuracy of 1% error the reconstructed data is largely usable for qualitative analysis. This is very interesting for long term storage as the required amount of data for this dataset is reduced to 0.2% of the original 634MB i.e. 1.2MB. Here, Fig. 4.8a shows the original vorticity field of $Re = 10000$ at $t = 1900$ while Fig. 4.8b proposes the same field from reconstructed ST-HOSVD in the vectorized layout and Fig. 4.8c is the difference between these two fields. One can see that the structures are well captured as well as the minimum and maximum value. The central region vorticity level is off by a few percent. However the lower amplitude structures are captured with less accuracy. Finally, the difference map shows that locally, the error can reach values comparable to the central region [6] but the frame remains mostly green i.e. the difference is below 1. As one would expect, most of the error is contained in large gradient regions near the boundaries of the domain.

Finally, in order to acquire a better grasp of the decomposition obtained, Fig. 4.9 shows the first modes associated with Re and time. In both cases, the first mode plays

[6] -3 to 4 are the min and max values of the error field as indicated at the extremities of the colormap



(a) First 6 Reynolds modes for $Re \in [10000, 10100]$



(b) First 5 modes closeup on $t \in [1900, 1905]$

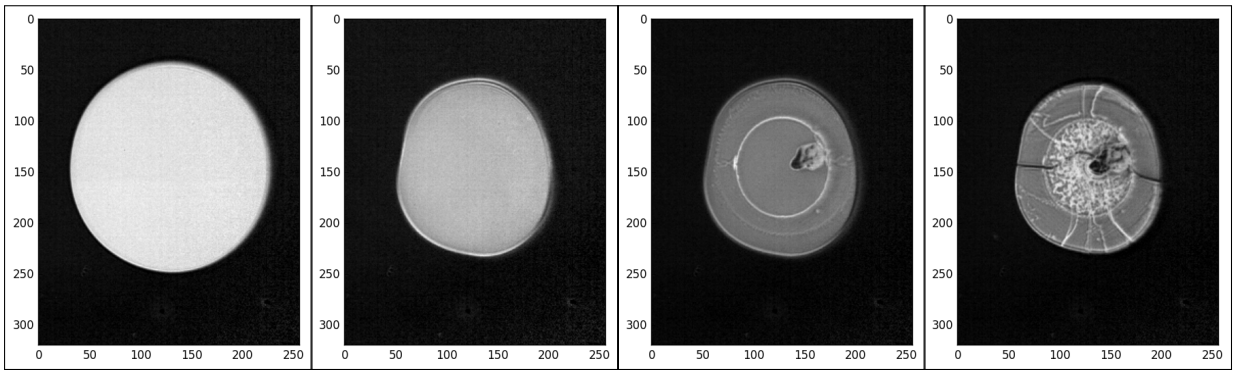
Figure 4.9: Time and Reynolds modes of lid driven cavity during limit cycle ($t \in [1900, 1905]$) in for $Re \in [10000, 10100]$.

a special role of virtually applying a constant offset, it can be referred as a mean mode. Indeed this kind of mode is observed whenever the data has not been centered beforehand, the decomposition "naturally" separate the mean field from the fluctuations. A simple averaging of the data suppresses it and it is often advocated to do so in the literature[7] as it should improve the decomposition. Next, Fig. 4.9b displays well orginized modes, these pairs of modes (2-3, 4-5) are seperated by a phase shift of $\pi/4$ and the frequency of pair 2 is double the frequency of pair 1. This pattern is studied in greater details in section 5.1, yet it interesting to note that the same pattern is observed for multivariate decomposition involving Re as a parameter as well as usual bivariate POD. It is then possible to infer that the time behavior is the same for each Re in the chosen range. At the other hand of the regularity spectrum, one finds Re associated modes in Fig. 4.9a. These modes appear to be a mean to exclude each other from combinations, no clear pattern emerges. This observation indicates low feasibility Re based interpolated ROM.

[7] Also, the data can be normalized in order to improve the decomposition. In fact, these processing aim at recovering the hypotheses behind POD, PCA, etc.

4.3.2 Experimental data : droplets evaporation

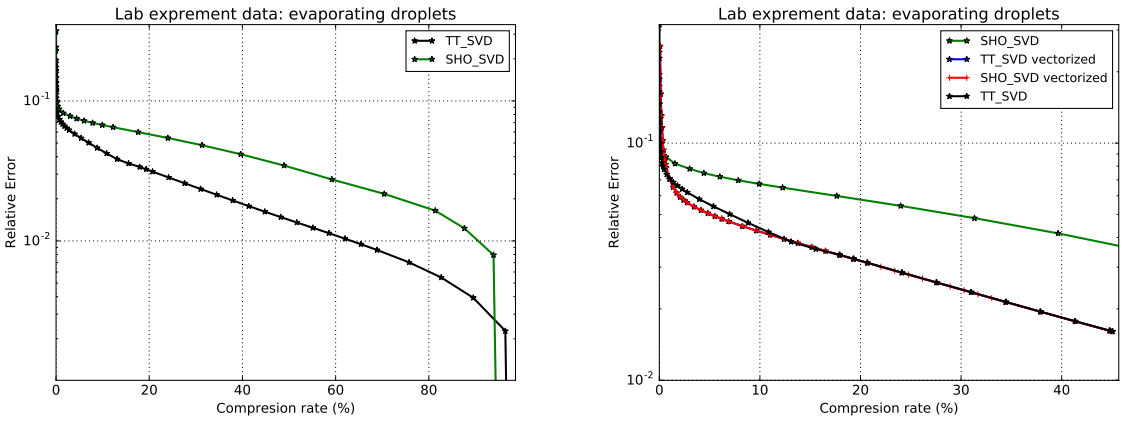
In this second example, a scalar field obtained by a lab experiment is studied. The goal here is to emphasize that very little knowledge of the data is necessary to perform decomposition. This dataset was kindly provided by C. Pradère, from I2M-TREFLE laboratory. It is a study of droplets evaporation during 29 timesteps with recording at 51 different wavelength to evaluate the density field. The camera resolution is 320×356, no further detail on the technology used is required. Finally, a matlab "`.mat`" binary file off 800MB was given. Once again, the remarkable aptitude of python for IO handling is exhibited as only one line of code[8] was required to obtain the $29 \times 51 \times 320 \times 356$ array. Fig. 4.10 pro-



(a) t= 1/29 (b) t= 23/29 (c) t= 24/29 (d) t= 29/29

Figure 4.10: Visualization of 4 snapshots of the density field at the 21st tabulated wavelength. Data kindly provided by C. Pradère (I2M Bordeaux).

vides insight on the phenomenon studied, the circular drop at initial time evaporates and shrinks gradually up to frame 23. Cracks appear at t=24 (different wavelength may not show these cracks) and the droplet is completely shattered at t=29. One may infer that the droplet has solidified but this information (not given by C. Pradère) is not necessary for data decomposition.



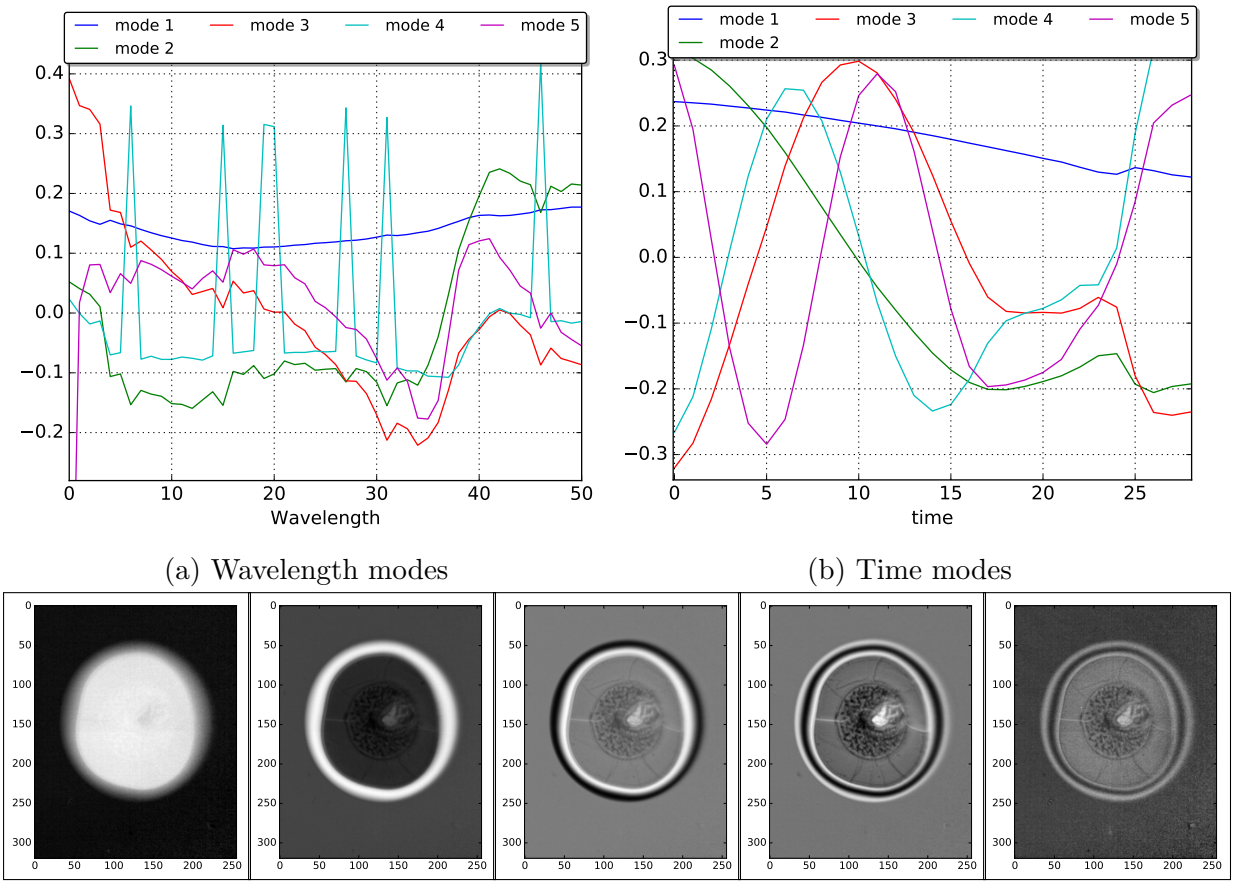
(a) Space as 2 dimensions, tensor shape: $29 \times 51 \times 320 \times 356$.

(b) Space is vectorized, tensor shape: $29 \times 51 \times 81920$.

Figure 4.11: Decomposition of experimental data kindly provided by C. Pradère (I2M Bordeaux). The density is given as a function of time, wavelength and space

[8] A call to the `h5py` library allows "natural" reading of data and selection of the required set/variable before actually loading it into RAM

Given that the data is obtained experimentally and that it is likely that many physics are happening during the experiment, one needs to assess the separability of the array. In the absense of any information about the parameter spaces at stake, for instance we don't know if the snapshots or the wavelength are equispaced, then Frobenius norm based decomposition is used. Fig. 4.11 shows that with both ST-HOSVD and TT-SVD, very little compression is achieved. Indeed, Fig. 4.11a shows that more than 60% compression rate to reach a relative error of 1%. Yet, one can see that the error drops (actually down to machine error) with a compression rate slightly below 100% which means that the density field is represented "exactly" with a slight datasize reduction. Fig. 4.11b proves that attempts at vectorizing data provide no improvement in the error decay rates. This zoomed in view, informs us that a reduction to a few percents of error is attained within a few modes, thus some of the behavior is separable. But the complexity of this phenomenon lies in nonlinear physics, such as transport or phase change, that is known to cause poor convergence of the SVD/POD.



(a) Wavelength modes

(b) Time modes

(c) Spatial modes from vectorized data layout (1 to 5, left to right).

Figure 4.12: First 5 modes study for droplets evaporation experiment obtained through ST-HOSVD.

A brief, analysis of the first 5 modes of time and wavelength is proposed. Fig. 4.12 shows the first five modes of each parameter, space was taken as a single dimension (Fig. 4.12c) since interpreting 1D space modes not relevant given the circular shape of the studied phenomenon. Although Tucker format allow extra-diagonal correlations in the core tensor, a clear link between the first mode of each dimension appear. Contrary to the mean value of LDC example, here we observe a slowly decaying size of the droplet from circular to shrinked ovoid. This behavior is helped by mode two that can be clearly associated with the size reduction in time (green line) as well as in space with the characteristic annular

structure. Mode interpretation among the wavelength (Fig. 4.12a) is more complex and may require additional knowledge. Mode 1 seems to correspond to a mean value whiles modes 2,3 and 5 have in common a visible demarcation around wavelength 35. Mode 4 has a distinct behavior, with marked spikes from one to the next, higher modes (not shown) do not reproduce this erratic evolution. Spatial modes 3 to 5 are clearly associated with cracking of the droplet, as well as part of the shrinkage. Indeed time modes show associated decay oscillations followed by a sharp rise or fall at t=24, the onset of cracking.

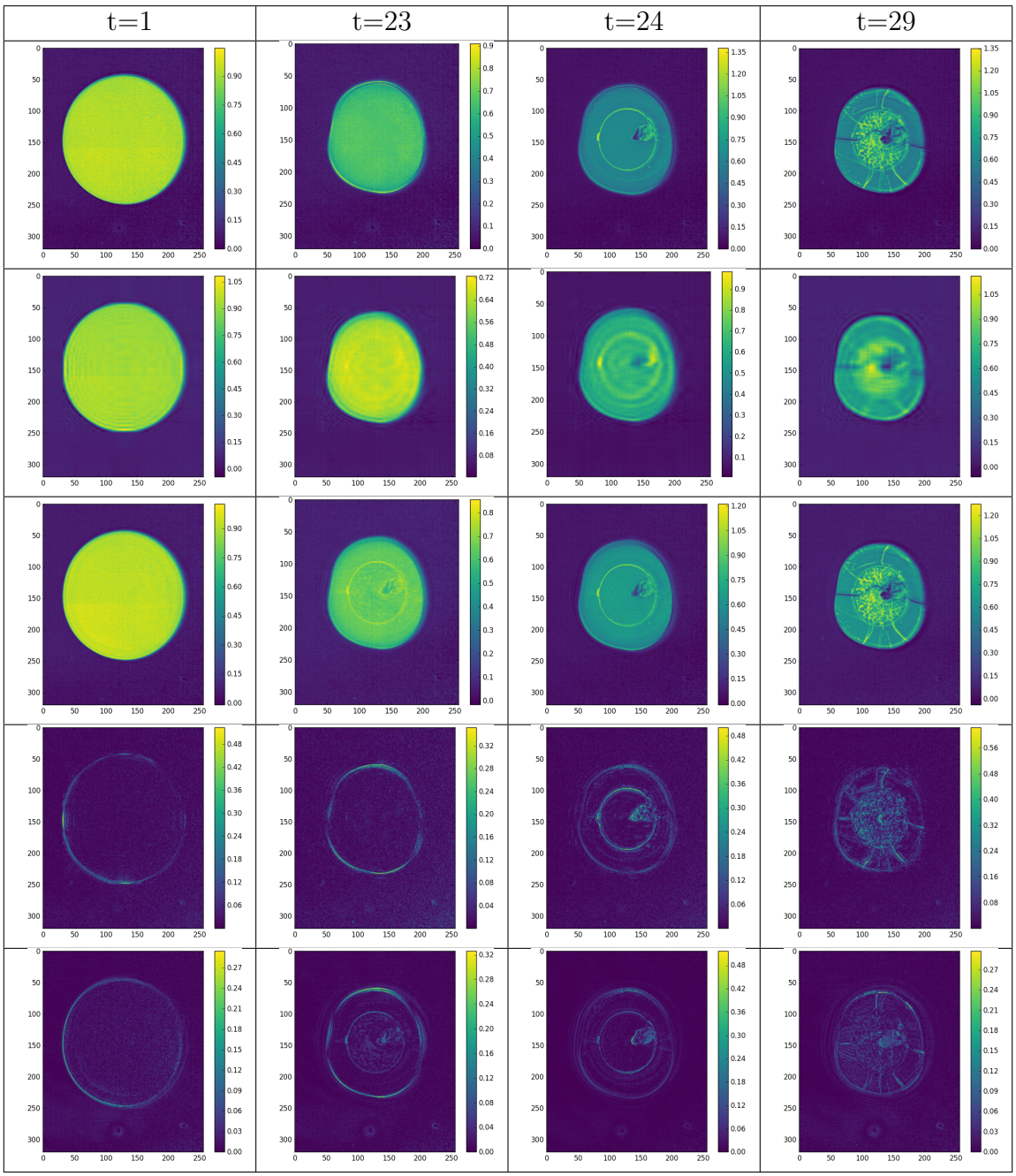


Figure 4.13: Synoptic view of reconstructed decomposition, tolerance, $\varepsilon = 10^{-2}$, wavelength 21. Each line represent a different dataset, namely: original, ST-HOSVD reshape, ST-HOSVD vectorized, difference ST-HOSVD reshape, difference ST-HOSVD vectorized.

Finally, Fig. 4.13 provides a synoptic view of the STHOSVD decompositions with a prescribed error of 10^{-2} in both vectorized and reshaped layout. This means an actual error of 6% for the vectorized layout with a compression rate of 1.5% while its space separated counterpart global error is 9% for a compression rate of 0.3%. This partial choice of low

accuracy high compression is aimed at showing that this kind of representation is sufficient for qualitative analysis. First, in spite of high global error level, the sequence of droplet evaporation is well captured by both methods, the crack appears at the expected frame in each decomposition. The main difference between the two layout lies in the sharpness of the spatial representation, indeed the vectorized approach produces a sharp edged representation while the separated space dimension lead to a "blurry" phenomenon. This is confirmed by the bottom frames, in which on clearly sees that the error is located at high density gradient regions. In conclusion vectorized layout produces less efficient decomposition but allows for a sharp and easy to interpret reconstructed field while the separated space dimensions yields a blurry image, yet with lower global error.

4.3.3 A vectorial simulation : breaking wave

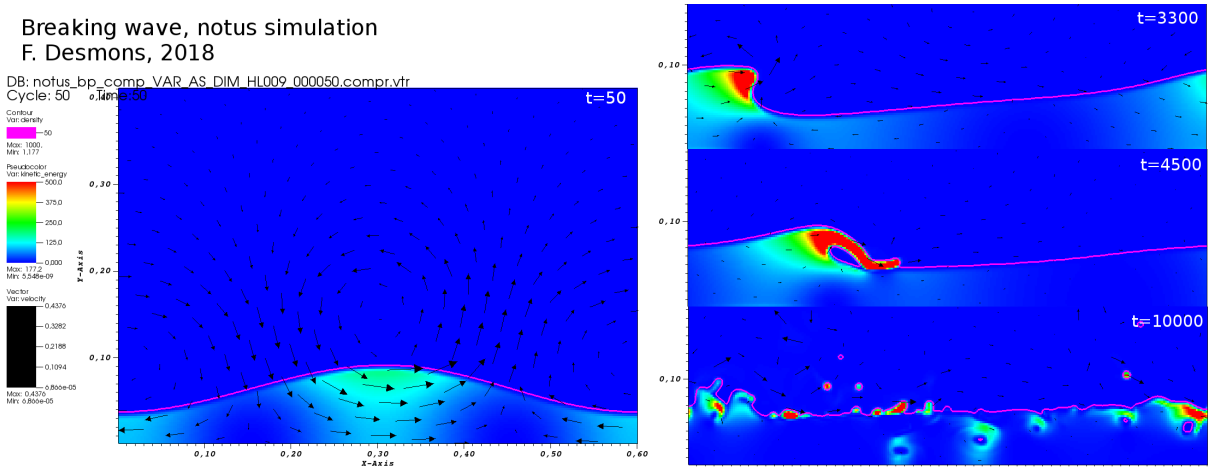


Figure 4.14: Breaking wave simulation computed with notus CFD code, wave height of 9cm and length of 10cm. The wave is going rightward from the initial state (left frame), crosses the periodic boundary (top right), breaks at $t \approx 4500$ follows to an unphysical chaotic state. Pink lines represent the water/air interface, arrows size are proportional to the velocity amplitude and the colormap accounts for kinetic energy.

In this last example, we study a 2D simulation of a breaking wave which provides 5 output variables : density, pressure, vorticity, velocity along each dimension. This simulation was performed by Florian Desmons,a fellow PhD student at I2M-TREFLE, with notus CFD (see boxed description). It is not intended to be a state of the art breaking physics simulation, the goal here is to provide a complex physics two phases flow computed with a validated HPC code.

notus CFD

Notus open-source CFD software is an initiative of Institut d'Ingénierie et de Mécanique - Bordeaux (I2M, Bordeaux University, CNRS UMR 5295) developed since 2015. It is dedicated to the modelisation and simulation of incompressible fluid flows in a massively parallel context. Its numerical framework is the Finite Volume method on Cartesian staggered grids with a methodological focus on interfaces treatment (on going works on fluid-fluid interface advection, surface tension computation, immersed boundary methods, etc.).

Extract from : notus-cfd.org

The Navier Stokes equation with two fluids is solved thanks to a level set methods with a velocity pressure scheme. The spatial domain $\Omega = [0, 0.6] \times [0, 0.6]$ is discretized on a 256×256 cartesian grid, while the time is solved with small times steps which are sampled in 201 equispaced snapshots. The third parameter is the ratio wave height over wave length, the latter being fixed for the whole set of simulation to 10cm, 3 heights are given: 9, 10 and 11 cm. In each case, the boundary conditions are periodic and the velocity field is initiated with an adapted velocity. Finally, the density field is equal to 1000 in the liquid phase and 1 in the gas phase. For stability reasons, the transition is smoothed on a few cells. Simulation with wave height of 9cm is provided in Fig. 4.14 where one can see four typical snapshots of the breaking wave.

Data layout. The previous examples have shown that in spite of providing sharper spatial description, a vectorized space is not the most efficient configuration in terms of storage cost. Additionally, the physics of the studied problem clearly has two seperate domains, air and water which remain in the same region with respect to coordinate Y. Only a small portion of the Y range is affected by phase change. In conclusion, a space separated layout is used. Furthermore, this dataset provides 5 different output fields which are correlated since they solve the same Navier-Stokes equation system. But they possess very different mathematical properties, for instance, density field is representing as sharply as possible an inherently discontinuous field whereas the pressure field is naturally smooth and continuous in spite of following the same interface. The velocity field is represented by two scalar values but has been solved at the same time. Finally, the vorticity field is post-processed from velocity but the field itself is much more sharp due to the rotational operator, thus making decomposition less efficient. In conclusion, two data layouts are studied, both with separated X and Y axes.

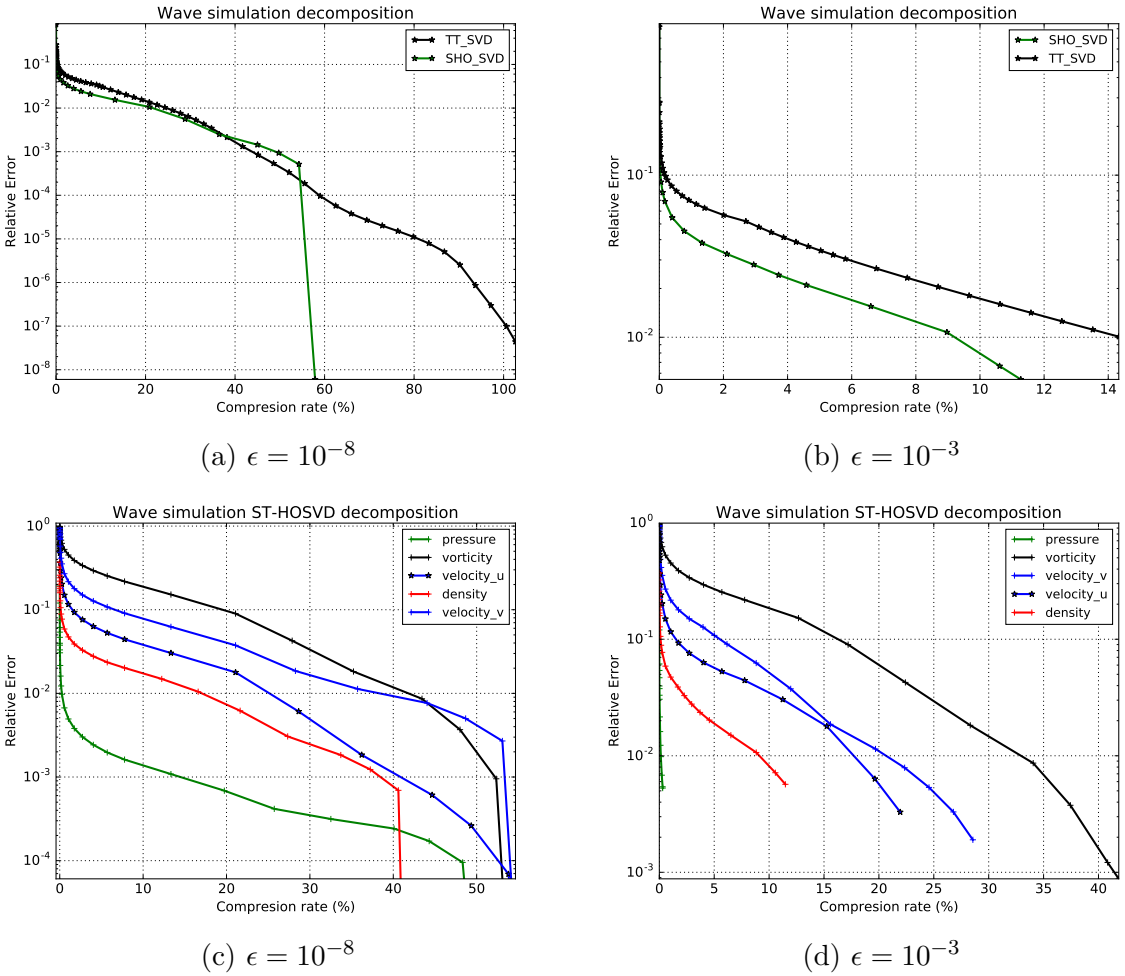
a. Output data for each variables are processed sequentially. Five order 4 tensor of shape $3 \times 201 \times 256 \times 256$ are decomposed.

b. Output data for each variable is assembled into a new dimension that intends to account for embedded correlation among variables. One order 5 tensor of shape $5 \times 3 \times 201 \times 256 \times 256$ is decomposed.

Scalar product. As for any decomposition problem, choosing the base scalar product and associated norm is thought carefully. Time and space support both L^2 and l^2 decomposition while wave height could also accommodate both, it's not clear whether the usual measure is suited for such parameter. Finally for case b., there is no natural measure to integrate over different variables, leaving l^2 as the only option. In conclusion, a full[9] l^2 decomposition is used, with two methdos, namely TT-SVD and ST-HOSVD.

Fig. 4.15 provides the error versus compression rate graphs for layouts a. and b. First we focus on the top frames in which the truncation error of the decomposed tensor is shown versus the compression rate. Separability of the dataset with layout b. sits in the separable range. Once again, a sharp decay is observed for large scale evolution i.e. for error levels down to a few percent. Then a clear inflection is observer around 0.5% compression rate for both method. Yet, it still appears that the error decay follow an exponential trail. It is interesting to notice that ST-HOSVD yeilds the best approximation at low compression levels (see Fig. 4.15d) and represents to machine error the data with a compression rate of 60% as seen in Fig. 4.15c. No such convergence is observed for TT-SVD.

[9]The implementation of `pydecomp` allows mixed scalar product with easy use. Still, it was shown that L^2 and l^2 scalar product produce almost identical decomposition for regular Cartesian grids. Thus, the increased complexity does not seem justified in this context.



(a) $\epsilon = 10^{-8}$ (b) $\epsilon = 10^{-3}$

(c) $\epsilon = 10^{-8}$ (d) $\epsilon = 10^{-3}$

Figure 4.15: Compression of breaking wave simulation data from `notus`. Parameters : 5 output variables, 3 wave heights, $n_t = 201$, $n_x = 256$, $n_x = 256$. Top frames are decomposition with output variable taken as an additional dimension with $n = 5$, bottom frames is the same dataset but each variable is seen as a seperate scalar decomposition problem.

Regarding separate decompositions of variables through ST-HOSVD, lower frames of Fig. 4.15, it is observed in Fig. 4.15c that every single variable is represented to machine error within 50% of the original data size (per variable). It is actually uncommon for complex simulation data to present an "exact" tucker rank i.e. for each variable, machine error is reached for a tucker rank of $r = (3, 201, \approx 130, 256)$. Next, for small truncation error levels, all variable decrease at the same slope, only the depth of the initial drop varies. Fig. 4.15d provides a bigger truncation criterion in order to better grasp the moderate accuracy decomposition. Once again, large differences between variables is observed, with pressure field being extremely separable while the vorticity field occupies the other end of the spectrum. Table 4.3 emphasizes the great variation of ranks among variable for an identical tolerance. In conclusion, if one is interested specifically in an "easily" separable field, then the best choice is to treat variables separately. On the other hand, when interested in several variables, it is a better option to compress all the data together.

Graphs discrepancies. One may notice that these graphs are not exactly the same, this is because the truncation value ε is applied to the ST-HOSVD itself i.e. to each SVD. This leads to some mode combination to disappear from the larger ε although the actual

Field	Rank
density	[3,174,58,147]
pressure	[3,52,14,44]
velocity_u	[3,184,79,256]
velocity_v	[3,179,101,158]
vorticity	[3,195,114,246]

Table 4.3: Breaking ST-HOSVD ranks with the same prescribed cutoff value $\epsilon = 10^{-3}$ (last point in Fig. 4.15d).

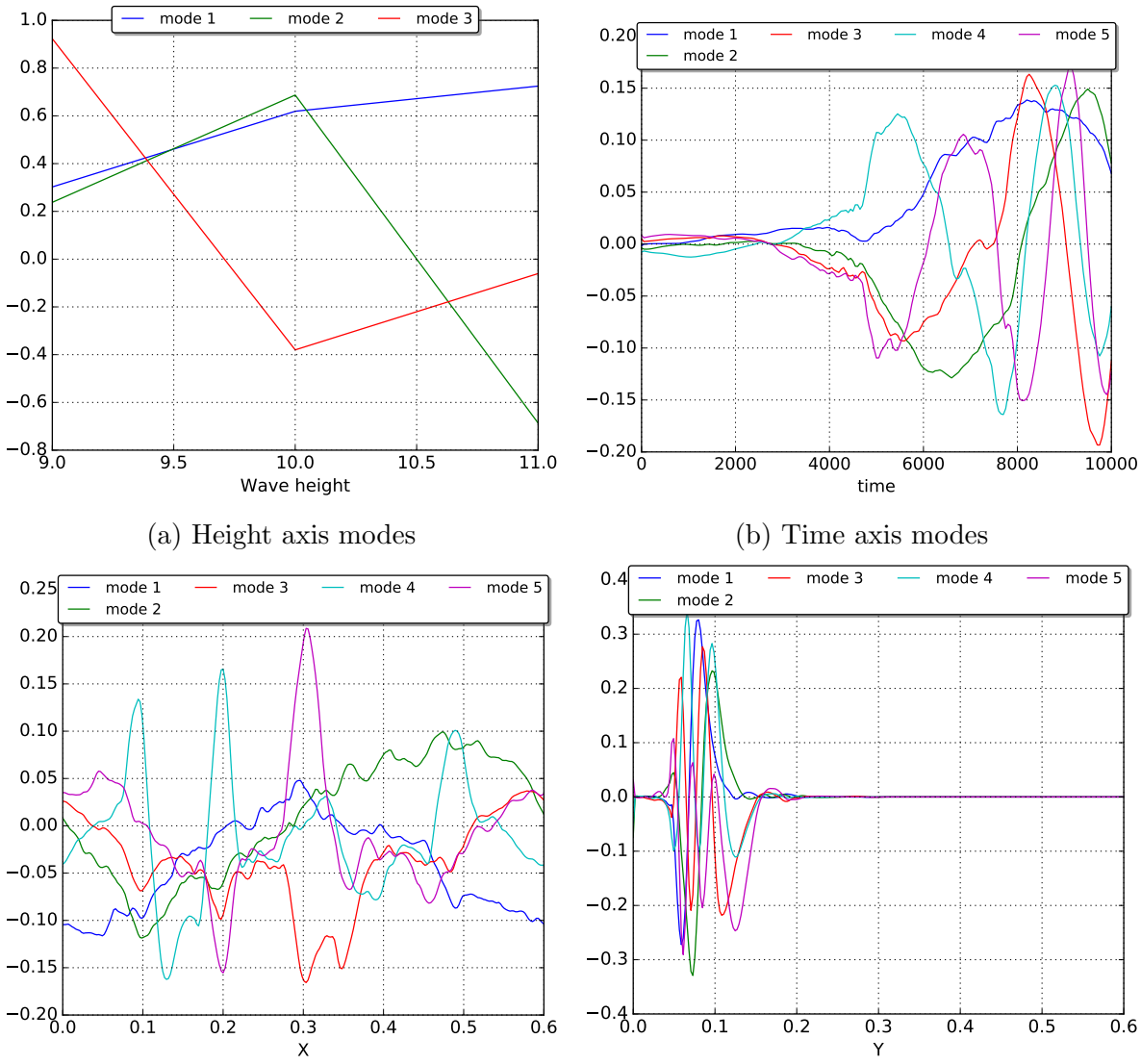
projection norm is of the same order as ε. For instance let us pretend that $\varepsilon = 10^{-3}$ yields a rank (3,7,27,35), there is no warranty that modes (3,**8**,27,32) from the full rank decomposition is associated with a weight $\omega_{3,8,27,32} < \varepsilon$.

Another possible cause to discrepancies between Fig. 4.15c and Fig. 4.15d lies in the building of the graph itself. The retained algorithm increases the rank at the same rate for each dimension while taking care of respecting the available rank. This is justified by the extreme complexity and variability of modes projection weight and the fact that the smallest dimension reach (in my experience) very quickly the original shape size, for instance wave height rank is maximum.

Breaking wave vorticity modes. Given that this example physics is a lot different than the previous ones, it is interesting to look at the first five modes of each dimension (see Fig. 4.16) for the vorticity field. The top left frame, Fig. 4.16a, shows the modes associated with the initial height of the wave. No clear pattern is distinguishable and the sharp variation mostly indicates that they would be better considered as discrimination function rather than modes in the usual sense. Consequently, there is very little prospect for interpolated ROM on this parameter when the user have only 3 instances available. Times modes (Fig. 4.16b) can be interpreted as being activated by the breaking of the wave (time range is approximately [3300,4500]) and further agitation. As expected, space modes along dimensions X and Y produce remarkably contrasting patterns. On one hand, X modes describe global agitation with distint patterns at impact ($x = 0.2$) and splash region ($x = 0.3$). On the other hand Y modes show an intense activity near the interface and close to 0 value elsewhere. The same pattern is observed for other variables (not shown) but vorticity provides the most readable graphs.

Reconstructed fields. Finally, a quick overview of the reconstruction is given by means of the density field and levelset reconstruction. Indeed, this is a very sensitive variable and it is required to capture correctly the interphase for any interpretation of the stored results. Fig. 4.17 shows the same snapshots as Fig. 4.14 where the black line is the original isoline 50 of the density field and the green dotted line is its reconstructed counterpart from ST-HOSVD($\varepsilon = 10^{-3}$). The background color maps the difference between both density fields. In spite of marked error field, the reconstructed levelset fits perfectly with the original one, no bubble is omitted and the shapes are well captured. Still, some parts of the density field are negative (deep blue color in the air corresponds to $\rho < -20$). This is obviously non physical and this issue should be addressed in order to prevent misinterpretation for cases in which the analysis is more complicated.

It should be noted that with this precision of $\varepsilon = 10^{-3}$, it is almost *impossible to distinguish* the reconstructed field from the original mode. Some slight oscillations may be spotted but are easily discarded by the observe as their amplitude is a few percent of



(a) Height axis modes

(b) Time axis modes

(c) X axis modes

(d) Y axis modes

Figure 4.16: The first vorticity modes for separated variables layout.

the maximum field value.

4.4 Standard interpolation techniques for reduced basis ROM

Given the discrete basis that we have obtained in the previous section, it is quite natural to try to build simple interpolation ROM using standard 1D interpolation methods. We do not intend to supersede specialized methods such as the empirical interpolation method (EIM) proposed by Maday et al. [MNPP09] or its discrete version DEIM [Cha08]. Neither do we try to compete with sophisticated approaches such as Grassmann manifold interpolation [AF08, AF11] proposed by Amsallem and Farhat. Indeed this technique relies on the particular topology of EDP solution space to provide accurate parametric interpolation. Here, we simply assume that as a first approximation, standard interpolation techniques such as Lagrange interpolation yield acceptable ROM if the sampling of the solution space is dense enough relative to the solution smoothness. Using 1D interpolation on separated basis is interesting since it prevents the occurrence of beat phenomenon and is very cheap

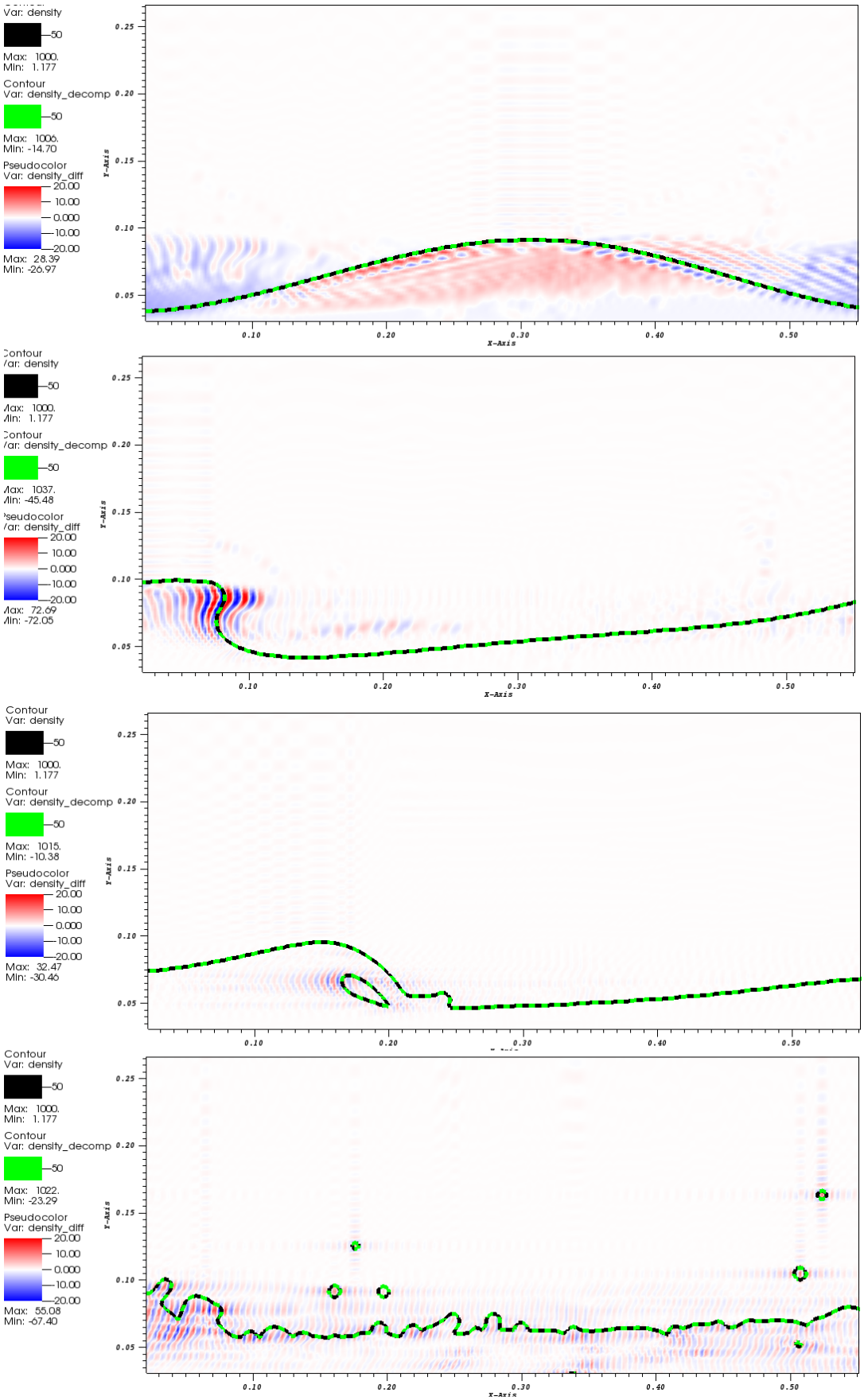


Figure 4.17: Levelset 50 of the reconstructed density field at 4 time steps (same as Fig. 4.14) with the difference field between the original and reconstructed data.

as compared with multidimensional interpolation.

We are interested in the following toy problem for the LDC flow separated in section 4.3.1. The vorticity data $\omega(\boldsymbol{x}, t, Re)$ defined at grid points $\{x_i\}_{i=1}^{N}$, $\{t_j\}_{j=1}^{n_t}$ and $Re = \{Re_k\}_{k=1}^{K}$ has been separated into the following Tucker representation $\omega(\boldsymbol{x}, t, Re) \approx [\boldsymbol{\mathcal{W}}, \boldsymbol{\Phi}, \boldsymbol{A}, \boldsymbol{U}]$ where base matrices form an orthonormal basis[10]. The goal here is to find an approximation of $\omega(\boldsymbol{x_i}, t_j, Re_t)\ \forall i, j$ at target Reynolds Re_t. We propose to interpolate on Re modes U. To do so, one can rely on standards interpolation techniques (and their limitations) such as the few examples listed below. A complete presentation of standards interpolation methods is available in many books including [ARF07].

Lagrange interpolation. Usually the first example in interpolation chapters, this polynomials method is exact at nodes and provides interpolation polynomials of the same order as the number of points. These are defined as

$$l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j} \quad \forall i < N \tag{4.4.1}$$

Then the interpolation polynomial reads

$$\Pi_n(x) = \sum_{i}^{n} y_i l_i(x) \tag{4.4.2}$$

with $\Pi_n(x_i) = y_i$. A complete numerical analysis is available in the literature [ARF07]. The main drawback of this method is known as Runge phenomenon, when the number of interpolation point grows, so does the order of the interpolation polynomial which results in oscillations at the edges of the domain.

Composite methods. In order to overcome Runge phenomenon, a solution is to use composite methods. As Runge phenomenon is observed only when many interpolation points are used, the idea is to cut the domain into small parts with $k + 1$ nodes for Lagrange interpolation and join the obtained interpolations into one. It possesses the following error estimate for $f \in C^{k+1([a,b])}$ for the global interval

$$||f - \Pi_k^h f||_\infty \leq C h^{k+1} ||f^(k+1)||_\infty \tag{4.4.3}$$

where h is the maximum distance between consecutive nodes. Consequently, the interpolation error is low as long as h is 'small enough' in spite of k being small. Obviously, there is a variety of ways to interpolate on the small segments in order to ensure regularity properties. In this thesis, we restrain to linear (2 points intervals) examples and splines.

Splines. Spline interpolation methods are often found in softwares as they provide global continuity properties while exhibiting the inherent stability of composite methods. In this manuscript, the *cubic* splines are chosen as they are the lowest order splines to ensure C^2 approximation and possess good regularity properties. Additionally they are natively proposed in `scipy`[11] which make them very easy to use. Once again, the reader can find more information in [ARF07].

[10] Orthonormality is useful for building stable ROM, if possible, prefer these bases.

[11] www.scipy.org

Table 4.4: Reconstruction RMS error of the interpolated ROM as

Linear	2.94%
Lagrange	1.49%
Spline	1.45%

These methods have been applied to the following LDC dataset, on a regular Cartesian of 257×257 computed with the previous code with 201 equispaced snapshots for $t \in [1900, 1940]$, Re=10000, 10020, 10040, 10060, 10080, 10100 have been evaluated with DNS. $Re_t = 10040$ is removed from the training set and an ST-HOSVD (tolerance of 10^{-2}) is applied to the data. It yields a reduced basis of 5 Re function $\{U_p\}_{p=1}^{5}$ which is interpolated at Re_t. Finally the interpolated ROM is reconstructed at target Re and the RMS error is compared using these three methods following the definition

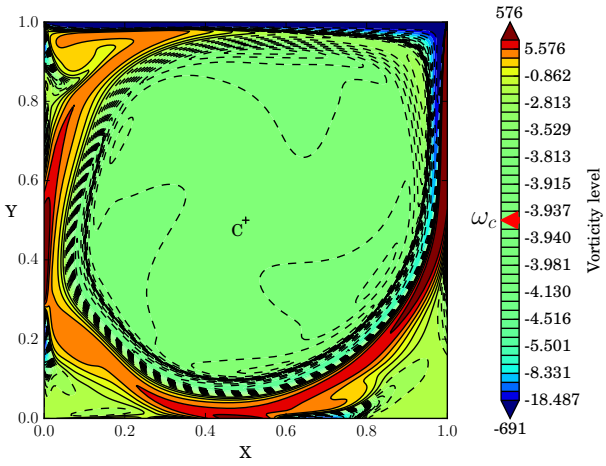
$$E = \frac{||\omega_{DNS}(Re_t) - \omega_{ROM}(Re_t)||_F}{||\omega_{DNS}(Re_t)||_F} \tag{4.4.4}$$

One can see in table 4.4 that the interpolated ROM provides a relatively good approximation of the vorticity field for all three tested methods. Not surprisingly, linear interpolation provides the largest RMS error while Lagrange and cubic spline interpolations perform equally well from this point of view. A better grasp of the ROM reconstructed vorticity field is given in Fig. 4.18 where the top frame is the DNS vorticity field at $t = 1900.199$ and subsequent pairs of frames show ROM reconstructed field for each interpolation method together with the difference map on the right hand side. Again the error map Fig. 4.18c shows that linear interpolation performs worst with marked error near the cavity boundaries. In this region, the vorticity is captured equally well by cubic splines and Langrange interpolation. But one notices that the central portion is represented with much more accuracy by Lagrange interpolation Fig. 4.18d both in shape and actual value than it is by cubic splines (Fig. 4.18f). It is as though the first pair of mode (see Fig. 5.10b), does not have enough weight in the ROM which would explain why a 6 branches structure is noticed for splines. The good accuracy of Lagrange interpolation is attributed to the reduced number of sampling Re points as finer sampling would result in Runge phenomenon. In this context however, Lagrange interpolation represents an efficient and easy to implement method for basis interpolation ROM. It should be noted that this approach provides good stability as long as the underlying interpolation method is stable. It should be noted that running all three steps of decomposition, interpolation and reconstruction of the slice requires less than 10 seconds for the studied case, interpolation (online) cost being virtually nil compared to the others.

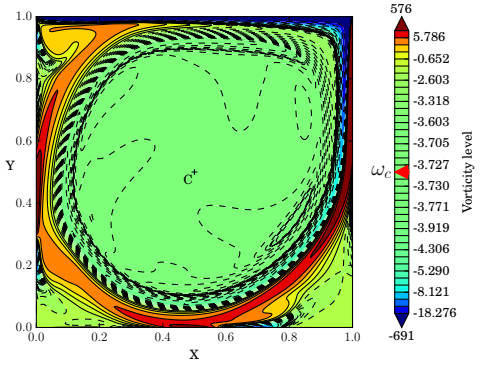
The drawback of this approach lies in the loose sampling of Re parameter, this leads to irregular modes, of which the characteristic are deeply affected by removing one sampling point. This impairs greatly the ability of such ROMs to produce better accuracy outputs. For instance, using much more refined ST-HOSVD decomposition such as $\epsilon = 10^{-4}$ does not improve at all the ROM accuracy since the maximum number of Re modes was already attained for $\epsilon = 10^{-2}$. To overcome such difficulties, one needs to rely on methods that inherently capture the complexity of the PDEs solution space which are for most cases curbed manifolds.

Conclusion

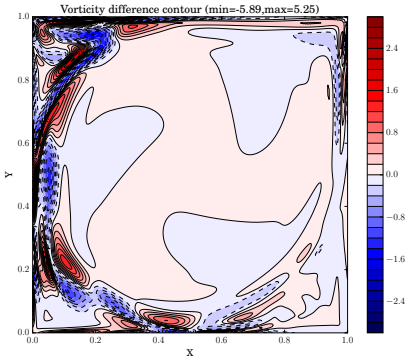
In this chapter, I tried to answer to a common question in mechanics laboratories and especially for numerical physics:



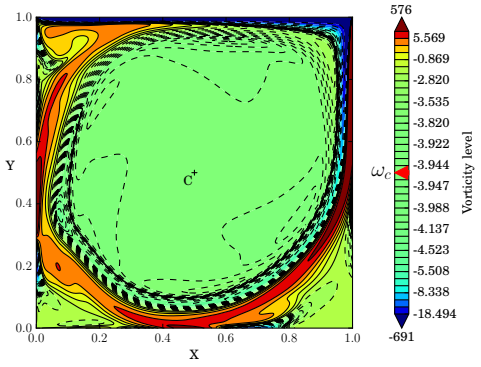
(a) DNS



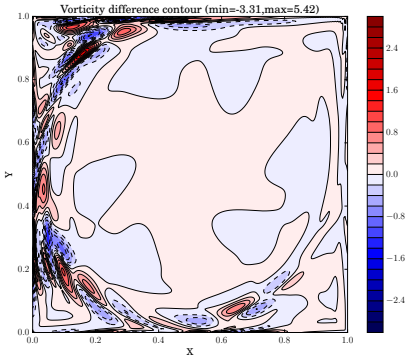
(b) Linear



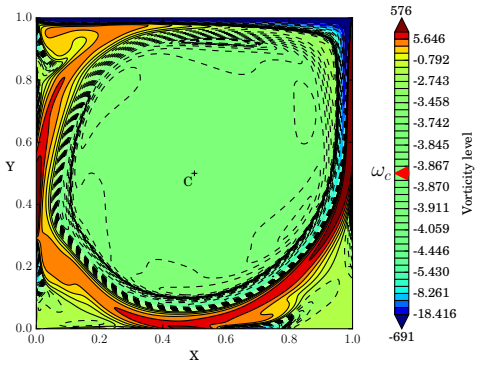
(c) $\omega_{lin} - \omega_{DNS}$



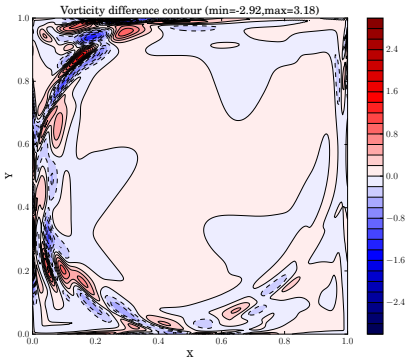
(d) Lagrange



(e) $\omega_{Lagrange} - \omega_{DNS}$



(f) Cubic



(g) $\omega_{cubic} - \omega_{DNS}$

Figure 4.18: Reconstructed vorticity and DNS vorticity field for target Re=10040 with donor points at Re = 10000, 10020, 10060, 10080, 100000 at $t = 1900.199$.

"Which one of the numerous decomposition method should be used to reduce mechanics data?"

To do so, I have developed two decomposition libraries, the second one, `pydecomp`, takes advantage of python numerous libraries for scientific computing, visualization and data I/O handling. Benchmark cases are proposed to test and compare each of the available methods : PGD, RPOD, T-HOSVD, ST-HOSVD and TT-SVD.

In section 4.2, these benchmarks have been put to use. The clear conclusion is that PGD cannot be used as a multidimensional decomposition method for it is extremely slow and compression power is excessively poor. Yet, it should not be dismissed as it makes a relatively efficient 2D iterative methods which allows the user to compute only the required modes. The T-HOSVD has been dismissed as its results are indistinguishable from ST-HOSVD while being several times slower for large datasets. RPOD has also been classified as not pertinent for three reasons. First, its decomposition performance is far poorer than TT-SVD and ST-HOSVD, second, the computing time is much higher than its contenders. Finally, the recursive mathematical nature of the methods translation to code is not natural and leads to flat trees that are slow to scan. Extensive discussion on scalar product selection has lead to the conclusion that default should be "blind" eulerian scalar product, especially for Cartesian grids. But some cases could benefit from L^2 scalar product such as Gaussian quadrature points, or contexts in which the physics is well known and requires a special scalar product such as integrating a vector field.

In section 4.3, a close attention was given to the two most efficient methods for tackled actual data. Two scalar fields were used, one from an experiment, the other from a DNS simulation. The experimental data, taken from a droplet drying with 4 parameters (time, wavelength, 2D space) was found weakly separable with both methods. This is attributed to the nonlinear nature of the studied physics. Yet the reconstructed field for a tolerance of $\varepsilon = 10^{-2}$ seems sufficient for qualitative interpretation. The high error levels seem to lie in the cracks representation as shown brief analysis of the modes. The DNS data was chosen to present regularities that were accurately captured. In each of these methods two data layouts for space decomposition were studied, one separates X and Y dimension while the other vectorize so that space is viewed as a single dimension. For both datasets, the separate dimension produces lower compression rate for a fixed error level but the inherent bi-dimensionality of the structured is captured with less accuracy (to the human eye) in spite of lower error. Indeed mode combination leads to oscillations that reduce when the rank is increased. The last example was a breaking wave simulation computed using `notus CFD`. It proved the versatility of `pydecommp` implementation in handling data from several sources with different characteristics. I have shown that different variable from a single simulation present remarkably diverse separability levels. As expected, the smoother the field, the more separable it is. Another, layout question was raised for this example, one may wonder whether these output variables should be treated as a distinct problems or as a single variable. Once again, there is no definitive answer and the user must adapt the layout to its need. The global decomposition allows easy handling but the compression rate is dominated by the least separable variable. Thus, for this case, a distinct processing of each variable is preferable. Moreover, we have seen that, surprisingly, this dataset tensor has a finite Tucker rank at machine error.

Finally a simple interpolated ROM of reduced basis (obtained by ST-HOSVD) has been proposed with standard interpolation techniques. This ROM produces solution with 1% RMS while being very simple to implement. The main drawback is that we were not able to improve the accuracy below this mark since the Re basis functions are not smooth. Improving interpolation may require a finer approach as we will discuss in chapter 6.

Summary and conclusion on data decomposition

In this first part of the manuscript, we have first presented the bivariate *a posteriori* data reduction technique. It was observed that they are mostly equivalent from an analytical point of view. However the algorithms display strong differences that may be used to improve computing efficiency depending on the studied problem.

Then a detailed presentation of the tensors, their representation formats (Canonical, Tucker, TT and HT) and the associated decomposition techniques was proposed in formalisme as close as possible to CFD community standards. They are well suited to reduce numerical simulation data. However it should be noted that some properties of the flow may be lost.

Subsequently, the multivariate function approximation/decomposition was studied proposing two methods, the PGD which is best suited for directly solving problems on a reduced base and the RPOD, a natural extension of POD with recursive bivariate decomposition. The constructed recursive tree structure differs notably from the tensor approximation that we have adapted to the continuous framework.

Finally, most of these methods were implemented (`pydecomp`) and numerical test were conducted. They showed the dramatic variation of CPU time and number of mode efficiency between these multivariate methods. For both the continuous and discrete frameworks, the main result of this analysis is that sequentially truncated Tucker decompositions present the most efficient for $d < 5$ while it is superseded by TT decompositions for higher d. These efficiency comparison have been performed with compression rate against approximation error that provides a fair ground when the rank does not account the same memory use. Finally, in view of building ROM, one should consider the orhtonormality property of the bases obtained via these methods, especially numerically.

To close the first part, a summary of the formats and their decomposition is given. A special care is given to provide insight for both continuous and discrete format. Anytime POD or SVD appear, they can be switched as we have shown that they are equivalent approaches with different scalar products. We have also noted that the correlation approach (POD and SVD by EVD) are much faster than full methods by limit the accuracy of the decomposition to the square root of machine precision, e.g. with double precision $\sqrt{10^{-16}} = 10^{-8}$.

Canonical Format

$$\begin{aligned} \text{cont.} \quad f(x_1, ..., x_d) &\approx \sum_{k}^{d} \prod_{i=1}^{d} X_i^k(x_i) \\ \text{disc.} \quad \boldsymbol{\mathcal{F}} &\approx \sum_{i=1}^{r} \bigotimes_{\mu=1}^{d} \tilde{\boldsymbol{x}}_\mu^i \end{aligned}$$

This decomposition is obtained by iterative approximation enriching algorithms such as PGD and ALS (see algorithm 2 and 3). Storage cost is linear in d ($\mathcal{O}(drn)$) but it was found that these decompositions are numerically inefficient. Additionally, convergence is not ensured. *These methods are not recommended if data approximation is the sole objective.*

Tucker

$$\begin{aligned} \text{cont.} \quad f(x_1,...,x_d) &\approx \sum_{k_1=1}^{r_1} \cdots \sum_{k_d=1}^{r_d} w_{k_1,...,k_d} \prod_{i=1}^{d} X_i^k(x_i) \\ \text{disc.} \quad \boldsymbol{\mathcal{F}} &\approx \sum_{k_1=1}^{r_1} \cdots \sum_{k_d=1}^{r_d} w_{\boldsymbol{k}} \bigotimes_{\mu=1}^{d} \tilde{\boldsymbol{x}}_\mu^{k_\mu} \end{aligned}$$

Many algorithm are available to compute Tucker format decomposition (HOOI, T-HOSVD) but ST-HOSVD has been found to be the most efficient in the production stage. At the moment, it is also the best decomposition method for small number of dimension $d \leq 4$ regarding compression versus approximation error. Storage cost is quasi-linear in d ($\mathcal{O}(r^d + drn)$), the main drawback is the number of the exponential growth of the core tensor ($\mathcal{O}(r^d)$). Converge of the method is certain together with quasi-optimality property and production of orhtonormal basis. *Preferred compression method for $d \leq 4$. Good candidate for multidimensional ROM.*

Recursive-POD

$$\text{cont.} \quad f(x_1,...,x_d) \approx \sum_{k_1}^{R_1} \cdots \sum_{k_{d-1}}^{R_{d-1}(r_1,...,r_{d-2})} X_1^{r_1}(x_1) X_2^{(r_1,r_2)}(x_2) ... X_d^{(r_1,...,r_d)}(x_d)$$

Not a format, rather a recursive generalization of POD. The recursive tree structure can also be used with any bivariate method (SVD, PGD,...) and allows easy truncation to prescribed accuracy. But this structure severely impairs orthogonality of the basis (only within leaves). Storage cost is rather difficult to evaluate but converge (thus compression rate) is good for regular functions. Numerically speaking, it is far less efficient than TT and ST-HOSVD methods. *A natural extension of POD with limited performance and a unique data structure.*

Tensor Train

$$\begin{aligned} \text{cont.} \quad f(x_1,...,x_d) &\approx \sum_{k_1,...,k_{d-1}} G_1(x_1,k_1) G_2(k_1,x_2,k_2) \cdots G_d(k_{d-1},x_d) \\ \text{disc.} \quad \boldsymbol{\mathcal{F}}(i_1,...,i_d) &\approx G_1(i_1) G_2(i_2) \cdots G_d(i_d), \quad \forall (i_1,...,i_d) \in \mathcal{I} \end{aligned}$$

This ordered product of order 3 tensors format allows easy implementation of decomposition for large dimension problems i.e. $d \geq 5$. Storage cost is d-linear in $\mathcal{O}(r^2 dn)$. Many decomposition algorithms including natural `TT-SVD` and blackbox methods such as `TT-DRMG-cross` that require evaluation of the field a few points. It is so efficient in large dimension that increasing artificially the number of dimension, a process known as tensorization, has become a new strategy to process large scale data. The main drawback of TT is the partial orhtonormality of the modes in transfer tensors ($\boldsymbol{\mathcal{G}}_i$) that is problematic for ROM building. *The most efficient method for $d \geq 5$ that does not provide orhtonormal bases for ROM.*

Hierarchical Tensor All previous formats and decomposition algorithms are subcases of this format. Formalism is much more complex and does not come with innovative algorithm. Then it was not studied numerically. *A promising format that contains other formats. Review and implementation by Grasedyck et al. [GKT13].*

Part II

Complex fluid dynamics and Reduced Order Modeling

We have seen in the general introduction that the need for cheap numerical models has remained undimmed in spite of the explosion of computing power. Finding the simplest equation to capture the physics of a problem is the first step. But extreme simplification of equation leads to model that inherently capture limited features. Typical example are steady state solutions or RANS are typical fluid dynamics examples while LES is able to represent much more complex phenomena. Still, they remain enable to describe properly complex flows at high Reynolds number such as singular lid driven cavity, for this kind of problem high accuracy direct numerical simulation is necessary [Sen13, SLV09, SVB09, Sen12]. These methods require intensive computing, for instance, 2D simulations with relatively coarse grid (257×257) requires 2 days on a single modern CPU to reach the limit cycle that is studied in chapter 5. In this context, adding a third dimension implies to use a parallel implementation to maintain computing time below a week.

These constraints prohibits any attempt at controlling or optimizing the flow that would require cheap evaluation of the solution field. These problems are one of the reasons motivating the use of ROM. Some of these methods, such as PGD [CKL13, CLA+09] try to solve problems and constructed a reduced basis at the same time until the required precision is reached. These approaches are generally very efficient at solving elliptic problems [FN11, FHMM13] but results hyperbolic equations are precarious. Another way to build ROM is to first compute reduced basis. There are many ways to obtain these basis, such as dynamic modes decomposition (DMD) [Sch10] or Fourier transform but the most used one in fluid dynamics is POD [Lum81, Sir87] as it is now routinely used for data analysis (see chapter 1) and ROM building. Indeed this decomposition provides, among the many possible bases [IR98], an orthogonal basis of the functional space in which the solution problem lives.

Since the POD bases are orthogonal it is possible to use Galerkin projection to build ROM. From the 1980s attempts at building such POD Galerkin ROM have flourished [Sir87, DKKO91, CVVI98, Fah01, Ber04] with relative success. Indeed, in this approach, the weak form EDP is solved against test function in the selected basis. But, in order to reduce the size of the problem, one has to truncate the basis to a relatively small rank which means, in the case of fluid flows, that the small structures are lost as we have seen in chapter 1. Yet these structures correspond to turbulence and viscosity for high Re number whose role is to dissipate energy. This is why Iollo et al. [ILD00] showed this approach is inherently unstable. Thereafter many stabilisation techniques have been proposed [BBI09, ANR09, IW14] and continue to be an active field of research [BGW15, LCLR16, SR18]. Generally, this approach has motivated substantial amount of work crystallized in various books in recent years [QR13, QMNI] under the name reduced bases (RB) popularized in the early 2000s [MMPR00, PRV+02]. Hesthaven, Rozza and Stamm have recently published a book [HRS16] that covers certified RB.

ROMs can also be built using interpolation in the parametric space of arrival of PDEs. Indeed, one can build a set of data for several parameters with FOM and later ask the data base for a point that was not previously sampled and interpolate to this new location. Given the large size of the full data, brute force multidimensional interpolation through standard techniques (Lagrange, splines,...) is not an option. Additionally, dynamic systems, even if they are relatively similar may produce beat phenomenon [LBS+18]. Consequently, numerous methods were proposed to build interpolated ROMs for nonlinear problems. The empirical interpolation method (EIM) has been introduced in 2004 by Barrault et al. [BMNP04]. The idea here is to sample the parametric space by greedy algorithm [MNPP09] that is particularly well suited for non-linear problems. It was later adapted into a discrete version (DEIM) using POD modes as a basis instead of samples [CS10]. Amsallem and Farhat proposed a Grassmann manifold interpolation

ROM [AF08, ACCF09, AF11, AZW15] that rely on the structure of PDE arrival space. The idea is to project the solutions from the manifold to a tangential plane to perform the interpolation (by standard means) and the return to the manifold. In his thesis manuscript [Mos18], Mosquera reviewed this family.

Part I has provided numerous tools for decomposition of bivariate and multivariate data. In the second part of this thesis manuscript, these decompositions are used to produce bases in the context of fluid dynamics. It is organized as follow. Chapter 5 proposes a complete analysis of a complex CFD flow through, inter alia, POD decomposition. The aim is two folds: first show that complex CFD problems require specific care at all stage, involving a detailed analysis and should lead to reasonable ROM expectation. Second we shall show decomposition methods such as POD can help this analysis and consolidate conclusions. Then, relying on previous chapters, an interpolated ROM is proposed with a new physics based interpolation methods for instability flows in chapter 6.

Chapter 5

Complex flow analysis using decomposition

Contents

5.1 Singular lid driven cavity: analysis of flow behavior with high accuracy direct simulation 134

5.1.1 Governing equations and numerical methods 135

5.1.2 Dynamics of singular LDC flow 136

5.1.3 Vorticity dynamics and polygonal vortex in LDC 137

5.1.4 Multiple Hopf bifurcations 138

5.1.5 Numerical sensitivity of the problem 140

5.2 POD analysis 143

5.2.1 Analysis through POD modes 144

In the era of ROM, the study of flow behavior is easily forgotten. ROM researchers usually compare their work with validated simulations. Due to the sheer difficulties inherent to reduced order modeling, only limited attention is devoted to full order simulation. Thus, it is not infrequent that commercial softwares be used for inputting data to the ROM building process. I support the idea that the accumulation of error incurred by ROM should lead researchers to give extra-attention to input data. Such a preliminary work should focus both on the physics studied and the scientific computing methods used to produce the data. Often, this step has been overlooked because of the simplicity of problems tackled (elliptic problem are less sensitive to these considerations). In the context of fluid dynamics, most application involved highly non-linear Navier Stokes equation or derived hyperbolic equations. The complexity of such flows is illustrated in this chapter by a single example: the singular lid driven cavity (LDC) flow. Despite its basic geometrical setup, the flow is known to produce Hopf bifurcations. The first section will describe the richness of the LDC physics and characterize it for further ROM building.

It is also well known that this analysis can be enriched though processing and decomposition. Historical methods like Fourier transform, eigenvalue problems can be enriched by POD as it was specially developed for fluid dynamics [Sir87]. The second section of this chapter will provide such POD analysis of the LDC flow. Higher order modes interpretation are more delicate, thus they have been restrained to straightforward comments in section 4.3.

5.1 Singular lid driven cavity: analysis of flow behavior with high accuracy direct simulation

In this section, we present the detailed study of singular lid driven cavity flow. It has already been presented in previous sections (1.4.3.1 and 4.3.1) and a detailed literature review has been provided in [LBA+18, LBS+18]. Thus, a short overview of the state of the art is given.

State of the art. 2D flow in a square LDC (of side L) is a popular problem to test new algorithms for incompressible Navier-Stokes equation (NSE) due to its unambiguous boundary conditions, coupled with its very simple geometry. As the lid is given a constant-speed translation (U), this gives rise to corner singularities on the top wall, as depicted in Fig. 5.1.The role of such singularities is to give rise to Gibbs' phenomenon, as reported by pseudo-spectral computation of NSE [AQV02].While it is possible to compute steady flow at low Re by various methods including lowest order spatial discretization, it is not so at higher Re, where the flow displays inherent tendencies to unsteadiness. One of the central activities in studying the problem of LDC is to show that the onset of unsteadiness is related to flow instability. Viewed in this perspective, the primary goal is then predict the correct equilibrium flow for global instability studies. Such questions can be studied through linear instability analysis but this branch is not explored in this work. In DNS one directly proceeds to obtain the unsteady flow. In this work we rely on the same sixth order CCD scheme as in [SLV09, SVB09], to discretize both the convection and diffusion terms of the vorticity transport equation. In this work, they indicate creation of a transient polygonal vortex at the core, with permanent gyrating satellite vortices around it.

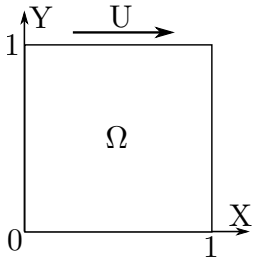


Figure 5.1: Schematic view of the LDC

A steady solution has been reported by many [BG09, ECG05, GGS82] for Re far exceeding the values reported in the literature for the first Hopf Bifurcation (Re_{cr1}), due to the excessive diffusion of the discretization. On the other hand, simulations of full time-dependent NSE [GGH90, OI11] reveal that the flow loses stability via a Hopf bifurcation with respect to increasing Re. Critical Re_{cr1} and frequencies obtained from DNS and eigenvalue (linear instability) analysis do not match and such differences are noted for different DNS results too for various reasons, some of these will be explained here.

It is shown in [SVB09, OI11, SVS11] that Re_{cr1} depends upon the accuracy of the method and how the flow is established in DNS. Physically, impulsively started flow is ideal to study the dynamics, as it triggers all frequencies at $t = 0$ [SLV09, SVB09]. Such an analysis is preferred and is superior to normal mode analysis of eigenvalue approach. Multiple Hopf bifurcations have been reported in the literature by researchers for LDC flow. Authors in [APQ02] have talked about a second bifurcation, while Sengupta et al. [SVS11] have described multiple Hopf bifurcations for flow in LDC. In recent times, Girault et al. [GGC12] have talked about multiple Hopf bifurcations for LDC flow using compact scheme. Thus, in this section a description of multiple Hopf bifurcation is based on overall dynamics of the flow field is proposed.

Appearance of unsteadiness with variation in parameter value(s) studied by bifurcation theory [Sey94] is due to flow instabilities [Sen12]. Various researchers noted different value of Re_{cr1}: 8031.93 in [SO03], 7972 in Cazemier *et al.* [CVVI98] using a finite volume method. Bruneau and Saad [BS06] noted this to be in the range of 8000 to 8050 using a third order upwind scheme, using (1024×1024) grid. The roles played by different numerical sources in triggering flow unsteadiness by Re_{cr1}, that explains the scattering of reported Re_{cr1}. Of

specific interest are for methods using very high accuracy methods which report relatively high values of Re_{cr1}.

The role of various sources of errors, including aliasing error for flow inside LDC has been described in [SVB09]. Here we will discuss the roles of other sources of errors. As these simulations are extremely sensitive to operating conditions, we rely only on sequential computing in order to capture the weak transient core vortex when the major sources of errors are removed. This aspect of hyper-sensitivity of omputed solution on background disturbance is further exploited here to explain why Re_{cr1} are different for different numerical methods.

5.1.1 Governing equations and numerical methods

DNS of the 2D flow is carried out by solving NSE in stream function-vorticity formulation given by,

$$\nabla^2\psi = -\omega \tag{5.1.1}$$

$$\frac{\partial\omega}{\partial t} + (\vec{V}\cdot\nabla)\omega = \frac{1}{Re}\nabla^2\omega \tag{5.1.2}$$

where ω is the non-zero out-of-plane component of vorticity for the 2D problem. The velocity is related to the stream function by $\vec{V} = \nabla\times\vec{\Psi}$, where $\vec{\Psi} = [0\ 0\ \psi]$. Reynolds number is defined by L and the constant lid velocity, (U), which are also used as length and velocity scales for nondimensionalization. This formulation is preferred due to inherent solenoidality of the velocity and vorticity for 2D flows. The numerical methods and the dynamics of the flow for $Re = 10000$ are given in greater details elsewhere [SLV09,SVB09] and is not repeated here.

Equations (5.1.1) and (5.1.2) are solved subject to the following boundary conditions. On all the four walls of LDC, $\psi = constant$ is prescribed which helps in satisfying no-slip condition; the wall vorticity is $\omega_b = -\frac{\partial^2\psi}{\partial n^2}$, with n as the wall-normal co-ordinate chosen for the four segments of the cavity to obtain the boundary vorticity. This is calculated using Taylor's series expansion at all the walls with appropriate velocity conditions at the boundary segments. The top lid moves horizontally with a unit nondimensional velocity, with all other walls as stationary. To solve the discretized form of Eq. (1), Bi-CGSTAB method has been used here, which is a fast and convergent elliptic PDE solver [dV92]. The convection and diffusion terms are discretized using the sixth order accurate NCCD method [SLV09, SVB09], which obtains both first and second derivatives simultaneously. All other details about NCCD and other compact schemes can be also found in Sengupta [Sen13] and hence are not reproduced here. For time advancing Eq. (2), four-stage, fourth-order Runge-Kutta (RK4) method is used that is tuned to preserve dispersion relation. The NCCD scheme has been analyzed for resolution and effectiveness in discretizing the diffusion terms along with the dispersion relation preservation properties for 1D convection equation [SLV09, SVB09]. It is noted that the NCCD method is efficient, providing high resolution and effective diffusion discretization. Additionally, the method has built-in ability to control aliasing error. The only drawback of NCCD scheme is that it can be used only with uniform structured grids. All computations are performed with nondimensional time-step of $dt = 0.001$. The final limit cycle behaves in a similar fashion, when time step is changed. Only the instability of the limit cycle appears at different time range with change in dt. A (513×513) grid is used for most of this work but some additional set of computations using a finer grid with (513×513) points are presented. Vorticity time series at a sampling point, qualitatively remains the same, with only the mean value shifted by

a small fraction. The sampling point location being at (0.95,0.95) has been thoroughly justified in [LBA+18].

5.1.2 Dynamics of singular LDC flow

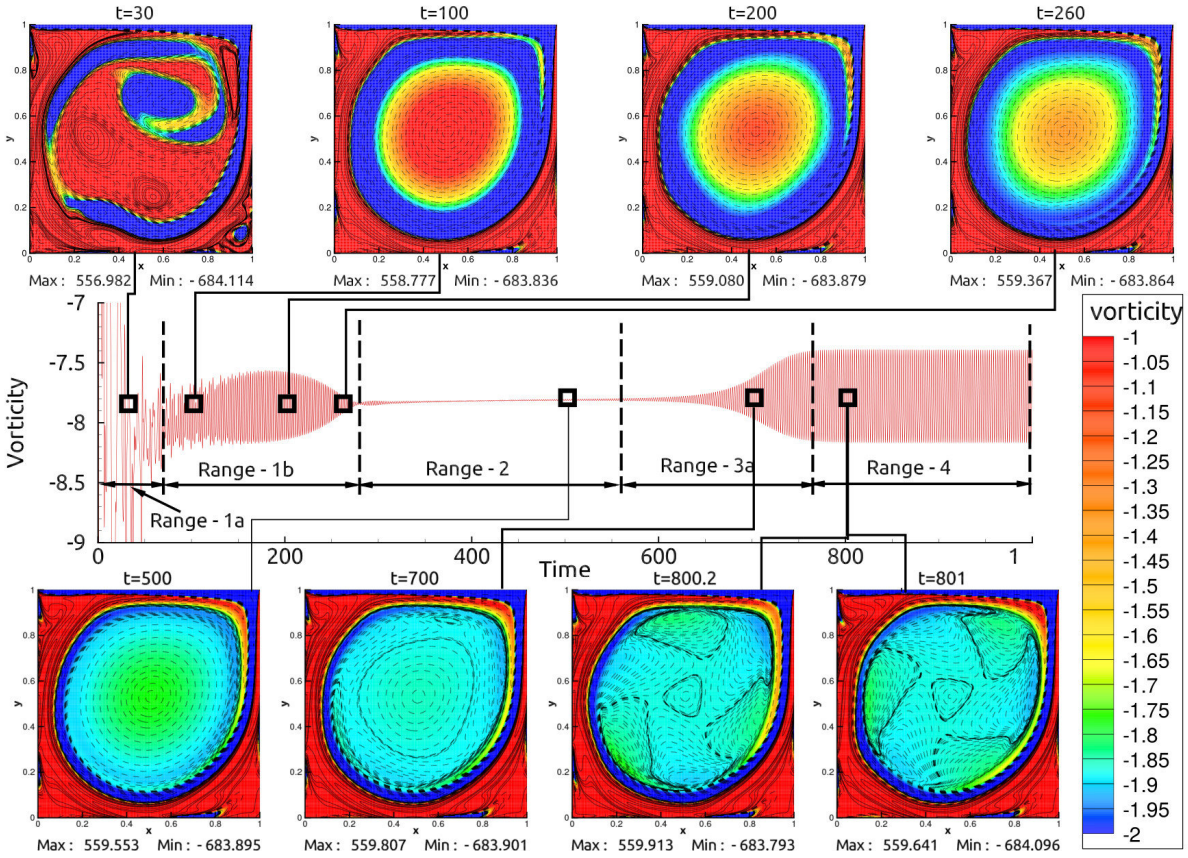


Figure 5.2: The vorticity time series at the sampling point $(x = 0.95, y = 0.95)$ obtained for $Re = 8800$ with vorticity contour plots shown at the indicated time instants. Solution of Navier-Stokes equation is obtained using (257×257) grid.

In this section, the dynamics of singular LDC is presented. To do so, we rely on Fig. 5.2 for $Re = 8800$, which shows the vorticity time series at ($x = 0.95$, $y = 0.95$) in the central frame. In the time series, we have identified various regimes of time-variation. For example, in Range-1a of Fig. 5.2, plotted vorticity displays high frequency transient variations, followed by banded relatively lower frequency variations of the vorticity in Range-1b. The time series shows the decay of the signal near the terminal time of Range-1b, the vorticity fluctuation reduces and settles down to a steady value and which is maintained throughout in Range-2. This period is followed by Range-3a, where the vorticity variation displays growth and which is presumably due to linear temporal instability. Finally, in Range-4 one notices nonlinear saturation of the growth noted in Range-3a. This is the typical variation of vorticity with time for lower Re cases, which are above Re_{cr1}.Range-4 is where the dynamical system settles down to its limit cycle.

The study of a wide Reynolds range has brought to light the variety of behaviors produced by singular LDC. A reduced overview is proposed here as an extensive analysis is proposed in [LBA+18]. Here we want to highlight the main features and infer Hopf bifurcation from the time series of vorticities near the top right corner (0.95,0.95). With our numerical setup, the first Hopf bifurcation happens undoubtedly for $8660 < Re_{cr1} <$

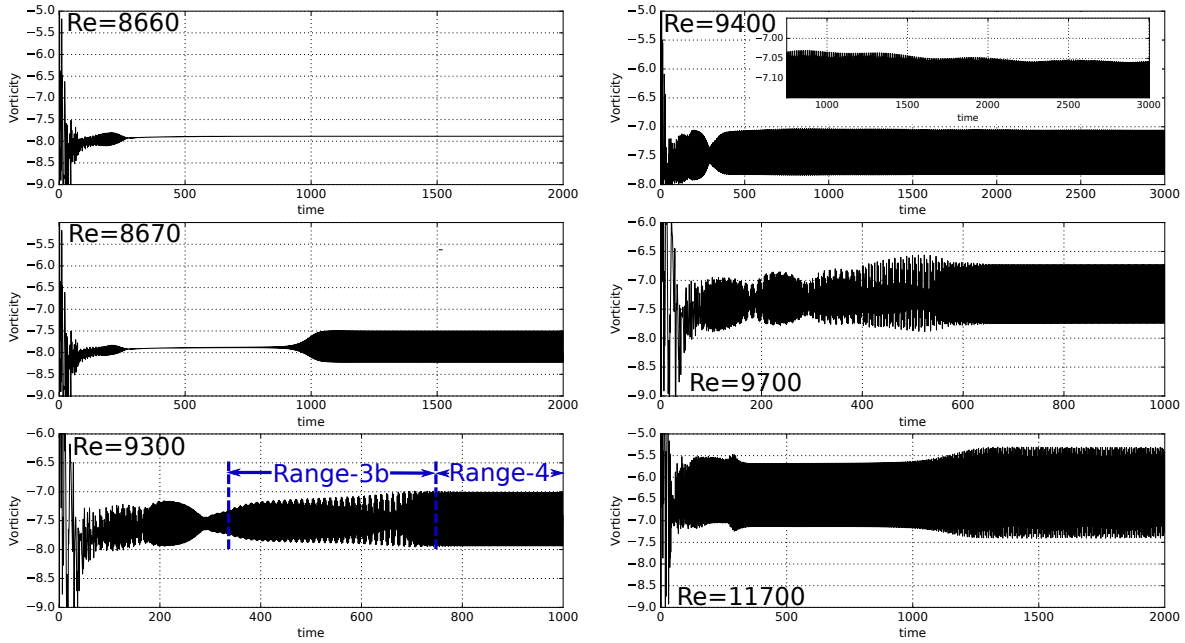


Figure 5.3: The vorticity time series for a point located at $x = 0.95$, $y = 0.95$, near top right corner for the displayed Reynolds numbers, obtained from solution of unsteady Navier- Stokes equation.

8670 as the steady state observe for Re=8660 in Fig. 5.3 is replaced by a stable limit cycle for Re=8670 after instability ends range 2. Increasing Re leads to more unsteady flows as shown for Re=9300, range 2 has disappeared and a new range (3b) is observed with large pulsations. $Re = 9400$ displays a peculiar pattern bypassing previously mentioned ranges to offer an special limit cycle which will be emphasized with spectrum analysis. This value marks a demarcation with higher Re exemplified by Re=9700. Finally, in the highest Reynolds range, limit cycle are subject to pulsations with many vibration modes at stake.

5.1.3 Vorticity dynamics and polygonal vortex in LDC

From the time series shown in Fig. 5.3 for different Re's at the stable limit cycle stage, we have noted the feature of periodicity of the solutions in the final limit cycle. Here, we investigate further about the flow field for $Re = 10300$ to describe the flow evolution in terms of vorticity dynamics. In Fig. 5.4, we show the vorticity contours inside the cavity at the indicated time instants, while the vorticity time series at $(x = 0.95, y = 0.95)$ is shown as the central panel in Fig. 5.4, to understand the choice of the time instants.

In the early stages of flow evolution, the inner core develops in conformity with the shape of the cavity, due to the action of the wall jet impinging near the top right corner. Thus, the lighter shaded contours shown in the form of a rounded rectangle, while the inner contour lines morph into a circular shape, as noted at $t = 200$. From the time series, one notes this stage to belong to beyond the early transient, where the coherent motion corresponds to an apparent neutral stage which is followed by decay of the disturbance. This continues up to $t = 280$, when the time series indicate the termination of decay and beyond this time, the disturbance once again grows. The vorticity contours show two distinct layers with sharp gradient and this motion continues, as shown in the frame for $t = 660$, where the gradient is really sharp. In subsequent flow evolution, the outer layer transforms into satellite vortices while the inner core shrinks to the triangular vortex,

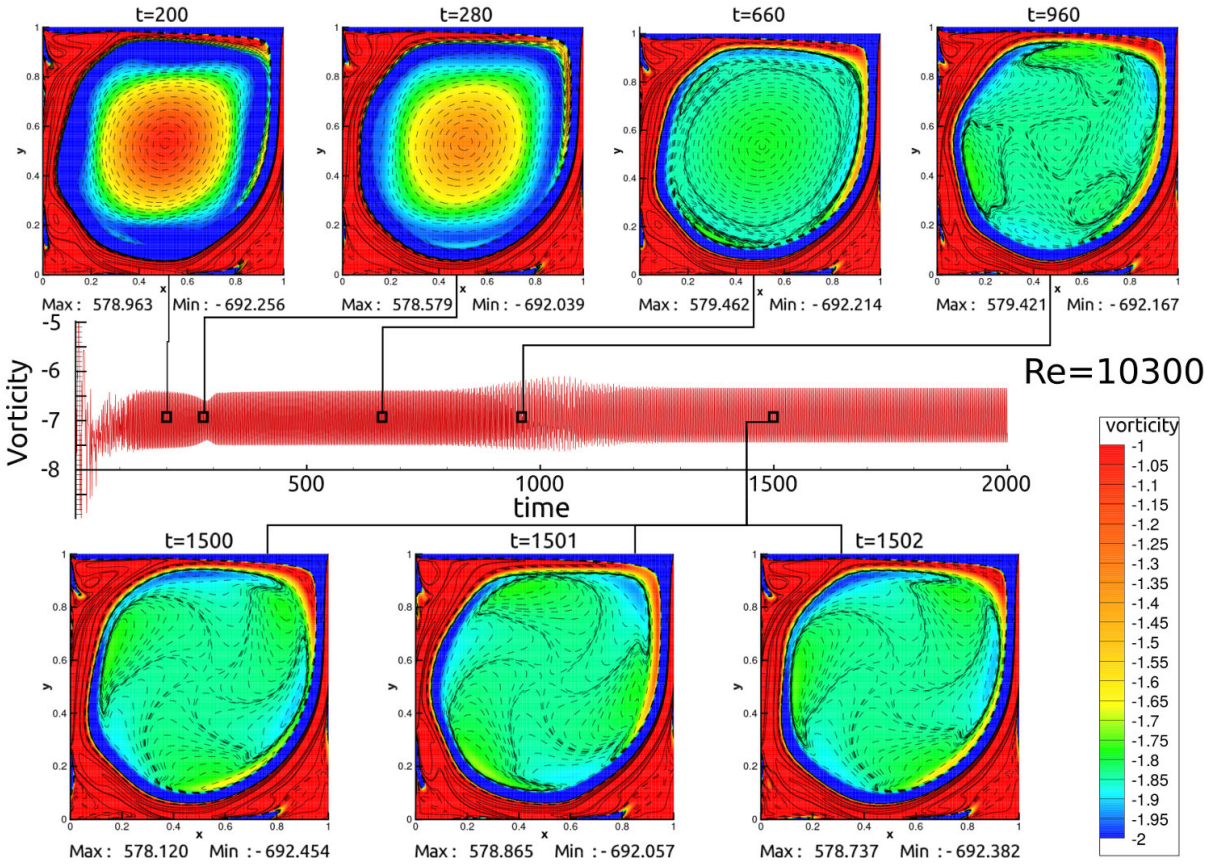


Figure 5.4: Vorticity contour is shown at different time instants from start till attainment of limit cycle for $Re = 10300$. Time series of the vorticity at point $(0.95, 0.95)$ is shown in the center.

as noted in the frame for $t = 960$. Such triangular vortices have been shown earlier for $Re = 10000$ [SLV09, SVB09] and it is noted here also. The triangular core vortex forms after the linear stability phase, only once the nonlinear saturation has taken place. Hence, one can conclude that its presence is essentially due to nonlinear dynamics of the flow field guided by the presence of six gyrating satellite vortices. However, with passage of time the central core vortex looses strength and identity. Thereafter, one notices these six gyrating satellite vortices to rotate about the center of the cavity. This is the terminal state of the limit cycle. One such cycle is shown in the bottom three frames.

5.1.4 Multiple Hopf bifurcations

The vorticity time series described in section 5.1.2 indicate different qualitative dynamics for different Re and that in turn is suggestive of multiple bifurcations in the range of computed solutions. Here, we address bifurcations for the LDC flow based on DNS performed following an impulsive start. To do so the limit cycle amplitude is studied first, then the analysis of the frequencies property allows a better comprehension of the underlying mechanisms.

5.1.4.1 New equilibrium state via stable limit cycle

The amplitude of the limit cycle A_e is defined as half of the maximum excursion of the vorticity time-series describing a constant width envelope, by sampling the vorticity at

$(0.95, 0.95)$. Different time evolution at the sampling point for different Re are presented in Fig. 5.3. For some higher Re cases, computed flow field display significant modulation even when the flow is computed up to $t = 2000$ and above. The Stuart-Landau model states that $A_e^2 \propto |(Re - Re_{cr1})|$ for the limit cycle cases with single dominant mode and this is useful for the flow past a circular cylinder approximately. Correspondingly, Fig. 5.5 displays the plot of A_e^2 as a function of Re for the range $8660 \leq Re \leq 12000$ obtained using a grid with (257×257) points. Unlike the nonlinear dynamical systems for bluff bodies, here the Hopf bifurcation [Sey94] starts very sharply, as shown in Fig. 5.5, which occurs between $Re = 8660$ and 8670. Each zoomed view in Fig. 5.5 shows a segment in which relation between Re and A_e^2 is compared with its linear variation. The linear regression coefficients can be found in Table 5.1.

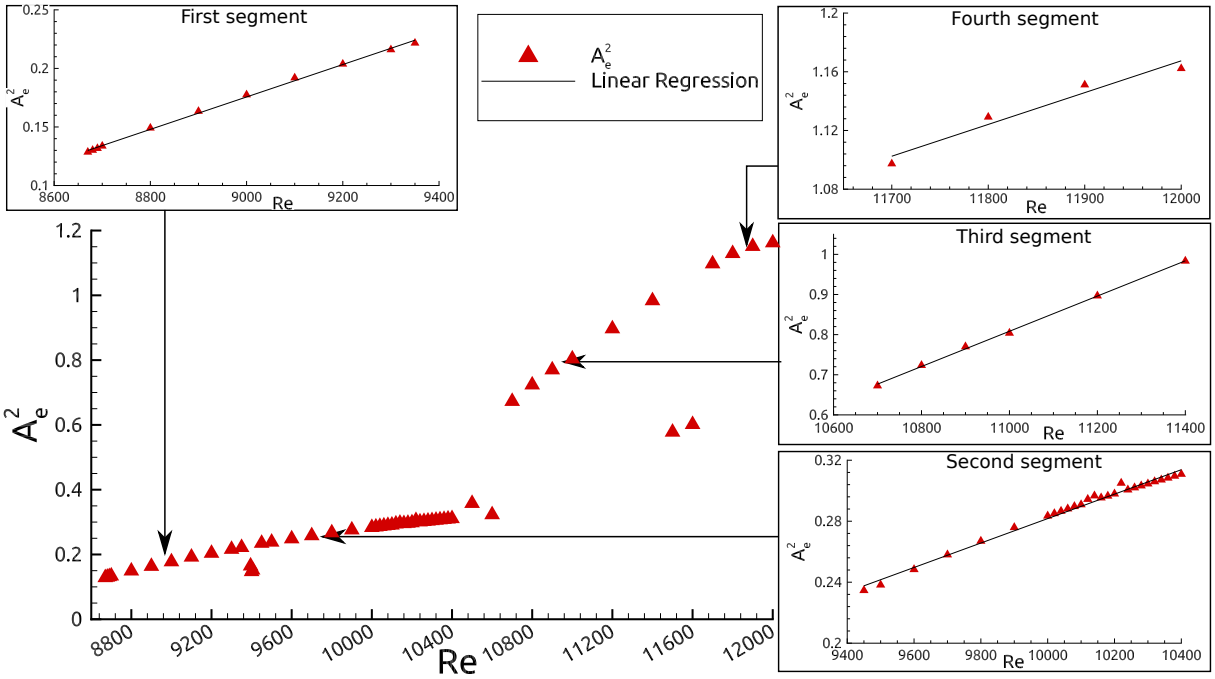


Figure 5.5: Multiple Hopf-bifurcation shown with respect to the vorticity time series data shown for Fig. 5.2. All the simulated Reynolds numbers data are used to plot the amplitude of the final stable limit cycle data against Reynolds number.

Table 5.1: Coefficients of linear regression equation of the form : $A_e^2 = aRe + b$ with regression correlation coefficient R.

Segment	a	b	R^2
1	1.384672e-4	-1.070575	0.998
2	8.017815e-5	-0.520059	0.992
3	4.389279e-4	-4.019874	0.999
4	2.166881e-4	-1.432799	0.956

In the range $8670 \leq Re \leq 9350$, one can see in Fig 5.5 that the linear regression fits the data well. This is confirmed by the value of the regression coefficient (R^2) being really close to 1. The amplitude then suddenly drops around $Re = 9400$, as noted in Fig. 5.5. To ascertain the correctness of this value, additional simulations have been performed for

$Re = 9350$, 9395, 9405 and 9450 and all these data are marked in the figure. It is noted that the value for $Re = 9350$ falls on the linear segment shown to the left of $Re = 9400$. For $Re = 9450$, the amplitude belongs to the next linear segment which ends at $Re = 10400$, as shown in Fig. 5.5 in the second box. However it should be noted that the correlation coefficient is lower on this range, mainly due to its larger extent. Another natural break in the curve is noted between $Re = 10500$ and 10600. Once again a linear segment is plotted for the data points for $Re = 10700$ to 11400. $Re = 11500$ and 11600 show a particular behavior in the higher Re range since A_e^2 values fall abruptly and then the amplitude again rises sharply at $Re = 11700$ defining the fourth bifurcation. A new range up to 12000 is presented in the fourth box of Fig. 5.5, however the correlation coefficient is low, implying that A_e^2 does not vary linearly with Re.

It has been noted [SSS10,SVS11] that the presence of such discontinuities is indicative of multiple Hopf bifurcations in (A_e^2, Re)-diagram, as in Fig. 5.5. The fact that the flow behaves qualitatively different in different range of Re is indicative of discrete change in A_e^2 with respect to Re, as indicated in Fig. 5.5. Along with such changes in the physical plane, one would expect to notice qualitative changes of the spectrum of the time series already shown in Fig. 5.3. From Fig. 5.5 and Tab. 5.1, one can infer the presence of four such Hopf bifurcations. In order to provide a better understanding of the phenomena at work here, the next sub-section will focus on spectral analysis of the vorticity time series at point $(0.95, 0.95)$.

5.1.4.2 Frequency spectrum analysis

In Fig. 5.6, we show few Fourier transforms of the time series shown in Fig. 5.3. Fourier analysis is applied over the last 100 cycles, i.e., after the stable limit cycle is reached. In order to provide accurate plots, the average on that time span has been removed from each time series. It is clear that for $Re = 8800$, the dynamics is governed mostly by three harmonics, with subsequent ones being more than a decade lower than the lowest of these top three frequencies, as noted in Fig. 5.6. The frequencies and pattern are identical up to Re=9400 (see table 2 of [LBA+18]) which marks the second Hopf bifurcation. For this Re, the stable limit cycle is not reached replace by triplets of frequency spikes which are attributed to interact with the first low amplitude spike. As noted in Fig. 5.5, the flow behavior above $Re \approx 9450$ resembles flows noted for lower post-critical Reynolds numbers. This is clearly seen in Fig. 5.6 for $Re = 9800$ with six peaks in the spectrum. For the higher $Re = 11700$ shown in Fig. 5.6, one notices a large numbers of spectral peaks with more than one dominant comparable peaks. This leads to pulsating limit cycles observed in Fig. 5.3.

5.1.5 Numerical sensitivity of the problem

The singular LDC problem is very sensitive to numerical setup. In this section we discuss two major issues affecting the solution, namely start-up conditions and grid sensitivity of our numerical method.

Influence of grid resolution Computations have been performed on two different grids, i.e., (257×257) and (513×513), in order to assess effects of grid on the simulation. Figure 5.7a displays time series for two different grids which clearly behave differently. On the one hand, the coarse grid exhibit a secondary instability around $t = 1200$, that leads to the final limit cycle. Because of finer wall resolution, calculated wall vorticity is higher for the finer grid calculation. Yet, the numerical excitation caused by sources

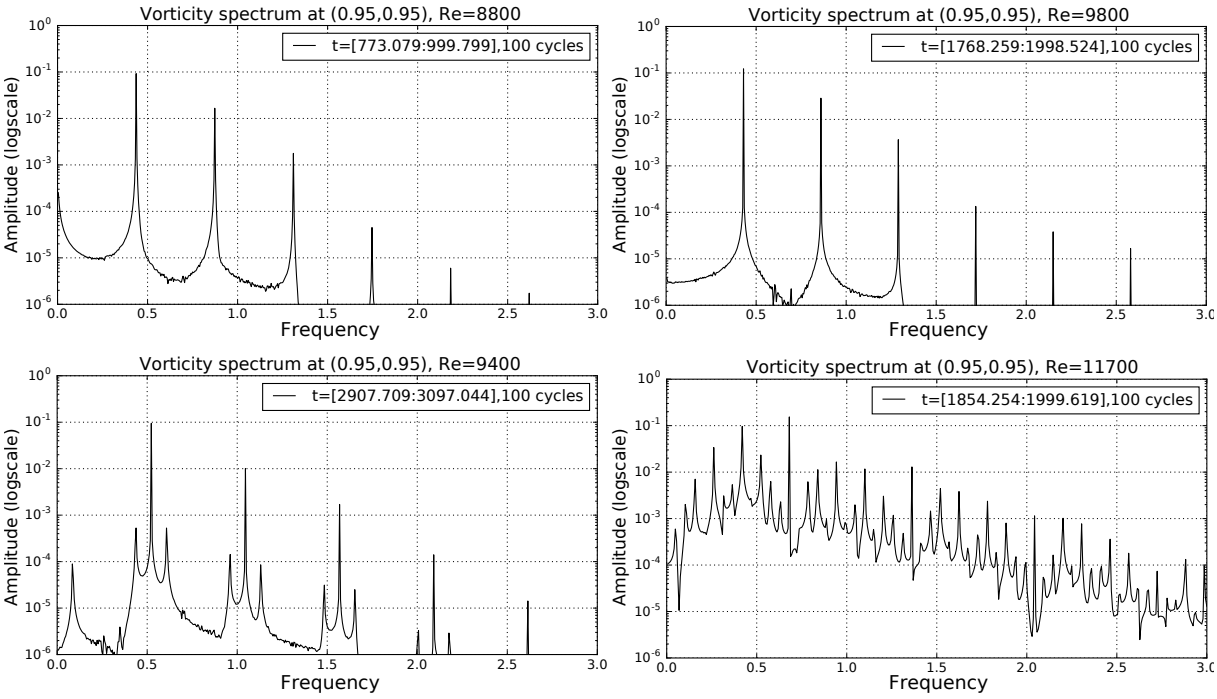
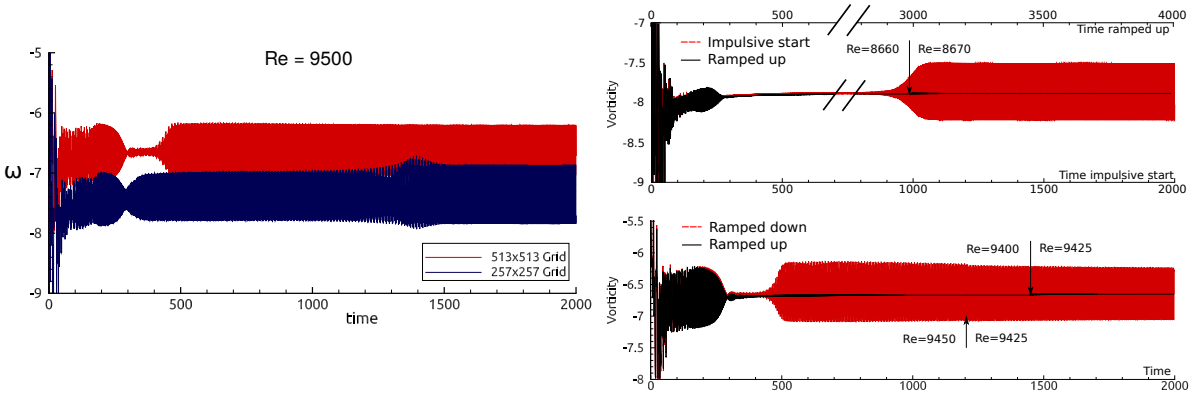


Figure 5.6: The frequency spectrum of the vorticity time series shown for all the simulated Reynolds numbers, for the solution obtained from unsteady Navier-Stokes equation and the data are for $x = 0.95$ and $y = 0.95$.

of error is lower for the finer grid. As a consequence, both the mean and fluctuation of disturbance vorticity is lower for the finer grid, which causes upward shift of the mean vorticity line, i.e., reduction of mean vorticity of disturbance. No secondary instability is seen for the finer grid and still a similar limit cycle is reached with marginal difference in amplitude and frequency of the fluctuating component of vorticity. Moreover, final state is stable for $Re \leq 9400$ when computations are carried on the finer (513×513) grid, i.e., $Re_{cr1} \in [9400, 9450]$. It emphasizes that the flow is driven by the receptivity aspect of the problem, with coarser grid (and less accurate numerical methods) having larger excitation due to implicit error, shows early onset of first Hopf bifurcation. This will be further discussed in the subsection 5.1.5.1.



(a) Grid resolution. (b) Impulsive vs. ramped Startup

Figure 5.7: Numerical sensitivity of the singular LDC problem.

Effect of start-up conditions The top sub-figure of Fig. 5.7b depicts the time series stored for $(0.95, 0.95)$ on (257×257) grid for $Re = 8670$ with two different initial conditions. The dashed line corresponds to the usual impulsive start whereas the solid line corresponds to the solution obtained by ramping up from $Re = 8660$ equilibrium solution. We note that the projected solution starting from lower Re remains quiescent (negligibly small variations), while the solution started impulsively shows non-zero values at the sampling point.

The bottom sub-figure of Fig. 5.7b is for (513×513)-grid in the vicinity of the bifurcation obtained for this grid near $Re = 9400$. Two different start-up cases are presented : (a) when the solution is obtained for $Re = 9425$ starting from an equilibrium solution obtained for $Re = 9450$ and (b) when the initial solution is projected from the case of $Re = 9400$. For the latter case, the vorticity field does not show any disturbance, while the former case shows significant disturbance vorticity. This justifies, *a posteriori*, the use of impulsive start-up which is known to excite all modes of oscillation simultaneously by equal magnitude.

5.1.5.1 Computational bifurcation analysis: Is there a universal critical Reynolds number for primary bifurcation?

In introduction, we have noted that different researchers have reported different critical Re_{cr1}, ranging from $7763 \pm 2\%$ to 10,500, with a marked clustering around Re_{cr1} in the vicinity of 8000. As stated earlier, the chosen NCCD scheme is known to achieve near spectral accuracy. Also Sengupta et al. ([SLV09, SVB09]) have reasoned that the trigger for the unsteadiness is the aliasing error originating near the top right corner of the LDC, while the truncation, round-off and dispersion error is extremely negligible. Thus, one needs to apply manually a disturbance to the vorticity field. Here, we chose a pulsating vortex ω_s at a location $r_0 = (0.015625, 0.984375)$ whose spread is defined by the exponent α,

$$\omega_s = A_0(1 + \cos(\pi(r - r_0)/0.0221)) \sin(2\pi f_0 t) \quad \text{for } (r - r_0) \leq 0.0221$$

where in the presented results here we have taken $f_0 = 0.41$ for different amplitude cases.

For $Re = 8660$ and below, we start with $A_0 = 1.0$. Once the excitation is started, one notices the vorticity to grow and saturate to a limit cycle. After, the limit cycle is set up, the excitation source is switched off,yet the limit cycle continues. The saturated limit cycle amplitude for decreasing Reynolds numbers are shown in Fig. 5.8 along with the unexcited cases (shown by hollow triangle facing towards left, up to $Re = 8670$) for the sampling point at $x = 0.95, y = 0.95$. Below this $Re = 8025$, increasing strength of pulsating vortex does not produce stable limit cycle. We note that the imposed vortical perturbation in the limiting amplitude case of $A_0 = 10.0$, constitute a perturbation level of around 20 percent of the maximum vorticity in the domain. Thus, this computational exercise indicate that the first critical Reynolds number (Re_{cr1}) lies between 8020 and 8025 and similar range of the value noted by many researchers as noted in the previous paragraph.

Conclusion

In this subsection, the direct numerical simulation singular LDC flow has been performed. Thanks to high accuracy NCCD schemes, it was shown that impulsively started flows go through a series of Hopf bifurcation starting from $Re_cr1 \approx 8665$. The complex behavior of the flow Figs. 5.2 and 5.4 is efficiently captured by monitoring the vorticity in the vicinity of the top right corner. Analysis of these time series and their spectrum lead us

to propose a series of 4 Hopf bifurcations emphasized in Fig. 5.5. In order to address the scattering of Re_cr1 found in the literature, the extreme sensitivity of the problem was discussed. In particular grid sensitivity and startup conditions have been shown to be exert tremendous influence on flow stability in addition to well known sensitivity to numerical schemes. Finally, artificial excitation was used to recover the *universal* range of first Hopf bifurcation to $Re_cr1 \in [8000, 8025]$.

In the next section, we show that the analysis of this complex flow can benefit abundantly of POD analysis.

5.2 POD analysis

A classification of POD modes based on the properties of the amplitude functions has been performed in [SSS10, SVS11], in terms of regular and anomalous modes. In Ref. [SSS10], the POD modes have been related with the instability modes for the first time, readying the field of flow instability study by POD analysis. The regular POD modes occur in pairs for the amplitude functions, separated by quarter cycle and the resultant instability modes obey the Stuart-Landau equation [Sen12]. The anomalous modes, on the other hand do not obey Stuart-Landau equation. Also, Stuart-Landau equation is of use for fluid dynamic system with a single dominant mode. This approach of obtaining POD eigenfunctions and amplitude functions in describing nonlinear instability of fluid flow has been described in Ref. [SBB] and is routinely used for incompressible flows [SHPP15, SG16].

Here, enstrophy is preferred over those in Refs. [NAM+03, HLB96, RF94, Sir87], where kinetic energy is used for POD analysis. In vortex dominated inhomogeneous flows, rotational energy is a better descriptor of POD over translational kinetic energy, as highlighted in Refs. [Sen12, SDS03, SSS10]. Authors in Ref. [SVS11], used enstrophy based POD approach to study both external and internal flows to show universality of POD modes in terms of amplitude functions.

Fig. 5.8 shows the variation of the equilibrium amplitude A_e with Re, for simulations performed using two grids, with (257×257) and (513×513) points. The onset of unsteadiness for this grid is the point marked as 'O' in the figure. The points shown by filled rhombus and square are obtained using the (513×513)-grid points. For the refined grid, onset of unsteadiness occurs for Re slightly lower than 9450, for the case of $A_0 = 0$. For the coarser grid we have identified 'S' as the point ($Re = 9800$) displaying secondary instability, as already shown.

For the finer grid, we note that the primary Hopf-bifurcation between $Re = 8660$ and 8670 is bypassed. For this grid, the second and third bifurcations occur for $Re = 9600$ and 10000, respectively. Following the second bifurcation, we notice three data points with the middle one identified as P_1 in Fig.2, which show similar variation as for the (257×257) grid over an extended range of Re. Later on, we compare a representative point at P_2 with P_1. A similar qualitative variation between the two grids are noted which originate in a sequence starting from Q_1 and Q_2, which are also compared later.

Few of the distinctive features of Fig. 5.8 are the following: (a) The used methods for space-time discretization are so accurate that the onset of unsteadiness in the flow field is delayed, with finer grid. Even for (257×257)-grid, the onset is delayed up to $Re = 8670$. (b) For the finer grid of (513×513) points, the first critical Reynolds number is noted between 9400 and 9425, for the case of no excitation. With excitation this can be brought down to as low as $Re = 8250$ (as shown in the figure). (c) For Re above 10400 with the (257×257)-grid, one notices two branches of solution, as shown in the figure. The lower branch (marked as U-branch) is essentially unstable and the upper branch is the stable

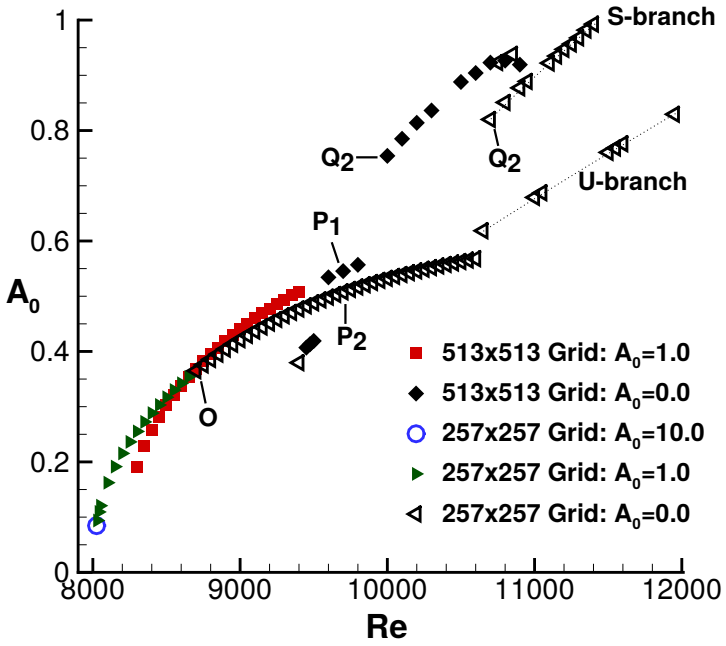


Figure 5.8: Variation of the equilibrium amplitude (A_e) with Reynolds number (Re) for the two grids, with (257×257) and (513×513) points. Note the points (P_1, P_2) and (Q_1, Q_2) have similar dynamics, as shown later. Additional points O and S represent the onset of unsteadiness ($Re = 8670$) and secondary instability ($Re = 9800$) of the flow field computed using (257×257) grid points.

branch, named as the S-branch. Upon application of slightest perturbations, the solution on the U-branch jumps to the S-branch.

5.2.1 Analysis through POD modes

5.2.1.1 Limit cycle POD modes

Here we use POD analysis to characterize flow fields obtained by the two grids. In Figs. 5.10b and 5.10a, we show the eigenfunctions obtained following the method of snapshots for the POD analysis is shown for the points, P_1 and P_2. We display only the first twelve modes the differences in Fig. 5.8 for the equilibrium amplitude and the associated maximum vorticity values in the domain, the first eight eigenfunctions have remarkable similarities, indicating the qualitative similarities of the associated flow fields obtained using two grids with significantly different points. The eigenfunction plots of Figs. 5.10b and 5.10a also show a definitive pattern, with the first and second modes are regular modes [SVS11], defined for classification of POD modes. In this case, one notices three pairs of similar vortical structures with opposite signs. In the same way, the third and fourth modes are composed of six such pairs; fifth and sixth modes similarly have nine pairs of structures.

This multiplicity of vortical structures are extended to higher mode pairs also. However, their contributions are negligibly small in terms of enstrophy content, as the first eight modes in Figs. 5.10b and 5.10a, account for nearly all of the enstrophy contents for both the grids. Such similarities are furthermore emphasized in Fig. 5.9, showing the cumulative enstrophy for the pairing of points shown in Fig. 5.8. For example, in discussing the flow dynamics for points P_1 and P_2, it has been mentioned that the flows would be similar. This is clearly brought out in the eigenfunction plots of 5.10b and 5.10a and the

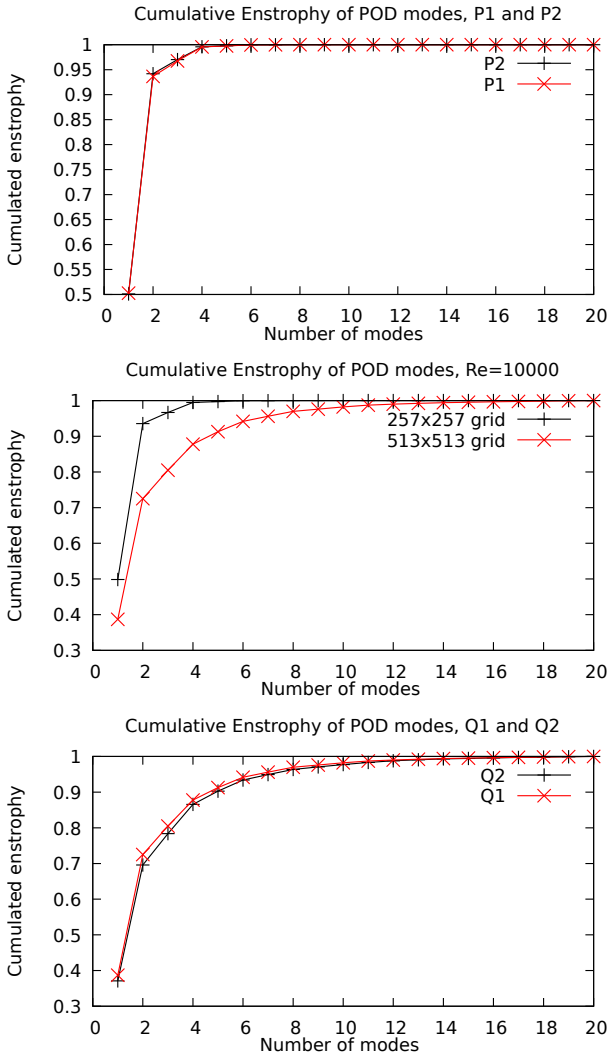
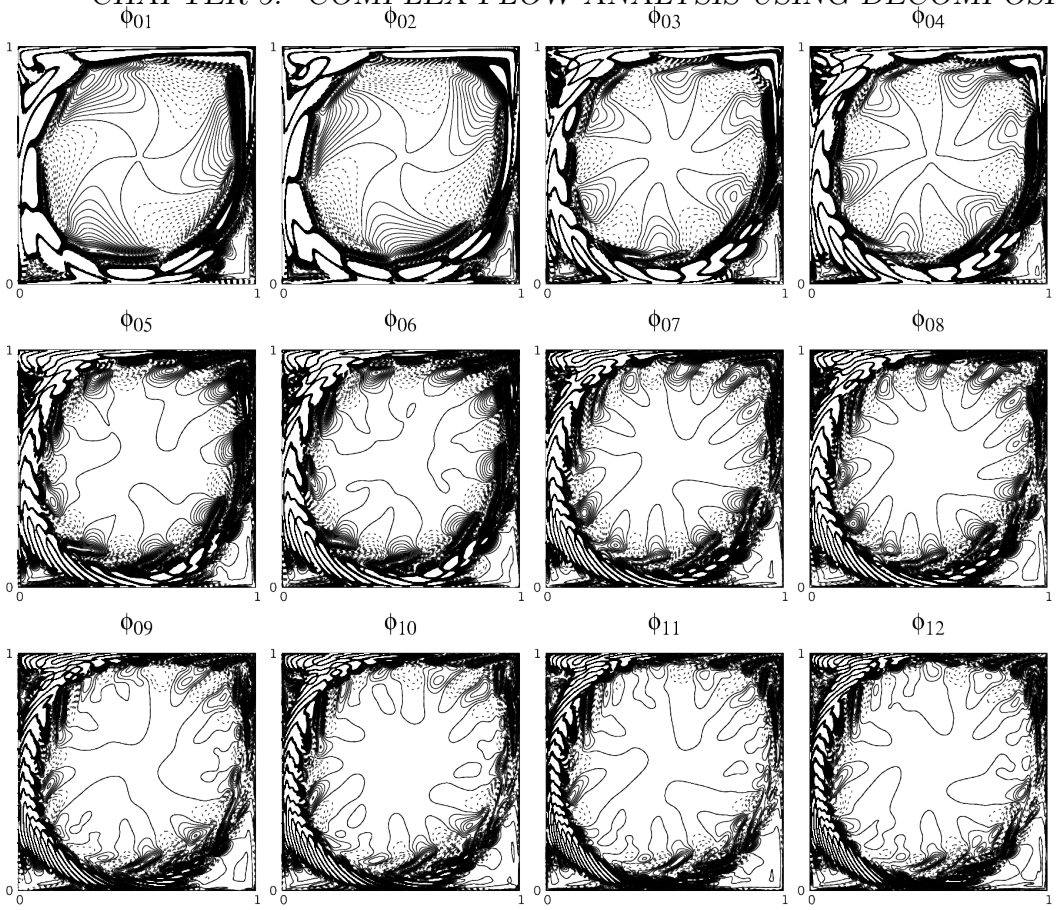


Figure 5.9: Cumulative enstrophy plots for the two grids shown for the indicated Re.

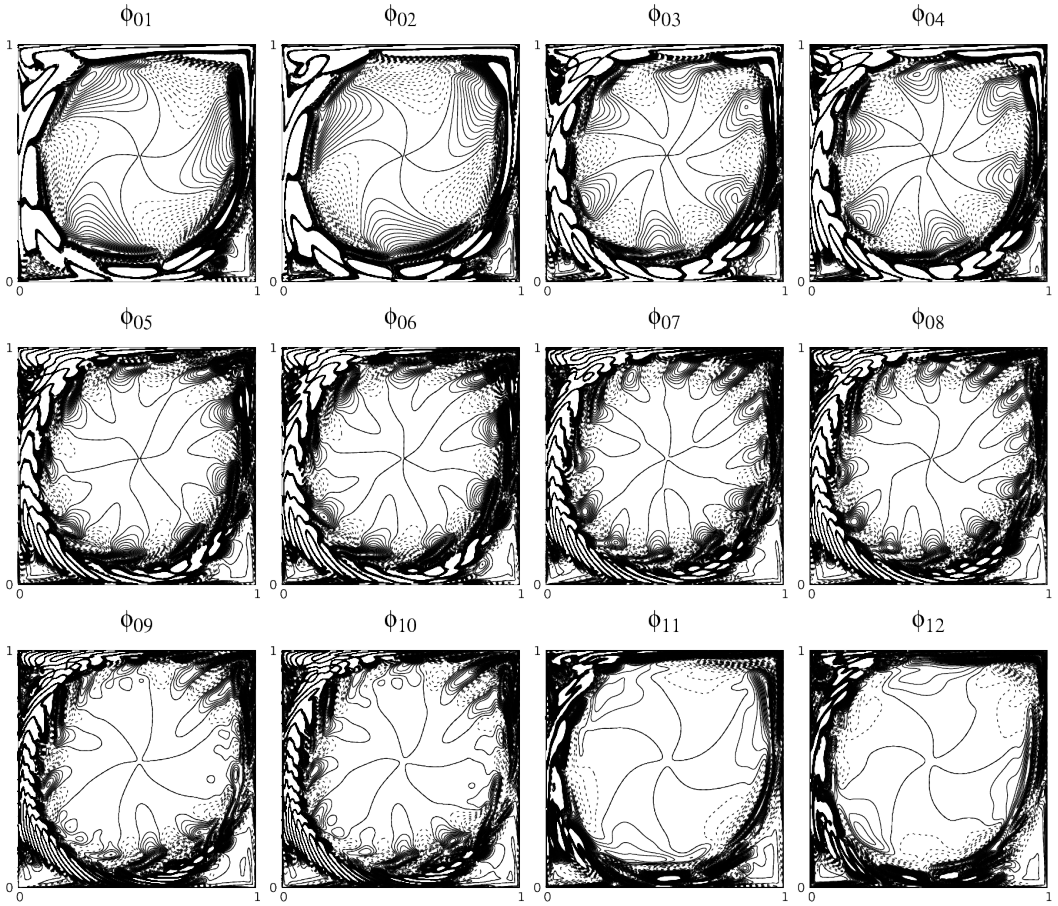
cumulative enstrophy shown in the top frames of Fig. 5.9. Similarities for the points Q_1 and Q_2 have been suggested, while discussing the bifurcation diagram (Fig. 5.8) and the cumulative enstrophy plot for this case shown in the bottom frame of Fig. 5.9 , strongly supports this. We also note that keeping the Reynolds number same with the two grids alone, does not ensure similarity of the flow, as noted from the cumulative enstrophy plot for $Re = 10000$ in the middle frame of Fig. 5.9.

The POD amplitude functions, their representative discrete Fourier Transform (DFT) plots are shown in Figs. 5.11a and 5.11b for $Re = 9700$ case, obtained using the two grids. These are shown pairwise, when the two constituents differ by a phase shift of quarter cycle. In Fig. 5.11a, amplitude functions are shown for P_1 obtained using (513×513) grid. The DFT of these time series is shown in the bottom frames for each pair. The top left frame indicates the fundamental frequency for the first and second modes ($f_0 = 0.43$), while the second, third and fourth mode pairs are the super-harmonics of this fundamental frequency (at $2f_0$, $3f_0$, $4f_0$). These amplitude functions and the frequencies are identical for both grids, as can be seen for the amplitude functions and their DFT shown for the point P_2 obtained using (257×257) grid. Once again the comparison between Figs. 5.11a and 5.11b supports the view that the flow dynamics is similar for P_1 and P_2.

Next, we investigate the flow fields for the points Q_1 ($Re = 10000$) and Q_2 ($Re =$

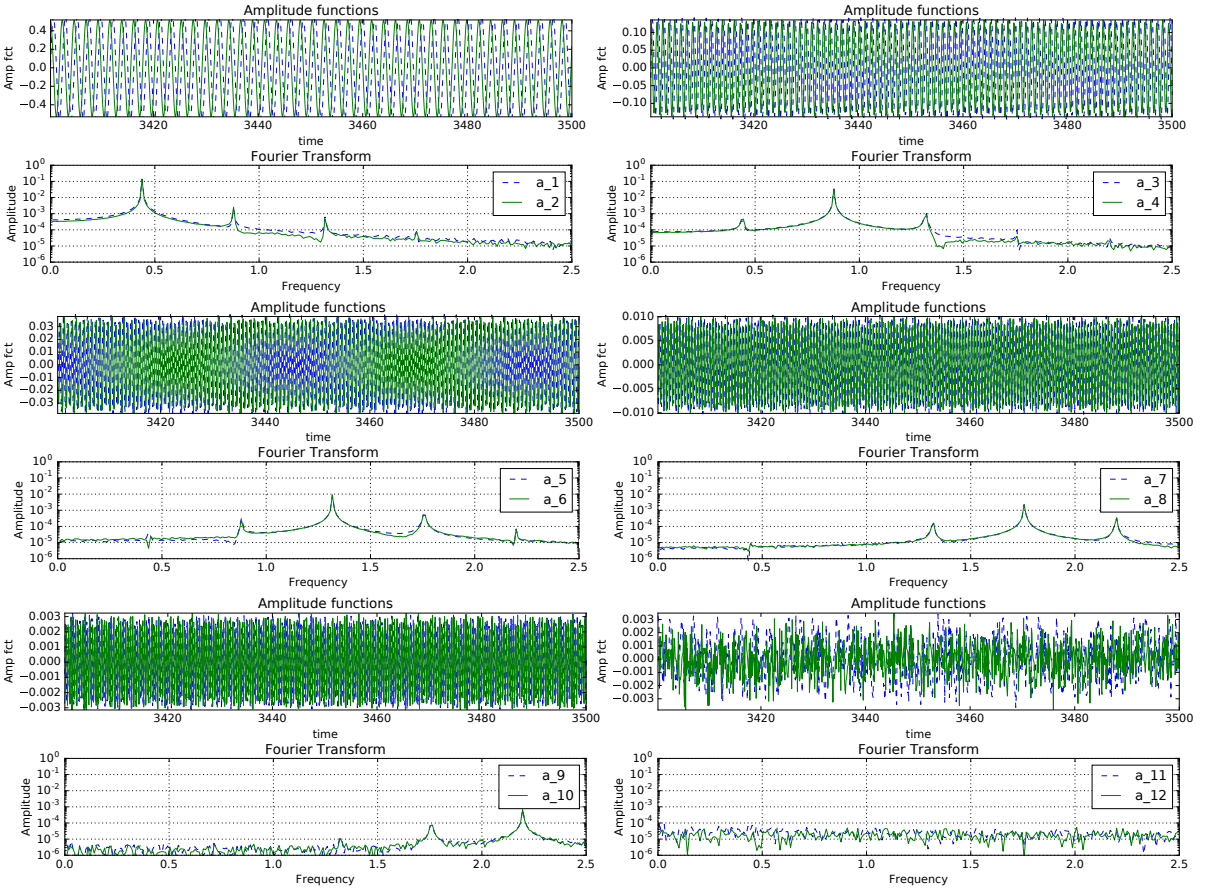


(a) P_1 (513×513 grid points).

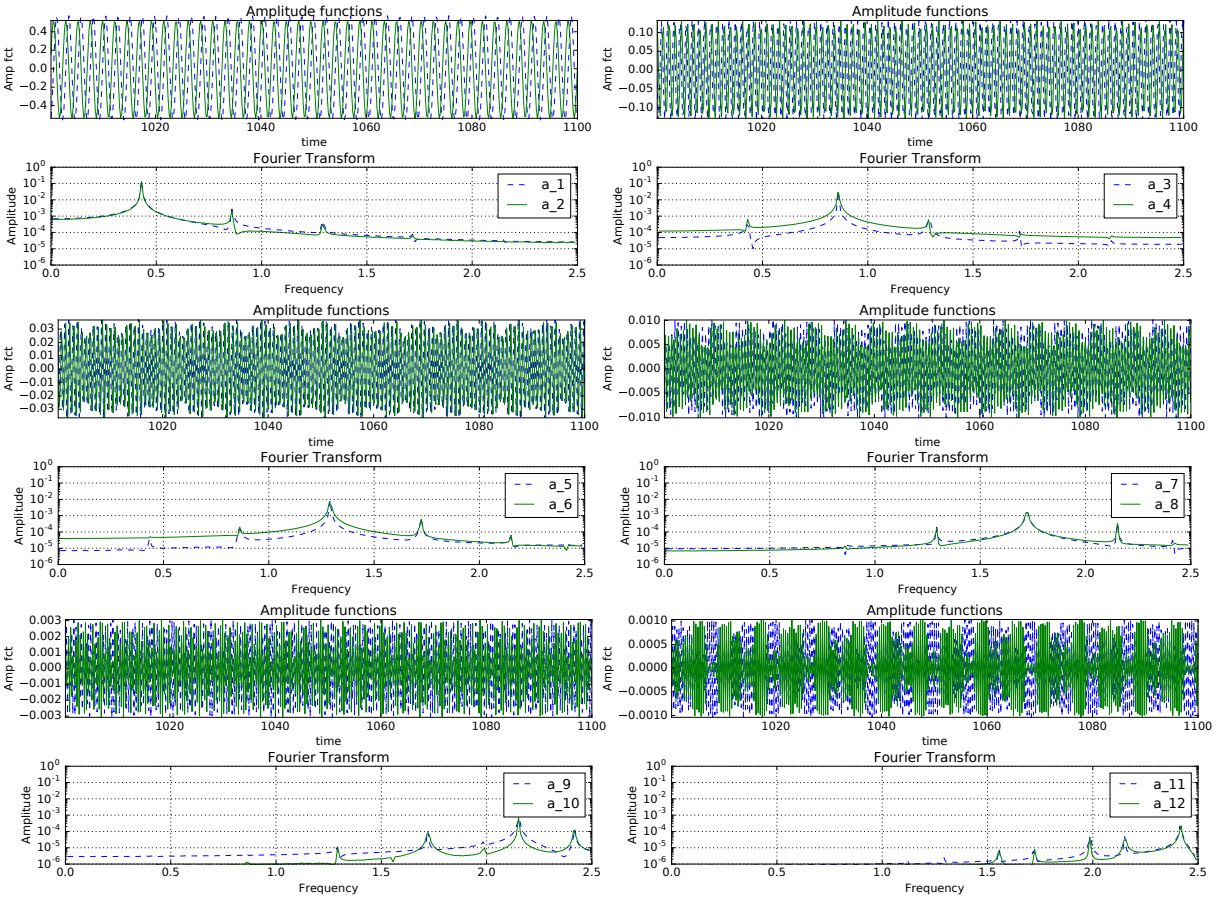


(b) P_2 (257×257 grid points).

Figure 5.10: Eigenfunctions of POD modes for $Re = 9700$ at points (P_1, P_2) in Fig. 5.8. $(\varphi_m)_m$ isolines are plotted in the $[-0.5, 0.5]$ range with 0.01 spacing. Solid lines are positive values, while dashed lines are negative value contour.

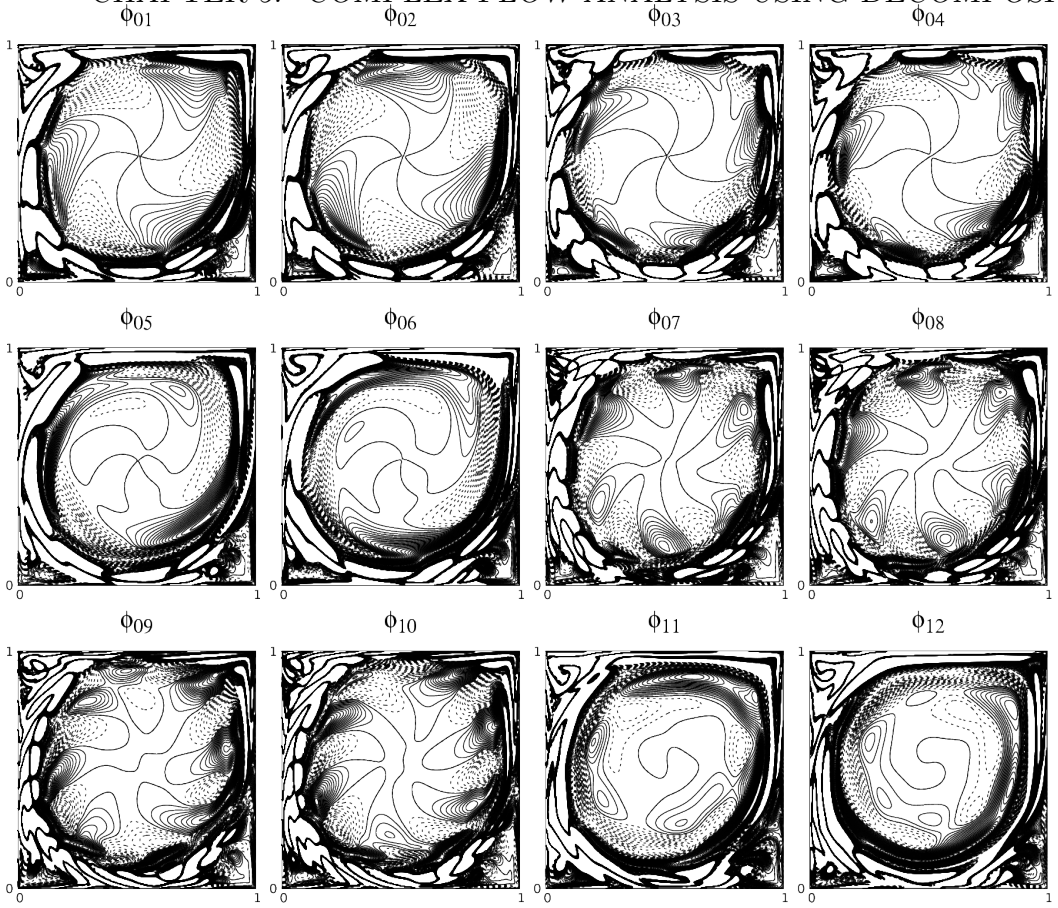


(a) $P_1(513 \times 513)$ grid

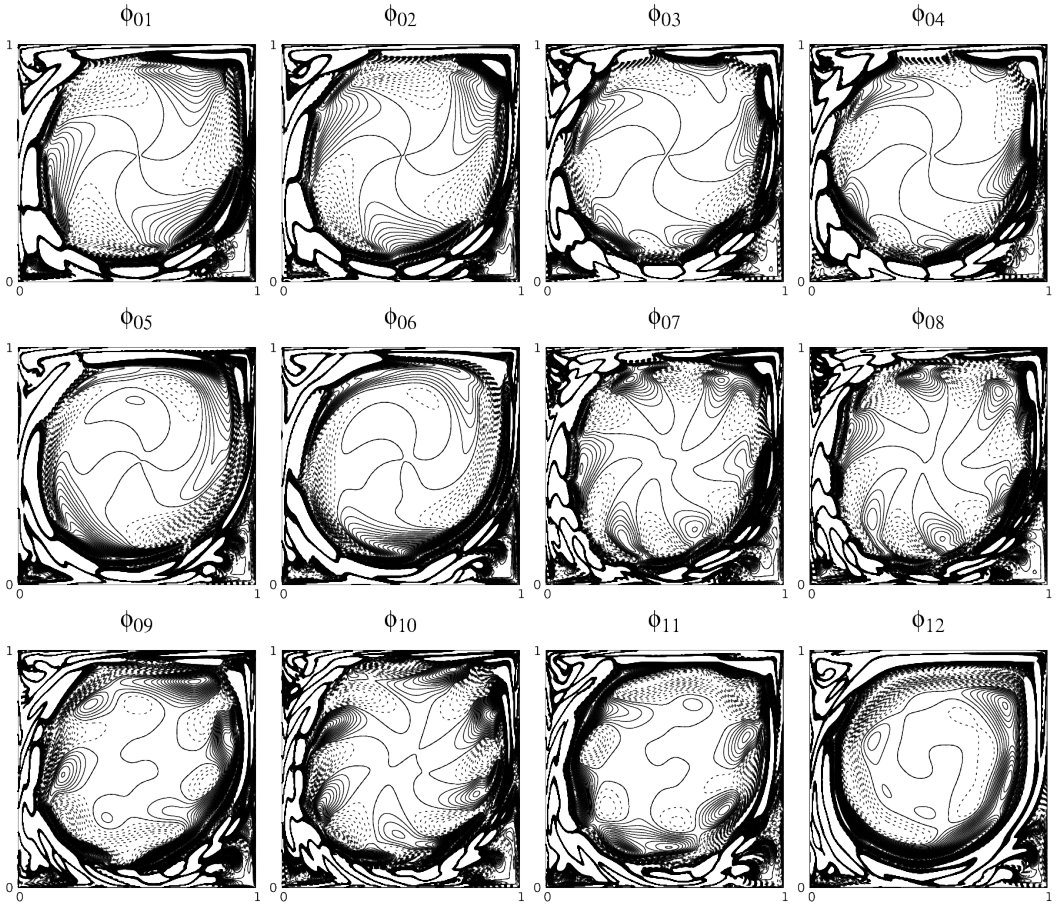


(b) P_2, $(257 \times 257$ grid)

Figure 5.11: Amplitude of POD modes and its DFT for $Re = 9700$ for P1 and P2.

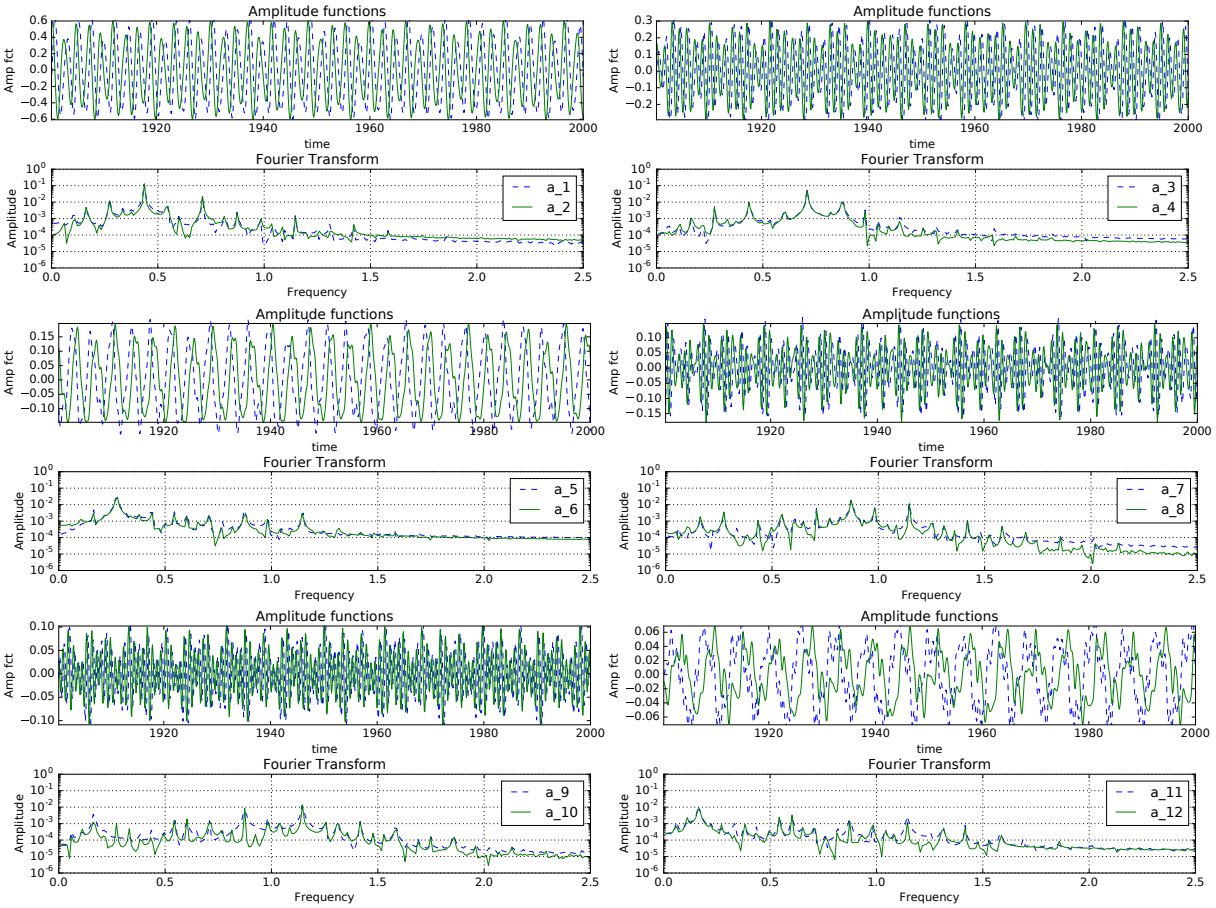


(a) Q_1, (513×513) grid.

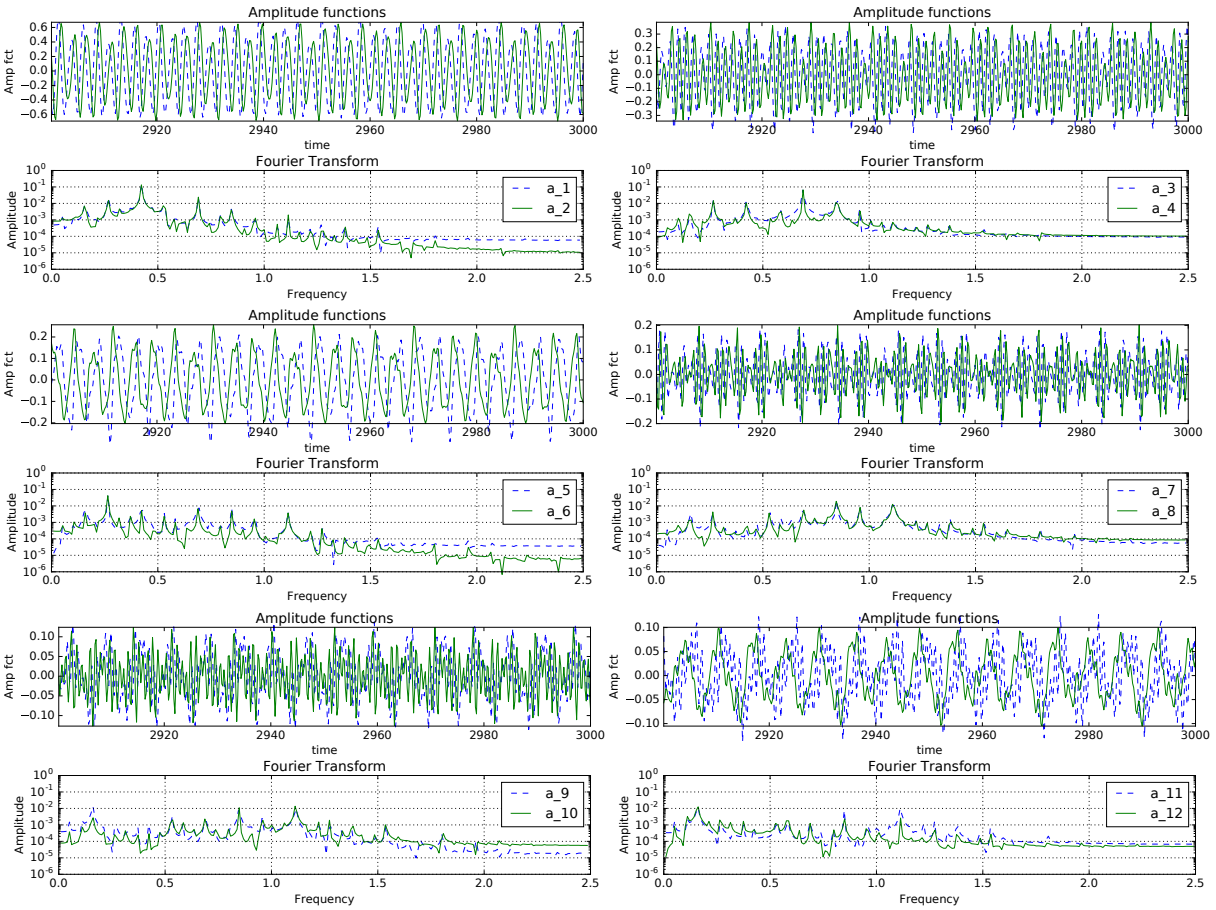


(b) Q_2, (257×257) grid.

Figure 5.12: Eigenfunctions of POD modes at points Q_1 (Re=10000) and Q_2 (Re=10700).$(\varphi_m)_m$ isolines are plotted in the $[-0.5, 0.5]$ range with 0.01 spacing. Solid lines are positive values, while dashed lines are negative value contour.



(a) $Q_1(513 \times 513)$ grid



(b) Q_2, $(257 \times 257$ grid)

Figure 5.13: Amplitude of POD modes and its DFT for Q1 (Re=10000) and Q2 (Re=10700).

10700) of Fig. 5.8, in Figs. 5.12a and 5.12b, respectively for the two grids with the help of POD eigenfunctions. Previously, we have noted that the flow fields for these points obtained by the two grids will be similar, while discussing the bifurcation diagrams in Fig. 5.8. Now the plotted eigenfunctions for the first twelve modes in Figs. 5.12a and 5.12b are also seen to be similar. This, added with the cumulative enstrophy plots shown in the bottom frame of Fig. 5.9, strongly support the view that the flow fields are indeed similar. This also shows that the view provided by the bifurcation diagram is a better descriptor of similarity of flow field in the diagram, whenever A_e^2 plotted against Re show identical slopes. The eigenfunctions have also similarity with the eigenfunctions shown in Figs. 5.10a and 5.10b for the first two pairs, with respect to qualitative features. The higher modes are distinctly different in Figs. 5.12a and 5.12b due to the flow fields belonging to different branches of the diagrams, as compared to the cases shown in Figs. 5.10a and 5.10b. Figures. 5.12a and 5.12b belong to branches in which the instability is higher due to multiple dominant frequencies interacting [LBA+18]. That causes the enstrophy to be distributed over larger number of modes, i.e., one should be interested in the higher modes beyond the number eight, as was the case for the lower Reynolds number. Even the symmetry for the eigenfunctions noted for $Re = 9700$ is lost from fifth mode onwards since two or more physical modes are interacting with the primary POD mode.

The features of eigenfunctions for Q_1 and Q_2 are also reflected in the amplitude functions shown in Figs. 5.13a and 5.13b. The first pair of amplitude functions displays identical peak for these two grid results, which is different from the fundamental frequency (f_0) noted in Figs. 5.11a and 5.11b for $Re = 9800$ case. The second pair of amplitude functions in Figs. 5.13a and 5.13b are not the super-harmonic of the fundamental seen for the first pair of amplitude function. Thus, this segment of bifurcation diagram for Figs. 5.13a and 5.13b, is qualitatively different from the lower Reynolds number parts shown in Figs. 5.11a and 5.11b. Between the two points Q_1 and Q_2, the third and fourth modes have some differences at the lower frequencies, otherwise other significant peaks are collocated. The fifth and sixth amplitude functions of POD modes again have the same value of frequency for the peak, as is noted for the first pair. All the other modes have qualitative similarity between amplitude functions for points Q_1 and Q_2, and with the exception of eleventh and twelfth modes, all the modes appear as wave-packets, which have been called as the anomalous mode of second kind [SVS11, Sen12].

5.2.1.2 Primary and secondary instabilities POD modes

Here we study the dynamics of the unsteady flow field using two different grids, with the intention of highlighting the mathematical physics of this canonical problem with POD as the analysis tool. It is necessary also to characterize the flow during primary and secondary instabilities. For this purpose, in Fig. 5.14 we show the POD eigenfunctions obtained without excitation during the primary instability stage for $Re = 8670$ obtained using the (257×257) grid, which is indicated as 'O' in Fig. 5.8. This Re is a super-critical case that displays linear instability during $t = 900$ to 1100. The eigenfunctions show various polygonal core-vortex. For example, the eighth, fourteenth and seventeenth modes display triangular vortex at the core, as was shown for the flow field in Fig. 5.2. POD captures the presence of triangular core vortex caused by the primary instability.

For the eigenfunctions shown in Fig. 5.14 for $Re = 8670$, the corresponding amplitude functions are shown in Fig. 5.15. It is readily apparent that the first two modes form the regular pair [SVS11], while the third mode is the anomalous mode of first kind; with fourth and fifth modes again form a regular pair, but modulated with higher frequency components. The sixth and seventh modes appear as wave-packets and hence, would be

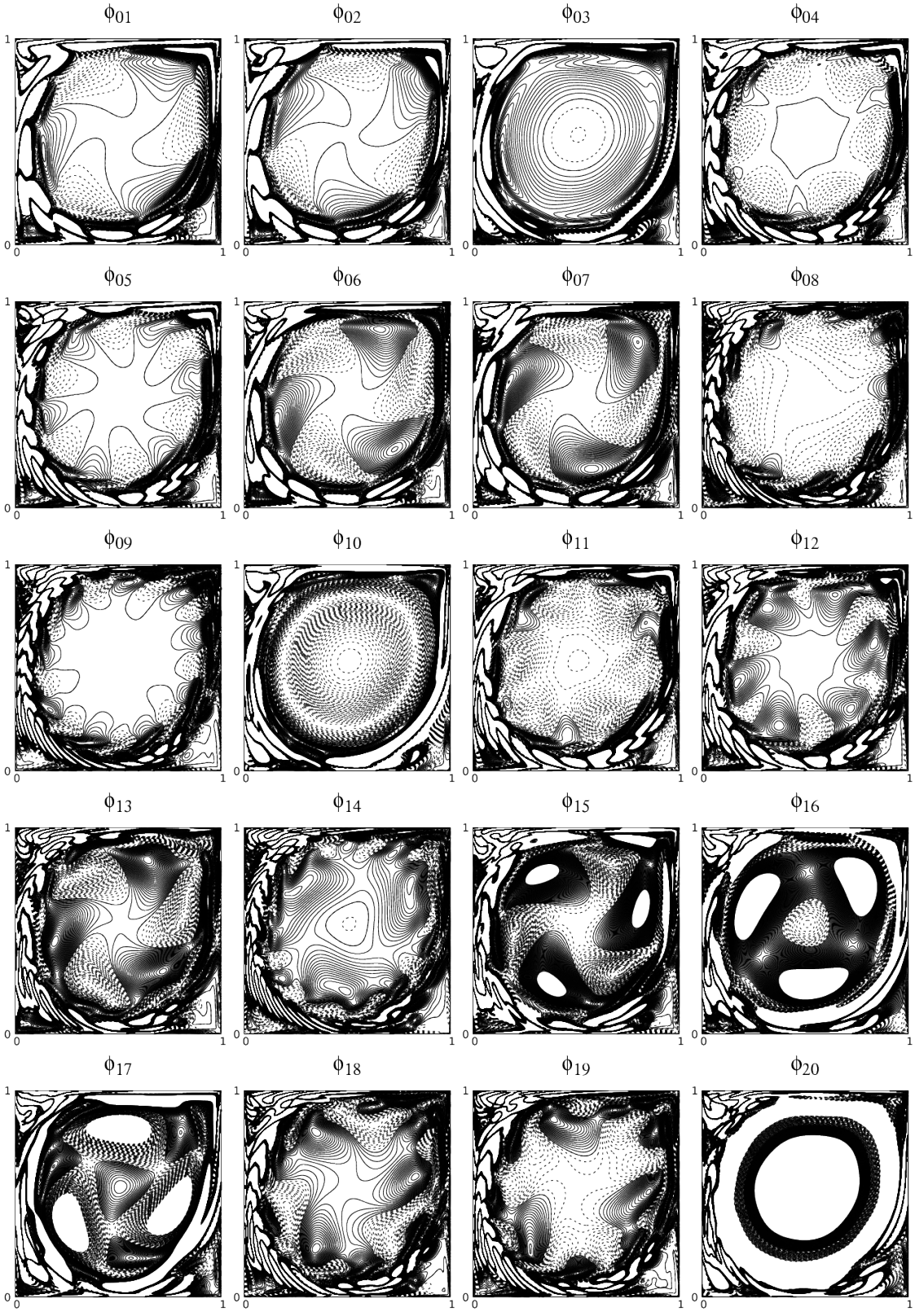


Figure 5.14: Eigenfunctions of POD modes for $Re = 8670$ obtained with (257×257) grid for the point O in Fig. 5.8 during the linear instability stage.

called the anomalous mode of second kind. The eighth and ninth modes are similar to fourth and fifth pair, i.e., regular modes which are highly modulated. The tenth mode is an anomalous mode of first kind, similar to the third mode. It has been explained

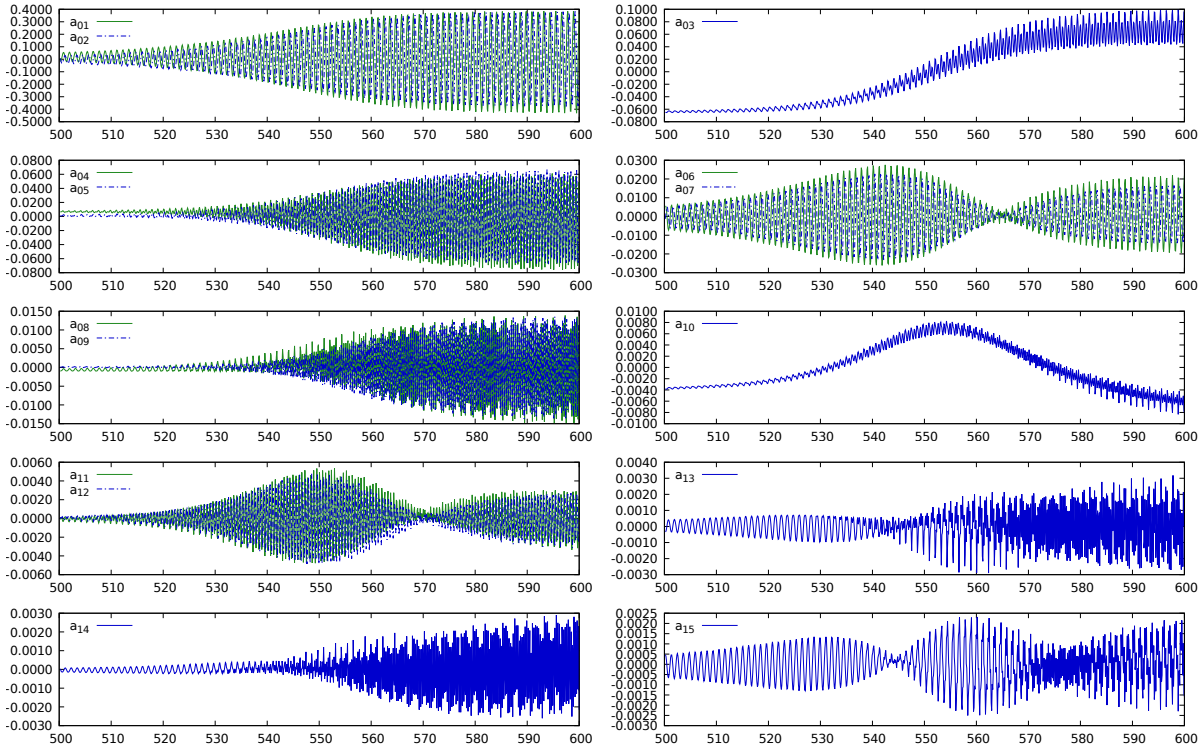


Figure 5.15: Amplitude functions for point 'O' linear instability grows. They are regrouped by pairs for regular modes while anomalous modes are shown alone.

in Refs. [SSS10, Sen12] that the anomalous mode of first kind, gives rise to equivalent stress term, like the Reynolds stress and alters the mean flow. In this respect, the third and the tenth modes have opposite effects on the mean flow, as is evident from the signs of the amplitude at the terminal time. One can similarly classify the other modes into these categories described. However, the sixteenth and seventeenth modes appear as combination of the two types of anomalous modes described. Another feature of the anomalous mode of first kind is the appearance of the eigenfunctions in Fig. 5.14, where one does not notice orbital motion of the vortices around the core, which gives rise to the polygonal vortex in the core.

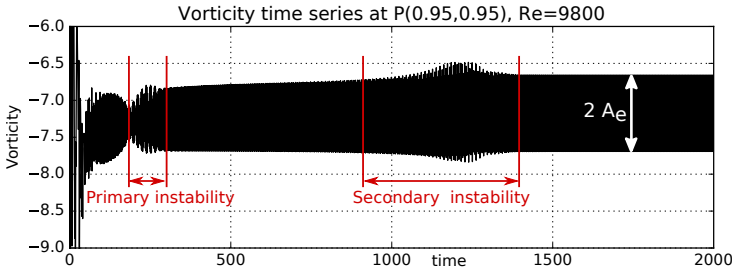


Figure 5.16: Time series for $Re = 9800$ obtained with (257×257) grid.

Fig. 5.16 provides the time series associated with $Re = 9800$, point 'S' in Fig. 5.8. The primary linear instability is followed by a quasi limit cycle (envelope is growing) before a second instability picks up and brings the flow to a stable limit cycle whose envelope does not change further with time. In the following, we give an interpretation of the results of POD analysis of one such secondary instability for point 'S' . In Fig. 5.17(a) we show the eigenfunctions obtained by POD analysis performed on data before the beginning

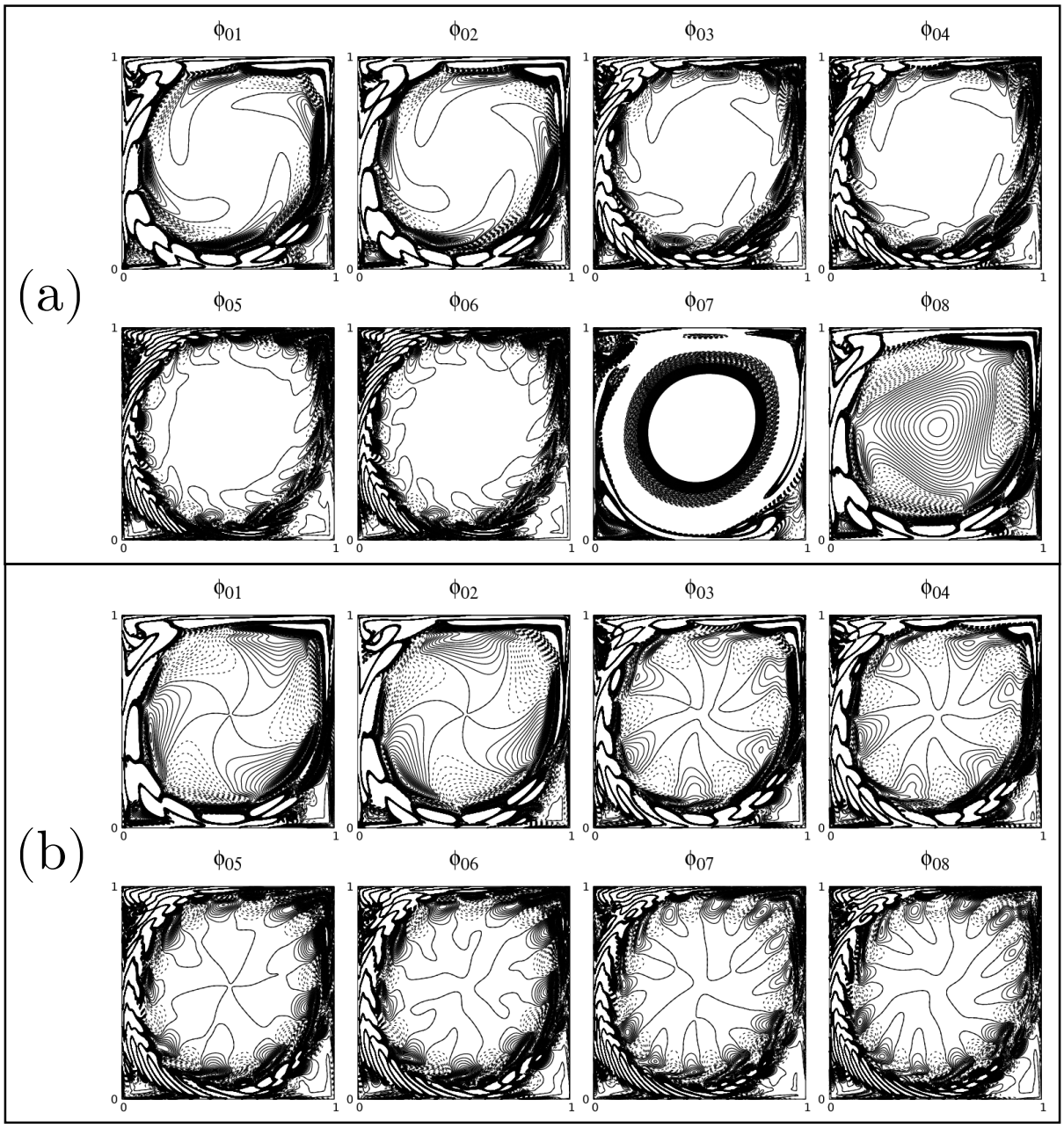


Figure 5.17: Eigenfunctions of POD modes for $Re = 9800$ obtained with (257×257) grid during (a) $t = 500$ to 600 before and during (b) $t = 1900$ to 2000 after the secondary instability.

of secondary instability during $t = 500$ to 600. At this stage, most of the enstrophy is contained in the first few modes which is why we show the first eight modes. One notices the onset of creation of the orbital vortices in the first six modes. The seventh mode is without any structure and is similar to the eigenfunction for the anomalous modes in Fig. 5.14. It is the eighth mode that shows the appearance of a large triangular vortex in the core, with three pairs of orbital vortices surrounding the core. In Fig. 5.17(b), we show the eigenfunctions for $Re = 9800$ after the occurrence of the secondary instability during $t = 1900$ to 2000.

The corresponding amplitudes and the DFT of various eigenmodes (as in Fig. 5.17), are shown in Fig. 5.18. In frames (a), the plotted amplitudes correspond to eigenfunctions shown in Fig. 5.17(a), in pairwise fashion. One can clearly note that the FFT is dominated

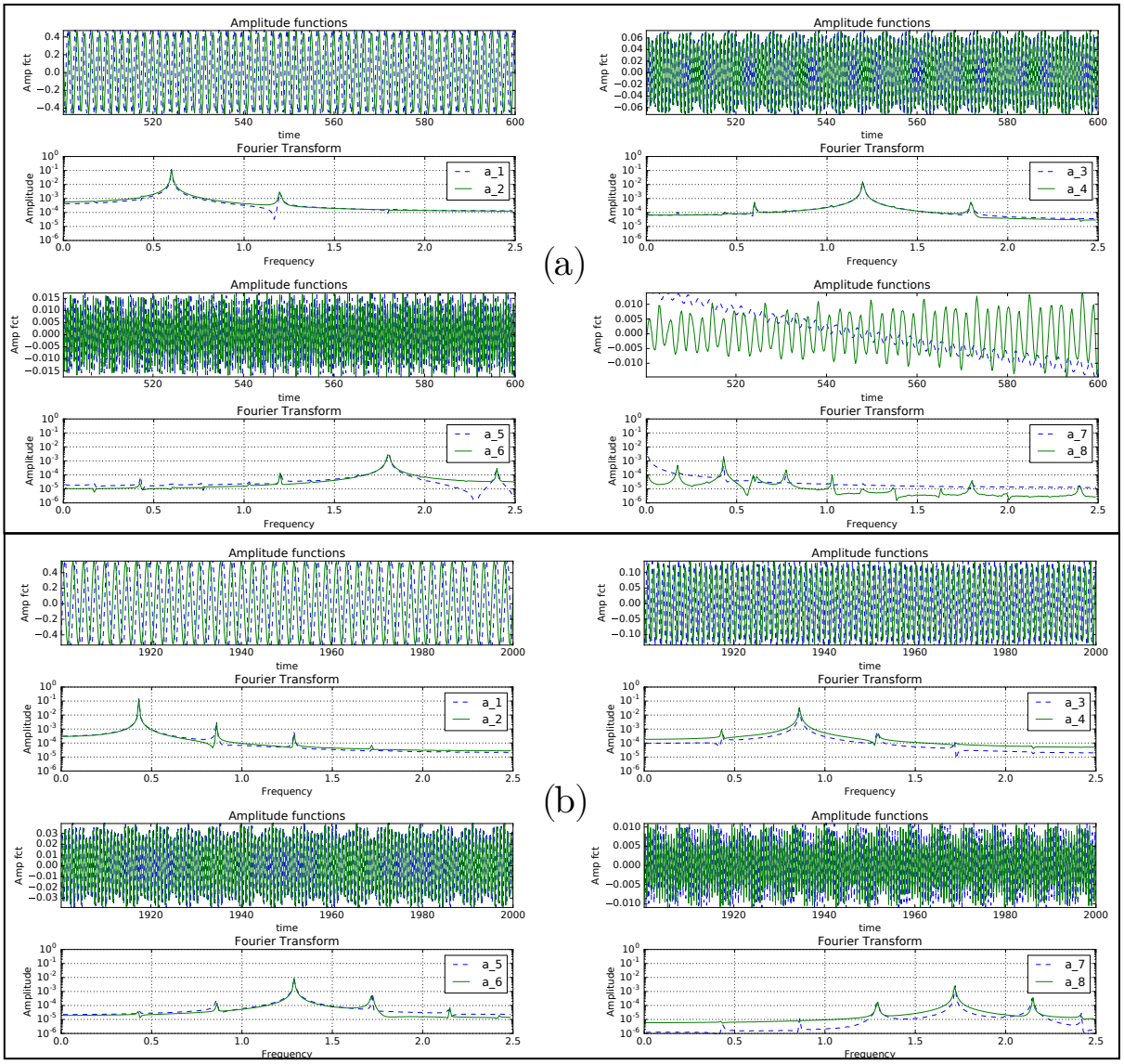


Figure 5.18: Amplitude of POD modes and its DFT for $Re = 9800$ using (257×257) grid (a) before $[t = 500$ to $600]$ and (b) after $[t = 1900$ to $2000]$ the secondary instability, for the case of Fig. 5.17

by a single mode and amplitudes are time-shifted by quarter cycle. While there is a distinct secondary mode, but its amplitude is orders of magnitude smaller. The third and fourth modes' amplitude shows the peak which has a value that is twice of that noted for the first pair. However, this mode-pair also shows modulation in the time plane, which is due to the secondary peak shown in the DFT, which is the fundamental for the first and second modes' amplitude. In the same way, the fifth and sixth modes have the peak at thrice the value noted for the first pair. The seventh and eighth modes have no correlation, as noted in Fig. 5.18(a).

In Fig. 5.18(b), we note the amplitude functions corresponding to the eigenfunctions shown in Fig. 5.17(b), obtained during $t = 1900$ and 2000, when one is in the final limit cycle stage. It is interesting to note that the action of the secondary instability is to shift the fundamental frequency for the first pair ($(f_0)_{before} = 0.60$) to a lower value ($f_0 = 0.43$), as noted in the DFT plots. The second and third pair of amplitude functions have peaks at $2f_0$ and $3f_0$, respectively. The seventh and eighth modes are characterized by very high

frequency fluctuations, and modulated at moderate frequencies, as a consequence one can categorize these as anomalous mode of second kind [SVS11]. This phenomenon is explained by similar amplitudes of the leading peak ($4f_0$), with the next peak in amplitude ($5f_0$) that interact to create modulations. This pattern is visible for each final state, however, it is weaker for the finer grid in Figs. 5.10a and 5.10b.

Conclusions

In this section, we have used POD to characterize LDC flow for a range of Re for simulations performed using two grids (257×257) and (513×513) points. The relative scaled amplitude of disturbance field is lower for the finer mesh, which explains why primary Hopf bifurcation is delayed for the refined grid. But, despite difference in bifurcation sequences in the two grids, the qualitative similarity of flow fields are noted for points in the bifurcation diagram. The flow in the two grids will be similar when A_e^2 versus Re curves have identical slope, even if the Re are different. This is supported first by comparing the POD modes of the flow field for $Re = 9700$ for the two grids at P_1 and P_2 but also for points Q_1 and Q_2 that do not share the same Re. This is attested by careful analysis of the POD eigenmodes corresponding amplitude functions and their DFT. These observations are strongly supported by the cumulative enstrophy plots in Fig. 5.9, for these four points, P_1, P_2, Q_1 and Q_2.

We were also able to characterize the primary temporal instability without excitation (point 'O') by POD analysis, showing eigenfunctions and amplitudes which shows clearly multi-periodic dynamics of the flow, with a single dominant fundamental frequency and its super-harmonics. Finally, we have characterized the secondary instability by showing POD computed onto $t = [500, 600]$ and $t = [1900, 2000]$ ranges. These time intervals correspond to before and after the secondary instability for $Re = 9800$. We note that such secondary instability does not occur for all Reynolds number cases, but when it does occur, the effect is to change the fundamental frequency from a higher value (0.60) to a lower value (0.43). The eigenfunctions are also completely different, before and after the secondary instability.

In this chapter, we have studied extensively the singular LDC flow. The exhibited complexity confirms that an extensive knowledge of the physics at work in complex flows is necessary. Ample evidence of extreme sensitivity of this problem has been given proving that a special attention is required for choosing discretization schemes. In the next chapters, the added understanding of this flow instability behavior will be relied upon to build reduced order models based on POD and the bifurcation diagram presented in Fig. 5.8.

A few remarks on building ROM for the singular LDC problem. It clearly appears that building a ROM accross ranges of Hopf bifurcations is a vain adventure as we have shown that DNS itself produces spread out results. Consequently, the framework in which the ROM is built should be defined precisely and limited to achievable goals. The ultimate goal of ROM for high Re LDC is to replicate the Hopf bifurcation sequence and anticipate Re numbers that do not belong to the set precomputed by DNS. Modeling the impulsive start and instability cascade seems out of reach for current ROM techniques. Thus, we have decided to focus on build ROM for limit cycle within Re segments delimited by Hopf bifurcation. The next chapter tackles this problem by means of interpolation ROM while the last one will explore projection ROM.

Chapter 6

Interpolated ROM

Contents

6.1 A physics based interpolation method: Time-scaling 158

6.1.1 Need for time scaling 160

6.1.2 Formulation and modeling of ROM 163

6.1.3 Time-scaling ROM algorithm 165

6.1.4 Time-shifting ROM applied to the LDC flow 166

6.1.5 Time-scaling ROM applied to the flow past a cylinder 169

6.1.6 POD and time-scaling. 172

We are now focusing on family of methods for building ROM that relies on interpolation. Using interpolation instead of solving the studied problem for a new set of parameters is a hot research topic [AF08, Cha08, CS10, SO11]. However direct interpolation is largely problematic. In order to fix ideas, without loss of generality, we suppose that a full order model is available and solves the problem with high accuracy within a few hours. The data generated amounts to a few GB for each simulation with reasonable time sampling. A fixed space and time discretization is available with N the number of space degrees of freedom and n_t the number of snapshots. The goal is to produce quick evaluation of the problem solution for a set of parameters for instance it can be used to solve optimization or control problems, or simply build virtual charts [CLB+17].

Typically in fluid dynamics applications, dimensionless numbers such as Re or St are used. Then we define the following toy problem:

Interpolation toy problem

Suppose a DNS solution produces a simulation for several donor Re number $\mathcal{R} = \{Re_1, ..., Re_q\}$ with a large number of spatial degrees of freedom N and stores n_t. The goal is to interpolate with as much precision as possible a solution to a target Reynolds such that $Re_t \notin \mathcal{R}$.

Given some preliminary sampling points have been computed, we want to dispose of interpolated values in space and time, this kind of feature is already available in most scientific computing code. However, it is quite expensive to interpolated high dimensional data with usual tools such as Lagrange interpolation. And parametrized interpolation is inherently problematic for unsteady flows. In order to solve this issue, many approaches have been proposed. Some of them rely on multiparameter decomposition

and simple interpolation methods (Lagrange, spline, etc.) while new methods have been devised specifically for such large problems EIM/DEIM [CS10, MNPP09] or Grassmann Manifold interpolation [ACCF09, AF11]. These methods have proven to be very efficient but require (especially for the latter) good knowledge of differential geometry. Recently, Rolando Mosquera [Mos18] provided an extensive review of these methods for fluid mechanics that confirms both the effiency of such approaches and the complexity that it implies. Our approach takes the opposite stance, we rely on physical observation and simple processing to construct a satisfying interpolation that we call *time-scaling interpolation*. Indeed, we have seen that even DNS with high accuracy schemes struggles to capture consistently the physics involved in complex incompressible flows such as LDC. This thorough analysis allows us to rely on physical observations rather than general PDE approach. The next section described in details time scaling interpolation.

6.1 A physics based interpolation method: Time-scaling

Flows governed by unsteady NSE presents the physical dispersion relation linking each length scale (wavenumber) with corresponding time scale (circular frequency). Thus, the ranges of time and length scales are important, even though a single Strouhal number (St) and Re are often used to describe the flow field. Multitude of length and time scales are inherent as noted in chapter 5 via POD modes and multiple Hopf bifurcations for flow in LDC. The existence of such ranges facilitates ROM development, i.e. when donor Re's are in the same range, where the target Re resides.

For a vortex dominated flow, the time scale is defined as St ($= fD/U_\infty$), relating dominant physical frequency (f) with flow velocity, (U_∞) and the length scale (D). However the flow does not display a single frequency, as one notices several peaks for both flows in figure 6.1. The time series of the vorticity data at indicated locations are shown in the left hand side frames. While the flow past a circular cylinder displays a single dominant peaks with side bands in the spectrum (shown on the right hand side frames), the flow inside LDC clearly demonstrates multiple peaks.

Specifically for flow past circular cylinders, an empirical relation of the type has been provided

$$St = St^* + m/\sqrt{Re} \tag{6.1.1}$$

in [FKE98] with experimental data, for variation of St with Re in the wide range of $47 < Re < 2 \times 10^5$, with values of St^* and m being different, for different ranges of Re. Instead of using such an algebraic additive relationship, here we propose a power law relation and test it for the range: $55 \leq Re \leq 200$. Consequently a relationship between Re and St is be proposed, in order to perform interpolation on the vorticity time series.

The existence of unique St for a fixed value of Re, as embodied in equation (6.1.1) implies that employing simple-minded interpolation strategies like Lagrange interpolation, will display unphysical wave-packets in reconstructed solution, as the time scales are function of Re at the target. This is clearly demonstrated in figure 6.3. The proposed ROM tackles this issue with the time scaling technique. Since the idea underlying the present method is based on observation of St-Re relation in the flow past a circular cylinder, this problem is studied together with the LDC that will provide insight for inside flows.

Numerical methods LDC DNS are obtained as in chapter 5 with (257×257) points are taken for the LDC problem. The same approach is applied to the flow past a circular

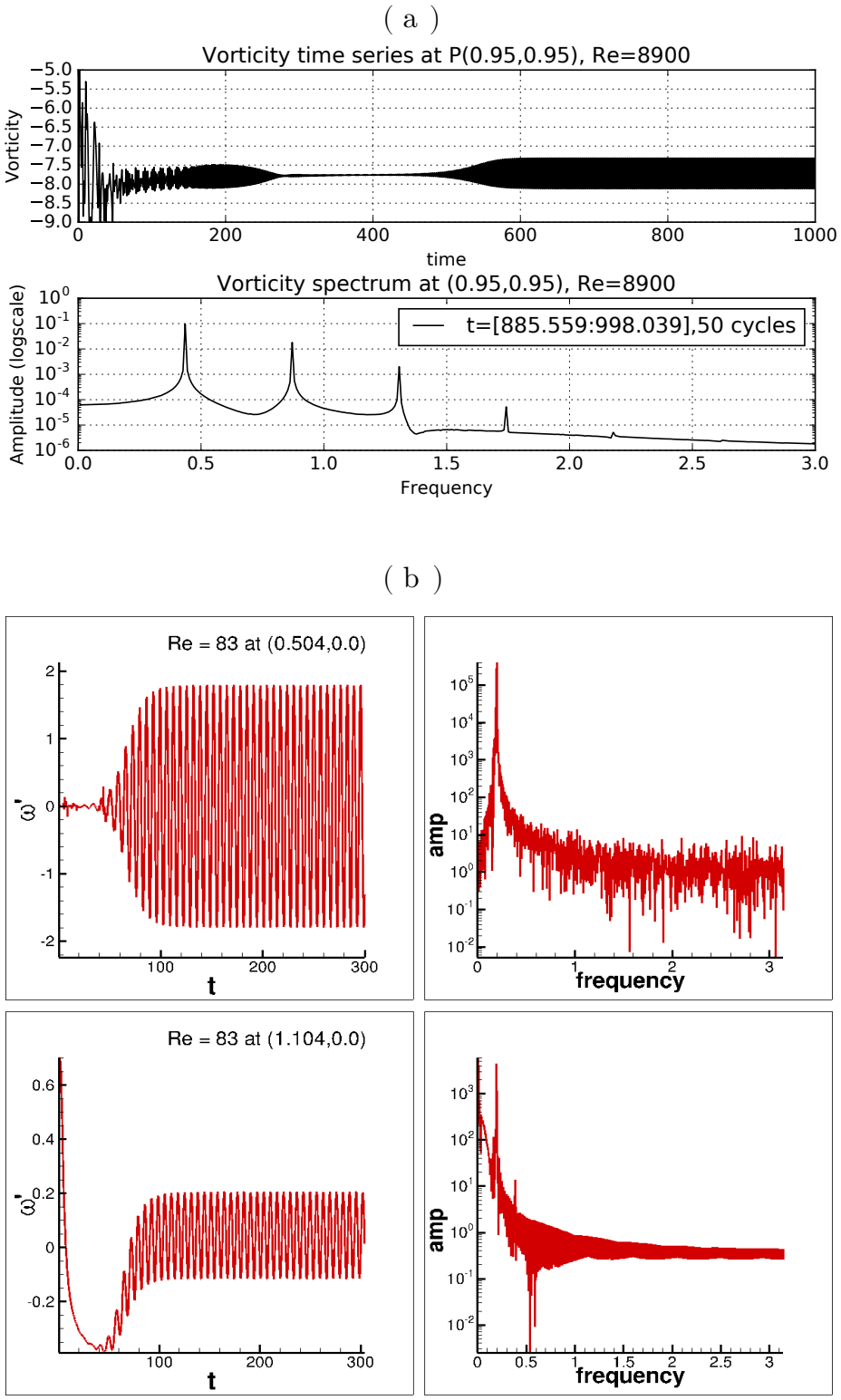


Figure 6.1: DNS time series and their associated DFT's are shown for (a) the flow inside a LDC and (b) the external flow past a cylinder, at indicated points in the flow. In (b), left frames are time series at different wake points and right frames are the associated spectra.

cylinder but requires the introduction of orthogonal curvilinear coordinates (ξ, η) to solve eqs. (5.1.1) and (5.1.2).

The governing equations in transformed plane are

$$\frac{\partial}{\partial \xi}\left(\frac{h_2}{h_1}\frac{\partial \psi}{\partial \xi}\right) + \frac{\partial}{\partial \eta}\left(\frac{h_1}{h_2}\frac{\partial \psi}{\partial \eta}\right) = -h_1 h_2 \omega \tag{6.1.2}$$

$$h_1 h_2 \frac{\partial \omega}{\partial t} + h_2 u \frac{\partial \omega}{\partial \xi} + h_1 v \frac{\partial \omega}{\partial \eta} = \frac{1}{Re}\left\{\frac{\partial}{\partial \xi}\left(\frac{h_2}{h_1}\frac{\partial \omega}{\partial \xi}\right) + \frac{\partial}{\partial \eta}\left(\frac{h_1}{h_2}\frac{\partial \omega}{\partial \eta}\right)\right\} \tag{6.1.3}$$

where h_1 and h_2 are the scale factors of the transformation given by: $h_1^2 = x_\xi^2 + y_\xi^2$ and $h_2^2 = x_\eta^2 + y_\eta^2$. The coordinate given by ξ is along azimuthal direction and η is in the wall-normal direction. No-slip boundary condition is applied on the wall via

$$\left(\frac{\partial \psi}{\partial \eta}\right)_{body} = 0 \quad \text{and} \quad \psi = constant$$

Uniform flow boundary condition (Dirichlet) is provided at the inflow and a convective condition (Sommerfeld) is provided for the radial velocity at the outflow.

The convection terms of equation (6.1.3) are discretized using the high accuracy compact OUCS3 scheme which provides near-spectral accuracy for non-periodic value of the convective acceleration terms, as explained in detail in [Sen13]. A central differencing scheme is used to discretize the Laplacian operator of equations (6.1.2) and (6.1.3) for the circular cylinder. An optimized four-stage, third-order Runge-Kutta (OCRK3) dispersion relation preserving method in [SRB11] is used for time marching. Equation (6.1.2) is solved using Bi-CGSTAB method as for LDC.

These same methods have been used earlier for validating and computing in [SSS10, SHPP15]. Here the simulations are performed in a fine grid, with (1001×401) points in the ξ and η directions. Finally, typical output of flow past a cylinder simulation is given in Fig. 6.2.

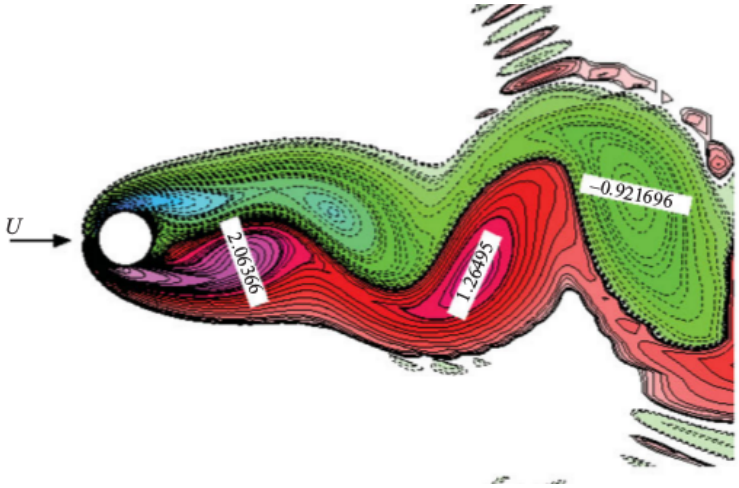


Figure 6.2: Vorticity field plot for the flow past a circular cylinder showing von Karman street at Re=75. Courtesy of T.K. Sengupta [SSS10].

6.1.1 Need for time scaling

The proposed ROM aims at interpolating vorticity fields at a target Re (Re_t) from pre-computed DNS at different donor Re's. If Lagrange interpolation is used directly, then it will not work due to variation of St with Re. Even with close-by donor Reynolds numbers

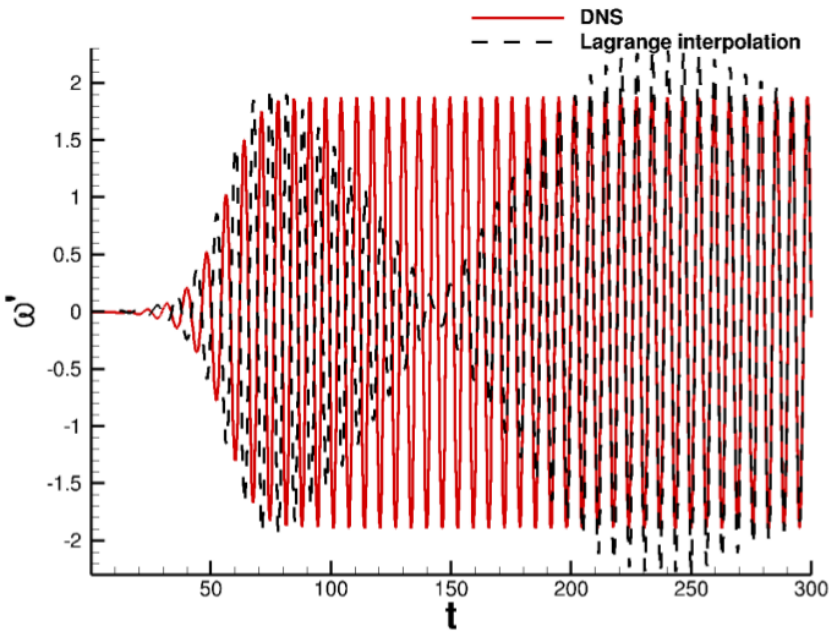


Figure 6.3: Direct Lagrange interpolation of DNS vorticity disturbance time series between Re causes wave packets in the cylinder wake at point (0.504,0.0).

data, upon interpolation, will produce wave-packets for flow past a cylinder as shown in figure 6.3. In this figure, results are shown for Re = 83, as obtained by DNS of NSE (shown by solid lines) and that is obtained by Lagrange interpolation of NSE solution donor data obtained for Re = 78, 80, 86 and 90.

The physical frequency (f) varies slowly with Re and superposition of time-series of donor data causes beat phenomenon observed by superposition of waves of slightly different frequencies. Thus, the knowledge of variation of St with Re is imperative in scaling out f-dependence of donor data before Lagrange interpolation and this is one of the central aspects of the present work. After obtaining frequency-independent data at target Re, one can put back the correct f-dependence via its variation with Re at the target Reynolds number.

It was noted in [SHPP15] that the flow past a circular cylinder suffers multiple Hopf bifurcations (experimentally shown in [Str86]) and in [SVS11] for flow inside LDC and flow over cylinder. Hence the accuracy of reconstruction naturally demands that the target and donor Re's should be in the same segments of figure 6.4, as the flow fields are dynamically similar. In figure 6.4, the equilibrium amplitude of disturbance vorticity are plotted as a function of Re for both flows. It is imperative that one identifies the target Re in the same segment of donor Re's for DNS-quality reconstruction for flow past circular cylinder as in [SHPP15] and for flow inside LDC as discussed in chapter 5. In each of these sectors of Re, the flow behaves similarly and the (St, Re)-relation is distinct.

In figure 6.4(a), the range of Re from 8000 to 12000 for the LDC is subdivided according to the bifurcation sequence (257×257) for the purpose of interpolation, Four ranges are defined with the first one given by: $\mathtt{R_I} = [8020 : 8660]$ that corresponds to externally excited range, which shows rapid variation of the amplitude. Range $\mathtt{R_{II}}$ and $\mathtt{R_{III}}$ are defined by hopf bifurcations and present stable limit cycles, thus making them good candidate for ROM building. Range $\mathtt{R_{IV}} = [10600 : 12000]$ is difficult for interpolation, as one can see two branches in this range, one of which is unstable (U-branch) with respect to any miniscule vortical excitation, as opposed to the stable one (S-branch). The flow past cylinder is also divided in ranges as shown in figure 6.4(b). For example, to reconstruct

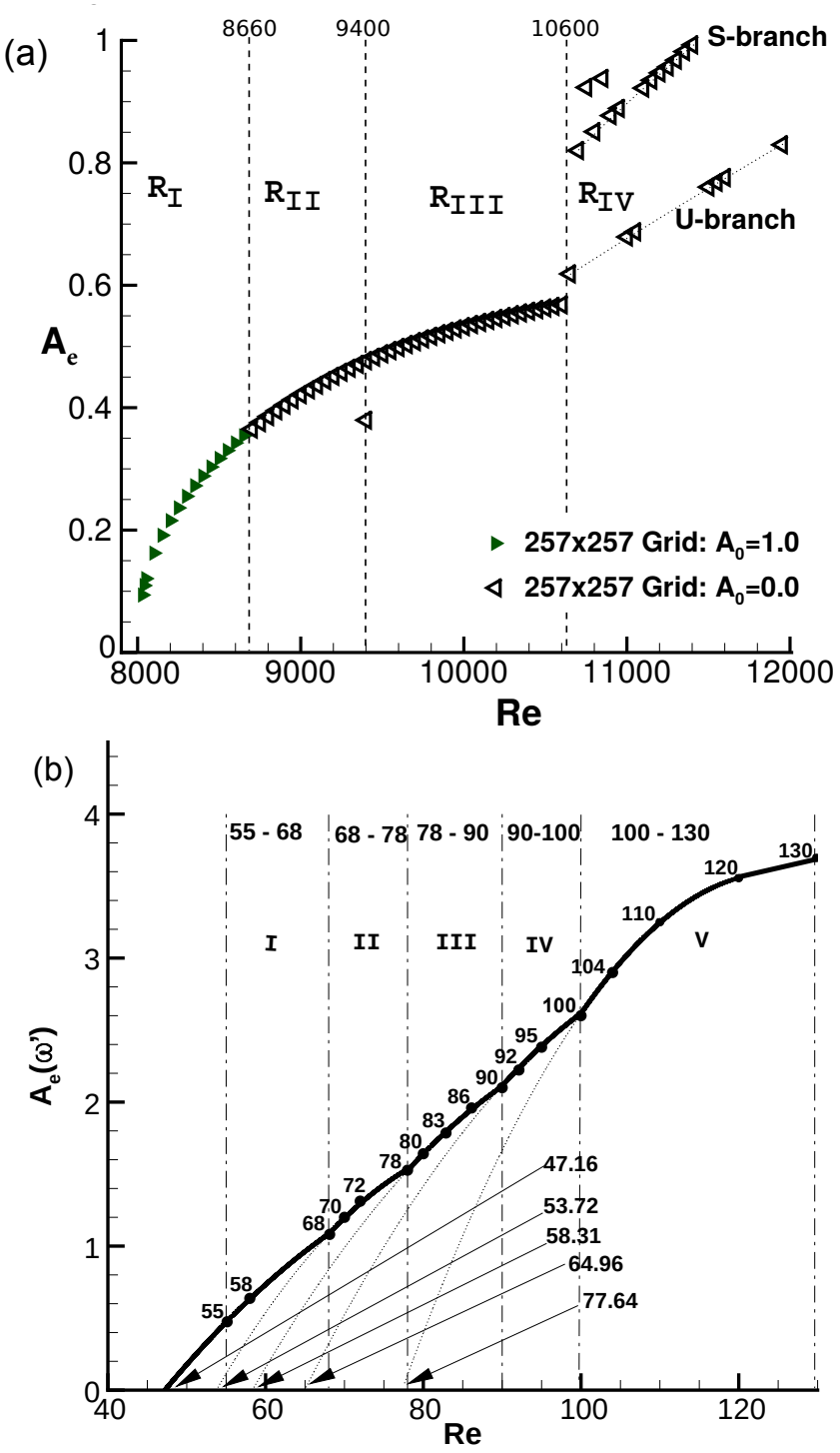


Figure 6.4: Variation of Equilibrium amplitude of disturbance vorticity with Re indicating the segments of Re with respect to bifurcation sequences for (a) flow in LDC and (b) for flow past a cylinder.

solution for Re=83, we have used data in the range of $78 \leq Re \leq 90$ for the most accurate ROM.

Table 6.1: Scaling constant and base Re_b for different ranges of Re_s

Re Range	Scaling Constant (n)	Basic Re (Re_b)
$55-68$	-0.49 ± 0.02	60
$68-78$	-0.41 ± 0.02	72
$78-90$	-0.37 ± 0.02	80
$90-100$	-0.32 ± 0.02	95
$100-130$	-0.28 ± 0.02	110

6.1.2 Formulation and modeling of ROM

In equation (6.1.1), a relation between St and Re is shown for a wide range, for the latter. One should scale out dependence of DNS data on f or St, for any Re, by a proposed power law scaling given below,

$$\frac{St(Re_s)}{St(Re_b)} = \left(\frac{Re_b}{Re_s}\right)^n \tag{6.1.4}$$

The exponent n will depend upon the segment of Re shown in figure 6.4, with Re_b denoting a base Reynolds number in each segment. In this equation, any donor Re is indicated as Re_s. Thus in a cluster of four donor Re's, one is identified as Re_b and the other three identified as Re_s. From equation (6.1.4) one deduces n, by the following,

$$n = \frac{\log(St(Re_s)/St(Re_b))}{\log(Re_b/Re_s)} \tag{6.1.5}$$

The scaling exponent n is a characteristic number of each segment and Re_b. In Table 6.1, we show five segments and the corresponding n, along with Re_b used in each range. For the flow past a circular cylinder, the value of n is obtained with the tolerance of ± 0.02 for all Re's in the respective segment. As discussed in [LBA+18], f is almost constant on each segment, so that we can set $n = 0$ for the LDC, individually in each segment. Having fixed n for any Re_s in the segment of choice, time-scaling is performed by the following,

$$t_s = t_b\left(\frac{Re_b}{Re_s}\right)^n + t_0(Re_b, Re_s) \tag{6.1.6}$$

To interpret equation (6.1.6), we plot the disturbance vorticity for the flow past a cylinder at a fixed location in the wake center-line ($x = 0.504$, $y = 0$), in figure 6.6. The same format of time scaling should apply to many other flows, including flow inside a LDC. It is noted that there exists a time-shift between the maximum of these two time series, shown as t_0 in the figure. This process is illustrated by a schematic view of the transformations applied by time scaling in Fig. 6.5. Let us consider the time for Re_b as t_b, and then to apply the proposed time-scaling for the data for Re_s, we change the physical time of Re_s, by the expression given in equation (6.1.6). Consequently, the left hand side of equation (6.1.6) is the scaled time. After obtaining t_0, it is needed to collapse the two time series for Re_s and Re_b, so that the maximum for these two time series coincide. Thus having fixed the base Reynolds number in each windows of bifurcation sequences, we can obtain the time-scaled abscissa for each Re_s in that range.

The search for t_0 is performed in such a way that the phases of both Re_b and Re_s match accurately. One should note that the effects of t_0 are significant, despite the fact

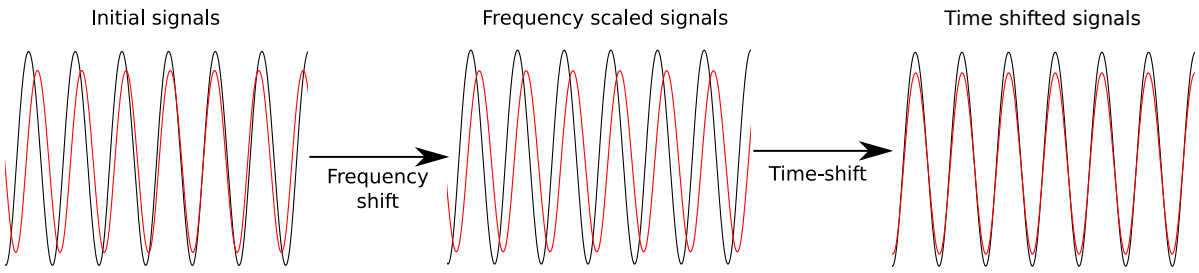


Figure 6.5: Schematic view of the time-scaling algoritmm (Algorithm 13). The black line is the base signal ω_s while the red line is another donor signal ω_s that is transformed by the algorihtm.

that it has a very small value. There are many ways to compute t_0, but accuracy must be very high in estimating it. A specific way is to view the time series in the spectral plane and using the imaginary part of DFT to be used as the accuracy parameter, as described in the next subsection.

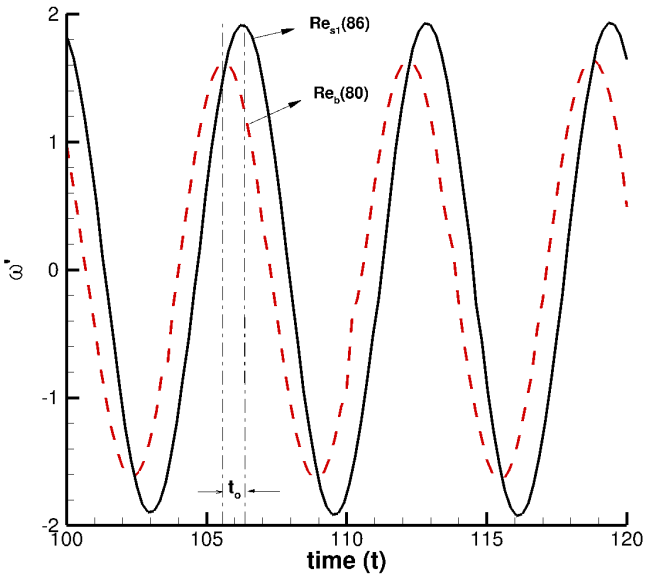


Figure 6.6: Variation of disturbance vorticity at a point (0.504, 0.0) with t_b and t_s for Re_b and Re_s, respectively, for the pair of $Re_b = 80$ and $Re_s = 86$ in the bifurcation sequence $78 \leq Re \leq 90$.

6.1.2.1 Computing the initial time-shift (t_0)

The present method is both accurate and computationally cheap, since it relies on the fast Fourier transform (DFT) that is provided in the `numpy` library. A DFT is applied to the vorticity time series at one relevant space point. On one hand, for the LDC problem $(0.95, 0.95)$ is used. On the other hand for the flow past a circular cylinder, point $(0.504, 0.0)$ in the cylinder wake is adequate. For each sampled frequency, a complex value $(z(f) = Ae^{i\theta})$ is obtained consisting of the modulus (A), which corresponds to the amplitude and a phase (θ). Consequently, we can recover the phase associated with the

leading frequency (L) for both signals θ_b and θ_s. Finally the time shift of signal s with respect to the signal b is given by

$$t_0 = \frac{\theta_b^L - \theta_s^L}{2\pi f^L} \tag{6.1.7}$$

Here, f^L is the lead frequency in the amplitude spectrum for both the signals as t_0 is computed only after the frequency scaling has been performed, with θ as the angle of the complex value of the DFT associated with the lead frequency for signal b or s. This method yields reliable and accurate values of t_0, as the ROM accuracy will prove in the following sections.

6.1.3 Time-scaling ROM algorithm

In this subsection, a short recap of the time shifting procedure for ROM building is given for the simple case of discrete signals $\omega_b(t_i)$ and $\omega_s(t_i)$ with $\{t_i\}_{i=1}^N$ indicating the time discretization. It can be directly applied to any space-time dependent field, with a reference signal chosen at a reference point. Fig. 6.7 provides a schematic view of the time scaling ROM which is built as follow:

a. Perform Algorithm 13 on all signals, except the base donor signal, in order to scale their oscillations. Fig. 6.5 provides a schematic view of Algorithm 13.

b. Perform Lagrange interpolation on the scaled donor signals at target Re_t for all discrete times t_i.

$$\bar{\omega}^\star(t_i) = \sum_{s \in \text{donors}} \hat{\omega}_s(t_i) l_s(Re_t) \tag{6.1.8}$$

where $\bar{\omega}^\star$ is the target signal and l_s are the Lagrange interpolation polynomials.

c. Scale-back $\bar{\omega}^\star$ to the physical time with $t^\star = \frac{t - t_0(Re_t)}{(Re_b/Re_t)^n}$.

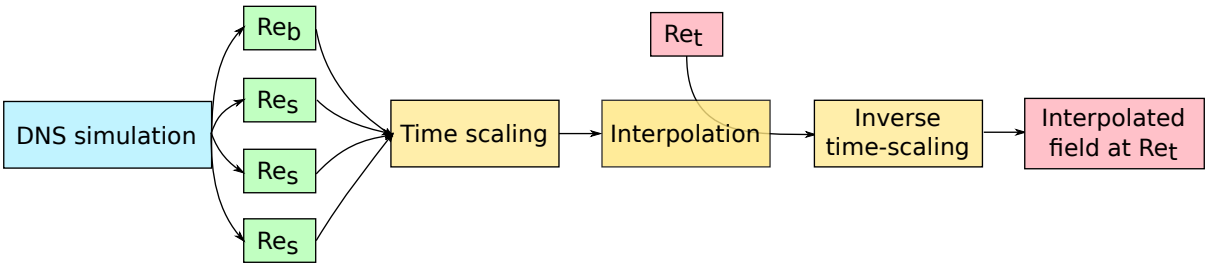


Figure 6.7: Schematic view of the time scaling interpolation method.

The last step of the ROM is to scale back $\bar{\omega}^{\star(t)}$ to the physical time, $t^\star$. Indeed, the interpolation is performed at grid points for t, which is actually the time-scaled representation of the target vorticity field. Thus the scale-back operation is computed to associate $\bar{\omega}^\star$ with the scaled-back time $t^\star$. One should note that the final domain is cropped according to the information lost after each shift, despite this the discrete time points match the original discretization.

Algorithm 13: Time-scaling algorithm for discrete signals

input : $\omega_b, Re_b, \omega_s, Re_s, t = \{t_i\}_{i=1}^{N}$
output: $\hat{\omega}_s$; `/* the time scaled signal. */`

1 Perform DFT on both signals
2 Scale frequencies $\left(C = \left(\frac{Re_b}{Re_s}\right)^n\right)$
3 Evaluate $t_0(Re_b, Re_s) = \frac{\theta_s^L - \theta_b^L}{f_s^L 2\pi}$
4 New time $t_s = Ct + t_0$ is associated with ω_s
5 Interpolate the *time-scaled signal* $\hat{\omega}_s(t)$ from $\omega_s(t_s)$
`/* At this point, one can perform Lagrange interpolation between the donor points to the target Re to obtain` $\bar{\omega}^*$ `*/`
return $\hat{\omega}_s$

6.1.4 Time-shifting ROM applied to the LDC flow

As it was discussed in chapter 5 (with frequency values table in [LBA+18]), the main frequency of the LDC flow is nearly constant across large ranges of Re. It emphasized here in Fig. 6.8. Thus, the time-scaling procedures simplify to a time-shifting procedure with $n = 0$, resulting in $t_s = t - t_0$ for the donor and target points, which have the same frequency in figure 6.8.

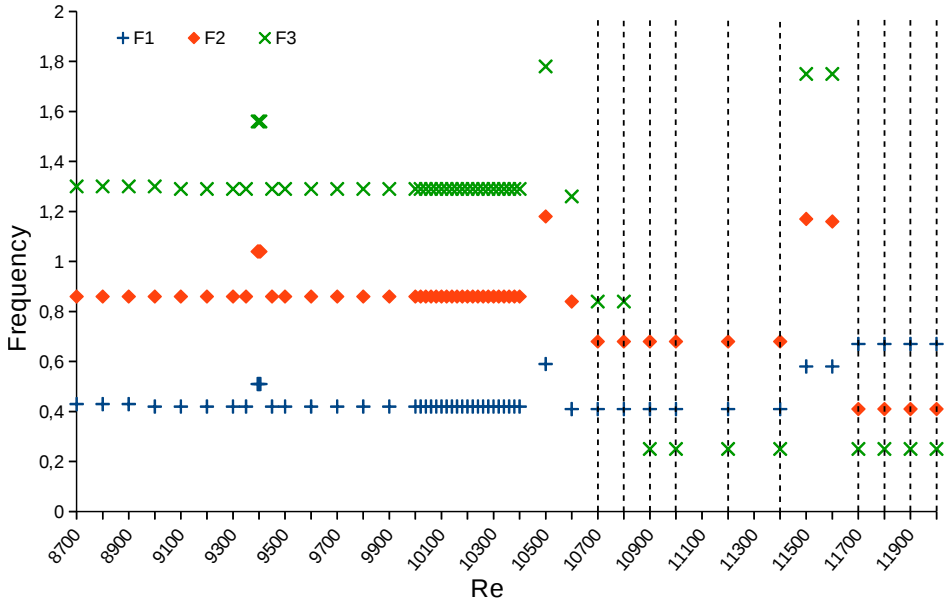
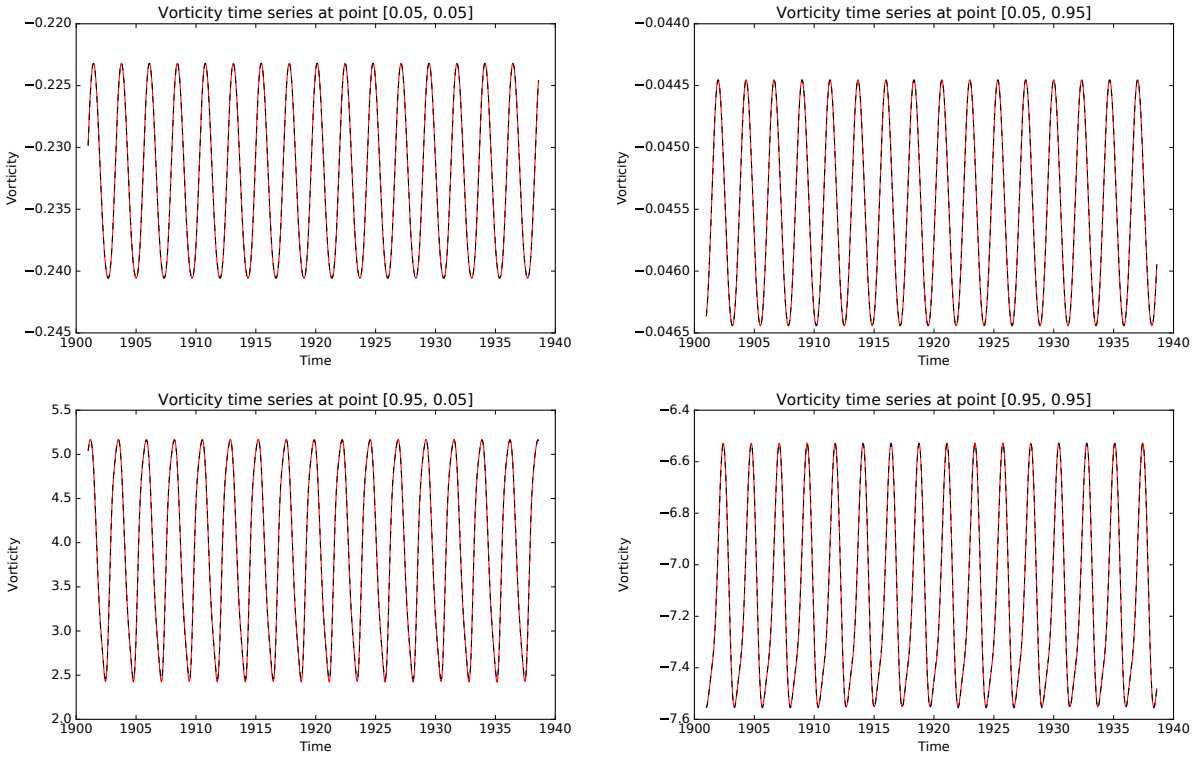


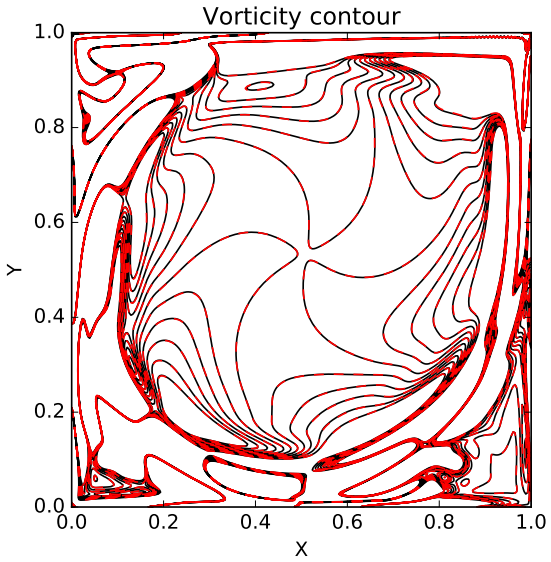
Figure 6.8: Frequency variation for Re=[8700,12000] for the first three leading frequencies of the vorticity time series at point $(0.95, 0.95)$ obtained for the last 50 periods. The dotted lines indicate the presence of multiple dominating peaks in the spectrum.

Following Algorithm 13, we have obtained the vorticity field for Re = 10040, using the donor points at Re = 10000, 10020, 10060 and 10080. From the reconstructed ROM data, we have shown the vorticity time series in figure 6.9a for four representative points near each corners. Despite the change in the vorticity magnitude by two orders, the accuracy of reconstruction is excellent and match almost exactly.

In figure 6.9b, the reconstructed vorticity contours inside the LDC is shown for Re = 10040, at the indicated time of $t = 1900.199$ by solid line, with the same donor data of



(a) Reconstructed vorticity field at points near each corner of the cavity.



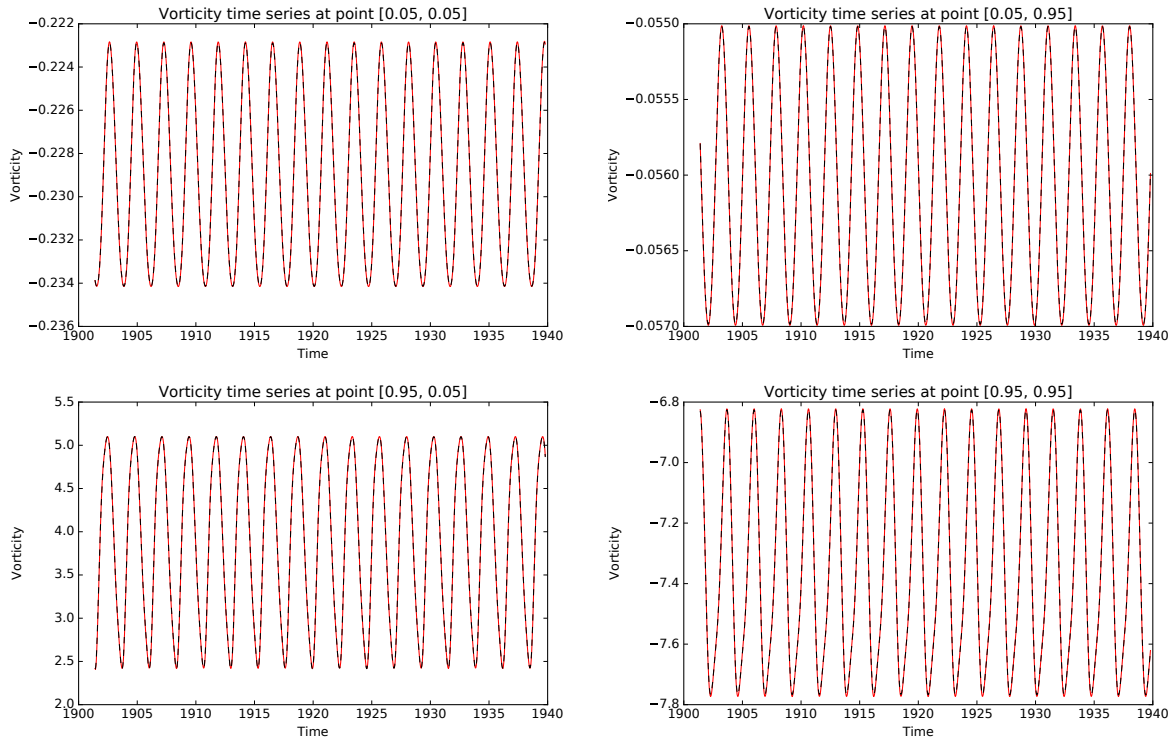
(b) Disturbance vorticity contour plot at nondimensional time $t = 1900.199$.

Figure 6.9: Reconstructed vorticity (solid lines) and DNS vorticity (dotted lines) field for target Re=10040 with donor points at Re = 10000, 10020, 10060 and 10080.

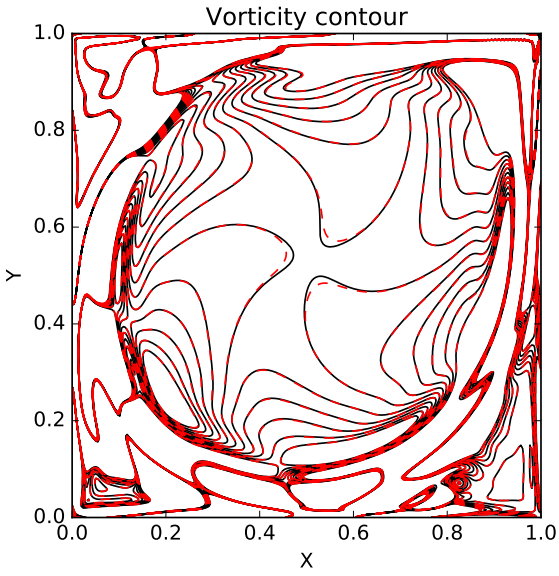
Re's for the use in the ROM following algorithm 13. The corresponding solution obtained by DNS of NSE-Solution for Re=10040 is shown in the same figure by dotted lines. It is readily observed that these exact and ROM solution overlap each other in the full domain with a relative root mean square (RMS) error of 7.1×10^{-4}.

It shows the special case of a flow, which is multi-periodic with respect to time, yet the predominant frequency remains constant over different ranges of Re, allowing one to use the special version of time scaling with power law exponent given by, $n = 0$ in equations

(6.1.4) and (6.1.5). Thus, one needs to simply apply a time-shift and reconstruct by the methods described in subsections 6.1.2.1 and 6.1.3.



(a) Reconstructed vorticity field at points near each corner of the cavity.



(b) Disturbance vorticity contour plot at nondimensional time $t = 1900.199$.

Figure 6.10: Reconstructed vorticity (solid lines) and DNS vorticity (dotted line) field for target Re = 9600 with donor points $\{9350, 9500, 9800 \text{ and } 10000\}$.

Next, ROM is performed for Re = 9600, with the donor points at Re = 9350, 9500, 9800 and 10000. The choice of the second target Re for LDC is made on purpose, as the bifurcation diagram in figure 6.4(a) shows that the flow has discontinuity in equilibrium amplitude in the chosen donors the bounds of R_{III} for Re = 9400 and 10600. The interpolated vorticity time series are compared with direct simulation results, as shown in

figure 6.10a, at those same sampling points used in figure 6.9a. Once again the match is excellent between interpolated results with DNS data with a very low RMS error of 5.6×10^{-4}.

In figure 6.10b, the interpolated vorticity contours for Re = 9600 are compared with those computed directly from NSE to show that interpolation works globally in the flow field and not merely at chosen sampling points. In this flow field, the power law exponent is zero and the strength of the interpolation is in obtaining the initial time shift (t_0) obtained using algorithm 13 , obtained from the DFT of the donor point vorticity with respect to the baseline Re chosen.

In the following, we study the case of flow past a circular cylinder to show the efficacy of the proposed time-scaling algorithm used here. For this flow also one notices presence of multiple time scales, but with a predominant frequency characterized by St, which follows the power law given by equation (6.1.4), with nonzero power law exponent, n.

6.1.5 Time-scaling ROM applied to the flow past a cylinder

All the time-scaled relation and corresponding power law exponent in equation (6.1.5), is applicable here for ROM with ω obtained by DNS. The time scaled interpolation of the ROM for disturbance vorticity for different combination of donor points, as indicated in Table 6.2, are obtained and RMS error with respect to DNS data are compiled in the table summed over all the points in the domain. Case I in the table corresponds to the case of donor points at Re = 78, 80, 86 and 90, which is noted as the most accurate based on RMS error for the ROM reconstruction for Re =83. When we choose the donors with Re = 55, 80, 86 and 130 for Case V in Table 6.2, the RMS error is again low, as compared to cases where only one donor point is taken from the same segment containing the target Re. For higher accuracy one must choose donor points from the same segment of target Re, as clearly shown in Table 6.2 in a quantitative manner.

Table 6.2: RMS Error estimates of interpolation for Re = 83

Cases	Re of donor points	Error for interpolation using donor points
I	(78,80,86,90)	0.0435
II	(72,80,86,90)	0.0439
III	(68,80,86,90)	0.0446
IV	(55,80,86,90)	0.0625
V	(55,80,86,130)	0.1409
VI	(55,68,72,86)	1.3160
VII	(55,68,72,130)	8.5224

We draw the attention on error estimates provided in Table 6.2 for different combinations of donor Re's. It is evident from the table that the best result is obtained when all four donor points are in the same segment of target Re, as in Case I. In Cases II to IV, we have taken the lowest Re, farther to the left with increase in RMS error, with lowering of the smallest donor Re. But in Case V, the extreme Re's are chosen as 55 and 130, and yet the RMS error is acceptable, as two of the donor Re's belong to the segment of target Re. In contrast, for the Case VI, only a single donor Re belongs to the same segment, resulting in RMS error increasing almost ten folds as compared to the Case V. The worst case (Case VII) occurs in Table 6.2, when all the donor Re's are outside the target Re

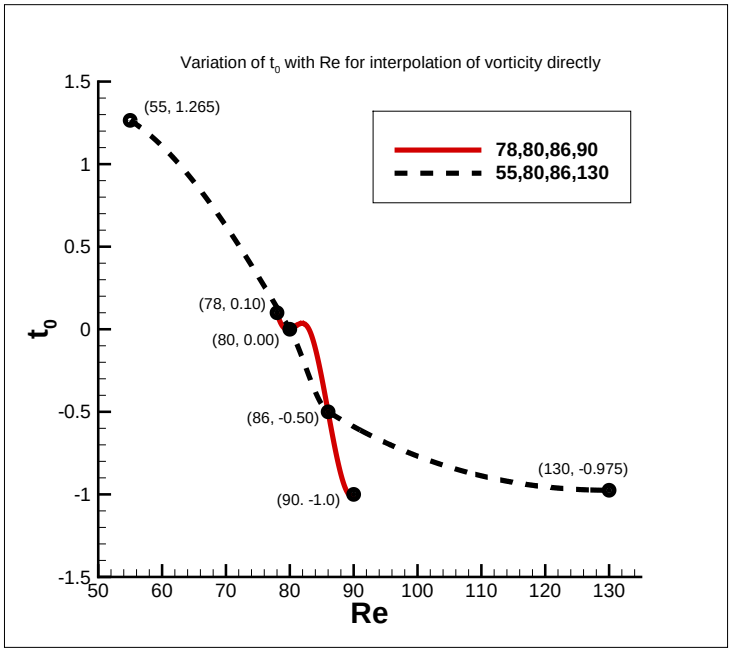


Figure 6.11: Variation of t_0 with Re_s for $Re_b = 80$ for Case I (solid line) and Case V (dashed line) of Table 6.2. Shown in parametric form are the pair of Reynolds number and corresponding optimal t_0.

segment. This justifies principle of the adopted ROM keeping the various ranges of Re punctuated by various Hopf bifurcations shown in figure 6.4(b).

The role of t_0 is also investigated here for ω' (the disturbance vorticity field) and the variation of t_0 with the Re is shown in figure 6.11 in the subrange $55 \leq \text{Re} \leq 130$. Here, we obtain t_0 for the data sets of (Re= $55, 80, 86, 130$) and (Re= $78, 80, 86, 90$), as indicated separately in the figure. Each of the discrete data are marked in the figure with Re and necessary time shifts in brackets, with $Re_b = 80$. It is noted that the finding of single t_0 is far easier and less time consuming for ω' for the present version of ROM, as compared to any method using POD or instability modes, which would require finding different t_0 for each retained modes.

In this method, ω' is reconstructed using the identical procedure of interpolation after time-scaling and initial time-shift, using equation 6.1.6 applied directly on ω obtained by DNS. Thus, this procedure even circumvents the need to use the time-consuming method of snapshots to obtain POD modes that is required for any POD based ROM e.g. POD-Galerkin, interpolated POD. The proposed time-scaling ROM requires storage of at most four DNS data sets in each segment for most accurate reconstruction. If one is willing to settle for lesser accuracy, then one can reduce the requirement of performing DNS for two Re only, in each segment of figure 6.4. Hence this ROM is not memory intensive.

Figures 6.12(a) and (b) show the comparison between DNS and the time-scaled interpolated ω' at two different points for Re=70, located along the wake-center line at $(0.504, 0.0)$ and at $(1.014, 0.0)$, respectively. Excellent match with the DNS data even in the transient state proves the efficacy of the time-scaling interpolation technique applied to vorticity data. It is to be noted that despite the presence of a dominant St, the physical variables demonstrate multiple time-scales as discussed in the introduction and shown in figure 6.1.

The case for Re= 83 are shown in figures 6.12(c) and (d), which compare the disturbance vorticity at the same two locations with DNS data. Once again, the reconstructed

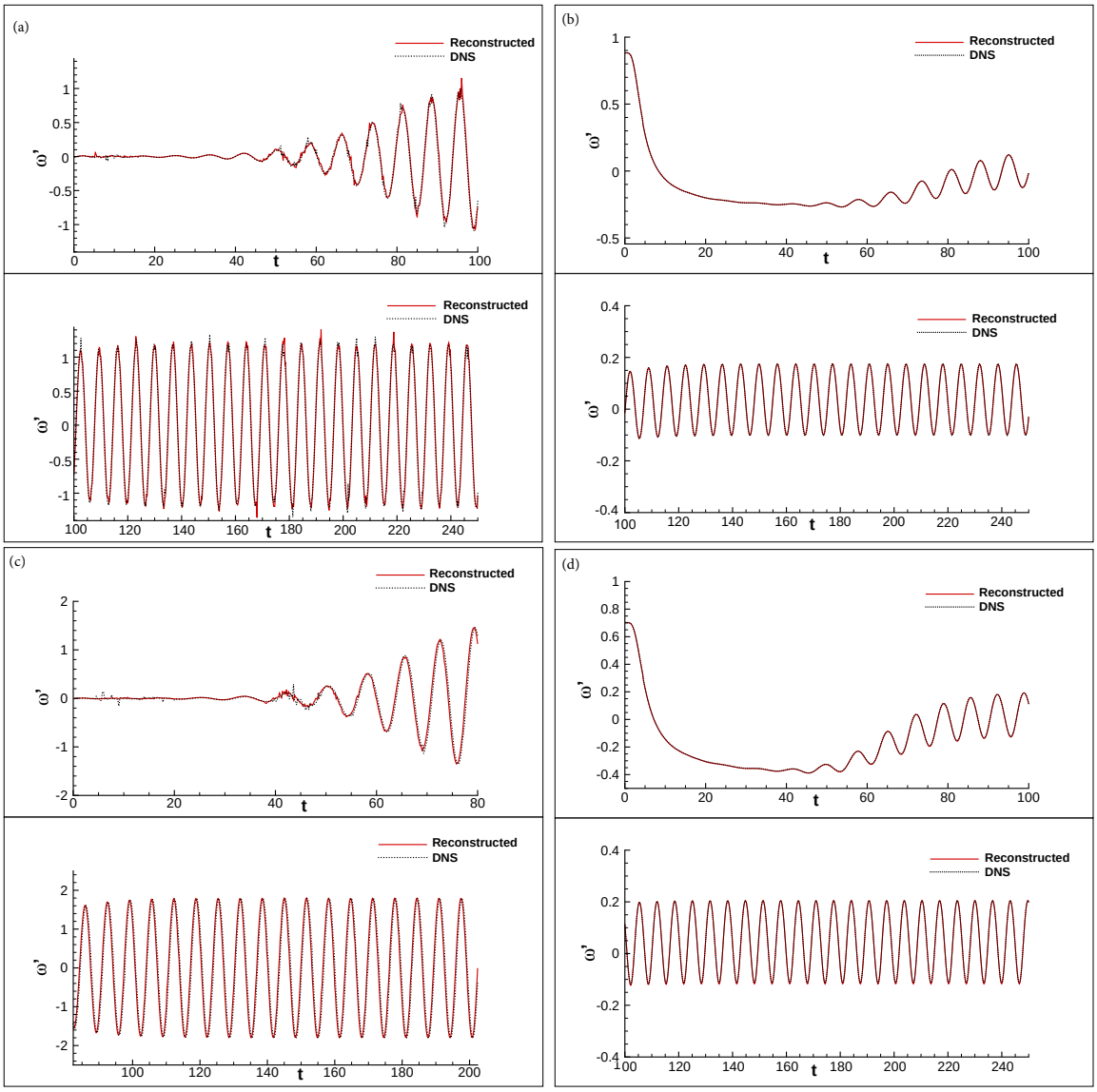


Figure 6.12: Reconstructed disturbance vorticity with time-scaling interpolation for (a) Re = 70 using Re = 68, 72, and 76 at $(0.504, 0.0)$, (b) at $(1.104, 0.0)$ and (c) Re = 83 using Re = 78, 80, 86 and 90 at $(0.504, 0.0)$ and (d) at $(1.104, 0.0)$. Within each subfigure, the top frame is for comparison at early times, while the bottom frame shows comparison at later times.

ROM solution is indistinguishable from the corresponding DNS data. Thus, it is evident that spectrum with multiple peaks can be handled by the presented approach of time-scaling with initial time-shift, utilizing the power law between Re with St.

6.1.6 POD and time-scaling.

One may object that storing 4 full resolution DNS is too expensive for large scale applications. A natural idea is to couple POD and time scaling. POD provides a reduced set of modes that can be interpolated and assembled into a new reduced order representation. This would provide cheap storage and cheap evaluation ROM. The process reads as follow for the simple case of discrete vorticity fields $\omega_b(\boldsymbol{x}_i, t_j)$ and $\omega_s(\boldsymbol{x}_i, t_j)$ with $\{t_j\}_{j=1}^{n_t}$ and $\{\boldsymbol{x}_i\}_{i=1}^{N}$:

a. Apply POD to each field $\omega(\boldsymbol{x}_i, t_j) = \sum_{k=1}^{r} \phi_k(\boldsymbol{x}_i) a_k(t_j)$ with $r \ll n_t$,

b. apply time scaling procedure to times modes of same index $((a_k^s, a_k^b))$,

c. interpolate space modes at target Re $(\phi_k^b(\boldsymbol{x}_i), \phi_k^s(\boldsymbol{x}_i))$,

d. Reconstruct interpolated field $\omega_t(\boldsymbol{x}_i, t_j) = \sum_{k=1}^{r} \phi_k^t(\boldsymbol{x}_i) a_k^t(t_j)$

POD applied to LDC has already been presented in details in section 5.2 so we directly focus on time scaling procedure applied to time POD modes.

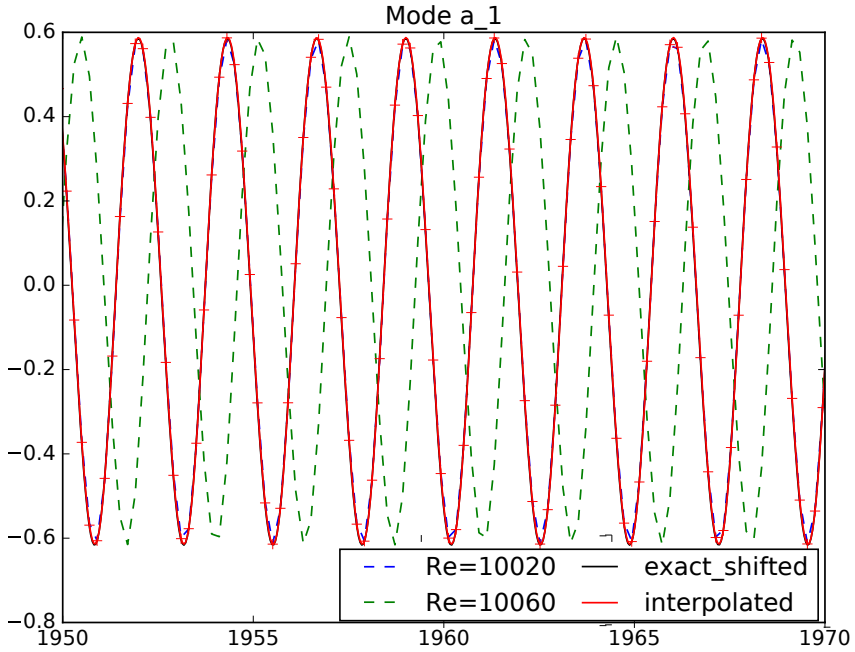
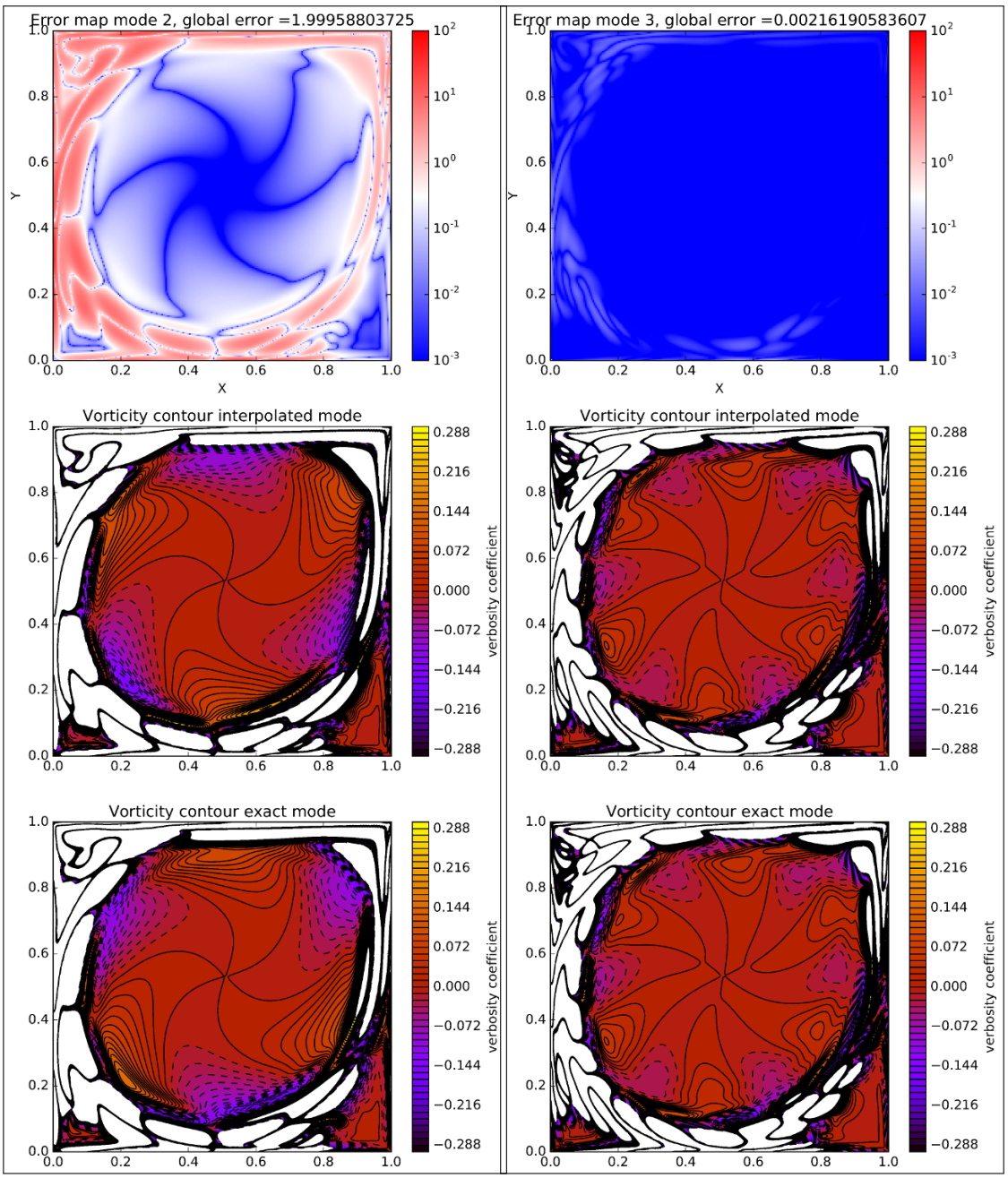


Figure 6.13: Time scaling applied to first time mode, target Re=10040 donor points at Re = 10000, 10020, 10060 and 10080. No back scaling applied.

Fig. 6.13 shows that time scaling procedure is very efficient at interpolating POD modes. Black line is the actual first time mode of target Re vorticity field (shifted to reference time of Re=10020) while the red line is the time scaled interpolation of donor Reynolds number {10000, 10020, 10060,10080}. One can see the very good match of the interpolated mode making it indistinguishable from the exact one. The shifting procedure is emphasized by presenting the green (Re=10060) and blue (Re=10020) dotted line for which direct interpolation would have resulted in canceling peaks.



(a) Mode 2 phase miss. (b) Mode 3 phase hit.

Figure 6.14: Lagrange spatial interpolation of target Re=10040 donor points at Re = 10000, 10020, 10060 and 10080.

Spatial modes interpolation can be achieved directly with Lagrange interpolation. But, as shown in Fig. 6.14, direct interpolation can result for the same simulation into vastly different outputs across the modes. The left frame shows that the interpolation of mode 2 produces a mode that looks like the expected modes but with the wrong sign, this leads to a global error near 2. Meanwhile many modes, including mode 3 (right frame), are interpolated with very good precision $\mathcal{E} = 2.2 \times 10^{-3}$. In order to circumvent the sign error, one can add a correlation evaluation $c = \int_\Omega \phi_k^s \phi_k^s$ and change the sign accordingly. But this is equivalent to a time shift of a quarter period for the associated time mode. How to account for this sign change in the reconstruction is not clear yet.

Scaling back. Actually, the biggest issue with time scaling applied to POD modes is to carry out the last step of scaling back. Indeed, for full DNS field it was satisfying enough to interpolate t_0 and f to reconstruct limit cycle data. Here one needs to reconstruct as many t_0 and f as there are modes, f is constant across ranges so interpolation is not an issue. t_0, however, is particularly tricky as it does not seam to present any regularity for LDC data. Additionally, one needs to ensure that the quarter phase shift between mode pairs (see section 5.2) is preserved. These difficulties have prevented us to construct a satisfying POD-Time-scaling interpolation ROM so far.

Summary and Conclusion

Here, we have proposed time-scaled ROM for reconstructing super-critical flow past circular cylinder and flow inside LDC using time-scaled Lagrange interpolation of vorticity data obtained by DNS for different donor data at Re's, largely located in the neighborhood of the target Re. In performing the interpolation, a time-scaling is performed following equation (6.1.6) along with an initial time-shift, as a direct consequence of (St, Re)-relations given in equations (6.1.4) and (6.1.5).

ROM reconstruction at a target Re is of DNS-quality, if all the donor points belong in the same Re subrange, identified by multiple Hopf bifurcations in figure 6.4(a), for flow inside the LDC in the range $8700 \leq Re \leq 12000$ and in figure 6.4(b) for flow past a circular cylinder, in the range of $55 \leq Re \leq 130$ and in Table 6.1.

Data requirement of present ROM is at most for four Re's located in the same subrange. If one wants to perform ROM with only three Re's, then the reconstructed data are of slightly lower accuracy, but of very acceptable quality (not shown here). The present procedure provides scientific and applied basis of ROM, depending upon the number and location of donor points of target Re.

The proposed ROMs can be used at any arbitrary Re on demand, by the proposed ROM performed with limited number of DNS at neighboring Re's. The novel procedure proposed here has been tested for the internal flow inside a LDC and an external flow over a circular cylinder, as proofs of concept.

Finally, it is tempting to couple POD and time scaling, preliminary results show that the interpolation of spatial and temporal modes is possible and accurate but the inverse time scaling and in particular the reconstruction of t_0 at target Re impairs our ability to propose a POD-time-scaling ROM.

Conclusion on CFD and ROM

In the second part of this manuscript, we have seen that complex flows can be very challenging simulation problems. Indeed, chapter 5 a complete study of singular lid driven cavity flow has been conducted. We have shown that this problem is particularly sensitive to numerical setup which is why high accuracy CCD schemes were used. This setup has allowed us to precisely describe the Hopf bifurcation sequence thanks to standard flow analysis tools (time series, discrete Fourier transform,...). POD analysis of the flow has confirmed these observation. Moreover it has provided qualitative analysis of the flows for different grids (257×257 and 513×513) as well as marked differences in pattern before and after secondary instabilities.

This kind of in-depth analysis has been used in chapter 6 to propose a "physics" based interpolation coined time-scaling. The idea originated from St-Re relation in flow past a cylinder experiments [FKE98]. Here, instead of using general Grassmann manifold interpolation, the time series are scaled and shifted to prevent interpolation induced beat phenomenon. The goal of this method was to provide interpolated solution at a target Re from a few donor Re (between two Hopf bifurcations) that were precomputed through DNS. This method has been successfully applied to LDC flow (limit cycle) and flow pas a cylinder (onset and limit cycle) with typical RMS errors in $\mathcal{O}(10^{-4})$ using a few as 3 donors.

Conclusion and perspectives

Conclusion

In the era of super computers, scientific computing is confronted more than ever to the curse of dimensionality. In this thesis we have explored a new paradigm that aims at solving this paradox. The general approach is to break the dimensionality with methods that turn exponential growth with respect to the number of dimension into linear growth. This approach is two fold. First, data decomposition techniques aims at reducing existing data in order to facilitate storage and manipulation. Second step is to build reduced order models that solve slightly different problems with acceptable loss of accuracy but for considerable decrease of computing time (at least in the on-line phase). Often, low rank bases obtained with data decomposition are used which is why it is often referred to as off-line phase. Obviously, complex problems require extended analysis prior to building such ROMs.

In the first part of this document, data low rank approximation was studied and programmed into a library, aiming both at compression and further use in ROM for CFD problems. In the second part, a complex benchmark flow, singular lid driven cavity flow at high Reynolds was studied and a novel *time-scaling* interpolation ROM was applied successfully to both LDC and flow past a circular cylinder. The contributions of this thesis can be summarized as follow.

Bivariate decomposition It was shown that bivariate decompositions are equivalent mathematically, they include matrix decomposition through SVD and function decomposition through POD or PGD. By equivalent, we mean that they perform the same operation on different spaces or norms. Their usual definitions involve different algorithms that can be tweaked into one another. This is supported by numerical implementation as long as convergence is reached. We have also shown that these decompositions can help and improve analysis of physical data. It was highlighted that some fields are more separable than other. Consequently, they have been deemed weakly separable and strongly or exponentially separable. Extensive insight on numerics has been provided.

Tensor approximation A broad review of tensor formats and decompositions was provided in order to contribute to the diffusion of this approach in computational fluid dynamics laboratories, starting with I2M. To do so, a complete description of these objects, their comparative advantages and algorithms have been provided. The theoretical aspect indicates that canonical decomposition, in spite of its d-linear storage cost, will produce poor approximation since the problem is NP-complex. Tucker decomposition is composed of modes and a correlation tensor core of the same order but of much smaller size than the original tensor. This structure makes it particularly suitable for decomposition of low order tensors by successive SVDs but larger dimension will cause exponential growth of the core tensor. Finally, TT and Hierarchical formats are recently introduce format that grow linearly with d while

presenting SVD based decomposition. That makes them good candidates for decomposition of high to very high number of dimension. $d = \mathcal{O}(1000)$ is perfectly accessible, which leads to the new practice of tensorization. Also, the distinction between formats an their associated decomposition has been highlighted to prevent prejudicial confusion.

Multivariate decomposition Tensors describe "blind" data and may not take advantage of the properties of fields or functions that are routinely encountered in CFD. This is why a particular care was given to decomposition methods in the continuous framework. First, a degraded version of the PGD, that we referred to as *a priori PGD*, was studied. It was concluded that this iterative method actually produces a canonical decomposition with an iterative least square enriching approach. It can be seen as a generalization of standard ALS algorithms. Then, a recursive generalization of POD to dimensions higher than 2 was presented. Although it can be written as a canonical format (by renumbering the sum indices) or as a tucker format (by introducing a sigma map tensor) decomposition, these manipulation introduce a lot of redundant information which is adverse to compression rate. Consequently, an atypical recursive tree structure has been proposed to represent RPOD data. It could be viewed a hierarchical tree, but usual definitions (see [GKT13]) involve binary trees only. Nevertheless, the RPOD recursive tree is a new format. Of course a SVD implementation is possible with the same properties. Finally a bridge was drawn between tensor decomposition algorithm and continuous equivalent versions. Notably, adaptation of Tucker format methods such as ST-HOPOD and TT-HOP are straightforward and produces orthonormal bases with respect to any POD compatible scalar product. Yet, they necessitate the introduction of integer parameters and associated measure.

Numerical comparison Most of the added value of this thesis on the question of data compression and decomposition lies in the comprehensive numerical study of these algorithms. Thanks to the computing library `pydecomp` developed for this purpose, experiments where conducted on synthetic, experimental and numerical simulation data. It was shown that ST-HOSVD and TT-SVD methods are the most efficient for comparing compression rate. In accordance with the theoretical study, the tipping point between these two methods is around 5 even though many parameters may counteract slight compression rate differences. For instance, for low precision compression, the rank is low enough for the core tensor to remain small. For physics related problem, the data layout of space related variables have been studied. It was concluded that separating space variables leads to better compression rate overall due to the global measuring of the error. But, conserving space as a single dimension provides a sharper description of spatial feature. Also, data representation of vectorial or multi-field data was discussed. Once again, there is no general rule emerging yet, It was witnessed that although correlated, variables decomposition possesses distinct compression rates. Still, for large data set it seems that introducing a new dimension that represents each variable is an efficient way to diminish memory use.

LDC analysis Next, singular lid driven cavity, a complex flow, has been studied thoroughly for $Re \in [8000, 12000]$. Standard means of investigation, such as time series analysis, discrete Fourier transform, linear regression, etc., have been used to propose a bifurcation series scenario. Additionally, it was pointed out that this flow is extremely sensitive to numerical setup which is why we relied on high accuracy CCD scheme for all DNS performed in this work. For instance, startup condition mod-

ify the onset of stable limit cycle and grid resolution changes the critical Reynolds values. This has motivated the introduction of manual excitation of the flow that enabled the construction of a more robust Hopf bifurcation scenario. Analysis of POD modes on this flow has confirmed the qualitative equivalence of bifurcation sequence for fine and coarse grid. Furthermore, it has provided additional clues for secondary instability, giving sufficient information to decide whether a stable limit cycle has been reached or not, even though direct analysis of time series would not provide enough information to conclude. A distinct feature of POD decomposition is that the time modes can be categorized relative to their physical role. The categorization given by Sengupta has been supported by this work. Indeed, regular modes, characteristic of limit cycle, are grouped by pair of identical frequency that DFT analysis have shown to correspond to a single peak in the spectrum. Anomalous modes are characteristic of transient behavior such as mean variation or instable oscillations. This complete analysis has contributed to the definition of attainable objectives for ROM building on LDC. Transient stage is far too unstable and sensitive for any attempt of ROM to reproduce DNS pattern. Consequently, a ROM of the limit cycle, between two bifurcations seemed a reasonable objective.

Time-scaling interpolation ROM Finally, a new interpolation ROM called *time–scaling* was proposed. It allows DNS vorticity field interpolation from a few donor Re to a target Re with high accuracy ($\mathcal{O}(10^{-4})$). In this ROM, instead of devising a technique based on the topology of the PDE solution manifold, it was proposed to rely on physical insight. Indeed, it was noted both experimentally and numerically that the Strouhal number can be related to Re by a power low rule. This rule is adapted into a transformation of time that scales (and shifts) time series to prevent beat phenomenon. Then any standard interpolation technique can be used. Since 3 or 4 donors are enough, Lagrange interpolation was chosen here. This new method was applied successfully on LDC within ranges defined during the analysis. Actually, the frequencies have been shown to remain constant in each range, thus, only shifting needs to be applied. Flow past a circular cylinder has also been interpolated successfully with full time-scaling algorithm, including the initial transient till the stable limit cycle (Von Karman street) for $Re \in [55, 130]$.

In the end, this work has lead to the publication of two articles and the submission of a third that are included at the end of the manuscript:

- [LBA+18] Lucas Lestandi, Swagata Bhaumik, G R K C Avatar, Mejdi Azaïez, and Tapan K Sengupta. *Multiple Hopf bifurcations and flow dynamics inside a 2D singular lid driven cavity.* Computers and Fluids, 166:86–103, 2018.

- [LBS+18] Lucas Lestandi, Swagata Bhaumik, Tapan K Sengupta, G R Krishna Chand Avatar, and Mejdi Azaïez. *POD Applied to Numerical Study of Unsteady Flow Inside Lid-driven Cavity.* Journal of Mathematical Study, 51(2):150– 176, 2018.

- Tapan K Sengupta, Lucas Lestandi, S. I. Haider, Atchyut Gullapalli and Mejdi Azaïez, *Reduced order model of flows by time-scaling interpolation of DNS data*, (*submitted to AMSES on April 30th 2018*).

Future work

As we have seen in the introduction, ROM and data decomposition is a rapidly expanding field, especially in the context of CFD. Many efforts are concentrated on applying

efficiently to the Navier Stokes equations methods that have proved efficient on simpler problem (elliptic, coercive,...). Regarding the initial step of data decomposition, recent works have been focusing of blackbox methods that allow to enormous memory savings. The present thesis has brought perspectives on its own, that I present hereunder.

`pydecomp` computing library

We have presented in section 4.1 a decomposition library was that developed during this thesis. It proposes various formats and decomposition methods. But, the main limiting factor is memory use. Indeed, current implementation is limited to a single computing node and its memory. Additionally, all available decompositions require knowledge of every entries of the field/tensor which amounts to very large datasets. Current capacity is limited to a few GB at a time. In order to overcome this limitation, two approaches are possible. First implementation of fully parallel solvers, typically iterative, would allow to distribute data on many computing nodes. Current cluster architectures provide hundreds of TB which is sufficient for processing of advanced large scale simulation data. The other approach is to consider that one can access, on-demand, any entry of the tensor without having to load the full tensor. This would imply huge memory savings and potentially cutting computing efforts since one only needs a limited number of evaluation in the parameter space to construct a general approximation of the tensor.

Also, from a practical point of view, interfaces with some common format (VTK, ADIOS, HDF5) have been developed; this effort should be continued with providing a user interface and easy-to-use reconstruction features.

Better interpolated ROM

Time-scaling interpolation ROM has been applied successfully onto full DNS vorticity data. However, it can be expensive to store such data for 3D refined cases with small time steps. Then, applying time-scaling ROM directly onto a decomposed representation, typically POD, is a short term goal. But as shown in section 6.1.6, having multiple time modes involves reconstructing many time shifts. This problem has not been circumvented yet. Other approaches such as direct interpolation of Re modes for full decomposition have been discussed but accuracy remains low. This suggests that methods accounting for the topology of the Grassmann manifold could improve as well as generalize the proposed method.

A stabilized POD Galerkin ROM

It was evoked several times that POD Galerkin projection ROM have been used in numerous work for the last two decades. But it turns out to be unstable due to truncation of lower energy modes. In collaboration with Instituto de Matemáticas Universidad de Sevilla, together with Samuele Rubino, we have undertaken work to propose a new stabilization scheme for POD Galerkin ROM (PODG).

In PODG, the idea is to project a set of PDEs onto a spatial reduced basis obtained by POD. For NS equations, with incompressible flows, the pressure term is dropped since POD preserves the divergence free property of the flow. If centered trajectories are used, additional terms appear associated with the mean field. Also, the trilinear form associated with advection introduces a third order tensor. A special processing may be required to evaluate this term, but the rank of the RB is usually small enough so that it can be kept entirely.

Thus, the idea is to introduce a new stabilization term that comply with the physics. Ongoing work stems from projection-based Variational Multi-Scale (VMS) ideas [IW14, Wel15] for the simulation of turbulent incompressible flows. In the FE context, stabilized formulations have been developed to deal with the numerical instabilities of the Galerkin method in strongly convection-dominated configuration. We are currently working on applying this approach to PODG stabilization.

Bibliography

[ABK16] Woody Austin, Grey Ballard, and Tamara G. Kolda. Parallel Tensor Compression for Large-Scale Scientific Data. In *Proceedings - 2016 IEEE 30th International Parallel and Distributed Processing Symposium, IPDPS 2016*, 2016.

[ABR+12] Jean Michel Alimi, Vincent Bouillot, Yann Rasera, Vincent Reverdy, Pier Stefano Corasaniti, Irène Balmès, Stéphane Requena, Xavier Delaruelle, and Jean Noel Richet. First-ever full observable universe simulation. In *International Conference for High Performance Computing, Networking, Storage and Analysis, SC*, 2012.

[ABR16] M Azaïez, F Ben Belgacem, and T Chacón Rebollo. Recursive POD expansion for reaction-diffusion equation. *Advanced Modeling and Simulation in Engineering Sciences*, (December), 2016.

[ACCF09] David Amsallem, Julien Cortial, Kevin Carlberg, and Charbel Farhat. A method for interpolating on manifolds structural dynamics reduced-order models. *International Journal for Numerical Methods in Engineering*, 80(9):1241–1258, 2009.

[ACDH10] A. Ammar, F. Chinesta, P. Diez, and A. Huerta. An error estimator for separated representations of highly multidimensional models. *Computer Methods in Applied Mechanics and Engineering*, 199(25-28):1872–1880, 2010.

[ACL15] Pierre-Eric Allier, Ludovic Chamoin, and Pierre Ladevèze. Proper Generalized Decomposition computational methods on a benchmark problem: introducing a new strategy based on Constitutive Relation Error minimization. *Advanced Modeling and Simulation in Engineering Sciences*, 2(1):17, 2015.

[ACM+15] M Azaiez, T Chacon Rebollo, M. G. Marmol, E Perracchione, and J M Vega. Real-time simulation of air-wall heat transfer in buildings using various tensor decomposition methods. *[PREPRINT]*, pages 1–15, 2015.

[AD81] C. J. Appellof and E. R. Davidson. Strategies for Analyzing Data from Video Fluorometric Monitoring of Liquid Chromatographic Effluents. *Analytical Chemistry*, 1981.

[AF08] D Amsallem and C Farhat. Interpolation Method for Adapting Reduced-Order Models and Application to Aeroelasticity. *AIAA Journal*, 46(7):1803–1813, 2008.

[AF11] David Amsallem and Charbel Farhat. An Online Method for Interpolating Linear Parametric Reduced-Order Models. *SIAM Journal on Scientific Computing*, 33(5):2169–2198, 2011.

[Ale15] Alen Alexanderian. A brief note on the Karhunen-Lo{è}ve expansion. 2015.

[ANR09] Imran Akhtar, Ali H. Nayfeh, and Calvin J. Ribbens. On the stability and extension of reduced-order galerkin models in incompressible flows : AAA numerical study of vortex shedding. *Theoretical and Computational Fluid Dynamics*, 23(3):213–237, 2009.

[APQ02] F. Auteri, N. Parolini, and L. Quartapelle. Numerical investigation on the stability of singular driven cavity flow. *Journal of Computational Physics*, 2002.

[AQV02] F. Auteri, L. Quartapelle, and L. Vigevano. Accurate ω-ψ spectral solution of the singular driven cavity problem. *Journal of Computational Physics*, 2002.

[ARF07] Quarteroni Alfio, Sacco Riccardo, and Saleri Fausto. *Méthodes Numériques, Algorithmes, analyse et applications*. Springer Milan, Milan, 2007.

[Aza] Mejdi Azaiez. High Order PGD Method Applied to some Elliptic Problems.

[AZW15] David Amsallem, Matthew J. Zahr, and Kyle Washabaugh. Fast local reduced basis updates for the efficient reduction of nonlinear systems with hyper-reduction. *Advances in Computational Mathematics*, 41(5):1187–1230, 2015.

[Bal12] Jonas Ballani. *Fast evaluation of near-field boundary integrals using tensor approximations*. Phd, University of Leipzig, 2012.

[BBI09] Michel Bergmann, Charles-Henri Bruneau, and Angelo Iollo. {E}nablers for robust {POD} models. *J. Comput. Phys.*, 228:516–538, 2009.

[BCIS06] M Buffoni, S Camarri, A Iollo, and M V Salvetti. Low-dimensional modelling of a confined three-dimensional wake flow. *J. Fluid Mech.*, 569:141–150, 2006.

[BEkM16] Daniele Bigoni, Allan P Engsig-karup, and Youssef M Marzouk. Spectral tensor-train decomposition. *SIAM Journal on Scientific Computing*, 38:1–32, 2016.

[Ber04] Michel Bergmann. *Optimisation aérodynamique par réduction de modèle POD et contrôle optimal. Application au sillage laminaire d'un cylindre circulaire*. PhD thesis, Institut National Polytechnique de Lorraine / LEMTA, 2004.

[BG09] V. B. L Boppana and J. S. B. Gajjar. Global flow instability in a lid-driven cavity. *Int. J. Num. Meth. Fluids*, (62):827–853, 2009.

[BG14] Jonas Ballani and Lars Grasedyck. Hierarchical tensor approximation of output quantities of parameter-dependent PDEs. 3:1–19, 2014.

[BGK10] Jonas Ballani, Lars Grasedyck, and Melanie Kluge. Black Box Approximation of Tensors in Hierarchical Tucker Format. *Linear algebra and its applications*, 438(2):639–657, 2010.

[BGW15] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):1–49, 2015.

[BHL93] Gal Berkooz, Philip Holmes, and Jl John L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1971):539–575, 1993.

[Big14] Daniele Bigoni. Uncertainty Quantification with Applications to Engineering Problems. 2014.

[BKO17] Brett W Bader, Tamara G Kolda, and Others. MATLAB Tensor Toolbox Version 3.0-dev. Available online, 2017.

[BMNP04] Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, and Anthony T. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 2004.

[BS06] Charles-henri Bruneau and Mazen Saad. The 2D lid-driven cavity problem revisited. *Computers & Fluids*, 35:326–348, 2006.

[BvH98] M Beckers and G J F van Heijst. The observation of a triangular vortex in a rotating fluid. *Fluid Dynamics Research*, 22(5):265, 1998.

[CALK11] 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*, 166(11):578–592, 2011.

[CB03a] Laurent Cordier and Michel Bergmann. Post-processing of experimental and numerical data: POD an overview. *von Karman Institute for Fluid Dynamics*, pages 1–46, 2003.

[CB03b] Laurent Cordier and Michel Bergmann. Two typical applications of POD: coherent structures eduction and reduced order modelling. *Post-Processing of Experimental and Numerical Data*, 2003.

[CC70] J. Douglas Carroll and Jih Jie Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[Cha00] Anindya Chatterjee. An introduction to the proper orthogonal decomposition. *Current Science*, 78(7):808–817, 2000.

[Cha08] Saifon Chanturantabut. *Dimension Reduction for Unsteady Nonlinear Partial Differential Equations via Empirical Interpolation methods*. PhD thesis, Rice University, 2008.

[Cic14] Andrzej Cichocki. Tensor Networks for Big Data Analytics and Large-Scale Optimization Problems. *arXiv preprint arXiv:1407.3124*, pages 1–36, 2014.

[CK94] G F Carnevale and R C Kloosterziel. Emergence and evolution of triangular vortices. *Journal of Fluid Mechanics*, 259:305–331, jan 1994.

[CKL13] Francisco Chinesta, Roland Keunings, and Adrien Leygue. *The Proper Generalized Decomposition for Advanced Numerical Simulations*. Springer, ©2013, 2013.

[CL93] J.-Y. Cognard and P Ladevèze. A large time increment approach for cyclic viscoplasticity. *International Journal of Plasticity*, 9(2):141–157, 1993.

[CL14] Francisco Chinesta and Pierre Ladavèze. *Separated Representations and PGD-Based Model Reduction*, volume 554. 2014.

[CLA+09] Francisco Chinesta, Pierre Ladeveze, Amine Ammar, ELias Cueto, and Anthony Nouy. Proper Generalized Decomposition in Extreme Simulations: Towards a Change of Paradigm in Computational Mechanics? *IACM Expressions*, (26/09):2–7, 2009.

[CLB+17] Francisco Chinesta, Adrien Leygue, Felipe Bordeu, Elias Cueto, David Gonzalez, Amine Ammar, and Antonio Huerta. PGD-Based Computational Vademecum for Efficient Design , Optimization and Control To cite this version : HAL Id : hal-01515083 PGD-based Computational Vademecum for efficient design , optimization and control. *Archives of Computa- tional Methods in Engineering*, 20(1):31–59, 2017.

[ÇMi09] Ali Çivril and Malik Magdon-ismail. On selecting a maximum volume submatrix of a matrix and related problems. *Theoretical Computer Science*, 410(47-49):4801–4811, 2009.

[CS10] Saifon Chaturantabut and Danny C Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM J. SCI. COMPUT.*, 32(5):2737–2764, 2010.

[CVVI98] W Cazemier, R W C P Verstappen, a E P Veldman, and I Introduction. Proper orthogonal decomposition and low-dimensional models for driven cavity flows. *Physics of Fluids*, 10(7):1685–1699, 1998.

[DDGS15] Wolfgang Dahmen, Ronald DeVore, Lars Grasedyck, and Endre Süli. Tensor-Sparsity of Solutions to High-Dimensional Elliptic Partial Differential Equations. *Foundations of Computational Mathematics*, 2015.

[DDV00] Lieven De Lathauwer, Bart De Moor, and Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[DI09] Alireza Doostan and Gianluca Iaccarino. A least-squares approximation of partial differential equations with high-dimensional random inputs. *Journal of Computational Physics*, 2009.

[DKKO91] a. E. Deane, I. G. Kevrekidis, G. E. Karniadakis, and S. a. Orszag. Low-dimensional models for complex geometry flows: Application to grooved channels and circular cylinders. *Physics of Fluids A: Fluid Dynamics*, 3(10):2337, 1991.

[dLdMV00] Lieven de Lathauwer, Bart de Moor, and Joos Vandewalle. On the best rank-1 and rank-(R1,R2,...,RN) approximation of higher order tensors. 21(4):1324–1342, 2000.

[DM13] Xiaoying Dai and Yvon Maday. Stable Parareal in Time Method for First- and Second-Order Hyperbolic Systems. *SIAM Journal on Scientific Computing*, 2013.

[DNS+12] J Du, I M Navon, J L Steward, A K Alekseev, and Z Luo. Reduced-order modeling based on POD of a parabolized Navier–Stokes equation model I: forward model. *International Journal for Numerical Methods in Fluids*, 69(3):710–730, 2012.

[dSL08] Vin de Silva and Lek-Heng Lim. Tensor Rank and the Ill-Posedness of the Best Low-Rank Approximation Problem. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1084–1127, 2008.

[dV92] H.A. der Vorst. Bi-{CGSTAB}: A fast and smoothly converging variant of {B}i-{CG} for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 13(2):631–644, 1992.

[ECG05] E. Erturk, T. C. Corke, and C. Gökçöl. Numerical solutions of 2-D steady incompressible driven cavity flow at high Reynolds numbers. *International Journal for Numerical Methods in Fluids*, 2005.

[Ett15] Simon Etter. Parallel ALS Algorithm for the Hierarchical Tucker Representation. 2015.

[EY36] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[Fah01] Marco Fahl. *Trust-region Methods for Flow Control based on Reduced Order Modelling*. PhD thesis, 2001.

[FHMM13] A. Falcó, L. Hilario, N. Montés, and M. C. Mora. Numerical strategies for the Galerkin-proper generalized decomposition method. *Mathematical and Computer Modelling*, 57(7-8):1694–1702, 2013.

[FHN15] Antonio Falco, Wolfgang Hackbusch, and Anthony Nouy. Geometric Structures in Tensor Representations (Final Release). pages 1–50, 2015.

[FKE98] Uwe Fey, Michael Konig, and Helmut Eckelmann. A new Strouhal–Reynolds-number relationship for the circular cylinder in the range 47<Re<20000. *Physics of Fluids*, 10(7):1547, 1998.

[FN11] A. Falco and A. Nouy. A Proper Generalized Decomposition for the solution of elliptic problems in abstract form by using a functional Eckart-Young approach. *Journal of Mathematical Analysis and Applications*, 376(2):469–480, 2011.

[FN12] Antonio Falco and Anthony Nouy. Proper generalized decomposition for nonlinear convex problems in tensor Banach spaces. *Numerische Mathematik*, 121(3):503–530, 2012.

[GGC12] G. Girault, Y. Guével, and J.M Cadou. An algorithm for the computation of multiple Hopf bifurcation points based on Pade approximants. *IJNMF*, (September 2017), 2012.

[GGH90] John W. Goodrich, Karl Gustafson, and Kadosa Halasi. Hopf bifurcation in the driven cavity. *Journal of Computational Physics*, 1990.

[GGS82] U. Ghia, K. N. Ghia, and C. T. Shin. High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method. *Journal of Computational Physics*, 1982.

[GHN11] Lars Grasedyck, Wolfgang Hackbusch, and Bericht Nr. An Introduction to Hierachical (H −) Rank and TT – Rank of Tensors with Examples. 11(329):291–304, 2011.

[GKM16] Alex Gorodetsky, Sertac Karaman, and Youssef Marzouk. Function-train: a continuous analogue of the tensor-train decomposition. 2016.

[GKT13] Lars Grasedyck, Daniel Kressner, and Christine Tobler. A literature survey of low-rank tensor approximation techniques. *GAMM Mitteilungen*, 36(1):53–78, 2013.

[GL96] Gene H Golub and Charles F Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 3rd edition, 1996.

[Gor16] Alex Gorodetsky. *Continuous low-rank tensor decompositions, with applications to stochastic optimal control and data assimilation*. PhD thesis, MIT, 2016.

[Gra10] Lars Grasedyck. Hierarchical Singular Value Decomposition of Tensors. *SIAM Journal on Matrix Analysis and Applications*, 31(4):2029–2054, 2010.

[Hac14] Wolfgang (Max-Planck-Institute for Mathematics in the Sciences) Hackbush. *Tensor spaces and numerical Tensor calculus*. Number 1. Springer Heidelberg Dordrecht London New York, Leipzig, Germany, 2014.

[Har70] Richard a Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. *UCLA Working Papers in Phonetics*, 16(10):1– 84, 1970.

[Hit27] FL Hitchcock. Multiple invariants and generalized rank of a p-way matrix or tensor. *J. Math. Phys*, 7:39–79, 1927.

[HK09] W. Hackbusch and S. Kühn. A new scheme for the tensor representation. *Journal of Fourier Analysis and Applications*, 2009.

[HL96] Richard Harshman and Margaret Lundy. Uniqueness proof for a family of models sharing features of Tucker's three-mode factor analysis and PARAFAC/candecomp. *Psychometrika*, 61(1):133–154, 1996.

[HLB96] P. Holmes, J. L. Lumley, and G. Berkooz. *Coherent Structures, Dynamical System and Symmetry*. Cambridge University Press, UK, 1996.

[Hot33] Harold Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417–441, 1933.

[HRS16] Jan S Hesthaven, Gianluigi Rozza, and Benjamin Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. Springer b edition, 2016.

[ILD00] Angelo Iollo, Stéphane Lanteri, and Jean-Antoine Désidéri. Stability Properties of POD – Galerkin Approximations for the Compressible Navier – Stokes Equations. *Theoret. Comput. Fluid Dynamics*, 13:377–396, 2000.

[IR98] K. Ito and S.S. Ravindran. A Reduced-Order Method for Simulation and Control of Fluid Flows. *Journal of Computational Physics*, 1998.

[IW14] Traian Iliescu and Zhu Wang. Variational Multiscale Proper Orthogonal Decomposition: Navier-Stokes Equations. *Numer. Methods Partial Differential Eq.*, 30(2):641—-663, 2014.

[KB09] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.

[Kho11] Boris N Khoromskij. O(logN)-Quantics Approximation of N-d Tensors in High-Dimensional Numerical Modeling. *Constructive Approximation*, 34(2):257–280, oct 2011.

[KKU16] Lars Karlsson, Daniel Kressner, and André Uschmajew. Parallel algorithms for tensor completion in the CP format. *Parallel Computing*, 57:222–234, 2016.

[Kol06] Tamara G Kolda. Multilinear operators for higher-order decompositions. *SANDIA Report*, (April):1–28, 2006.

[Kos43] D D Kosambi. Statistics in function spaces. *Journal of the Indian Mathematical Society*, 1943.

[KT11] Daniel Kressner and Christine Tobler. Low-Rank Tensor Krylov Subspace Methods for Parametrized Linear Systems. *SIAM Journal on Matrix Analysis and Applications*, 32(4):1288–1316, 2011.

[KT13] Daniel Kressner and Christine Tobler. htucker – A Matlab toolbox for tensors in hierarchical Tucker format. pages 1–28, 2013.

[LBA+18] Lucas Lestandi, Swagata Bhaumik, G. R. K. C. Avatar, Mejdi Azaiez, and Tapan K. Sengupta. Multiple Hopf bifurcations and flow dynamics inside a 2D singular lid driven cavity. *Computers and Fluids*, 166:86–103, 2018.

[LBS+18] Lucas Lestandi, Swagata Bhaumik, Tapan K. Sengupta, G. R. Krishna Chand Avatar, and Mejdi Azaïez. POD Applied to Numerical Study of Unsteady Flow Inside Lid-driven Cavity. *Journal of Mathematical Study*, 51(2):150–176, 2018.

[LCLR16] Stefano Lorenzi, Antonio Cammi, Lelio Luzzi, and Gianluigi Rozza. POD-Galerkin method for finite volume approximation of Navier–Stokes and RANS equations. *Comput. Methods Appl. Mech. Engrg.*, 311:151–179, 2016.

[LN03] P Ladevèze and A Nouy. On a multiscale computational strategy with time and space homogenization for structural mechanics. *Appl. Mech. Engrg,*, 192:3061–3087, 2003.

[Loè77] Michel Loève. *Probability Theory*, volume 9. 1977.

[LPN10] P. Ladevèze, J. C. Passieux, and D. Néron. The LATIN multiscale computational method and the Proper Generalized Decomposition. *Computer Methods in Applied Mechanics and Engineering*, 2010.

[Lum67] J. L. Lumley. The Structure of Inhomogeneous Turbulence. In *Atmospheric Turbulence and Wave Propagation*, pages pp. 166–178. Nauka, Moscow, a. m. yagl edition, 1967.

[Lum81] J L Lumley. Coherent Structures in Turbulence. In RICHARD E MEYER, editor, *Transition and Turbulence*, pages 215–242. Academic Press, 1981.

[MMPR00] L. Machiels, Y. Maday, A. T. Patera, and D. V. Rovas. Blackbox reduced-basis output bound methods for shape optimization. *Proceedings 12ˆ{th} International Domain Decomposition Conference*, pages 429–436, 2000.

[MNPP09] Yvon Maday, Ngoc Cuong Nguyen, Anthony T. Patera, and George S.H. Pau. A general multipurpose interpolation procedure: The magic points. *Communications on Pure and Applied Analysis*, 2009.

[Mos18] Rolando Mosquera Meza. *Interpolation sur les variétés grassmaniennes et application à la réduction de modèles en mécanique*. Phd. thesis, Université de La Rochelle, 2018.

[NAM+03] Bernd R. Noack, Konstantin Afanasiev, Marek Morzyński, Gilead Tadmor, and Frank Thiele. A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497(February 2016):335–363, 2003.

[Nou10] Anthony Nouy. A priori tensor approximations for the numerical solution of high dimensional problems: alternative definitions. *The Seventh International Conference on Engineering Computational Technology*, paper 44, 2010.

[Nou15] Anthony Nouy. Low-rank tensor methods for model order reduction. pages 1–73, 2015.

[ODS18] I.V. Oseledets, S. Dolgov, and D. Savostyanov. ttpy, 2018.

[OI11] Takuya Osada and Reima Iwatsu. Numerical simulation of unsteady driven cavity flow. *Journal of the Physical Society of Japan*, 80(9):1–11, 2011.

[Ose11] I V Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[Ose13] I V Oseledets. Constructive Representation of Functions in Low-Rank Tensor Formats. pages 1–18, 2013.

[Ose18] I.V. Oseledets. MATLAB TT-Toolbox Version 2.2, 2018.

[OT09] Ivan Oseledets and E. E. Tyrtyshnikov. Tensor tree decomposition does not need a tree. (October 2009), 2009.

[OT10] Ivan Oseledets and Eugene Tyrtyshnikov. TT-cross approximation for multi-dimensional arrays. *Linear Algebra and Its Applications*, 432(1):70–88, 2010.

[Pea01] Karl Pearson. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[PR07] Anthony T Patera and Gianluigi Rozza. Reduced Basis Approximation and A Posteriori Error Estimation for Parametrized Partial Differential Equations. *Foundations*, (January):251, 2007.

[PRV+02] C. Prud'homme, D. V. Rovas, K. Veroy, L. Machiels, Y. Maday, A. T. Patera, and G. Turinici. Reliable Real-Time Solution of Parametrized Partial Differential Equations: Reduced-Basis Output Bound Methods. *Journal of Fluids Engineering*, 2002.

[PS14] Bernard Philippe and Yousef Saad. Calcul des valeurs propres. 33(0):1–22, 2014.

[QMNI] Alfio Quarteroni, Andrea Manzoni, Federico Negri, and An Introduction. *Reduced Basis Methods for Partial Differential Equations*.

[QR13] Alfio Quarteroni and Gianluigi Rozza. *Reduced Order Methods for Modeling and Computational Reduction*. Springer Publishing Company, Incorporated, 2013.

[RF94] D. Rempfer and H. F. Fasel. Evolution of Three-Dimensional Coherent Structures in a Flat-Plate Boundary Layer. *J. Fluid Mech.*, 260:351–375, 1994.

[SBB] T. K. Sengupta, S. Bhaumik, and Y. G. Bhumkar. Nonlinear receptivity and instability studies by {POD}. *6th AIAA Theoretical Fluid Mechanics Conf., Honululu, Hawaii,USA,*.

[Sch10] Peter J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 2010.

[Sch15] Martin Daniel Schatz. *Distributed Tensor Computations: Formalizing Distributions, Redistributions, and Algorithm Derivation*. PhD thesis, The University of Texas at Austin, 2015.

[SDS03] T. K. Sengupta, S. De, and S. Sarkar. Vortex-induced instability of an incompressible wall-bounded shear layer. *Journal of Fluid Mechanics*, (493):277–286, 2003.

[Sen12] T. K. Sengupta. *Instabilities of flows and transition to turbulence*. Taylor & Francis, 2012.

[Sen13] T. K. Sengupta. *High Accuracy Computing Methods: Fluid Flows and Wave Phenomena*. Cambridge Univ. Press, USA, 2013.

[Sey94] R. Seydel. *Practical Bifurcation and Stability Analysis from Equilibrium to Chaos*. Springer : Berlin, 1994.

[SG16] T. K. Sengupta and A. Gullapalli. Enstrophy-based proper orthogonal decomposition of flow past rotating cylinder at super-critical rotating rate. *Physics of Fluids*, 2016.

[SHPP15] T. K. Sengupta, S. I. Haider, M. K. Parvathi, and G. Pallavi. Enstrophy-based proper orthogonal decomposition for reduced-order modeling of flow past a cylinder. *Phys. Rev. E*, 91(4):1–23, 2015.

[Sir87] L Sirovich. Turbulence and the dynamics of coherent structures. I - Coherent structures. II - Symmetries and transformations. III - Dynamics and scaling. *Quarterly of Applied Mathematics (ISSN 0033-569X)*, 45(July):561, 1987.

[SLV09] T. K. Sengupta, V. Lakshmanan, and V.V.S.N. Vijay. A new combined stable and dispersion relation preserving compact scheme for non-periodic problems. *Journal of Computational Physics*, 228(8):3048–3071, 2009.

[SO03] Mehmet Sahin and Robert G. Owens. A novel fully-implicit finite volume method applied to the lid-driven cavity problem?Part II: Linear stability analysis. *International Journal for Numerical Methods in Fluids*, 2003.

[SO11] Dmitry Savostyanov and Ivan Oseledets. Fast adaptive interpolation of multi-dimensional arrays in tensor train format. 2011.

[SR18] Giovanni Stabile and Gianluigi Rozza. Finite volume POD-Galerkin stabilised reduced order methods for the parametrised incompressible Navier-Stokes equations. *Computers and Fluids*, 0:1–12, 2018.

[SRB11] T. K. Sengupta, M. K. Rajpoot, and Y. G. Bhumkar. Space-time discretizing optimal {DRP} schemes for flow and wave propagation problems. *Comput. Fluids.*, 47(1):144–154, 2011.

[SSS10] T. K. Sengupta, N. Singh, and V. K. Suman. Dynamical system approach to instability of flow past a circular cylinder. *Journal of Fluid Mechanics*, 656:82–115, 2010.

[Str86] P. J. Strykowski. *The control of absolutely and convectively unstable shear flows*. PHD thesis, Yale University, 1986.

[SVB09] T. K. Sengupta, V.V.S.N. Vijay, and S. Bhaumik. Further improvement and analysis of CCD scheme: Dissipation discretization and de-aliasing properties. *Journal of Computational Physics journal*, (228):6150—-6168 Contents, 2009.

[SVS11] Tapan K. Sengupta, V.V.S.N. Vijay, and N. Singh. Universal instability modes in internal and external flows. *Computers and Fluids*, 40(1):221–235, 2011.

[TLN14] L Tamellini, O Le Maître, and A Nouy. Model Reduction Based on Proper Generalized Decomposition for the. *SIAM J. SCI. COMPUT.*, 36(3):1–23, 2014.

[Tuc66] Ledyard R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[Tyr00] Eugene E Tyrtyshnikov. Incomplete Cross Approximation in the Mosaic-Skeleton Method. *Computing*, 64(4):367–380, 2000.

[VC06] F. Verstraete and J. I. Cirac. Matrix product states represent ground states faithfully. *Physical Review B - Condensed Matter and Materials Physics*, 2006.

[VVM11] Nick Vannieuwenhoven, Raf Vandebril, and Karl Meerbergen. On the truncated multilinear singular value decomposition. *Department of Computer Science, K.U.Leuven*, (March), 2011.

[VVM12] Nick Vannieuwenhoven, Raf Vandebril, and Karl Meerbergen. A New Truncation Strategy for the Higher-Order Singular Value Decomposition. *SIAM Journal on Scientific Computing*, 34(2):A1027—-A1052, 2012.

[Wel15] David Wells. *Stabilization of POD-ROMs*. PhD thesis, Virginia Tech, 2015.

[ZFXM] Guoxu Zhou, Andrzej Cichocki Fellow, Shengli Xie, and Senior Member. Decomposition of Big Tensors With Low Multilinear Rank. pages 1–12.

Included papers

Multiple Hopf bifurcations and flow dynamics inside a 2D singular lid driven cavity

Lucas Lestandi, Swagata Bhaumik*, G. R. K. C. Avatar*, Mejdi Azaiez, Tapan K. Sengupta*

I2M Laboratory, Univ. of Bordeaux, France
**HPCL, IIT Kanpur, Kanpur, India*

Abstract

Two-dimensional (2D) flow inside a lid driven cavity (LDC) is shown to display multi-modal behavior in a consistent manner following the first Hopf bifurcation with varying Reynolds numbers (Re), depending upon the chosen spatial and temporal discretization scheme. Direct numerical simulation (DNS) following impulsive start, is used to show spatio-temporal growth and its nonlinear saturation of disturbance growth. Despite the fact that researchers have produced different value of Reynolds number when first Hopf bifurcation occurs (Re_{cr1}), DNS fundamentally differs from classical bifurcation studies involving global instability study of an equilibrium flow due to adopted nonlinear approach and not restricting the analysis to temporal instability only. The accuracy attribute of the DNS adopted here has been shown conclusively earlier via demonstration of a weak transient polygonal core vortex surrounded by relatively stronger gyrating vortices, which appear as a constellation after the disappearance of the transient, in Sengupta *et al.* (J. Comput. Phys., **228**, 3048- 3071 and 6150-6168 (2009)). Investigated LDC flow is characterized by multiple time scales at any Re, which are weak function of Re in selective intervals, punctuated by multiple bifurcations. The present investigation achieves two primary goals. First, it proposes to reconcile that Re_{cr1} obtained by different numerical approaches can be shown to be in same range, provided the equilibrium flow obtained is of good quality, untainted by excessive diffusion. Secondly, we also show that for increasing Re following the first Hopf bifurcation, the flow during the limit cycle suffers a secondary instability, thus, requiring computation of the flow field over a longer time period. The first goal is met by exciting the flow field with a pulsating vortex inside the LDC for a very high accuracy scheme, we are able to show the universal nature of the primary bifurcation for Re in the range between 8020 and 8025. The flow at higher Re displays significantly increased spectral peaks, including broad-band spectrum and the understanding of all these have been aided by phase space portraits.

Keywords: Lid driven cavity, DNS, Multiple Hopf bifurcation, polygonal core vortex, phase space portrait

1. Introduction

The 2D flow in a square LDC (of side L) is a popular problem to test new algorithms for incompressible Navier-Stokes equation (NSE) due to its unambiguous boundary conditions, coupled with its very simple geometry. As the lid is given a constant-speed translation (U), this gives rise to corner singularities on the top wall, as depicted in the top frame of Fig. 1. The role of such singularities is to give rise to Gibbs' phenomenon, as reported by pseudo-spectral computation of NSE [2, 7]. Computing flow in LDC by other discrete computing methods [9, 17, 33], corner singularities do not cause any problem due to smoothly decaying spectrum created by spatial discretization [37] near the cutoff wavenumber. While it is possible to compute steady flow at low Re by various methods including lowest order spatial discretization, it is not so at higher Re, where the flow displays inherent tendencies of unsteadiness. One of the central activities in studying the problem of LDC is to show that the onset of unsteadiness is related to flow instability. Viewed in this perspective, the primary goal is then predict the correct equilibrium flow for global instability studies. However, in DNS one directly proceeds to obtain the unsteady flow. This latter approaches can thus cause confusion, as is noted in the published literature. Many low order methods are incapable of computing unsteady flows at high Re ($= UL/\nu$), where ν is the kinematic viscosity. In Ghia *et al.* [17], results for a wide range of Re up to 10000 are presented. The flow is steady for $Re = 10000$ in [17], while numerical results obtained by high accuracy combined compact difference (CCD) scheme presented in [26, 27] indicate creation of a transient polygonal vortex at the core, with permanent gyrating satellite vortices around it. In these references, sixth order CCD scheme has been used to discretize both the convection and diffusion terms of the vorticity transport equation. It is well known [1, 36] that compact schemes for spatial discretization filters minimally, as compared to other methods.

For the LDC problem at $Re \leq 1000$, researchers [2, 3, 7] have tried to circumvent the singularity by subtracting the contribution due to singularity (divergence of pressure and vorticity at the top corners) to obtain a steady flow solution

Email addresses: llestandi@u-bordeaux.fr (Lucas Lestandi), swagata@iitk.ac.in (Swagata Bhaumik*), krishnachand.beaero14@pec.edu.in (G. R. K. C. Avatar*), azaiez@enscbp.fr (Mejdi Azaiez), tksen@iitk.ac.in (Tapan K. Sengupta*)

by pseudo-spectral methods. The singularity diverges as $1/r$, with r as the radial distance from the corner for the flow, by including inertial effects [20]. This method has not been used for Re exceeding 1000 and instead for higher Re, singularity is removed by altering the velocity boundary condition on the lid - a process known as the *regularization* [25].

A steady solution has been reported by many [6, 14, 17] for Re far exceeding the values reported in the literature for the first Hopf Bifurcation (Re_{cr1}), due to the excessive diffusion of the discretization. If this steady solution is treated as equilibrium solution, then its global instability will not be predicted in a unique manner, as have been attempted by solving numerically the bifurcation problem [6, 16, 30]. Use of lower order methods in obtaining equilibrium flow results in contaminated eigenvalues. On the other hand, simulations of full time-dependent NSE [18, 19, 28] reveal that the flow loses stability via a Hopf bifurcation with respect to increasing Re. Critical Re_{cr1} and frequencies obtained from DNS and eigenvalue analysis do not match and such differences are noted for different DNS results too for various reasons, some of these will be explained here.

It is shown in [27, 28, 38] that Re_{cr1} depends upon the accuracy of the method and how the flow is established in DNS. Physically, impulsively started flow is ideal to study the dynamics, as it triggers all frequencies at $t = 0$ [26, 27]. Such an analysis is preferred and is superior to normal mode analysis of eigenvalue approach. We note that obtaining a limit cycle at a different Re from the limit cycle solution obtained at another Re is not appropriate. While this may result in faster computations [28], this also produces different Re_{cr1}, as compared to the results obtained by impulsive start [26, 27, 38]. This is highlighted here by the high accuracy study of Hopf bifurcations by DNS performed by reducing sources of error.

Multiple Hopf bifurcations have been reported earlier by researchers for LDC flow. Authors in [3] have talked about a second bifurcation, while Sengupta et al. [38] have described multiple Hopf bifurcations for flow in LDC by plotting the bifurcation diagram using FFT data of vorticity time series. In recent times, Girault et al. [46] have talked about multiple Hopf bifurcations for LDC flow using compact scheme. Thus, the present effort in reporting multiple Hopf bifurcation is based on relating it with overall dynamics of the flow field, as computed by the high accuracy method, which is being used for LDC flow in [26, 27, 38] and in the present study.

The role of various sources of errors, including aliasing error for flow inside LDC has been described in [27]. Here we will discuss the roles of other sources of errors, based on the model convection-diffusion error [41]. As this simulations are extremely sensitive to operating conditions, the present work relies only on sequential computing in order to capture the weak transient core vortex when the major sources of errors are removed. This aspect of hyper-sensitivity of computed solution on background disturbance is further exploited here to explain why Re_{cr1} are different for different numerical methods.

Appearance of unsteadiness with variation in parameter value(s) studied by bifurcation theory [39] is due to flow instabilities [35]. Linear instability of equilibrium flow and DNS have been used in the literature to evaluate the onset of unsteadiness, providing scattered values of Re_{cr1} for flow in LDC. The authors in [3], using a second order projection method along with second order backward difference for time integration, obtained this value to be bracketed between 8017.6 to 8018.8. The authors furthermore added that their preliminary analysis beyond the first bifurcation led them *to suppose that the system passes through a second Hopf bifurcation for a second critical Reynolds number located in the interval [9687, 9765).* Various researchers noted different value of Re_{cr1}: As 8031.93 in [32], 7972 in Cazemier *et al.* [11] using a finite volume method. Bruneau and Saad [8] noted this to be in the range of 8000 to 8050 without showing the relevant bifurcation diagram, using a third order upwind scheme, using (1024×1024) grid. However, the use of three time-level Gear method, produces a spurious mode to affect results. We highlight the roles played by different numerical sources in triggering flow unsteadiness by Re_{cr1}, that explains the scatter of reported Re_{cr1}. Of specific interest are for methods using very high accuracy methods which report relatively high values of Re_{cr1}. It is somewhat paradoxical that very diffusive upwind methods produce very high Re_{cr1} also, by attenuating disturbances to delay unsteadiness, as reported in [6, 14, 17].

High accuracy compact schemes have been used in Sengupta *et al.* [38] and described multiple Hopf bifurcations, showing $Re_{cr1} = 7933$ and the second at 8187, using the FFT amplitude of the vorticity time series obtained using sixth order accurate combined compact difference (NCCD) scheme on a uniform (257×257) grid. Osada and Iwatsu [28] have identified this value at $7987 \pm 2\%$ - in similar range using compact scheme on non-uniform (128×128) and (257×257)-grids. This limit is obtained based on linear interpolation of data from these two grids, in Fig. 3 of [28] for u-component of velocity at the core of LDC. The figure clearly shows grid dependence of the computed results as the A_e^2 (equilibrium amplitude) versus Re curves have diverging slopes. Thus, the authors obtained grid-dependent results, as seen in the present computations. This is due to the fact that the wall vorticity is different for different wall resolution, and here we explain the reason for this. Here all computations are obtained by starting from quiescent condition at $t = 0$, following an impulsive start. Several results in [28] are obtained by projecting results from one Re to another. Here, we will demonstrate that projecting a solution from one Re to another is essentially flawed to obtain Re_{cr1}. In [28], the authors also provide the second critical Reynolds number as $Re_{cr2} = 9575 \pm 3\%$, which is quite different from that is given in [3].

There are also studies which report widely different values of Re_{cr1}. For example, Shen [34] reported Re_{cr1} in the range of 10000 to 10500. Poliashenko and Aidun [30] on the other hand reported a value of $Re_{cr1} = 7763 \pm 2\%$ using a commercial FEM package. Peng *et al.* [29] reported a value of $Re_{cr1} = 7402 \pm 4\%$ using FDM by Marker and Cell (MAC)

method.

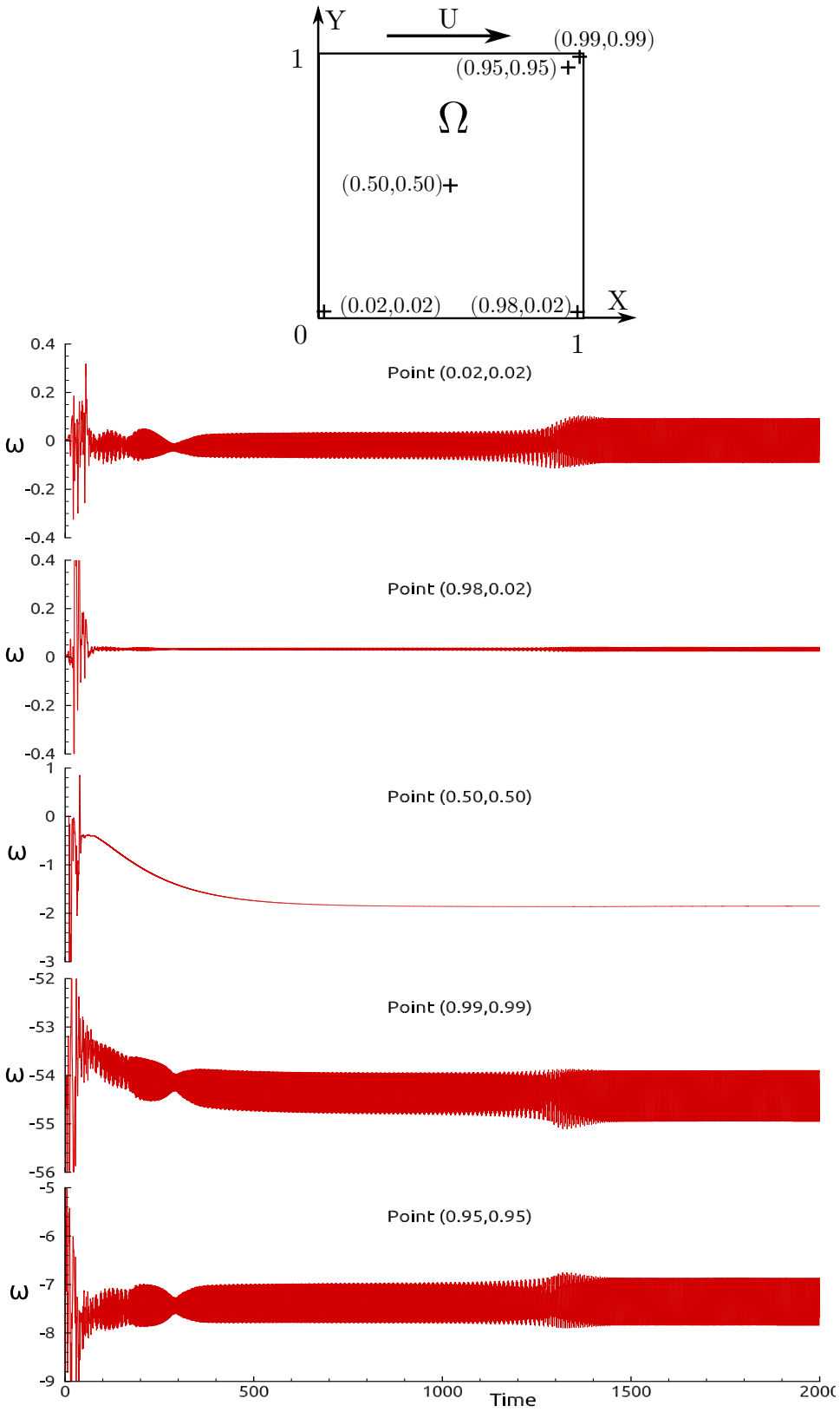


Figure 1: The schematic of the computational domain and location of sampling points (top) and vorticity time series at the sampling points obtained for $Re = 9500$ obtained from solution of Navier-Stokes equation using a (257×257) grid.

To obtain Re_{cr1}, DNS is preferred over eigenvalue analysis, as in the latter a temporal analysis assumes all points to

have identical time variation. In DNS, the true spatio-temporal dynamics is traced. In the schematic shown on top of Fig. 1, five points are identified where time variation of vorticity is stored, as shown in the other frames of the figure. Apart from the center of the LDC, there are two points at the bottom embedded inside corner vortices at $(x = 0.02, y = 0.02)$ and $(x = 0.98, y = 0.02)$. Other two points located near the top right corner of the cavity at $(x = 0.95, y = 0.95)$ and $(x = 0.99, y = 0.99)$, display higher unsteadiness, strongly affected by aliasing error [27]. This is noted in the vorticity time series shown in Fig. 1. The core suffers least perturbation and the next higher disturbance amplitude is noted at the bottom right corner. While bottom left corner point registers significant vorticity disturbance, the top right corner points log higher disturbance vorticity. The point at $(x = 0.99, y = 0.99)$ being closest to the corner singularity, displays highest variations and takes longer to attain the limit cycle. Hence the point $(x = 0.95, y = 0.95)$ is preferred for analysis purpose [26, 27, 38]. In [28], the time series has been sampled at a point near the bottom right corner at $(x = 13/16, y = 1/16)$.

The paper is formatted in the following manner. In the next section, a very brief recap of the governing equation and the numerical methods used are provided. In section 3, the flow field is characterized by vorticity field obtained by DNS with Re. Vorticity dynamics and polygonal core vortex are described in section 4. This is followed by description of multiple Hopf bifurcations in section 5. Apart from describing new equilibrium states, we also show the frequency spectrum of the flow field as a function of Re and provide the phase space trajectory to describe the flow dynamics. In the following section 6, we discuss about extreme sensitivity to the grid resolution and projection of solution for one parameter to another. Most importantly we explain receptivity of the flow to imposed excitation and trace back the universal Re_{cr1}, which we consider as the main results in the present work. In the end, summary and conclusions are provided.

2. Governing Equations and Numerical Methods

DNS of the 2D flow is carried out by solving NSE in stream function-vorticity formulation given by,

$$\nabla^2 \psi = -\omega \tag{1}$$

$$\frac{\partial \omega}{\partial t} + (\vec{V} \cdot \nabla)\omega = \frac{1}{Re}\nabla^2 \omega \tag{2}$$

where ω is the non-zero out-of-plane component of vorticity for the 2D problem. The velocity is related to the stream function by $\vec{V} = \nabla \times \vec{\Psi}$, where $\vec{\Psi} = [0\ 0\ \psi]$. Reynolds number is defined by L and the constant lid velocity, (U), which are also used as length and velocity scales for nondimensionalization. This formulation is preferred due to inherent solenoidality of the velocity and vorticity for 2D flows. It also allows one to circumvent the pressure-velocity coupling problem. The numerical methods and the dynamics of the flow for $Re = 10000$ are given in greater details elsewhere [26, 27] and is not repeated here.

Equations (1) and (2) are solved subject to the following boundary conditions. On all the four walls of LDC, $\psi = constant$ is prescribed which helps in satisfying no-slip condition; the wall vorticity is $\omega_b = -\frac{\partial^2 \psi}{\partial n^2}$, with n as the wall-normal co-ordinate chosen for the four segments of the cavity to obtain the boundary vorticity. This is calculated using Taylors series expansion at all the walls with appropriate velocity conditions at the boundary segments. The top lid moves horizontally with a unit nondimensional velocity, with all other walls as stationary. To solve the discretized form of Eq. (1), Bi-CGSTAB method has been used here, which is a fast and convergent elliptic PDE solver [42]. The convection and diffusion terms are discretized using the sixth order accurate NCCD method [26, 27], which obtains both first and second derivatives simultaneously. All other details about NCCD and other compact schemes can be also found in Sengupta [36] and hence are not reproduced here. For time advancing Eq. (2), four-stage, fourth-order Runge-Kutta (RK4) method is used that is tuned to preserve dispersion relation. The NCCD scheme has been analyzed for resolution and effectiveness in discretizing the diffusion terms along with the dispersion relation preservation properties for 1D convection equation [26, 27]. It is noted that the NCCD method is efficient, providing high resolution and effective diffusion discretization. Additionally, the method has built-in ability to control aliasing error. The only drawback of NCCD scheme is that it can be used only with uniform structured grids. All computations are performed with nondimensional time-step of $dt = 0.001$. The final limit cycle behaves in a similar fashion, when time step is changed. Only the instability of the limit cycle appears at different time range with change in dt. We also report an additional set of computations using a finer grid with (513×513) points. Vorticity time series at a sampling point, qualitatively remains the same, with only the mean value shifted by a small fraction. The sampling point location being at (0.95,0.95) has been explained in the previous section with respect to sampling point taken in [28].

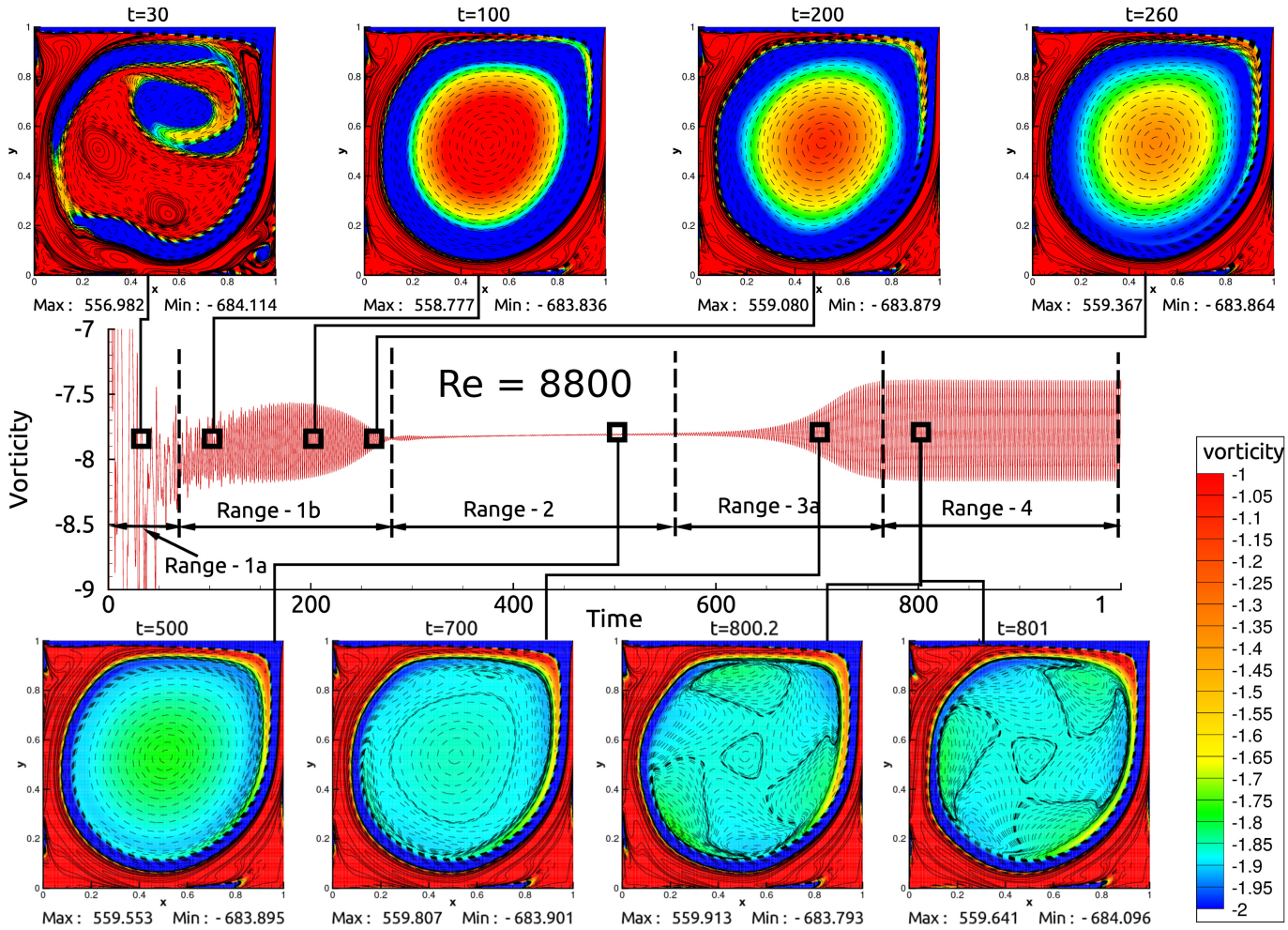


Figure 2: The vorticity time series at the sampling point ($x = 0.95, y = 0.95$) obtained for $Re = 8800$ with vorticity contour plots shown at the indicated time instants. Solution of Navier-Stokes equation is obtained using (257×257) grid.

3. Dynamics of singular LDC Flow

To understand why the eigenvalue analysis and direct solution of NSE do not match, we explain this further with the help of Fig. 2 for $Re = 8800$, which shows the vorticity time series at ($x = 0.95$, $y = 0.95$) in the central frame. The results are obtained by solving unsteady NSE using NCCD scheme [26, 27, 38] for spatial discretizations of first and second derivatives. In the time series, we have identified various regimes of time-variation. For example, in Range-1a of Fig. 2, plotted vorticity displays high frequency transient variations, followed by banded relatively lower frequency variations of the vorticity in Range-1b. In Range-1b, it is possible to see coherent vortices inside the cavity. However, such structures at the core are highly transitory and the time series shows the decay of the signal near the terminal time of Range-1b, the vorticity fluctuation reduces and settles down to a steady value and which is maintained throughout in Range-2. This period is followed by Range-3a, where the vorticity variation displays growth and which is presumably due to linear temporal instability. Finally, in Range-4 one notices nonlinear saturation of the growth noted in Range-3a. This is the typical variation of vorticity with time for lower Re cases, which are above Re_{cr1}.

For such a time series shown in Fig. 2, the linear growth in Range-3a is followed by nonlinear saturation in Range-4. Results obtained by high accuracy solution of time-dependent NSE in Range-2 and that is strictly obtained as solution of steady NSE may not match. Due to this, in the following linearly unstable range, solutions obtained by time-dependent NSE in Range-3a would also not necessarily be the same, which is obtained from the eigenvalue analysis of steady NSE solution. Also, the mismatches in Ranges-2 and -3a, can be due to differences in accuracy of numerical methods employed. The steady state solution obtained in the unstable range is essentially due to the diffusive nature of numerical methods. Such steady solutions have been reported for a high Reynolds number of 20000 [6, 14]. The sensitive dependence of solution of a nonlinear dynamical system to initial condition (here the equilibrium state obtained by two ways) is well known and for fluid dynamical system governed by NSE is recorded in the literature [36]. The Range-4 is where the dynamical system settles down to its limit cycle. We shall note later that the transition from Ranges-2 and -3a to Range-4 can be quite

complicated, punctuated by intermediate quasi-equilibrium states which suffer instabilities to take the system to newer equilibrium state. This will be discussed again with respect to higher Re cases.

One of the major aspects of the work reported by Sengupta *et al.* [26, 27] is the multi-modal frequency spectrum of the flow inside LDC for $Re = 10000$. It has been noted that with the use of NCCD scheme, the aliasing error near the top right corner is held in control and only five distinct frequencies are noted in the spectrum for $Re = 10000$. In contrast, when another combination of compact schemes were used for the same problem, the spectrum was seen to be broad-band with presence of fluctuations at multiple scales. This difference between the two methods is due to better diffusion discretization by NCCD scheme [27, 36]. In the present exercise, the same method has been used to track accurately the phenomenon of multiple Hopf bifurcations in the range of Reynolds number: $8000 \leq Re \leq 12000$ and the associated dynamics. For the flow in the range: $8000 \leq Re \leq 8660$, the flow remains steady and would be of lesser interest to us, obtained using (257×257) grid.

In various frames of Fig. 3, we depict vorticity time series at point ($x = 0.95$, $y = 0.95$) for different Re from 8660 onwards. The time series for $Re = 8660$ only displays Ranges-1a, -1b and -2 for the simulation performed up to $t = 2000$. However, for $Re = 8670$, one notices all the ranges shown in Fig. 2. This implies that the first critical Hopf bifurcation occurs in the range $[8660, 8670]$. As Re increases, one notices that the Range-2 shrinks, i.e., the time over which flow remains steady decreases, before being destabilized. This steady state (an equilibrium solution) is unstable, and the flow suffers a temporal instability. For $Re = 8900$, the apparent steady state actually consists of low amplitude oscillations and the flow suffers temporal instability, which can be studied by Floquet theory [5], provided this solution strictly periodic having a single frequency. For such periodic equilibrium flow the eigenvalue analysis for steady flow, as in [6, 30, 16] is not possible. However, the spectrum of periodic solutions are populated with more than one incommensurate frequencies and Floquet analysis is not an option. Thus to avoid such complexities of linear instability studies, we advocate high accuracy solutions of unsteady NSE, as has been practiced also in the literature [8, 18, 19, 26, 27, 38] and here.

With increase in Re above 8800, one notices that the onset of Range-3a also advances, which is noted by comparing the frames in Fig. 3. The Ranges-1a and -1b are seen to change at a slower rate with increase in Re. Also, in the time series for $Re = 9000$ onwards, one notices significant modulations during the growth and nonlinear saturation stages of the vorticity evolution. For $Re = 9100$ onwards, one notices that the Range-2 is completely absent, implying that the flow does not achieve steady state at all. The temporal growth starts from the transient stage (Range-1) itself, for $Re = 9100$. For such cases, eigenvalue analysis of steady state [6, 16, 30] is of no value. This situation is further compounded by multi-modal interactions, as seen for $Re = 9000$ with distinct wave-packets forming. Intensification of modulations and their extension in nonlinear saturation stage is also noted for $Re = 9300$. This is a new unreported instability for LDC flow, which starts from the nonlinear modulation stage and will be referred to as secondary instability henceforth, while the time extent is marked as Range-3b. The demarcation between early transient stage and regular temporal growth stage is visible as a neck formation in the time series near $t = 300$, for $Re = 9100$ onwards. One of the visible signature of the dynamical system having reached a stable equilibrium state is the presence of constant amplitude limit cycle beyond $t = 700$ onwards for $Re = 9300$. We have already noted that depending upon Re, one may or may not see the presence of the five ranges indicated in Fig. 2 for $Re = 9100$.

The marked secondary nonlinear instability is obtained here by high accuracy solution of NSE using NCCD scheme [26, 27]. Even if linear instability were to be a valid option in the earliest phase in Range-3a, one cannot resort to normal mode analysis [12, 21], since the visible modulation is due to multi-modal interactions. In case of single mode being present during the growth phase, one can use Stuart-Landau equation [24, 40] and explain the nonlinear saturation noted in Range-4, as due to self-interaction only. An objective discussion on applicability of this model is given in [38]. A theoretical approach to multi-modal interactions has been advanced [23, 35], where the eigenfunction expansion formalism proposed by Eckhaus [13] has been utilized to derive the more general Stuart-Landau-Eckhaus (SLE) equation [23, 35]. SLE equation is a tool to explain the limit cycle stage with multi-modal interactions. Additionally, use of proper orthogonal decomposition (POD) helps one to obtain the instability modes. The governing SLE equations have been used to explain bluff body flow instability, as well as developing reduced order model (ROM) for flow past circular cylinder for low Reynolds numbers in [23, 31].

An exceptional case is also noted for $Re = 9400$, in which the presence of multi-modal interactions is noted even when the computations are carried out till $t = 2000$, as shown in Fig. 3. Only when the computation is extended beyond $t = 3000$, one notices the slow disappearance of modulation. However, the amplitude of the final limit cycle is significantly lower, as compared to neighboring Re cases. This will be further described in presenting multiple Hopf bifurcations. The implications of sustained modulation is explained shortly with the help of spectrum of data after the dynamical system has settled down to an apparent equilibrium state.

With further increase in Re, a secondary instability occurs, as seen for $Re = 9700$, with the final limit cycle setting in before $t = 700$ in Fig. 3. In this case, after Range-1a a series of secondary instabilities are noted, culminating in the final limit cycle very early on. For $Re = 9800$ the first instability is followed by nonlinear action which leads to the amplitude continuously increasing and a single secondary instability is noted around $t = 1200$, which leads to the final limit cycle being firmly established by $t = 1500$. This is an atypical behavior, not seen for lower Re. The vorticity time series for $Re = 9900$, 10000, 10080, 10180 and 10400 (not shown here), follow similar time variation as that of $Re = 9700$.

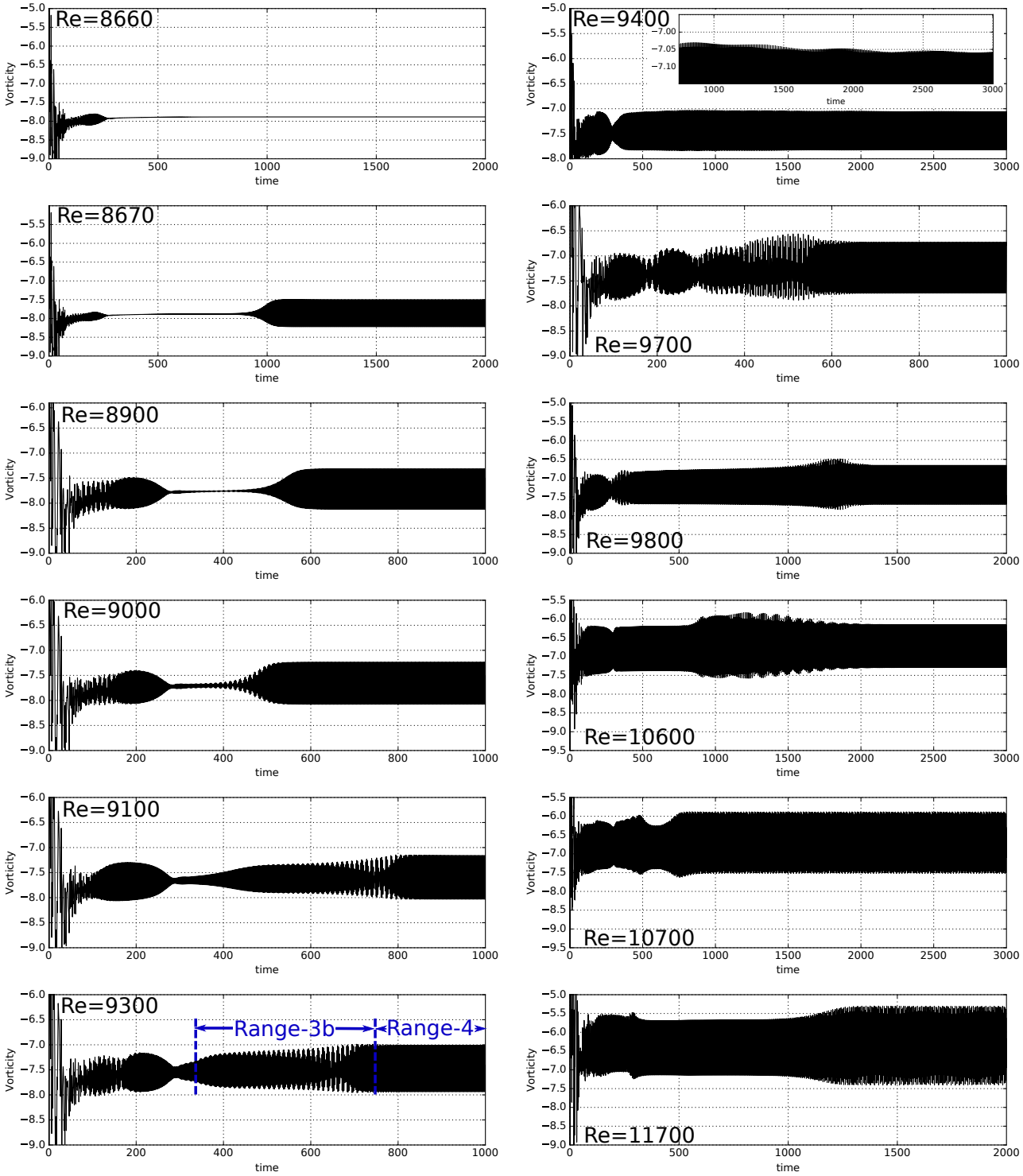


Figure 3: The vorticity time series for a point located at $x = 0.95$, $y = 0.95$, near top right corner for the displayed Reynolds numbers, obtained from solution of unsteady Navier- Stokes equation.

For $Re = 10500$, qualitatively different vorticity dynamics is noted as compared to the case for $Re = 10400$. This time series has resemblance with the case for $Re = 9400$. The vorticity time series for $Re = 10600$ shows qualitatively similar behavior as noted for $Re = 9800$. Above this Reynolds number, computed vorticity field up to $t = 2000$, show continuous modulations in the time series. This is with the exception of $Re = 11500$ and 11600 cases, for which one notices stable limit cycle after a very small time interval following the formation of neck around $t = 200$.

The described multiple mode interactions and consequent modulations can take a long time (T_{lc}) before the dynamical system settles down to a stable limit cycle. Additionally, this process is very sensitive to the Reynolds number as one can infer from Fig. 4. Indeed T_{lc} behaves irregularly during the very short range $10000 \leq Re \leq 10400$ that is displayed here.

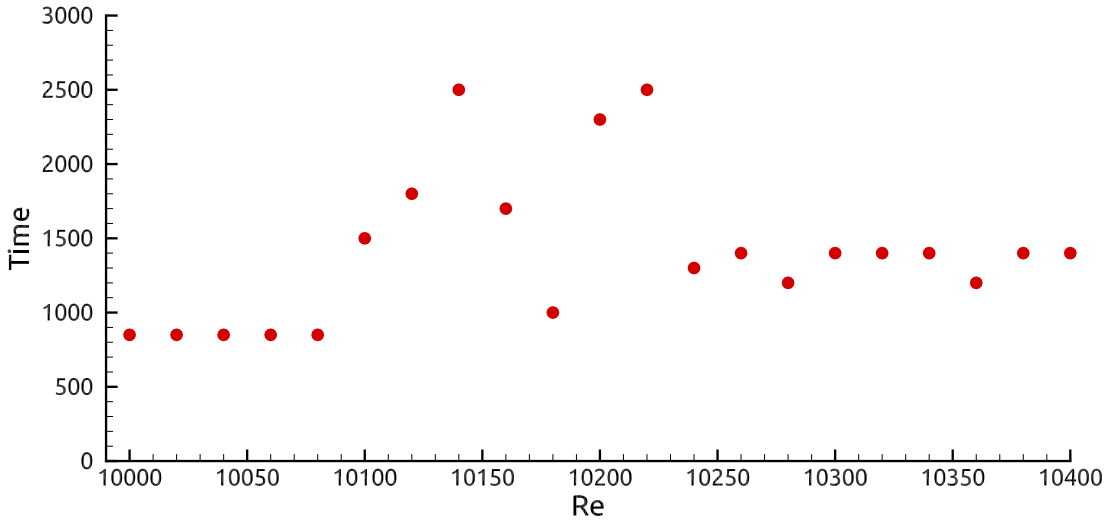


Figure 4: The onset time of final stable limit cycle of the vorticity time series for the displayed Reynolds numbers, with solution shown for the location given by, $x = 0.95$ and $y = 0.95$.

4. Vorticity dynamics and polygonal vortex in LDC

From the time series shown in Fig. 3 for different Re's at the stable limit cycle stage, we have noted the feature of periodicity of the solutions in the final limit cycle. Here, we investigate further about the flow field for $Re = 10300$ to describe the flow evolution in terms of vorticity dynamics. In Fig. 5, we show the vorticity contours inside the cavity at the indicated time instants, while the vorticity time series at $(x = 0.95, y = 0.95)$ is shown as the central panel in Fig. 5, to understand the choice of the time instants.

In the early stages of flow evolution, the inner core develops in conformity with the shape of the cavity, due to the action of the wall jet impinging near the top right corner. Thus, the lighter shaded contours shown in the form of a rounded rectangle, while the inner contour lines morph into a circular shape, as noted at $t = 200$. From the time series, one notes this stage to belong to beyond the early transient, where the coherent motion corresponds to a apparent neutral stage which is followed by decay of the disturbance. This continues up to $t = 280$, when the time series indicate the termination of decay and beyond this time, the disturbance once again grows. The vorticity contours show two distinct layers with sharp gradient and this motion continues, as shown in the frame for $t = 660$, where the gradient is really sharp. In subsequent flow evolution, the outer layer transforms into satellite vortices while the inner core shrinks to the triangular vortex, as noted in the frame for $t = 960$. Such triangular vortices have been shown earlier for $Re = 10000$ [26, 27] and it is noted here also. The time series also indicates that there is no steady state for this flow to perform linear stability analysis, as attempted by other researchers [6, 30]. The triangular core vortex forms after the linear stability phase, only once the nonlinear saturation has taken place. Hence, one can conclude that its presence is essentially due to nonlinear dynamics of the flow field guided by the presence of six gyrating satellite vortices. However, with passage of time the central core vortex looses strength and identity. Thereafter, one notices these six gyrating satellite vortices to rotate about the center of the cavity. This is the terminal state of the limit cycle. One such cycle is shown in the bottom three frames.

5. Multiple Hopf Bifurcation

The vorticity time series described in section 3 indicate different qualitative dynamics for different Re and that in turn is suggestive of multiple bifurcations in the range of computed solutions. Here, we address bifurcations for the LDC flow

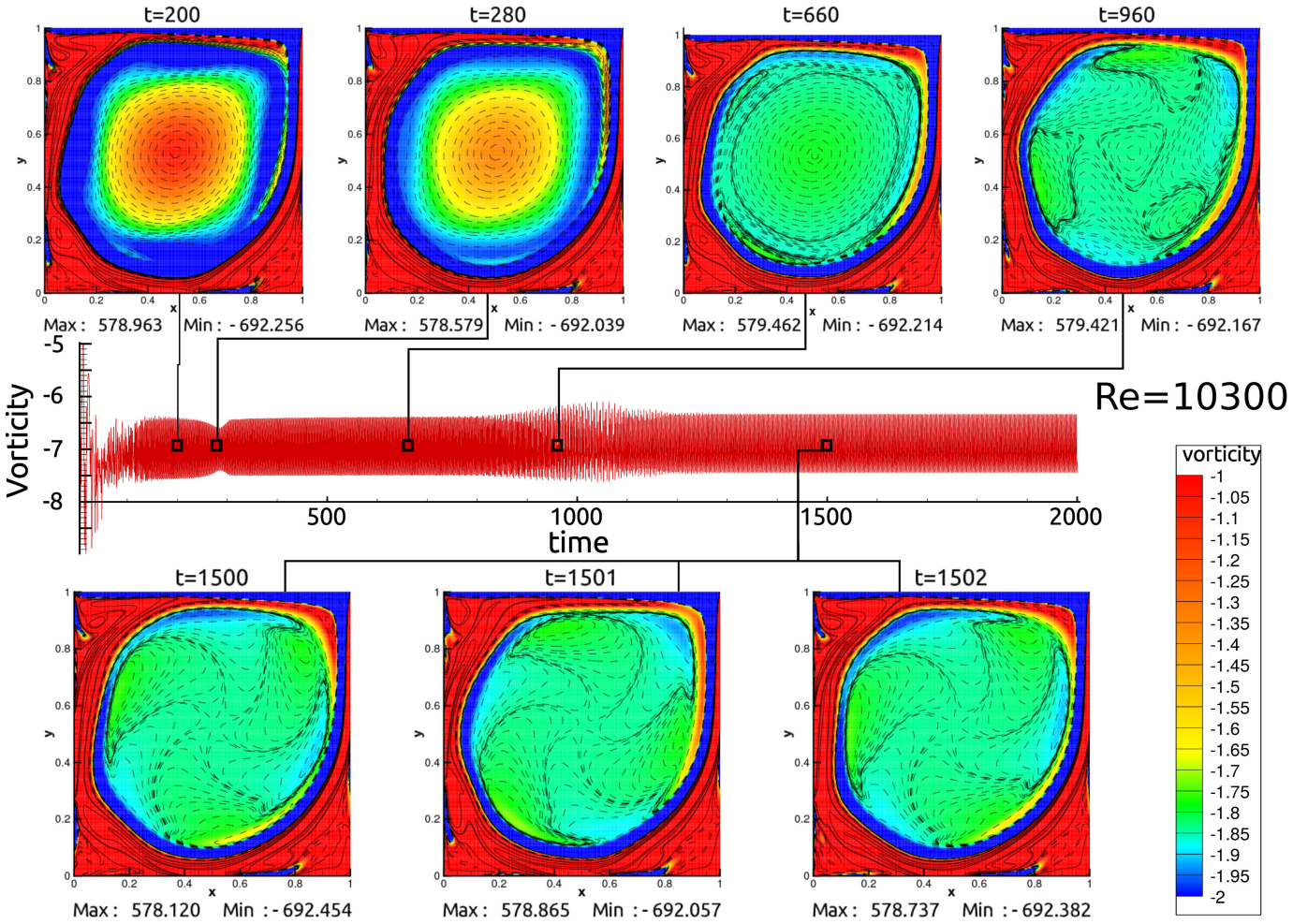


Figure 5: Vorticity contour is shown at different time instants from start till attainment of limit cycle for $Re = 10300$. Time series of the vorticity at point $(0.95, 0.95)$ is shown in the center.

based on DNS performed following an impulsive start. To do so the limit cycle amplitude is studied first, then the analysis of the frequencies property allows a better comprehension of the underlying mechanisms.

5.1. New Equilibrium State via Stable Limit Cycle

The amplitude of the limit cycle A_e is defined as half of the maximum excursion of the vorticity time-series describing a constant width envelope, by sampling the vorticity at $(0.95, 0.95)$. Different time evolution at the sampling point for different Re are presented in Fig. 3. For some higher Re cases, computed flow field display significant modulation even when the flow is computed up to $t = 2000$ and above. The Stuart-Landau model states that $A_e^2 \propto |(Re - Re_{cr1})|$ for the limit cycle cases with single dominant mode and this is useful for the flow past a circular cylinder approximately. Correspondingly, Fig. 6 displays the plot of A_e^2 as a function of Re for the range $8660 \leq Re \leq 12000$ obtained using a grid with (257×257) points. Unlike the nonlinear dynamical systems for bluff bodies, here the Hopf bifurcation [39] starts very sharply, as shown in Fig. 6, which occurs between $Re = 8660$ and 8670. For flow past a circular cylinder, DNS based Hopf bifurcation studies and corresponding results are given in [23]. Each zoomed view in Fig. 6 shows a segment in which relation between Re and A_e^2 is compared with its linear variation. The linear regression coefficients can be found in Table 1.

In the range $8670 \leq Re \leq 9350$, one can see in Fig 6 that the linear regression fits the data well. This is confirmed by the value of the regression coefficient (R^2) being really close to 1. The amplitude then suddenly drops around $Re = 9400$, as noted in Fig. 6. To ascertain the correctness of this value, additional simulations have been performed for $Re = 9350$, 9395, 9405 and 9450 and all these data are marked in the figure. It is noted that the value for $Re = 9350$ falls on the linear segment shown to the left of $Re = 9400$. For $Re = 9450$, the amplitude belongs to the next linear segment which ends at $Re = 10400$, as shown in Fig. 6 in the second box. However it should be noted that the correlation coefficient is lower on this range, mainly due to its larger extent. Another natural break in the curve is noted between $Re = 10500$ and

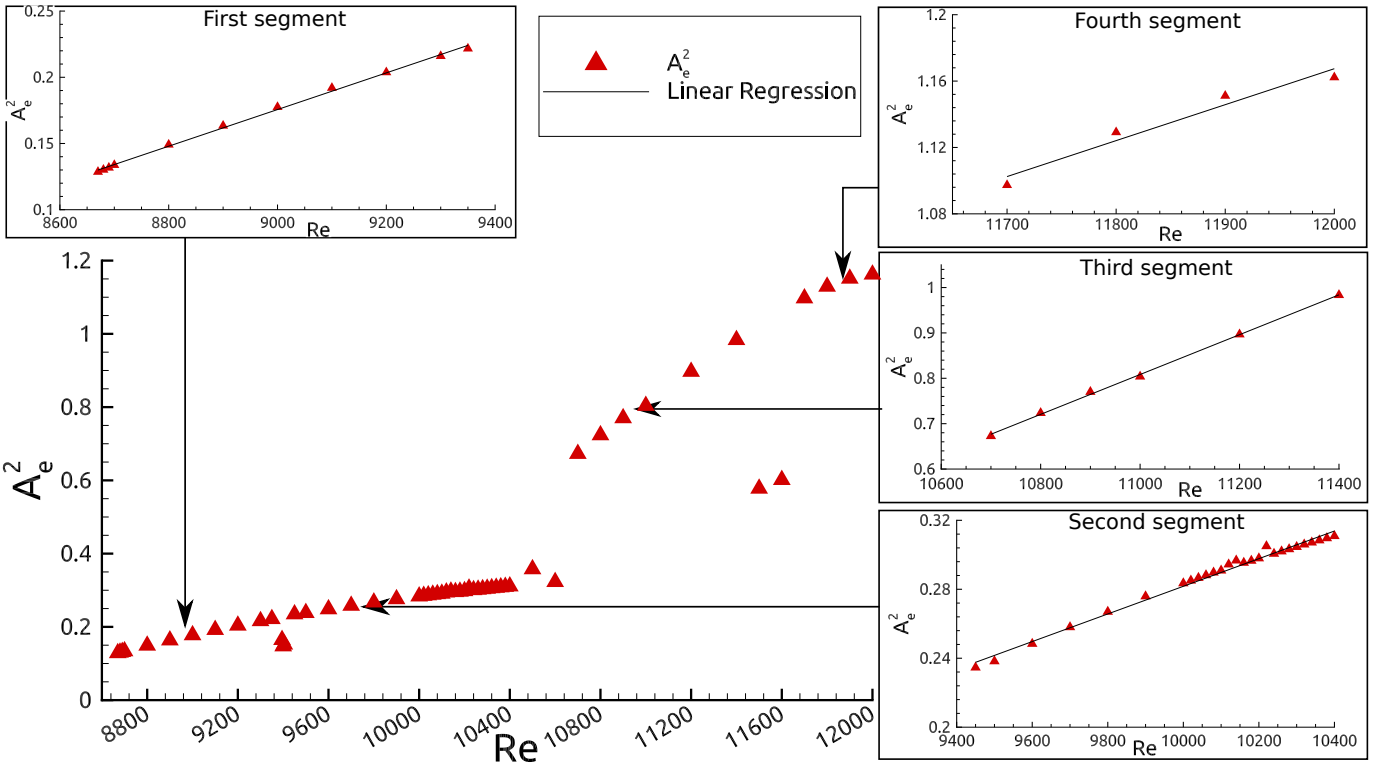


Figure 6: Multiple Hopf-bifurcation shown with respect to the vorticity time series data shown for Fig. 2. All the simulated Reynolds numbers data are used to plot the amplitude of the final stable limit cycle data against Reynolds number.

Table 1: Coefficients of linear regression equation of the form : $A_e^2 = aRe + b$ with regression correlation coefficient R.

Segment	a	b	R^2
1	1.384672e-4	-1.070575	0.998
2	8.017815e-5	-0.520059	0.992
3	4.389279e-4	-4.019874	0.999
4	2.166881e-4	-1.432799	0.956

10600. Once again a linear segment is plotted for the data points for $Re = 10700$ to 11400. $Re = 11500$ and 11600 show a particular behavior in the higher Re range since A_e^2 values fall abruptly and then the amplitude again rises sharply at $Re = 11700$ defining the fourth bifurcation. A new range up to 12000 is presented in the fourth box of Fig. 6, however the correlation coefficient is low, implying that A_e^2 does not vary linearly with Re.

A similar curve was drawn using the highest amplitude of spectrum (obtained by the FFT of the time series) for each Reynolds number [38]. Also, the simulations have been performed here over a significantly longer time interval, till we obtained the stable final limit cycle. It has been noted [23, 38] that the presence of such discontinuities is indicative of multiple Hopf bifurcations in (A_e^2, Re)-diagram, as in Fig. 6. The fact that the flow behaves qualitatively different in different range of Re is indicative of discrete change in A_e^2 with respect to Re, as indicated in Fig. 6. Along with such changes in the physical plane, one would expect to notice qualitative changes of the spectrum of the time series already shown in Fig. 3. These are now shown in Fig. 7, testifying the qualitative changes in the flow field following the appearance of a new bifurcation. From Fig. 6 and Tab. 1, one can infer the presence of four such Hopf bifurcations. In order to provide a better understanding of the phenomena at work here, the next sub-section will focus on spectral analysis of the vorticity time series at point $(0.95, 0.95)$.

5.2. Frequency Spectrum Analysis

In Fig. 7, we show few Fourier transforms of the time series shown in Fig. 3. Fourier analysis is applied over the last 100 cycles, i.e., after the stable limit cycle is reached. In order to provide accurate plots, the average on that time span has been removed from each time series. It is clear that for $Re = 8800$, the dynamics is governed mostly by three harmonics, with subsequent ones being more than a decade lower than the lowest of these top three frequencies, as noted

Table 2: Frequencies of the six leading harmonics (only shown if amplitude (given in parenthesis) is larger than 10^{-4}). Asterisk (*) for the $Re = 9400$ and $Re = 10500$ indicates the presence of doublet peaks accompanying the main spike. Shaded lines indicate that many peaks are found in the Fourier analysis, only the first six in amplitude are given.

Re	F_1	F_2	F_3	F_4	F_5	F_6
8670	0.437 (8.61e-2)	0.874 (1.44e-2)	1.312 (1.46e-3)	1.749 (3.83e-5)	2.186(4.14e-6)	2.623 (1.20e-6)
8680	0.437 (8.66e-2)	0.874 (1.46e-2)	1.311 (1.49e-3)	1.748 (3.90e-5)	2.186 (4.02e-6)	2.623 (1.16e-6)
8690	0.437 (8.71e-2)	0.874 (1.48e-2)	1.311 (1.51e-3)	1.748 (3.97e-5)	2.185 (3.86e-6)	2.622 (1.18e-6)
8700	0.437 (8.76e-2)	0.874 (1.49e-2)	1.311 (1.54e-3)	1.749 (3.86e-5)	2.186 (4.50e-6)	2.627 (1.56e-6)
8800	0.437 (9.23e-2)	0.873 (1.66e-2)	1.310 (1.78e-3)	1.746 (4.50e-5)	2.183 (6.00e-6)	2.620 (1.74e-6)
8900	0.436 (9.64e-2)	0.871 (1.81e-2)	1.307 (2.00e-3)	1.742 (5.13e-5)	2.178 (6.00e-6)	2.614 (2.33e-6)
9000	0.435 (1.00e-1)	0.870 (1.95e-2)	1.305 (2.22e-3)	1.740 (5.66e-5)	2.174 (8.26e-6)	2.609 (3.00e-6)
9100	0.435 (1.04e-1)	0.871 (2.10e-2)	1.306 (2.47e-3)	1.739 (6.29e-5)	2.173 (9.89e-6)	2.608 (4.88e-6)
9200	0.434 (1.07e-1)	0.867 (2.22e-2)	1.301 (2.65e-3)	1.740 (6.89e-5)	2.168 (1.29e-5)	2.602 (5.51e-6)
9300	0.433 (1.10e-1)	0.866 (2.35e-2)	1.299 (2.86e-3)	1.733 (7.55e-5)	2.166 (1.62e-5)	2.599 (7.39e-6)
9350	0.433 (1.12e-1)	0.866 (2.41e-2)	1.299 (2.95e-3)	1.731 (7.94e-5)	2.164 (1.85e-5)	2.597 (7.91e-6)
9395	0.085 (1.12e-3)	0.523* (9.30e-2)	1.046* (9.74e-3)	1.569* (1.61e-3)	2.092* (1.29e-4)	2.615 (1.11e-5)
9400	0.085 (8.91e-5)	0.523* (9.45e-2)	1.046* (1.01e-2)	1.569* (1.70e-3)	2.091* (1.40e-4)	2.614 (1.42e-5)
9405	0.086 (1.08e-4)	0.523* (9.48e-2)	1.046* (1.02e-2)	1.568* (1.72e-3)	2.091* (1.41e-4)	2.614 (1.50e-5)
9450	0.432 (1.14e-1)	0.865 (2.53e-2)	1.297 (3.16e-3)	1.730 (8.63e-5)	2.162 (2.16e-5)	2.594 (8.93e-6)
9500	0.432 (1.15e-1)	0.864 (2.58e-2)	1.296 (3.22e-3)	1.728 (9.30e-5)	2.160 (2.41e-5)	2.591 (1.05e-5)
9600	0.431 (1.18e-1)	0.862 (2.68e-2)	1.294 (3.38e-3)	1.725 (1.05e-4)	2.156 (2.86e-5)	2.587 (1.25e-5)
9700	0.431 (1.20e-1)	0.861 (2.78e-2)	1.292 (3.53e-3)	1.722 (1.20e-4)	2.153 (3.31e-5)	2.583 (1.43e-5)
9800	0.430 (1.23e-1)	0.860 (2.87e-2)	1.290 (3.67e-3)	1.720 (1.35e-4)	2.150 (3.81e-5)	2.580 (1.65e-5)
9900	0.429 (1.25e-1)	0.858 (2.94e-2)	1.287 (3.78e-3)	1.716 (1.61e-4)	2.145 (4.32e-5)	2.574 (1.75e-5)
10000	0.429 (1.27e-1)	0.857 (3.02e-2)	1.286 (3.89e-3)	1.714 (1.75e-4)	2.143 (4.88e-5)	2.572 (2.07e-5)
10100	0.428 (1.29e-1)	0.856 (3.09e-2)	1.284 (3.98e-3)	1.712 (1.99e-4)	2.140 (5.52e-5)	2.568 (2.32e-5)
10200	0.427 (1.31e-1)	0.855 (3.16e-2)	1.282 (4.05e-3)	1.709 (2.27e-4)	2.136 (6.25e-5)	2.564 (2.55e-5)
10300	0.427 (1.33e-1)	0.853 (3.22e-2)	1.280 (4.11e-3)	1.706 (2.57e-4)	2.133 (7.02e-5)	2.560 (2.83e-5)
10400	0.426 (1.34e-1)	0.852 (3.27e-2)	1.278 (4.15e-3)	1.704 (2.89e-4)	2.130 (7.81e-5)	2.556 (3.12e-5)
10500	0.167 (6.79e-5)	0.597* (1.37e-1)	1.191* (6.50e-3)	1.786* (3.02e-3)	2.384* (2.48e-4)	2.980 (2.17e-6)
10600	0.425 (1.38e-1)	0.849 (3.36e-2)	1.274 (4.21e-3)	1.698 (3.60e-4)	2.123 (9.56e-5)	2.547 (3.68e-5)
10700	0.159 (1.29e-2)	0.265 (2.71e-2)	0.424 (1.27e-1)	0.530 (1.04e-2)	0.689 (7.09e-2)	0.849 (2.74e-2)
10800	0.160 (1.14e-2)	0.264 (2.86e-2)	0.424 (1.24e-1)	0.528 (1.08e-2)	0.688 (8.20e-2)	0.848 (2.55e-2)
10900	0.160 (1.13e-2)	0.264 (3.09e-2)	0.424 (1.17e-1)	0.528 (1.39e-2)	0.688 (9.36e-2)	0.848 (2.05e-2)
11000	0.160 (1.08e-2)	0.264 (3.19e-2)	0.423 (1.18e-1)	0.527 (1.45e-2)	0.687 (1.01e-1)	0.846 (2.20e-2)
11100	0.264 (3.43e-2)	0.423 (1.12e-1)	0.527 (1.61e-2)	0.687 (1.11e-1)	0.846 (1.81e-2)	0.950 (1.17e-2)
11200	0.263 (3.48e-2)	0.422 (1.12e-1)	0.526 (1.78e-2)	0.686 (1.17e-1)	0.845 (1.79e-2)	0.949 (1.24e-2)
11300	0.263 (3.39e-2)	0.422 (1.05e-1)	0.526 (2.02e-2)	0.685 (1.29e-1)	0.844 (1.50e-2)	0.947 (1.35e-2)
11400	0.262 (3.44e-2)	0.421 (1.07e-1)	0.525 (2.18e-2)	0.684 (1.32e-1)	0.842 (1.54e-2)	0.946 (1.36e-2)
11500	0.587 (1.94e-1)	1.175 (7.91e-3)	1.762 (4.26e-3)	2.349 (6.39e-4)	2.937 (1.17e-5)	
11600	0.586 (1.98e-1)	1.173 (8.32e-3)	1.759 (4.47e-3)	2.346 (6.89e-4)	2.932 (1.40e-5)	
11700	0.261 (3.39e-2)	0.420 (9.64e-2)	0.523 (2.33e-2)	0.681 (1.54e-1)	0.942 (1.65e-2)	1.362 (1.29e-2)
11800	0.261 (3.29e-2)	0.419 (9.23e-2)	0.522 (2.20e-2)	0.680 (1.61e-1)	0.941 (1.71e-2)	1.360 (1.43e-2)
11900	0.261 (3.15e-2)	0.420 (8.70e-2)	0.521 (1.80e-2)	0.679 (1.68e-1)	0.939 (1.70e-2)	1.358 (1.54e-2)
12000	0.260 (3.03e-2)	0.418 (7.96e-2)	0.521 (1.79e-2)	0.678 (1.76e-1)	0.939 (1.84e-2)	1.357 (1.76e-2)

in Fig. 7. Other higher Re cases display similar spectra for $Re = 8800$ to 9300 (not shown here). The displayed harmonics for different Re in this range, also show the primary modal frequencies to remain constant up to $Re = 9300$, as shown in Tab. 2 (the features of which is discussed in details later). In the range of Re around 9400, the fundamental and higher harmonics display variations of frequencies that is typically different with two lower peaks appearing on each side of a central spike and there are more than three spikes, including one with very low frequency. Also the spectrum for $Re = 9400$ in Fig. 7 shows rightward shift of all the major harmonics. As noted in Fig. 6, the flow behavior above $Re \approx 9450$ resembles flows noted for lower post-critical Reynolds numbers. This is clearly seen in Fig. 7 for $Re = 9800$ with six peaks in the spectrum. This is the state of flow behavior till $Re = 10400$, these frequencies revert back to the lower constant value, that was noted for up to $Re = 9300$; with the number of peaks increasing, as shown in Fig. 7 for $Re = 9800$. For the higher $Re = 10700$ shown in Fig. 7, one notices a large numbers of spectral peaks with more than one dominant comparable peaks. This feature of two dominant modes are noted up to $Re = 11400$. We also remark that unlike the spectra for lower Re cases, these higher Re cases have many peaks in the spectrum. The spectrum for $Re = 11500$ and 11600 (not shown here) again displays very clean spectrum with very few peaks, as was noted for $Re = 8800$ to 9300. For $Re = 11700$ and higher (not shown here), once again the spectrum represent two dominant modes, with larger number of peaks present. The only difference between this case and the one for $Re = 11400$ is that the peak amplitude is higher for the higher frequency, as shown in Fig. 7.

5.3. Phase Portrait Analysis

For the 2D LDC flow, the governing NSE in primitive variables are given by two evolution equations for velocity components (u, v), apart from the mass conservation equation. Thus, the phase portrait of this dynamical system should ideally be given in the (u, v)-plane with time as the parameter. However in our approach we have solved NSE in the (ψ, ω)-formulation thereby exactly satisfying the mass conservation equation. Also, this formulation avoids the pressure-velocity coupling problem. Thus, we propose to depict the phase space portrait by plotting vorticity and its time rate in Fig. 8 for the sampling point at $(x = 0.95, y = 0.95)$. The time series depicting the history of vorticity evolution for these cases have been already shown in Fig. 3. We have plotted the phase space portrait for the indicated time ranges, where the time series indicates the existence of stable limit cycle-like behavior. This is confirmed by the phase space

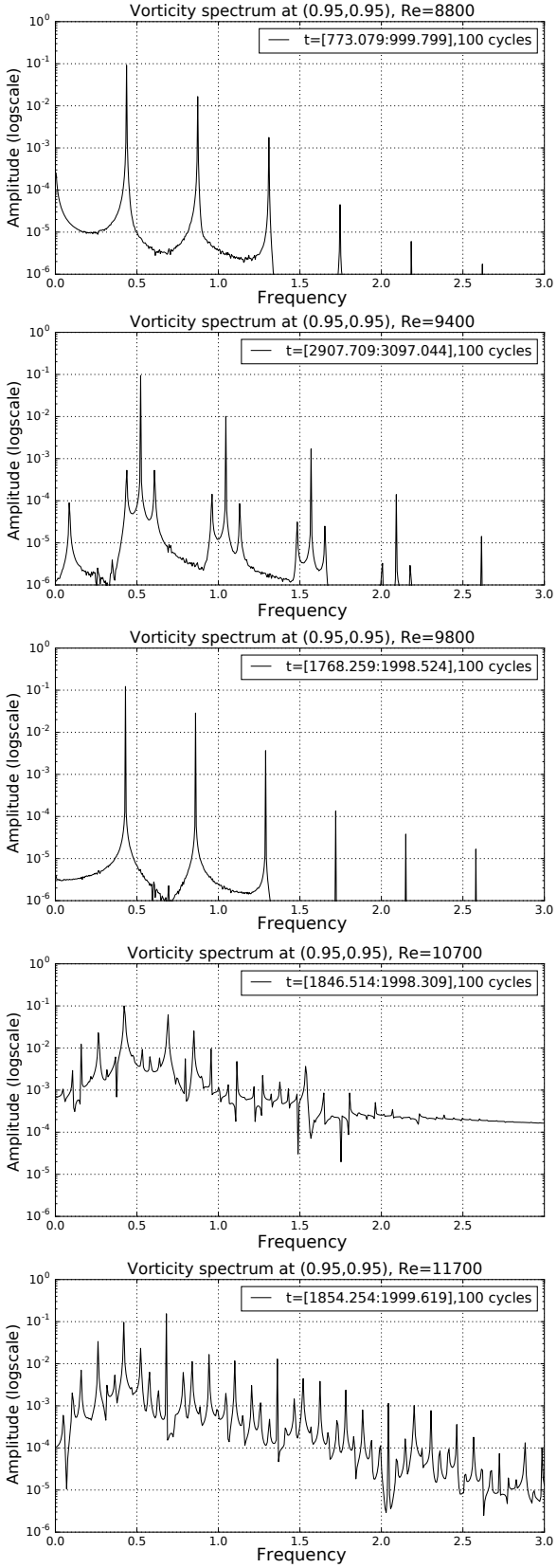


Figure 7: The frequency spectrum of the vorticity time series shown for all the simulated Reynolds numbers, for the solution obtained from unsteady Navier-Stokes equation and the data are for $x = 0.95$ and $y = 0.95$.

portrait for the relatively lower post-critical Re values. Except for $Re = 9400$ case, all the other three cases for $Re = 8800$, 10000 and 10300 show almost identical limit cycles. Excursion of ω about the mean value is almost similar and there is a characteristic dip in the limit cycle in all the three cases, except for $Re = 9400$, for which the phase portrait almost shows oblong oval shape. The width of the limit cycle is slightly wider for $Re = 9400$, due to the typical frequency spectrum shown in Fig. 7 with dominant spikes accompanied by side-bands on either side. As the second dominant cluster in the spectrum is one decade higher than the global maximum for this case, explains the near-oval shape of the phase space portrait. In contrast, for the other three Re cases, the global maximum and the nearest higher peaks are not widely separated and such multi-modal behavior, whose mutual interaction can reduce $\frac{d\omega}{dt}$ for a particular combination of phase (i.e., at a particular time) during each cycle.

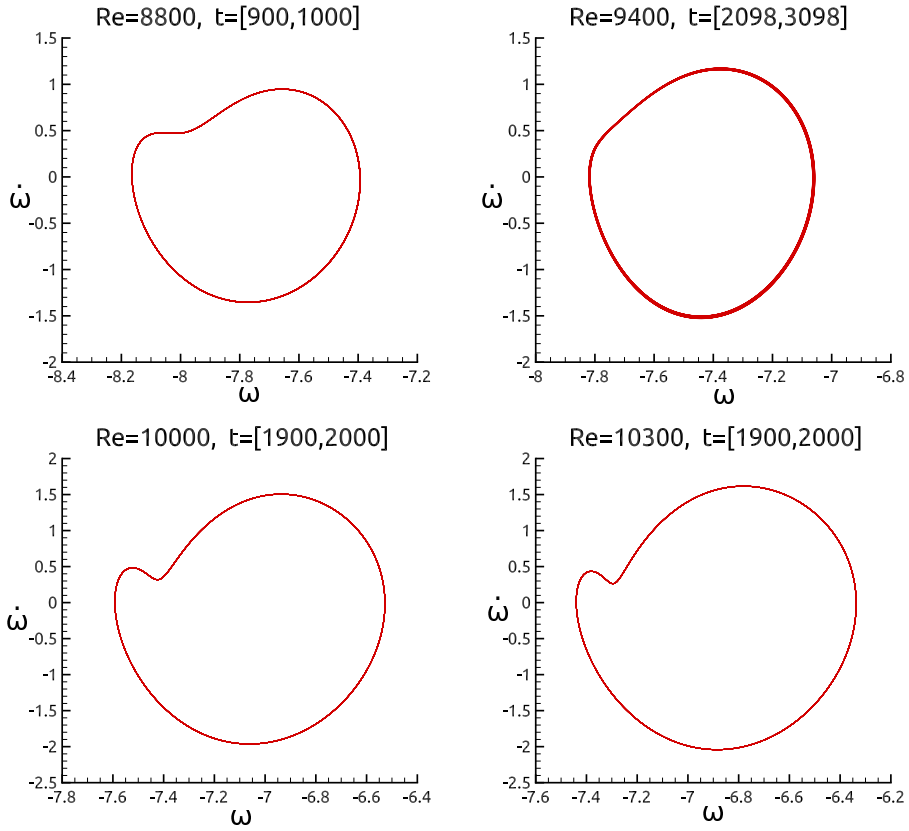


Figure 8: Phase portrait of the vorticity $(\omega, \dot{\omega})$ at $(0.95, 0.95)$ once the last 100 time interval for $Re = 8800$, 9400, 10000, 10300. Trajectories are closed since stable limit cycle has been reached.

5.4. Multi-periodic State

Beyond the third Hopf bifurcation, i.e., beyond $Re = 10600$, the flow field changes qualitatively, as noted from the time series in Fig. 3 and the frequency spectrum in Fig. 7 for Re = 10700. The time series is characterized by the significant modulation noted for time greater than $t \approx 1000$. The same characteristic is noted better in the spectrum, where one can see many dominant frequencies of comparable magnitude. Presence of multiple time scales lead to vorticity contours, as shown in top frames of Fig. 9 for $Re = 11000$ and $Re = 12000$ which shows the irregular six gyrating vortices and their center is not aligned with the geometric center of the cavity. The irregularity of gyrating vortices is more pronounced at the higher Reynolds number case. However, this aspect is seen very graphically from the phase space portrait shown in the bottom frames of Fig. 9. The lower Re case show distinct limit cycles with larger basin of attractor and quasi-periodicity of the flow field is evident. However, for $Re = 12000$ the phase space portrait is characterized by large continuous basin of attractor with its appearance given by a Mobius strip.

6. Numerical sensitivity of the problem

The singular LDC problem is very sensitive to numerical setup. In this section we discuss two major issues affecting the solution, namely start-up conditions and grid sensitivity of our numerical method.

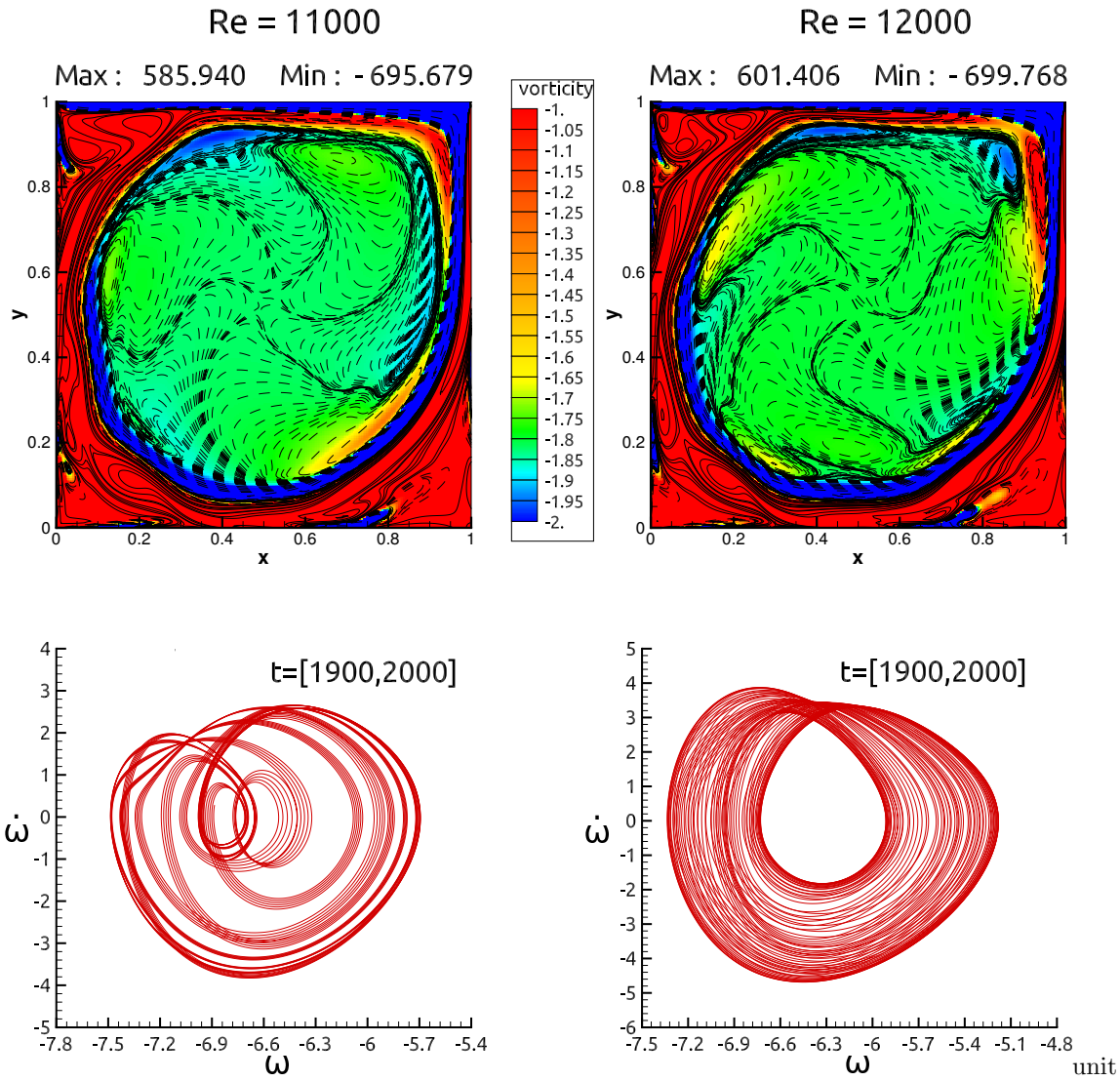


Figure 9: Two examples of unstable limit cycle for $Re = 11000$ (left) and $Re = 12000$ (right). Vorticity contour plot is shown in the first row at $t = 2000$ and phase portrait in the second.

6.1. Influence of grid resolution

Computations have been performed on two different grids, i.e., (257×257) and (513×513), in order to assess effects of grid on the simulation. Figure 10 displays time series for two different grids which clearly behave differently. On the one hand, the coarse grid exhibit a secondary instability around $t = 1200$, that leads to the final limit cycle. Because of finer wall resolution, calculated wall vorticity is higher for the finer grid calculation. Yet, the numerical excitation caused by sources of error is lower for the finer grid. As a consequence, both the mean and fluctuation of disturbance vorticity is lower for the finer grid, which causes upward shift of the mean vorticity line, i.e., reduction of mean vorticity of disturbance. No secondary instability is seen for the finer grid and still a similar limit cycle is reached with marginal difference in amplitude and frequency of the fluctuating component of vorticity. Moreover, final state is stable for $Re \leq 9400$ when computations are carried on the finer (513×513) grid, i.e., $Re_{cr1} \in [9400, 9450]$. It emphasizes that the flow is driven by the receptivity aspect of the problem, with coarser grid (and less accurate numerical methods) having larger excitation due to implicit error, shows early onset of first Hopf bifurcation. This will be further discussed in the subsection 6.3.

6.2. Effect of start-up conditions

The top sub-figure of Fig. 11 depicts the time series stored for $(0.95, 0.95)$ on (257×257) grid for $Re = 8670$ with two different initial conditions. The dashed line corresponds to the usual impulsive start whereas the solid line corresponds to the solution obtained by ramping up from $Re = 8660$ equilibrium solution. We note that the projected solution starting from lower Re remains quiescent (negligibly small variations), while the solution started impulsively shows non-zero values at the sampling point.

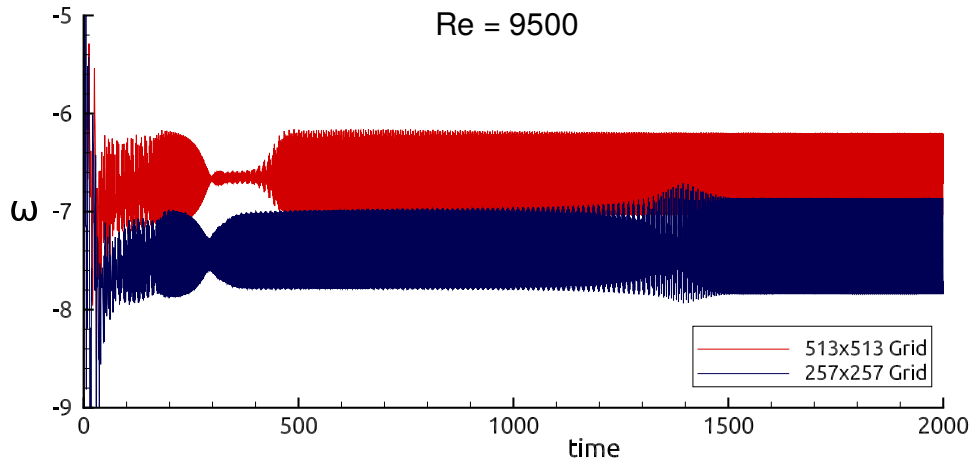


Figure 10: Vorticity time series are shown for $Re = 9500$ for two different grid spacing 257×257 and 513×513.

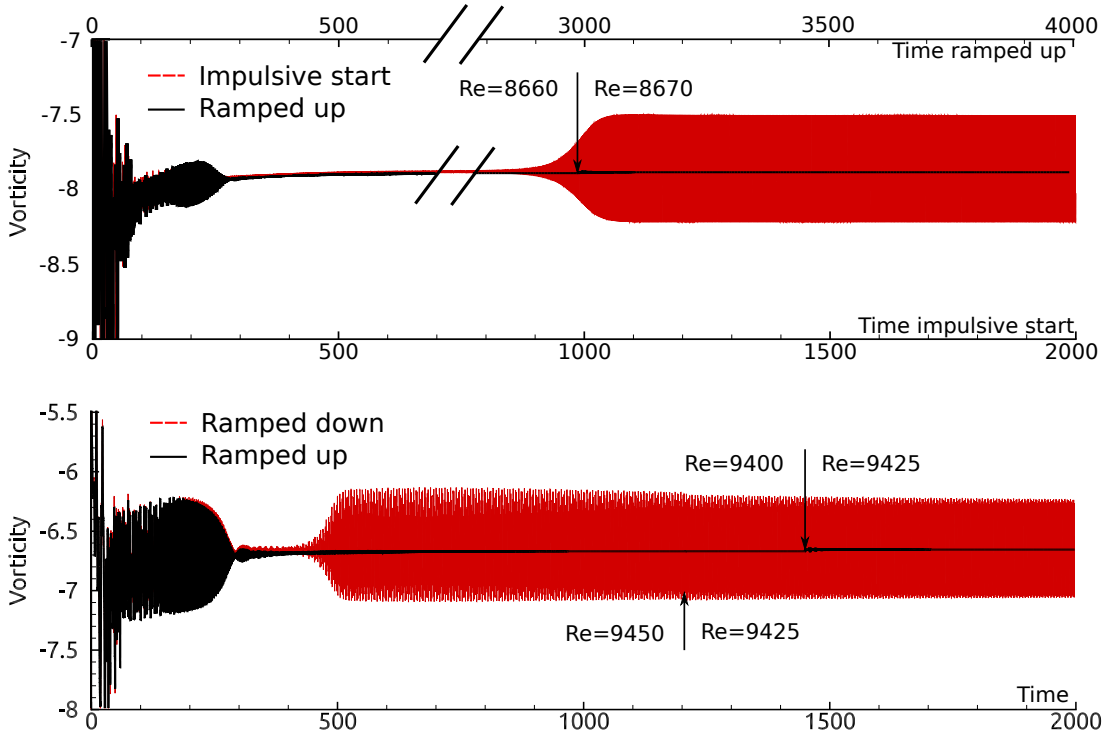


Figure 11: The figure in top shows $Re = 8670$ (257×257 grid) being initialized by impulsive start-up (dashed) and ramp up from the limit cycle solution of $Re = 8660$ at t=2000. The bottom figure shows result with the fine grid of (513×513) size for $Re = 9425$ computed from the limit cycle solutions of $Re = 9400$ (solid) and $Re = 9450$ (dashed).

The bottom sub-figure of Fig. 11 is for (513×513)-grid in the vicinity of the bifurcation obtained for this grid near $Re = 9400$. Two different start-up cases are presented : (a) when the solution is obtained for $Re = 9425$ starting from an equilibrium solution obtained for $Re = 9450$ and (b) when the initial solution is projected from the case of $Re = 9400$. For the latter case, the vorticity field does not show any disturbance, while the former case shows significant disturbance vorticity. This justifies, *a posteriori*, the use of impulsive start-up which is known to excite all modes of oscillation simultaneously by equal magnitude.

6.3. Computational bifurcation analysis: Is there a universal critical Reynolds number for primary bifurcation?

In introduction, we have noted that different researchers have reported different critical Re_{cr1}, ranging from $7763 \pm 2\%$ to 10,500, with a marked clustering around Re_{cr1} in the vicinity of 8000. For example, $Re_{cr1} = 8018$ in [3] and 8031.93 in [32]. Cazemier *et al.* [11] reported Re_{cr1} at 7972, while Bruneau and Saad [8] suggested this to be in the range of $8000 \leq Re_{cr1} \leq 8050$. Sengupta *et al.* [38] have described multiple Hopf bifurcations, showing the first one at 7933 and

the second at 8187, using uniform (257×257) grid, with bifurcation diagram drawn using the amplitude of the primary mode only. In [28], this value is reported at $7987 \pm 2\%$. In light of these scattered values, we furthermore investigated why the present simulation using NCCD scheme using the uniform (257×257) grids produces Re_{cr1} in the narrow range of 8660 and 8670, when no explicit excitation is applied.

One of the attributes of the used NCCD scheme is its near-spectral accuracy and it has been reasoned in [26, 27], the trigger for the unsteadiness is the aliasing error originating near the top right corner of the LDC, while the truncation, round-off and dispersion error is extremely negligible [44, 45]. To circumvent the issue of lower numerical excitation in the present work (which is based on the method in [26, 27, 38]), we position a pulsating vortex ω_s at a location $r_0 = (0.015625, 0.984375)$ whose spread is defined by the exponent α,

$$\omega_s = A_0(1 + \cos(\pi(r - r_0)/0.0221))\sin(2\pi f_0 t) \quad \text{for } (r - r_0) \leq 0.0221$$

where in the presented results here we have taken $f_0 = 0.41$ for different amplitude cases. For $Re = 8660$ and below, we start with $A_0 = 1.0$. Once the excitation is started, one notices the vorticity to grow and saturate to a limit cycle. Once the limit cycle is set up, the excitation source is switched off and yet the limit cycle continues. The saturated limit cycle amplitude for decreasing Reynolds numbers are shown in Fig. 12 along with the unexcited cases (shown by hollow triangle facing towards left, up to $Re = 8670$) for the sampling point at $x = 0.95, y = 0.95$. The excited cases with $A_0 = 1.0$ and $f_0 = 0.41$ are shown with filled triangles facing towards right, up to $Re = 8030$. Below this Reynolds number value, the vortex source strength has to be increased to obtain self-sustained limit cycle, as shown by the upright hollow triangle for $Re = 8025$ and $A_0 = 10.0$.

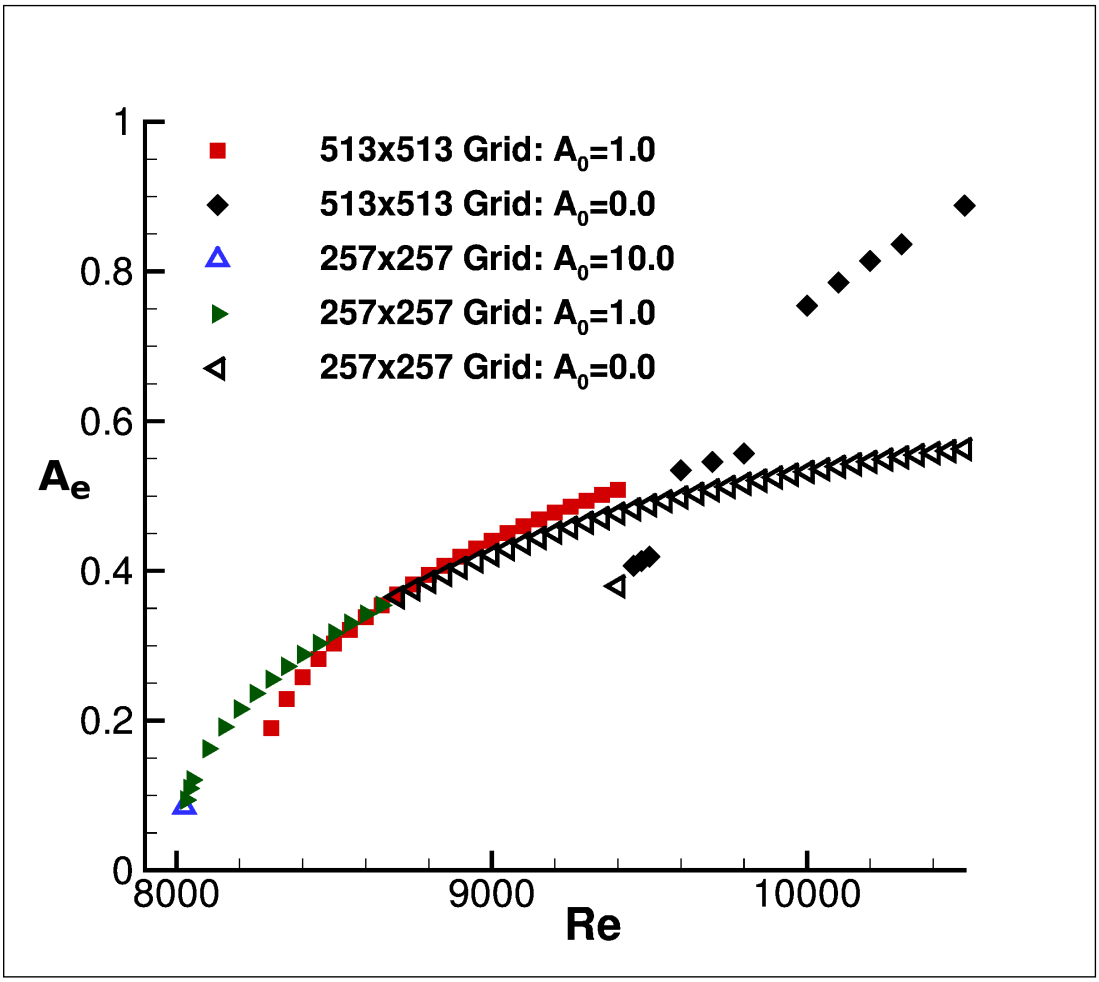


Figure 12: The extended bifurcation diagram obtained using the pulsating vortex source near the top left of the LDC. It is noted that as Reynolds number decreases, the imposed pulsating vortex strength is to be increased. For $Re = 8020$, no excitation with higher amplitude produces stable final limit cycle.

Below this Reynolds number, increasing strength of pulsating vortex does not produce stable limit cycle. We note that the imposed vortical perturbation in the limiting amplitude case of $A_0 = 10.0$, constitute a perturbation level of around

20 percent of the maximum vorticity in the domain. Thus, this computational exercise indicate that the first critical Reynolds number (Re_{cr1}) lies between 8020 and 8025 and similar range of the value noted by many researchers as noted in the previous paragraph. The present DNS does not require linearization or making assumptions pertaining to growth in time or space.

The fact that the flow in LDC cannot be made unsteady for $Re = 8000$ is demonstrated in Fig.13. In the top frame, the vorticity time series are shown at the sampling point $(x = 0.9922, y = 0.9922)$ for the indicated Re values, which is decreased by a step of 100, starting with $Re = 8400$ up to 8100, for $A_0 = 1.0,\ f_0 = 0.41$. For this amplitude of excitation, one notices a modulated time series, due to mutual interference of the natural frequencies with the imposed time scale. However, when the excitation is switched off at $t = t_s$, a pure and stable limit cycle is obtained in Fig.13(a), up to $Re = 8100$. For the case of $Re = 8000$, the withdrawal of excitation causes the vorticity time series to decay. In frames (b) to (d), the results for the cases are shown for which the pulsating vortex amplitude is increased to $A_0 = 1$, 2 and 10 for $Re = 8000$, with the frequency kept the same at $f_0 = 0.41$. When the amplitude of excitation is doubled to 2.0, the modulated time series is noted, and thereafter the exciter is switched off at t_s. One notices that beyond t_s, with exciter switched off, the flow inside the LDC approaches a steady state. For the case of $A_0 = 10.0$, when the exciter is turned on, instead of the modulated time series, one notices a wide-band response of the vorticity field without any perceptible limit cycle. Even with such large excitation amplitude for such a Re, when the excitation is switched off at t_s, one notices the vorticity field to be quiescent again in a short time. Thus, this value of $Re = 8000$ shows that the flow is stable, even for a large perturbation, which is of the order of 20% of the maximum vorticity in the domain.

Finally in Fig.14 we show results for $Re = 8500$, for different amplitude of pulsating excitation source. For this Reynolds number we note the causation of stable limit cycle for $A_0 = 0.06$, while it goes back to steady state for $A_0 = 0.04$. It is also equally important to note that for all the amplitudes of excitation, the final limit cycle has always the same amplitude. This indicates that the present analysis produces results which are invariant of the way one excites the flow. It also shows that this flow shows the receptivity of the flow field, as has been highlighted earlier for external vortex dominated flow in [35].

7. Summary and Conclusion

Flow inside a LDC is shown to display multi-modal behavior following first Hopf bifurcation with varying Re, depending upon the discretization schemes. DNS following impulsive start, is used to show initial temporal growth followed by nonlinear saturation of disturbance. Researchers have reported different value of Re_{cr1} using direct simulation of Navier-Stokes equation. This approach differs from bifurcation studies using global instability study of an equilibrium flow due to adopted nonlinear approach and not restricting the analysis to temporal instability only. Here, flow in LDC is investigated using high accuracy NCCD scheme for DNS using stream function-vorticity, (ψ, ω)-formulation for the range of Reynolds numbers, $8000 \leq Re \leq 12000$.

The accuracy aspect of DNS adopted here has been shown conclusively via (a) demonstration of a very weak transient polygonal core vortex surrounded by relatively stronger gyrating vortices, which appear as a constellation, as shown earlier in [26, 27]. This requires extreme accuracy of resolving convection and diffusion terms, otherwise the created aliasing error affects the numerical stability of the method and also maintaining the complex equilibrium in creating the constellation and (b) the value of the first critical Reynolds number, Re_{cr1} depends strongly upon the error dynamics of the adopted discretization scheme for a model equation [36]. One of the major achievements of the present work is to show that Re_{cr1} can be further reduced by explicit excitation for very high accuracy numerical schemes. Thus, we distinguish between excited and unexcited LDC flow here. This difference is described in details in section 6.3, where it is established that the delay of onset for Re_{cr1} is an attribute of accuracy of the present method, and not due to excessive numerical diffusion of the methods, as in [6, 8, 14, 17].

The vorticity evolves here for the unexcited cases, as in Fig. 5, where one sees the presence of transient triangular vortex at the core is unsteady for $Re > Re_{cr1}$. Dynamics is further studied by time series analysis of vorticity at the point $(0.95, 0.95)$, chosen based on information in Fig. 1. Overall, the dynamics is characterized in Fig. 2, by an onset showing irregular transient behavior (in Range-1a), followed by coherent temporal decay of the time series in Range-1b. Range-2 represents quiescent state. In this figure, Range-3a is defined as the region where the flow experiences an instability, which in the literature is often analyzed by linearized temporal theory. Secondary instability marked as Range-3b in the time series for $Re = 9300$ has an onset from $Re = 9100$ onwards. Such secondary instability of limit cycle can also be regarded as nonlinear instability and is reported for the flow inside LDC. In Fig. 2, Range-4 demarcates the stable limit cycle for $Re = 8800$ case. It is noted that the time at which the final limit cycle is reached varies greatly with Re.

Present DNS-based approach to study Hopf bifurcations is distinct from that in the literature, where normal mode instability analysis is attempted for steady state solutions. Here, we show multiple Hopf bifurcations in the range $8000 \leq Re \leq 12000$, for the unexcited impulsively started cases. Also, we do not project solution from one Re to another, as this is shown to produce erroneous flows, as shown in subsection 6.2 (Fig. 11) especially near Hopf bifurcations. In [3], the authors have reported a possible second bifurcation for a supposed range of [9687, 9765).

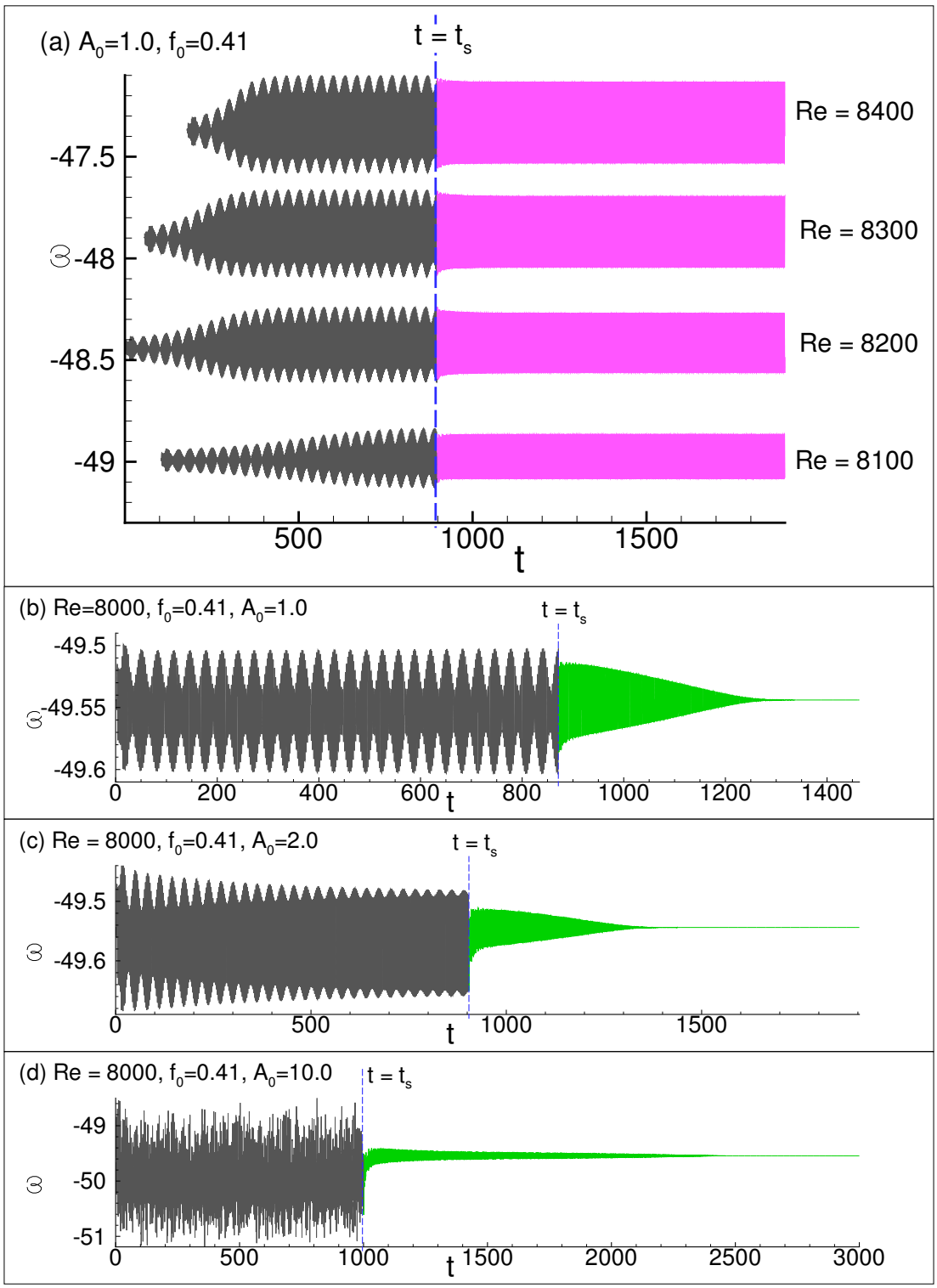


Figure 13: The vorticity time series at the indicated sampling station at $x = 0.9922,\ y = 0.9922$, for the indicated Reynolds numbers, for which stable limit cycle is obtained by the pulsating vortex source near the top left of the LDC.

The limit cycle amplitude variation with respect to Re has been used to characterize Hopf bifurcations. In Fig. 6, it is shown that A_e^2 is proportional to Re on different segments of Re for the unexcited cases, which are clearly separated from one to the next bifurcation. For the (257×257)-grid, the Hopf bifurcations for unexcited case are located at $Re_{cr1} = 8660$ followed by $Re_{cr2} = 9400$, $Re_{cr3} \in [10500, 10700]$ and $Re_{cr4} = 11700$.

Here $Re_{cr1} \approx 8660$ is noted for the case of no excitation, and such a high value is due to higher accuracy of the method

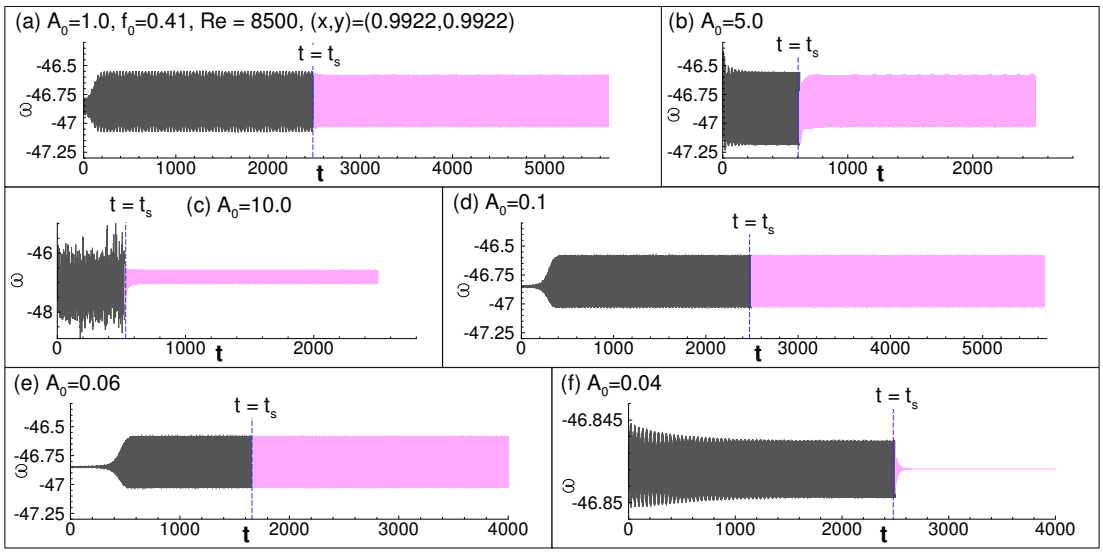


Figure 14: The vorticity time series at the sampling station for $Re = 8500$ shown for cases with varying amplitude of excitation, to find the limiting amplitude for which stable limit cycle is obtained by the pulsating vortex source near the top left of the LDC.

discussed in section 6.1, with low aliasing error responsible in triggering unsteadiness. This value has been lowered by the application of explicit excitation, in the form of a pulsating vortex placed near the top left corner. Following this procedure, it has been shown that Re_{cr1} can be lowered to as small a value in between 8020 and 8025, shown in Fig. 12 and described in section 6.3.

Investigated LDC flow is characterized by multiple time scales at any Re, which are weak function of Re in selective intervals, punctuated by multiple bifurcations, as shown in Table 2. Further investigation of bifurcations are made with FFT of time series results shown in Fig. 7. The lower post-critical Re cases show distinct harmonics, while the case for $Re = 9400$ has unique pattern of triplet of harmonics, as also noted in Table 2. The very high Re cases are characterized by broadband spectrum, with many peaks and having two incommensurate dominant frequencies as noted in the bottom frame of Fig. 7. Phase portraits are plotted over a time interval of 100 units, further showing vorticity and its time rate, which show trajectories for very stable limit cycles for low Re cases with very narrow basin of attractors, which are unlike the last two segments, where trajectories belong to larger dispersed bands with wider basin of attractors. This irregularity is observed in vorticity contours for higher Re, where one notices the basin as a continuous sheet resembling a Möbius strip.

In conclusion, we explain the universality of the primary Hopf bifurcation Reynolds number, Re_{cr1}, by showing the effects of pulsating a vortex at a fixed frequency near the top left corner of the LDC. Presented results in Figs. 12 to 14 show the universal value of Re_{cr1} to be within the range of 8020 and 8025. The present investigation achieves two primary goals: First, it reconciles that Re_{cr1} obtained by different numerical approaches can be shown in identical range, provided the equilibrium flow obtained is of good quality, untainted by excessive diffusion. The present high accuracy computation, on the other hand, require explicit excitation to obtain Re_{cr1} towards the *universal* value of the same. This is shown here for the first time reconciling high accuracy approach for DNS with results based on global stability analysis.

[1] Sengupta T. K., Ganeriwal G. and De S., Analysis of central and upwind compact schemes, *J. Comput. Phys.*, **192**, *677-694* (2003).

[2] Auteri F., Quartapelle L. and Vigevano L., Accurate ω-ψ spectra; solution of the singular driven cavity problem, *J. Comput. Phys.*, **180**, *597-615* (2002).

[3] Auteri F., Parolini N. and Quartapelle L., Numerical investigation on the stability of singular driven cavity flow, *J. Comput. Phys.*, **183**, *1-25* (2002).

[4] Beckers M. and van Heijst G. J. F., The observation of a triangular vortex in a rotating fluid, *Fluid Dyn. Res.*, **22**, *265-279* (1998).

[5] Bender C. M. and Orszag S. A., *Advanced Mathematical Methods for Scientists and Engineers I: Asymptotic Methods and Perturbation Theory*, Springer, USA (1999).

[6] Boppana V. B. L. and Gajjar J. S. B., Global flow instability in a lid-driven cavity, *Int. J. Num. Meth. Fluids*, **62**, *827-853* (2009).

[7] Botella O. and Peyret R., Benchmark spectral results on the lid-driven cavity flow, *Comput. Fluids,* **24***, 421-433* (1998).

[8] Bruneau C. H. and Saad M., The 2D lid-driven cavity problem revisited, *Comput. Fluids,* **35***(3), 326-348* (2006).

[9] Burgraff O. R. Analytical and numerical study of the structure of steady separated flows, *J. Fluid Mech.,* **24***, 113-151* (1966).

[10] Carnevale G. F. and Kloosterziel R. C., Emergence and evolution of triangular vortices, *J. Fluid Mech.,* **259***, 305-331* (1994).

[11] Cazemier W., Verstappen R. W. C. P. and Veldman A. E. P., Proper orthogonal decomposition and low-dimensional models for driven cavity flows, *Physics Fluids,* **10***(7) 1685-1699* (1998).

[12] Drazin P. G. and Reid W. H., *Hydrodynamic Stability*, Cambridge Univ. Press, UK (1981).

[13] Eckhaus W., *Studies in Nonlinear Stability Theory*, Springer: Berlin (1965).

[14] Erturk E., Corke T. C., Gökcöl C., Numerical solutions of 2-D steady incompressible driven cavity flow at high Reynolds numbers, *Int. J. Num. Meth. Fluids,* **48***(7), 747-774* (2005).

[15] U. Fey, M. König and H. Eckelmann, A new Strouhal-Reynolds-number relationship for the circular cylinder in the range 47<Re $< 2\times 10^5$, *Phys. Fluids,* **10***, 1547* (1998).

[16] Fortin A., Jardak M., Gervais J. J. and Pierre R., Localization of Hopf bifurcations in fluid flow problems, *Int. J. Num. Meth. Fluids,* **24***(11), 1185-1210* (1997).

[17] Ghia U., Ghia K. N. and Shin C. T., High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method, *J. Comput. Phys.,* **48***, 387-411* (1982).

[18] Goodrich J. W., Gustafson K. and Halasi K., Hopf bifuraction in the driven cavity, *J. Comput. Phys.,* **90***, 219-261* (1990).

[19] Gustafson K. and Halasi K., Vortex dynamics of cavity flows, *J. Comput. Phys.,* **64***, 279-319* (1986).

[20] Hancock C., Lewis E. and Moffatt H. K., Effect of inertia in forced corner flows, *J. Fluid Mech.,* **112***, 315-327* (1986).

[21] Schmid P. J. and Henningson D. S., *Stability and Transition in Shear Flows*, Springer Verlag: New York (2001).

[22] Jansson T. R. N., Haspang M. P., Jensen K. H., Hersen P. and Bohr, T., Polygons on a rotating fluid surface, *Phys. Rev. Lett.,* **96***, 174502* (2006).

[23] Sengupta T. K., Singh N. and Suman V. K., Dynamical system approach to instability of flow past a circular cylinder, *J. Fluid Mech.,* **656***, 82-115* (2010).

[24] Landau L. D. and Lifshitz E. M., *Fluid Mechanics*, Pergamon Press, UK (1959).

[25] E. Leriche, Direct Numerical Simulation in a Lid-Driven Cubical Cavity at High Reynolds Number by a Chebyshev Spectral Method, *J. Sci. Comput.* **27**, *335-345* (2006).

[26] Sengupta T. K., Lakshmanan V. and Vijay V. V. S. N., A new combined stable and dispersion relation preserving compact scheme for non-periodic problems, *J. Comput. Phys.,* **228***, 3048-3071* (2009).

[27] Sengupta T. K., Vijay V. V. S. N. and Bhaumik S, Further improvement and analysis of CCD scheme: Dissipation discretization and de-aliasing properties, *J. Comput. Phys.,* **228***, 6150-6168* (2009).

[28] Osada T. and Iwatsu, R., Numerical simulation of unsteady driven cavity flow, *J. The Phys. Soc. Japan.* **80***, 094401* (2011).

[29] Peng Y.-F., Shiau Y.-H. and Hwang R. R.. Transition in a 2-D lid-driven cavity flow, *Comput. Fluids,* **32***, 337-352* (2003).

[30] Poliashenko M., Aidun C. K., A direct method for computation of simple bifurcations. *J. Comput. Phys.,* **121***(2), 246-260* (1995).

[31] Sengupta T. K., Haider S. I., Parvathi M. K. and Pallavi G., Enstrophy-based proper orthogonal decomposition for reduced-order modeling of flow past a cylinder, *Phys. Rev. E,* **91***, 043303* (2015).

[32] Sahin M. and Owens R. G., A novel fully-implicit finite volume method applied to the lid-driven cavity problem. Part II. Linear stability analysis, *Int. J. Num. Meth. Fluids,* **42***, 79-88* (2003).

[33] Schreiber R. and Keller H. B., Driven cavity flows by efficient numerical techniques, *J. Comput. Phys.,* **49***, 310-333* (1983).

[34] Shen J., Hopf bifurcation of the unsteady regularized driven cavity flow , *J. Comput. Phys.,* **95***, 228* (1991).

[35] Sengupta T. K., Instabilities of Flows and Transition to Turbulence, CRC Press, USA (2012).

[36] Sengupta T. K., High Accuracy Computing Methods: Fluid Flows and Wave Phenomena, Cambridge Univ. Press, USA (2013).

[37] Sengupta T. K. and Nair M. T., Upwind schemes and large eddy simulation, *Int. J. Num. Meth. Fluids,* **31***(5), 879-889* (1999).

[38] Sengupta T. K., Singh N. and Vijay V. V. S. N., Universal instability modes in internal and external flows, *Comput. Fluids,* **40***, 221-235* (2011).

[39] Seydel R., *Practical Bifurcation and Stability Analysis from Equilibrium to Chaos*, Springer: Berlin (1994).

[40] Stuart J. T., On the nonlinear mechanics of wave disturbances in stable and unstable parallel flows. Part 1. The basic behaviour in plane Poiseuille flow, *J. Fluid Mech.,* **9***, 353-370* (1960).

[41] Suman, V.K., Sengupta, T. K., Prasad, C. J. D., Mohan, K. S. and Sanwalia, D., Spectral analysis of finite difference schemes for convection diffusion equation, *Computers and Fluids, 150, 95-114* (2017).

[42] Van der Vorst H. A., Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems, *SIAM J. Sci. Stat. Comput.,* **12***, 631- 644* (1992).

[43] Yu C. H., Bhumkar Y. G. and Sheu Tony W. H., Dispersion relation preserving combined compact difference schemes for flow problems, *J. Sci. Comput.* **62***, 482-516* (2015).

[44] Sengupta T. K., Dipankar, A. and Sagaut, P., Error dynamics: Beyond von Neumann analysis, *J. Comput. Phys.* **226(2), 1211-1218** (2007).

[45] C. David, P. Sagaut and T.K. Sengupta, A linear dispersive mechanism for numerical error growth: Spurious caustics, *Euro. J. Mechanics B/Fluids* **28***(1), 146-151* (2009)

[46] Girault, G., Guevel, Y., Cadou, J. M., An algortihm for the computation of multiple Hopf bifurcation points based on Padé approximants, *Int. J. Num. Methods Fluids* **68***, 1189-1206* (2012)

JOURNAL OF MATHEMATICAL STUDY
J. Math. Study, Vol. **x**, No. x (201x), pp. 1-27

POD applied to numerical study of unsteady flow inside lid-driven cavity

Lucas Lestandi[1,*], Swagata Bhaumik[2], Tapan K Sengupta[2], G.R. Krishna Chand Avatar[2], Mejdi Azaïez[3]

[1] *University of Bordeaux, I2M UMR 5295, France*
[2] *HPCL, IIT Kanpur, Kanpur, India*
[3] *Bordeaux Institut National Polytechnique, I2M UMR 5295, France*

Abstract. Flow inside a lid-driven cavity (LDC) is studied here to elucidate bifurcation sequences of the flow at super-critical Reynolds numbers (Re_{cr1}) with the help of analyzing the time series at most energetic points in the flow domain. The implication of Re_{cr1} in the context of direct simulation of Navier-Stokes equation is presented here for LDC, with or without explicit excitation inside the LDC. This is aided further by performing detailed enstrophy-based proper orthogonal decomposition (POD) of the flow field. The flow has been computed by an accurate numerical method for two different uniform grids. POD of results of these two grids help us understand the receptivity aspects of the flow field, which give rise to the computed bifurcation sequences by understanding the similarity and differences of these two sets of computations. We show that POD modes help one understand the primary and secondary instabilities noted during the bifurcation sequences.

Key Words: Lid driven cavity, POD, POD modes analysis, DNS, multiple Hopf bifurcation, polygonal core vortex

AMS Subject Classifications: 65M12, 65M15, 65M6,76D05, 76F20, 76F65

1 Introduction

The 2D flow in a square LDC (of side L) is a canonical problem to study flow dynamics numerically for incompressible Navier-Stokes equation due to its unambiguous boundary conditions and very simple geometry. The flow is essentially shear-driven, with the lid given a constant-speed translation (U), giving rise to corner singularities on the top wall, as depicted in the top frame of Fig. 1. Such singularity gives rise to Gibbs' phenomenon [1, 5], which is milder for low order methods [16, 29]. Low order highly diffusive methods [6,16] are incapable of computing unsteady flows at high Reynolds number

*Corresponding author. *Email addresses:* `llestandi@u-bordeaux.fr` (L. Lestandi)

 ©201x Global-Science Press

($Re=UL/\nu$, where ν is the kinematic viscosity). In Ghia *et al.* [16], results for a wide range of Re up to 10000 are presented as steady flow. However, numerical results obtained by high accuracy combined compact difference scheme indicate creation of a transient polygonal vortex at the core, with permanent gyrating satellite vortices around it [38,42], for the same Re. It is well known that compact schemes for spatial discretization behave properly as compared to other methods, and Gibbs' phenomenon [35] is not experienced for the singular LDC problem due to numerical smoothing of the derivatives near the Nyquist limit [31,39].

Steady solutions have been reported [14,16] for Re far exceeding the values reported in the literature for the first Hopf Bifurcation (Re_{cr1}). Unsteady flows have been obtained as a solution of bifurcation problem [26,43], by studying linear temporal instability of the steady solution obtained numerically. Simulations of full time-dependent Navier-Stokes equation [25,38] reveal that the flow loses stability via Hopf bifurcation, as Re increases. Critical Re and frequencies obtained from DNS and eigenvalue analysis do not match. Such differences are also noted for different DNS results. However, DNS approach is preferable, due to its superiority of spatio-temporal multi-modal analysis over normal mode analysis of eigenvalue approach. In the latter, one postulates explicitly that all points in the domain have identical variation with respect to time. This is strictly incorrect, as one is dealing with space-time dependent growth of disturbances during the onset of unsteadiness.

It is shown [25,41,42] that Re_{cr1} depends upon accuracy of the method and how the flow is established in DNS. Impulsive start of the flow triggers all frequencies at the onset and hence preferred [38,42]. Obtaining final limit cycle at one Re from the limit cycle solution from another Re [25] is inappropriate [22]. First Hopf bifurcation obtained by DNS is dependent upon source of numerical error, mainly on the aliasing error for flow inside LDC [42]. This also depends upon the discretization, which in turn determines the creation of wall vorticity. A finer grid will create larger wall vorticity, but will have lesser truncation error. For the same numerical method, using same time step, a finer grid will have lesser aliasing and truncation errors, and hence numerical Re_{cr1} will be higher for finer grid. However, this can also be studied with the help of explicit excitation to show the near universality of Re_{cr1}.

Linear instability of equilibrium flow and DNS have been used to evaluate the onset of unsteadiness, i.e., obtaining Re_{cr1} for LDC. These methods yield values of Re_{cr1} differently. For example, $Re_{cr1}=8018$ in Ref. [2] and 8031.93 in Ref. [28] have been reported. Cazemier *et al.* [8] reported Re_{cr1} at 7972 using a finite volume method. In Bruneau and Saad [6], the critical Re is suggested to be in the range of $8000 \leq Re_{cr1} \leq 8050$, obtained using a third order upwind finite difference scheme. The authors do not provide any bifurcation diagram to substantiate this observation. Sengupta *et al.* [41] have described multiple Hopf bifurcations, showing the first one at 7933 and the second at 8187, using uniform (257×257) grid, with these values obtained from the FFT of vorticity time series. Osada and Iwatsu [25] have identified this value at $7987 \pm 2\%$, obtained using compact scheme on non-uniform (128×128) and (257×257) grids. However, the authors do not

produce any evidence for grid independent data. Shen [44] reported Re_{cr1} in the range of 10000 to 10500 obtained using partial regularization of top-lid boundary conditions. Poliashenko and Aidun [26] on the other hand reported a value of $Re_{cr1} = 7763 \pm 2\%$ using a commercial FEM package. Using the present method [41] with a (257×257) grid, a value of $Re_{cr1} \approx 8665$ has been reported by Lestandi *et al.* [22], for the case of no explicit excitation applied. A major difference is that computations for all the cases presented here have been performed following an impulsive start.

The point located at $(x = 0.95, y = 0.95)$ is used here for sampling the data, which is very close to the singularity at the top right corner, and will log larger value of disturbances [22, 41, 42]. A recent study [22] highlights aspects of computing flow inside LDC based on study of time series at this point. Although, it is a valid way of studying the flow dynamics in LDC, it is desirable to use a global flow analysis tool like POD, which provides spatio-temporal information for the full domain. POD was introduced by Kosambi [21] to project a stochastic field on to a finite set of deterministic basis functions in the most optimum way possible. POD is also known as Karhenen-Loève decomposition, principal component analysis, etc. This method requires solving an optimization problem of variational calculus, whose discrete version is a linear algebraic eigenvalue problem that decomposes a stochastic field into a set of eigenfunctions. Once the eigenvalues and eigenfunctions are obtained, one can obtain the time dependent amplitude functions, which apportion disturbance field into different eigenmodes.

There are many versions of POD reported in the literature. The eigenvalues may be obtained through a variety of methods including direct [9] and iterative solvers such as a Lanczos procedure [34], with or without re-orthogonalization, as given by Cullum and Willoughby [10]. One of the advantage is that this method can be used locally, in a small zone of investigation, with the number of eigenvalues depending on the total number of points in that small part of zone investigated.

However, even such a local analysis can be very resource-intensive. Thus, one uses instead the alternative method of snapshots proposed by Sirovich [45]. In this case, the number of eigenvalues depends upon number of snapshots used for the investigation. The popularity of this method rests with the use of limited number of snapshots, thereby making the method very efficient. Like the classical method, the problem of optimization in projection used for method of snapshots also involves obtaining two-point correlation functions. POD with method of snapshots have been used in fluid mechanics originally with the idea of applying it to turbulent flows [19], with the number of modes decided upon capturing a very high percentage of kinetic energy. This has been followed in many early attempts [8, 11, 23, 24] to build POD based reduced order models (ROMs), where primitive variable formulations have been used to convert the governing PDEs into a set of coupled ODEs for the amplitude functions. In doing so, the pressure gradient terms are usually omitted. This is avoided in an alternative approach, where stream function -vorticity formulation is used for the governing 2D Navier-Stokes equation and the projection onto a deterministic basis is sought in capturing maximum enstrophy [33, 34, 36, 37, 40]. This does not entail omission of pressure information, as vor-

ticity transport equation is not directly coupled to pressure. Also, in this approach of using DNS, one directly obtains the amplitude functions up to the desired numbers with enhanced accuracy. This helped classifying POD modes based on the properties of the amplitude functions [40, 41], in terms of regular and anomalous modes. In Ref. [40], the POD modes have been related with the instability modes for the first time, readying the field of flow instability study by POD analysis. The regular POD modes occur in pairs for the amplitude functions, separated by quarter cycle and the resultant instability modes obey the Stuart-Landau equation [30]. The anomalous modes, on the other hand do not obey Stuart-Landau equation. Also, Stuart-Landau equation is of use for fluid dynamic system with a single dominant mode. Hence, an augmented eigenfunction approach due to Eckhaus [13] has been used in instability studies of fluid dynamic system with multiple modes. The resultant governing equations for instability modes have been termed as Stuart-Landau-Eckhaus equations. This approach of obtaining POD eigenfunctions and amplitude functions in describing nonlinear instability of fluid flow has been described in Ref. [32] and is routinely used for incompressible flows [36, 37]. In Ref. [24], the authors devised a new POD mode which was obtained through a Galerkin projection on Reynolds-averaged Navier-Strokes (RANS) equation, and called it a shift mode.

Here, enstrophy is preferred over those in Refs. [19, 24, 27, 45], where kinetic energy is used for POD analysis. In vortex dominated inhomogeneous flows, rotational energy is a better descriptor of POD over translational kinetic energy, as highlighted in Refs. [30, 33, 40]. Authors in Ref. [41], used enstrophy based POD approach to study both external and internal flows to show universality of POD modes in terms of amplitude functions.

The paper is formatted in the following manner. In the next section, we provide a very brief recap of the governing equation and numerical methods used. In section 3, with the help of computed Navier-Stokes Equation (NSE) solution, we characterize the flow field by bifurcation analysis. POD as a tool has been used in section 4, to relate vorticity dynamics in the LDC flow field about the sensitivity to grid resolution by solving NSE using two grids in describing primary and secondary instabilities. We close the paper by providing the conclusions arising out of this research.

2 Governing Equations and Numerical Methods

Direct simulation of the 2D time-dependent flow is carried out by solving NSE in stream function-vorticity formulation given by,

$$\nabla^2\psi = -\omega \tag{2.1}$$

$$\frac{\partial\omega}{\partial t} + (\vec{V}\cdot\nabla)\omega = \frac{1}{Re}\nabla^2\omega \tag{2.2}$$

where ω is the only non-zero, out-of-plane component of vorticity for the 2D problem considered here. The velocity is related to the stream function as $\vec{V} = \nabla \times \vec{\Psi}$, where $\vec{\Psi} = [0,0,\psi]^T$. The governing equations are non-dimensionalized with L as the length scale and the constant lid velocity, (U), as the velocity scale, so that the Reynolds number is $Re = \frac{UL}{\nu}$. Consequently, computational domain is the unit square, while the time evolution is continued up to desired flow development. Present formulation is appropriate for 2D incompressible flows due to its inherent satisfaction of solenoidality condition for velocity and vorticity. This allows one to circumvent the pressure-velocity coupling problem, which is otherwise an important issue in primitive variable formulation. Identical numerical methods have been used previously of the flow for $Re = 10000$ in Refs. [38, 42] and is not repeated here.

Equations (2.1) and (2.2) are solved using uniform grid of a Cartesian frame with the origin at the bottom left corner of the LDC. A schematic of the computational domain is shown in Fig. 1(a). The flow field is subjected to the following boundary conditions. On all the four walls of LDC, ψ=constant is prescribed, which satisfies no-slip condition and helps evaluating the wall vorticity as $\omega_b = -\frac{\partial^2 \psi}{\partial n^2}$, with n as the wall-normal coordinate chosen for the four segments of the LDC. This is calculated using Taylor series expansion at the walls with appropriate velocity conditions on the boundary segments, as given for the top wall by,

$$\psi(x, L-dy) = \psi(x,L) - dy\frac{\partial \psi}{\partial y} + \frac{dy^2}{2}\frac{\partial^2 \psi}{\partial y^2} + \mathcal{O}(dy^3)$$

Since, $U = \frac{\partial \psi}{\partial y}$ at the top wall, the wall vorticity can be written in truncated series form as

$$\omega_b(x) = \frac{2}{dy^2}\left[\psi(x,L) - \psi(x,L-dy) - dy\right] \tag{2.3}$$

In Eq. (2.3) on the right hand side, the last term is due to the top lid continuously moving at the constant speed, U, which is taken equal to one in non-dimensional form. One can similarly obtain the expression for the wall vorticity at other wall-segments, where we use $\frac{\partial \psi}{\partial n} = 0$ identically.

To solve the discretized form of Eq. (2.1), Bi-CGSTAB method has been used here, which is a fast and convergent elliptic PDE solver [47]. The convection and diffusion terms of Eq. (2.2) are discretized using the NCCD method [38, 42] to obtain both first and second derivatives, simultaneously. For time advancing Eq. (2.2), four-stage, fourth-order Runge-Kutta (RK4) method is used, that is tuned to preserve physical dispersion relation. The NCCD scheme has been analyzed for resolution and effectiveness in discretizing diffusion terms [38, 42]. It is noted that the NCCD scheme is particularly efficient, providing high resolution and effective diffusion discretization, as also has been shown with the help of model convection-diffusion equation [46]. Additionally, it has built-in ability to control aliasing error. The only drawback of NCCD scheme is that it can be used only

with uniform structured grids. All computations are performed with non-dimensional time-step of $\Delta t = 10^{-3}$. Additional details of the method for this problem is in Ref. [22], which explained the reason for the location where time-series for vorticity is stored for analysis. This is shown in Fig. 1(a) as P, with the coordinate $(x = 0.95, y = 0.95)$.

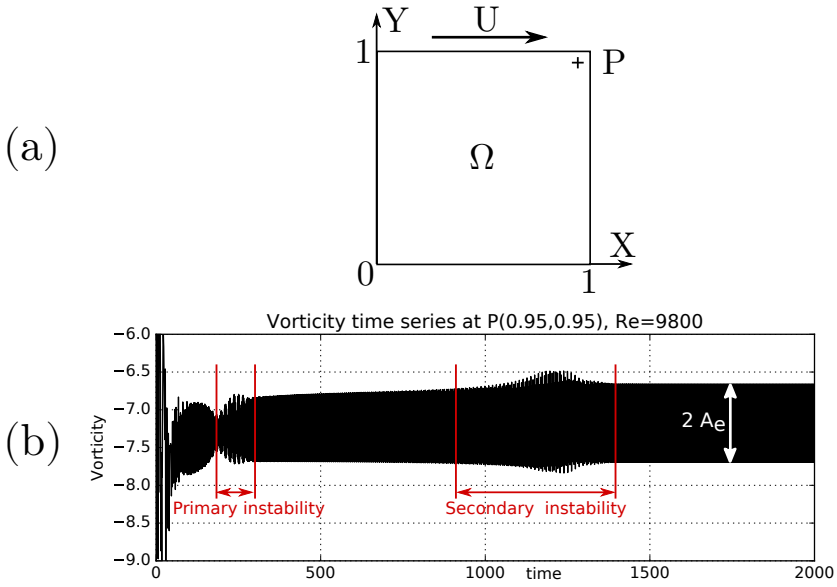


Fig. 1: (a) Schematic view of the LDC problem and (b) time series of the vorticity taken at point P (0.95, 0.95) obtained using (257×257) grid points for $Re = 9800$.

3 Flow dynamics in LDC: Bifurcation sequences

To understand how a steady flow inside the LDC becomes unsteady with increasing Re above critical value, we record the time variation of the vorticity in the domain at point P, as shown in Fig. 1(b). This is a typical time series, when we use the uniform grid with (257×257) points, for $Re = 9800$ with the flow unexcited.

The used combined compact difference (CCD) scheme has near-spectral accuracy and it has been explained in [38, 42], the onset of unsteadiness is due to aliasing error predominant near the top right corner of the LDC, while the truncation, round-off and dispersion errors are extremely negligible. To avoid the issue of lower numerical excitation in the present work [38, 41, 42], a pulsating vortex is placed having the form at $r_0 = (0.015625, 0.984375)$ whose spread is defined by $\alpha = 0.0221$ as given in the following,

$$\omega_s = A_0(1+\cos(\pi(r-r_0)/0.0221))\sin(2\pi f_0 t) \text{ for } (\mathrm{r}-\mathrm{r}_0) \leq 0.0221$$

where in the presented results here we have taken $f_0 = 0.41$ for different amplitude cases.

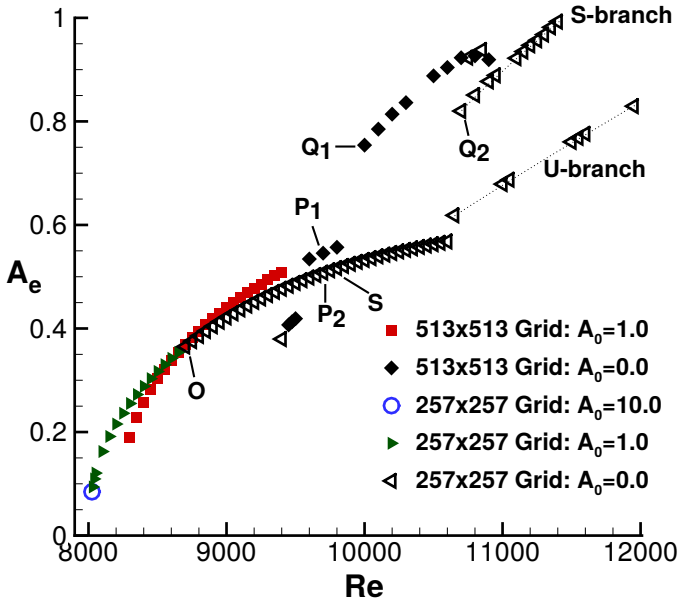


Fig. 2: Variation of the equilibrium amplitude (A_e) with Reynolds number (Re) for the two grids, with (257×257) and (513×513) points. Note the points (P_1, P_2) and (Q_1, Q_2) have similar dynamics, as shown later. Additional points O and S represent the onset of unsteadiness ($Re=8670$) and secondary instability ($Re=9800$) of the flow field computed using (257×257) grid points.

From Fig. 1(b) one notices a primary instability as marked in the frame, following subsidence of the initial transient. After this instability, one notices a regular time variation of vorticity (almost like a limit cycle, with slowly increasing amplitude). However, after some time, one notices rapidly growing envelope amplitude, caused by a secondary instability, following which one notes a final stable limit cycle, settling down to an equilibrium peak to peak amplitude indicated as $2A_e$.

Figure 2 shows the variation of the equilibrium amplitude A_e with Re, for simulations performed using two grids, with (257×257) and (513×513) points. The triangles correspond to the equilibrium amplitude obtained using (257×257) grid points, except the highest amplitude case of $A_0=10$ for this grid with open circle, for the lowest supercritical case. It shows the onset of unsteadiness for this grid to occur between $Re=8660$ and 8670 for the case of $A_0=0$, with the point marked as 'O' in the figure. The points shown by filled rhombus and square are obtained using the (513×513)-grid points. For the refined grid, onset of unsteadiness occurs for Re slightly lower than 9450, for the case of $A_0=0$. The (257×257) grid results also show a dip in A_e around $Re=9400$, which is identified as the second bifurcation point [22] for this grid. In this reference, different bifurcation sequences are identified by plotting A_e^2 versus Re and the segments are

identified by straight lines with different slope for the unexcited cases. This stems from the literature which identifies bifurcation with disturbance amplitude evolution following Stuart-Landau equation [30] to occur quadratically with respect to Reynolds number. However, this equation is valid only if there is a single dominant mode for the disturbance field. It is understood that for circular cylinder, presence of many POD modes and instability modes necessitates adoption of Stuart-Landau-Eckhaus (SLE) equation to account for multi-modal interactions [40], which show quadratic variation of disturbance with Re merely as an assumption. In Fig. 2, for the coarser grid we have identified 'S' as the point ($Re=9800$) displaying secondary instability, as already shown in Fig. 1(b).

For the finer grid, we note that the primary Hopf-bifurcation between $Re=8660$ and 8670 is bypassed. For this grid, the second and third bifurcations occur for $Re=9600$ and 10000, respectively. Following the second bifurcation, we notice three data points with the middle one identified as P_1 in Fig.2, which show similar variation as for the (257×257) grid over an extended range of Re. Later on, we compare a representative point at P_2 with P_1. A similar qualitative variation between the two grids are noted which originate in a sequence starting from Q_1 and Q_2, which are also compared later.

Few of the distinctive features of Fig. 2 are the following: (a) The used methods for space-time discretization are so accurate that the onset of unsteadiness in the flow field is delayed, with finer grid. Even for (257×257)-grid, the onset is delayed up to $Re=$ 8670. This has been explained here by performing the computations for lower Re, with an excitation applied at a single point by a pulsating vortex, with frequency of excitation of 0.41, which is distinctly different from the natural Strouhal number on 0.43. More details about the excitation is given in [22]. Following this process of excitation, one notices from Fig. 2 that the critical Re for this case can be brought down to between 8020 and 8025. (b) For the finer grid of (513×513) points, the first critical Reynolds number is noted between 9400 and 9425, for the case of no excitation. With excitation this can be brought down to as low as $Re=8250$ (as shown in the figure). (c) For Re above 10400 with the (257×257)-grid, one notices two branches of solution, as shown in the figure. The lower branch (marked as U-branch) is essentially unstable and the upper branch is the stable branch, named as the S-branch. Upon application of slightest perturbations, the solution on the U-branch jumps to the S-branch.

4 Proper Orthogonal Decomposition

4.1 Method overview

Here, we use the enstrophy-based POD, which is preferred over those in Refs. [19,24,45], where kinetic energy-based POD analysis have been performed. In vortex dominated flows, which are neither homogeneous nor periodic, rotationality is more important and enstrophy is a better descriptor of POD over translational kinetic energy, as has been used in Refs. [32,36,37,40,41]. Authors in Ref. [41], used enstrophy based POD approach to study both external and internal flows to show universality of POD modes in terms of

amplitude functions. In Ref. [24], the authors devised a reduced order model (ROM) that relied on POD mode and Galerkin projection of RANS solution. Thus, POD analysis is noted to be useful in studying internal and external flows of different kinds.

POD technique introduced among others by Kosambi [21] for a random field $v_i(\vec{x},t)$, where it is projected onto a set of deterministic vectors $\varphi_i(\vec{x})$, so that $\langle|(v_i,\varphi_i)|^2\rangle/||\varphi_i||_{L_2}$ is maximum. The outer angular brackets signify time-averaging and inner brackets signify an inner product. The computation of $\varphi_i(\vec{x})$ can be posed as an optimization problem in variational calculus,

$$\int_{\Omega_x} R_{ij}(\vec{x},\vec{x}')\,\varphi_j(\vec{x}')\,d\vec{x}' = \lambda\,\varphi_i(\vec{x}) \tag{4.1}$$

The kernel of the above is the two-point correlation function, $R_{ij} = \langle v_i(\vec{x},t)v_j(\vec{x}',t)\rangle$ of the random field. It is noted [32] that *classical Hilbert-Schmidt theory applies to flows with finite energy, and, therefore, denumerable infinite orthogonal POD modes can be computed.* Furthermore, Hilbert-Schmidt theory is applicable for flow instabilities, as the disturbance field derives its energy from the equilibrium flow. Disturbance vorticity field is thus, represented in POD formalism as

$$\omega'(\vec{x},t) = \sum_{m=1}^{\infty} a_m(t)\,\varphi_m(\vec{x}) \tag{4.2}$$

where $a_m(t)$ represents the amplitude function, which describes the spatio-temporal variation of the modal amplitude and $\varphi_m(\vec{x})$ is the corresponding spatial eigenfunction. It should be noted that the eigenfunctions are orthogonal [9], additionally they are taken of unit norm for practical reasons. Thus, these form an orthonormal basis [3] on which ω' can be projected, as in Eq. (4.2). Then, one can compute the corresponding amplitude functions $a_m(t)$ easily through spatial inner product

$$\forall m \in \mathbb{N}^*, a_m(t) = (\omega',\varphi_m)_{L_2(\Omega_x)} = \int_{\Omega_x} \omega'(\vec{x},t)\varphi_m(\vec{x})d\vec{x},$$

which emphasize the spatio-temporal nature of the POD. Equation (4.1) is an eigenvalue problem in the integral form, which becomes intractable even for moderate grid resolution. To overcome this difficulty, Sirovich [45] introduced the method of snapshots, which has an advantage of dealing with smaller data sets in multiple dimensions. Instead of solving Eq. (4.1), it is chosen to solve the equivalent problem on q_m which yields the same decomposition.

$$\int_{\Omega_t} C(t,t')q_m(t')\,dt' = \lambda_m q_m(t) \tag{4.3}$$

where Ω_t is the time interval and the autocorrelation function is defined as

$$C(t,t') = \frac{1}{T}\int_{\Omega_x} \omega'(\vec{x},t)\omega'(\vec{x},t')d\vec{x}.$$

Once Eq. (2.2) has been solved, we can recover the spatial POD modes $(\varphi_m)_m$ due to the following projection

$$\varphi_m(\vec{x}) = \int_{\Omega_t} q_m(t)\omega'(\vec{x},t)dt. \tag{4.4}$$

Finally, (φ_m) are normalized and the norm is passed to $a_m = \sqrt{\lambda_m}q_m$. This method produces the same basis that one would obtain through classical POD. The strength of the snapshot POD lies in the small size of the snapshots of DNS data, where N_t the number of snapshots (time frames) that is lot smaller than the number of grid points N_X. Discretization of the above operators is performed by trapezoidal integration rule for time (as well as space) with weights at time point i noted $m_i = dt/T$, half of that for $i = 1$ and $i = N_t$. The discrete version of the POD decomposition reduces to a simple matrix eigenvalue problem $[\bar{C}]\{\boldsymbol{q}\} = \lambda\{\boldsymbol{q}\}$, where $[\bar{C}]$ is given by

$$\bar{C}_{ij} = \sqrt{m_i m_j}\int_{\Omega_x} \omega'(\vec{x},t_i)\,\omega'(\vec{x},t_j)\,d\vec{x}. \tag{4.5}$$

The eigenvalues λ and eigenvectors $\{\mathbf{q}\}$ of $[\bar{C}]$ are computed using LAPACK eigenvalue problem solver for symmetric matrices (DSYEV). It should be noted that the calculations account for the differences between discrete L_2 inner product and vector scalar product. Consequently an extra step is required that reads $q_m = \boldsymbol{m}^{-1}\{\mathbf{q}\}$ where $m_i^{-1} = 1/m_i$.

In this paper the maximum number of snapshots is $N_t = 1000$ while the number of grid points is $N_X = 66049$ or 263169 (according to grid size), thus only the method of snapshots is used. Moreover, the spatial POD modes will be referred to as eigenfunctions for historical reasons while (a_m) will be called amplitude function or time POD modes.

4.2 DNS Data Analysis: Limit Cycle

Here we use POD analysis to characterize flow fields obtained by the two grids. In Fig. 3, the eigenfunctions obtained following the method of snapshots for the POD analysis is shown for the points, P_1 and P_2, shown in Fig. 2 for $Re = 9700$. We display only the first twelve modes obtained for the two grids in Figs. 3(a) and 3(b). It is noted that despite the differences in Fig. 2 for the equilibrium amplitude and the associated maximum vorticity values in the domain, the first eight eigenfunctions have remarkable similarities, indicating the qualitative similarities of the associated flow fields obtained using two grids with significantly different points. The eigenfunction plots of Fig. 3 also show a definitive pattern, with the first and second modes are regular modes [41], defined for classification of

POD modes. In this case, one notices three pairs of similar vortical structures with opposite signs. In the same way, the third and fourth modes are composed of six such pairs; fifth and sixth modes similarly have nine pairs of structures. This multiplicity of vortical structures are extended to higher mode pairs also. However, their contributions are negligibly small in terms of enstrophy content, as the first eight modes in Fig. 3, account for nearly all of the enstrophy contents for both the grids.

Such similarities are furthermore emphasized in Fig. 4, showing the cumulative enstrophy for the pairing of points shown in Fig. 2. For example, in discussing the flow dynamics for points P_1 and P_2, it has been mentioned that the flows would be similar.

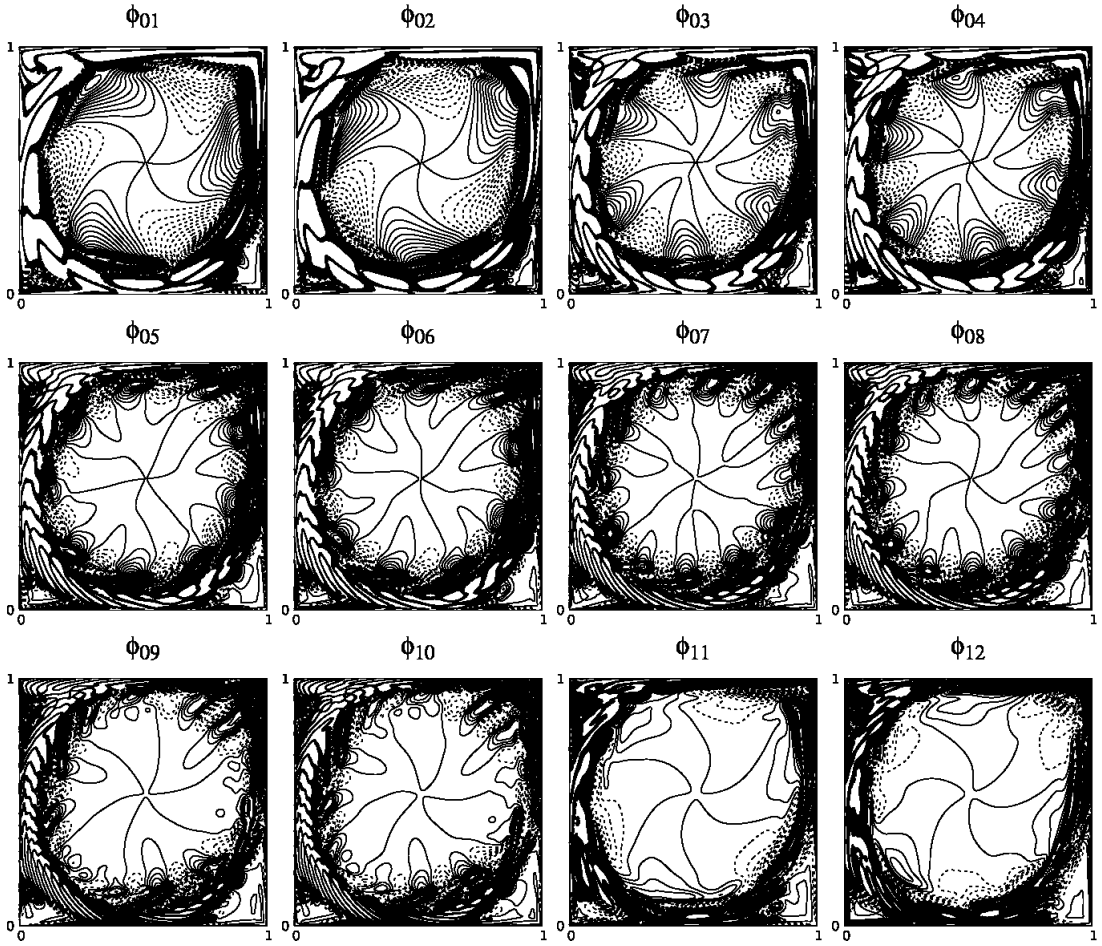


Fig. 3(a): Eigenfunctions of POD modes for $Re = 9700$ with (257×257) grid. These are for the two points (P_1, P_2) in Fig. 2. $(\varphi_m)_m$ isolines are plotted in the $[-0.5, 0.5]$ range with 0.01 spacing. Solid lines are positive values, while dashed lines are negative value contour. (Cont.)

This is clearly brought out in the eigenfunction plots of Fig. 3 and the cumulative enstrophy shown in the top frames of Fig. 4. Similarities for the points Q_1 and Q_2 have been suggested, while discussing the bifurcation diagram (Fig. 2) and the cumulative enstrophy plot for this case shown in the bottom frame of Fig. 4, strongly supports this. We also note that keeping the Reynolds number same with the two grids alone, does not ensure similarity of the flow, as noted from the cumulative enstrophy plot for $Re = 10000$ in the middle frame of Fig. 4.

The POD amplitude functions, their representative DFT plots are shown in Figs. 5(a) and 5(b) for $Re = 9700$ case, obtained using the two grids. These are shown pairwise, when the two constituents differ by a phase shift of quarter cycle. In Fig. 5(a), amplitude functions are shown for P_1 obtained using (513×513) grid. The FFT of these time series is shown in the bottom frames for each pair. The top left frame indicates the fundamental

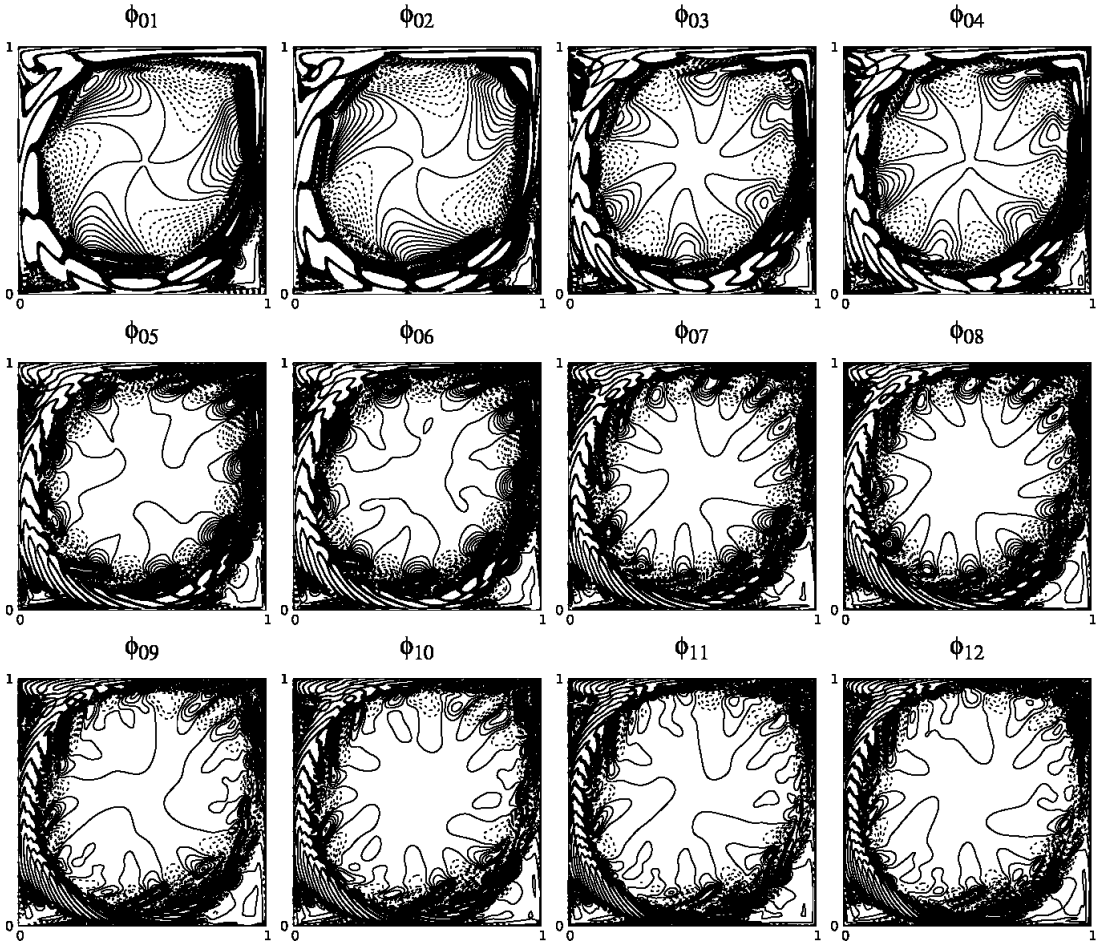


Fig. 3(b): same modes with (513×513) grid points.

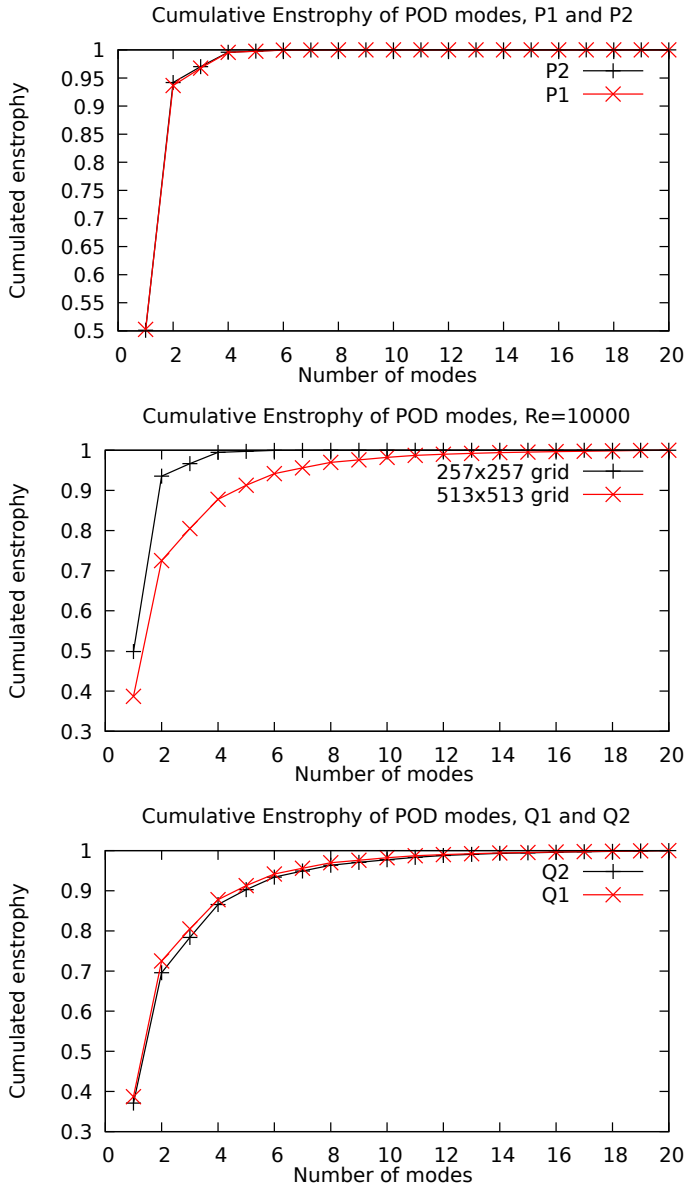


Fig. 4: Cumulative enstrophy plots for the two grids shown for the indicated Reynolds number, for the enstrophy based POD.

frequency for the first and second modes ($f_0 = 0.43$), while the second, third and fourth mode pairs are the super-harmonics of this fundamental frequency (at $2f_0, 3f_0, 4f_0$). These amplitude functions and the frequencies are identical for both grids, as can be seen for the amplitude functions and their DFT shown for the point P_2 obtained using (257×257)

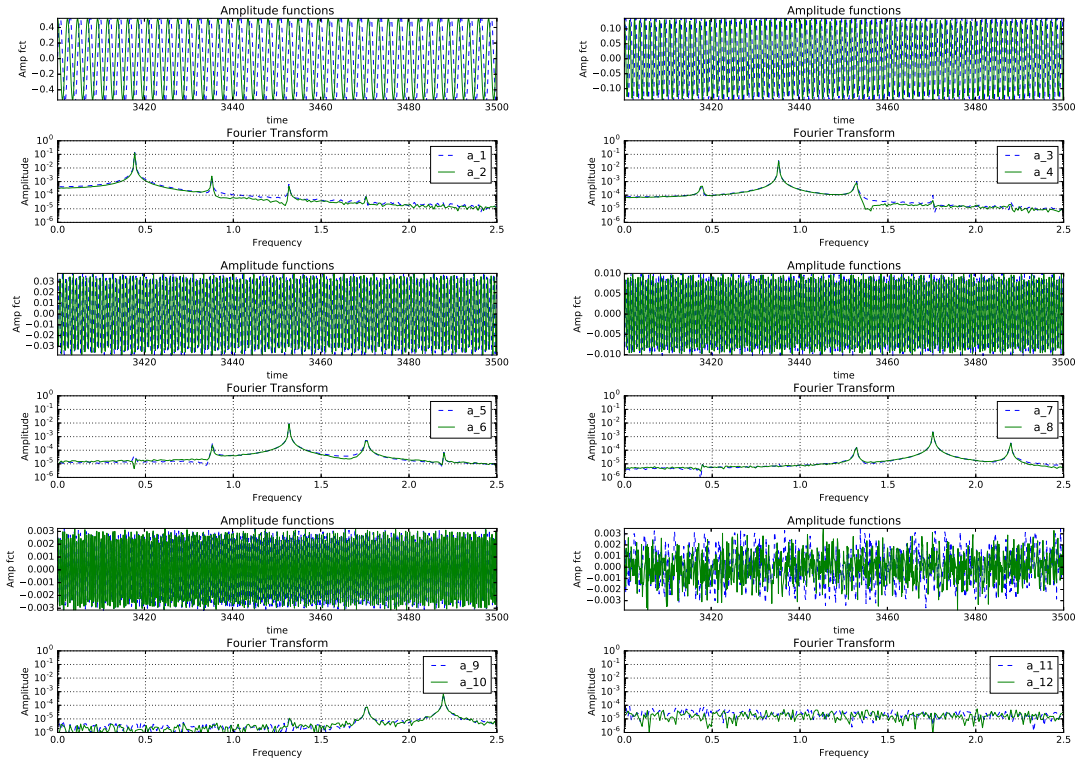


Fig. 5(a): Amplitude of POD modes and its DFT for $Re = 9700$ obtained for the (513×513) grid for the point P_1 in Fig. 2.

grid. Once again the comparison between Figs. 5(a) and 5(b) supports the view that the flow dynamics is similar for P_1 and P_2.

Next, we investigate the flow fields for the points Q_1 ($Re = 10000$) and Q_2 ($Re = 10700$) of Fig. 2, in Figs. 6(a) and 6(b), respectively for the two grids with the help of POD eigenfunctions. Previously, we have noted that the flow fields for these points obtained by the two grids will be similar, while discussing the bifurcation diagrams in Fig. 2. Now the plotted eigenfunctions for the first twelve modes in Figs. 6(a) and 6(b) are also seen to be similar. This, added with the cumulative enstrophy plots shown in the bottom frame of Fig. 4, strongly support the view that the flow fields are indeed similar. This also shows that the view provided by the bifurcation diagram is a better descriptor of similarity of flow field in the diagram, whenever A_e^2 plotted against Re show identical slopes. The eigenfunctions have also similarity with the eigenfunctions shown in Figs. 3(a) and 3(b) for the first two pairs, with respect to qualitative features. The higher modes are distinctly different in Fig. 6, due to the flow fields belonging to different branches of the diagrams, as compared to the cases shown in Figs. 3(a) and 3(b). Figures. 6(a) and 6(b) belong to branches in which the instability is higher due to multiple dominant frequen-

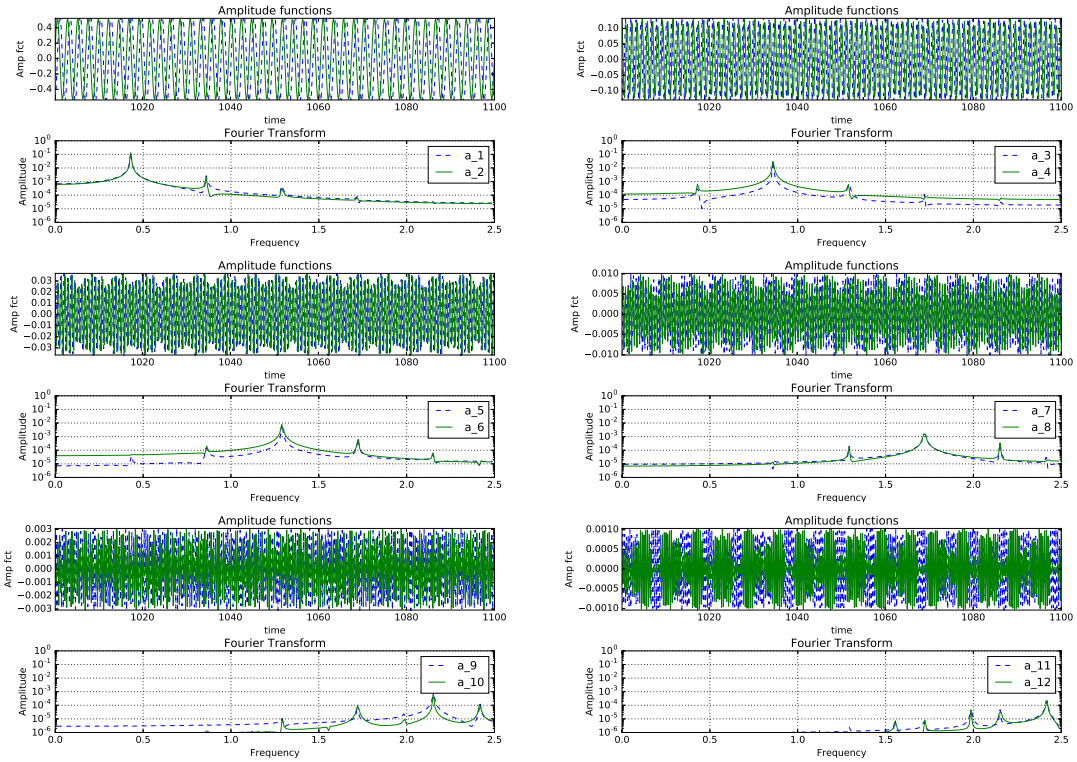


Fig. 5(b): Amplitude of POD modes and its DFT for $Re = 9700$ obtained for the (257×257) grid for the point P_2 in Fig. 2.

cies interacting [22]. That causes the enstrophy to be distributed over larger number of modes, i.e., one should be interested in the higher modes beyond the number eight, as was the case for the lower Reynolds number. Even the symmetry for the eigenfunctions noted for $Re = 9700$ is lost from fifth mode onwards since two or more physical modes are interacting with the primary POD mode.

The features of eigenfunctions for Q_1 and Q_2 are also reflected in the amplitude functions shown in Figs. 7(a) and 7(b). The first pair of amplitude functions displays identical peak for these two grid results, which is different from the fundamental frequency (f_0) noted in Figs. 5(a) and 5(b) for $Re = 9800$ case. The second pair of amplitude functions in Figs. 7(a) and 7(b) are not the super-harmonic of the fundamental seen for the first pair of amplitude function. Thus, this segment of bifurcation diagram for Figs. 7(a) and 7(b), is qualitatively different from the lower Reynolds number parts shown in Figs. 5(a) and 5(b). Between the two points Q_1 and Q_2, the third and fourth modes have some differences at the lower frequencies, otherwise other significant peaks are collocated. The fifth and sixth amplitude functions of POD modes again have the same value of frequency for the peak, as is noted for the first pair. All the other modes have qualitative similarity

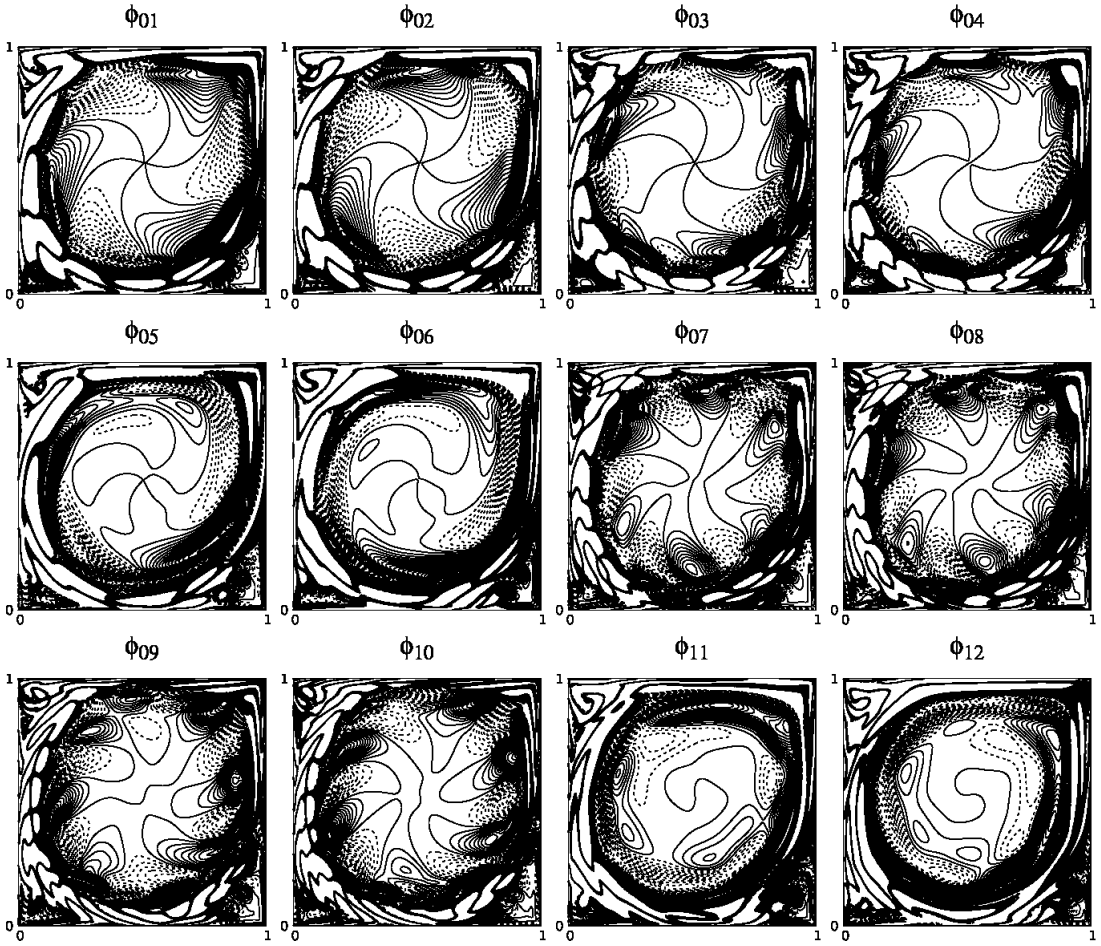


Fig. 6(a): Eigenfunctions of POD modes for $Re=10000$ obtained with (513×513) grid for the point Q_1 in Fig. 2.

between amplitude functions for points Q_1 and Q_2, and with the exception of eleventh and twelfth modes, all the modes appear as wave-packets, which have been called as the anomalous mode of second kind [30,41].

4.3 DNS Data Analysis: Primary and Secondary Instabilities

So far, we have reported POD analysis of flow fields after the time series reaches stable limit cycle for the sampling point $(x=0.95,y=0.95)$. We have previously reported DNS-based study of Hopf bifurcations using the (257×257) grid in Ref. [22], providing the numerical details of the methodology. Here we have studied the dynamics of the unsteady flow field using two different grids, with the intention of highlighting the mathematical

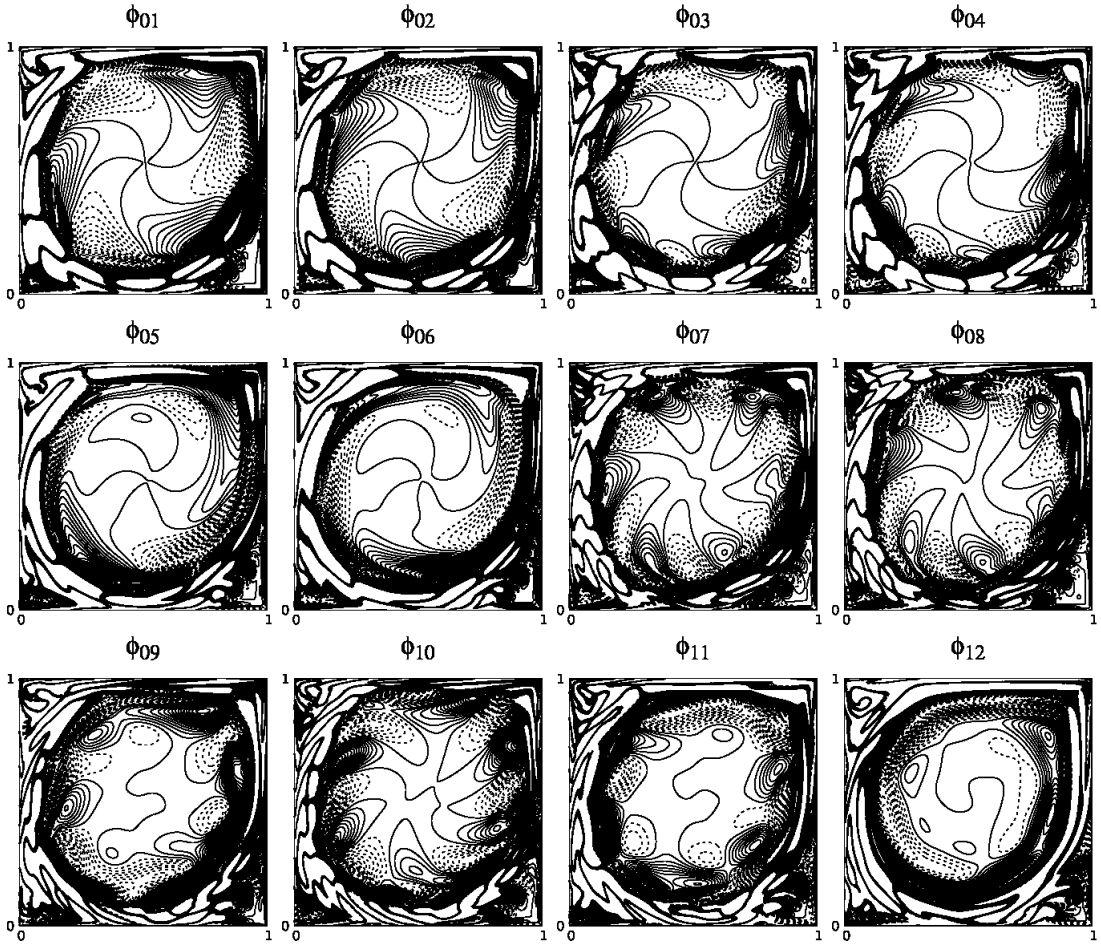


Fig. 6(b): Eigenfunctions of POD modes for $Re = 10700$ obtained with (257×257) grid for the point Q_2 in Fig. 2.

physics of this canonical problem with POD as the analysis tool. It is necessary also to characterize the flow during primary and secondary instabilities.

For this purpose, in Fig. 8(a) we show the POD eigenfunctions obtained without excitation during the primary instability stage for $Re = 8670$ obtained using the (257×257) grid, which is indicated as 'O' in Fig. 2. The first Hopf bifurcation obtained for this grid occurs between 8660 and 8670. Thus, this Re is a super-critical case that displays linear instability during $t = 900$ to 1100. The eigenfunctions show various polygonal core-vortex. For example, the eighth, fourteenth and seventeenth modes display triangular vortex at the core, as was shown for the flow field in Refs. [38, 42] for $Re = 10000$. Present simulation and its POD confirms the presence of triangular core vortex caused by the primary instability. This has also been advocated as the proof of accuracy of numerical schemes in

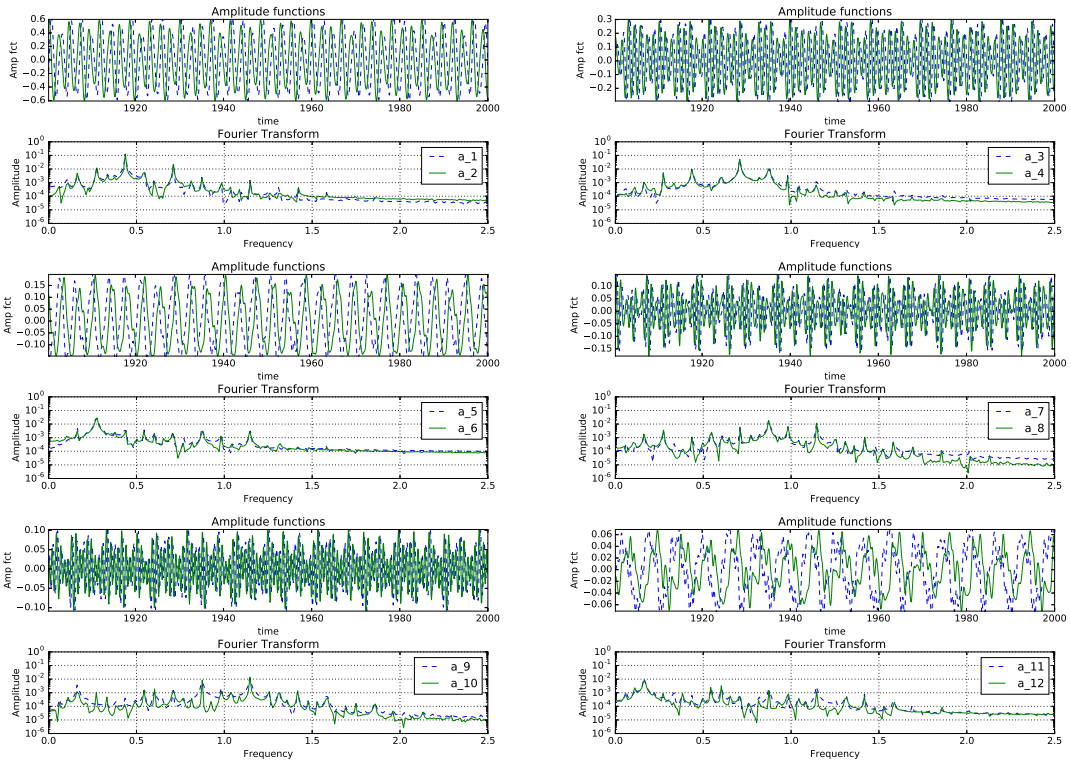


Fig. 7(a): Amplitude of POD modes and its DFT for $Re=10000$ obtained for the (513×513) grid for the point Q_1 in Fig. 2.

Ref. [22], in capturing the triangular vortex at the core, as has been experimentally shown in Refs. [4,7,20].

For the eigenfunctions shown in Fig. 8(a) for $Re=8670$, the corresponding amplitude functions are shown in Fig. 8(b). It is readily apparent that the first two modes form the regular pair [41], while the third mode is the anomalous mode of first kind; with fourth and fifth modes again form a regular pair, but modulated with higher frequency components. The sixth and seventh modes appear as wave-packets and hence, would be called the anomalous mode of second kind. The eighth and ninth modes are similar to fourth and fifth pair, i.e., regular modes which are highly modulated. The tenth mode is an anomalous mode of first kind, similar to the third mode. It has been explained in Refs. [30, 40] that the anomalous mode of first kind, gives rise to equivalent stress term, like the Reynolds stress and alters the mean flow. In this respect, the third and the tenth modes have opposite effects on the mean flow, as is evident from the signs of the amplitude at the terminal time. One can similarly classify the other modes into

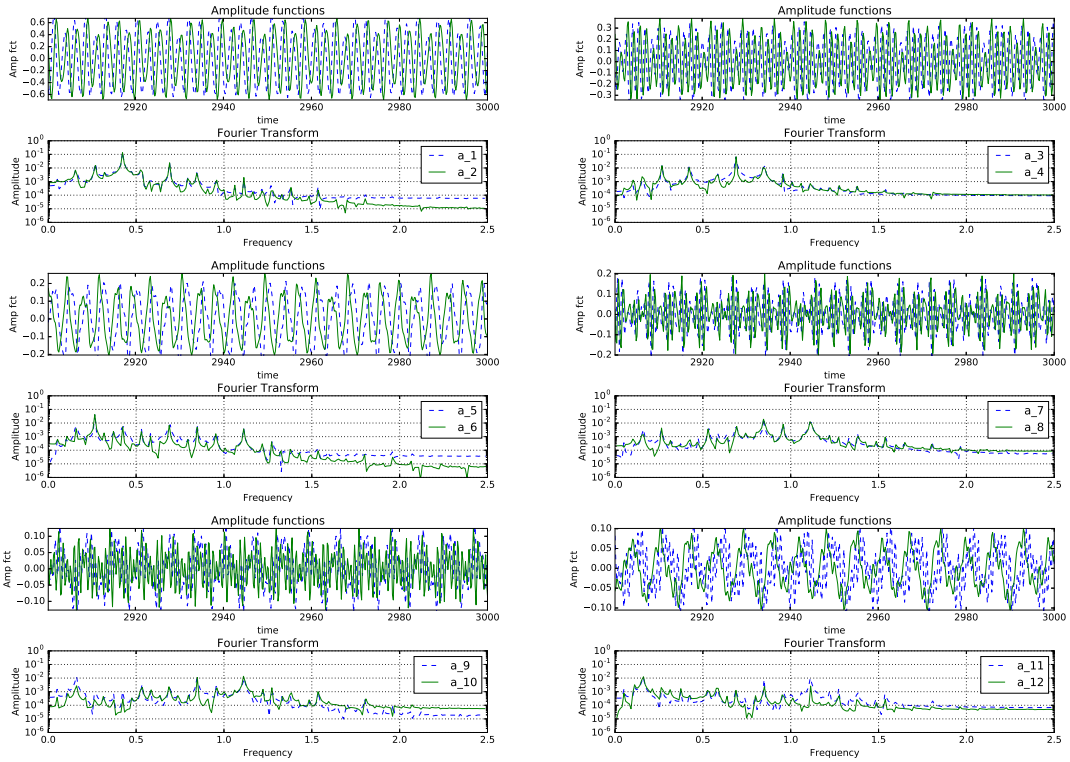


Fig. 7(b): Amplitude of POD modes and its DFT for Re=10700 obtained for the (257×257) grid for the point Q_2 in Fig. 2.

these categories described. However, the sixteenth and seventeenth modes appear as combination of the two types of anomalous modes described. It is worth remembering that the classification of POD modes like this is only feasible with DNS and not by RANS [24]. Authors in this latter reference introduced the so-called shift mode, which possibly happen, if we time average the anomalous mode of the first kind using URANS approach. One of the features of the present approach is that one does not require performing time averaged computations using closure models. Another feature of the anomalous mode of first kind is the appearance of the eigenfunctions in Fig. 8(a), where one does not notice orbital motion of the vortices around the core, which gives rise to the polygonal vortex in the core. In describing the dynamics of LDC flow in real time plane in Ref. [22], it was noted that for some cases, limit cycle behaviour is noted after the primary instability (as characterized in Figs. 8(a) and 8(b)), but with slowly varying amplitude of the envelope. Such variations continue till a secondary instability occurs, following which a stable limit cycle is noted whose envelope does not change further with time. In the following, we report results of POD analysis of one such secondary instability noted for $Re = 9800$, point 'S' in Fig. 2. The representative time series at $(x = 0.95, y = 0.95)$ has been already

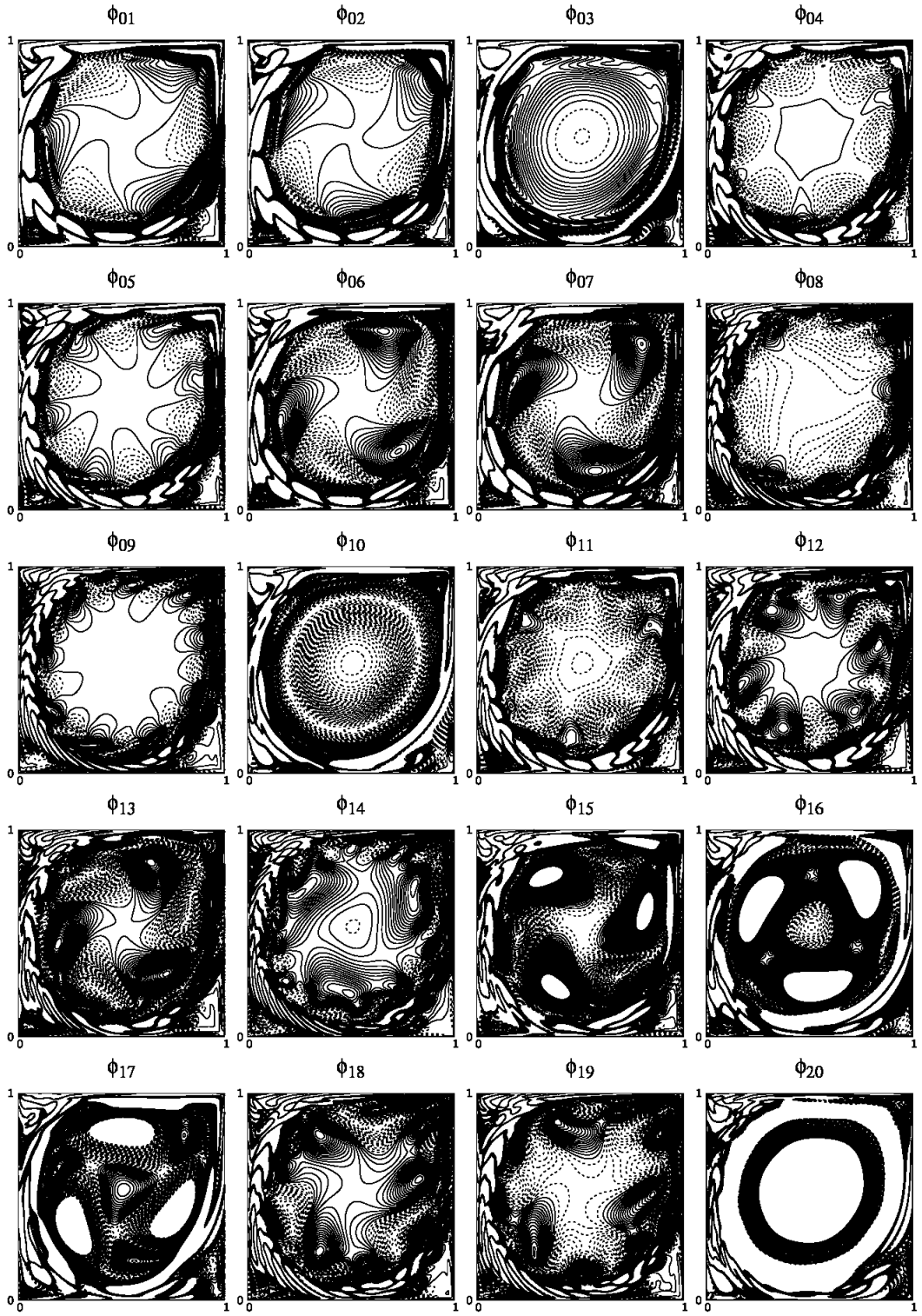


Fig. 8(a): Eigenfunctions of POD modes for $Re = 8670$ obtained with (257×257) grid for the point O in Fig. 2 during the linear instability stage.

shown in the bottom frame of Fig. 1, marking the primary and secondary instabilities. In Fig. 9(a) we show the eigenfunctions obtained by POD analysis performed on data before the beginning of secondary instability during $t=500$ to 600. At this stage, most of the enstrophy is contained in the first few modes and we show eight of these modes in Fig. 9(a). One notices the onset of creation of the orbital vortices in the first six modes. The seventh mode is without any structure and is similar to the eigenfunction for the anomalous modes in Fig. 8(a). It is the eighth mode that shows the appearance of a large triangular vortex in the core, with three pairs of orbital vortices surrounding the core.

In Fig. 9(b), we show the eigenfunctions for $Re=9800$ after the occurrence of the secondary instability during $t=1900$ to 2000. The first pair of eigenfunctions display three pairs of orbiting vortices, without any core vortex. This is typical of the behaviour of POD modes noted in the final limit cycle cases shown for higher Re. For the third and fourth modes, one notices six pairs of orbital vortices, without any core. The following two eigenmodes show nine pairs of orbital vortices and that is followed by the seventh and eighth modes, which show twelve pairs of orbital vortices.

The corresponding amplitudes and the DFT of various eigenmodes (as in Fig. 9), are shown in Fig. 10. In frames (a), the plotted amplitudes correspond to eigenfunctions

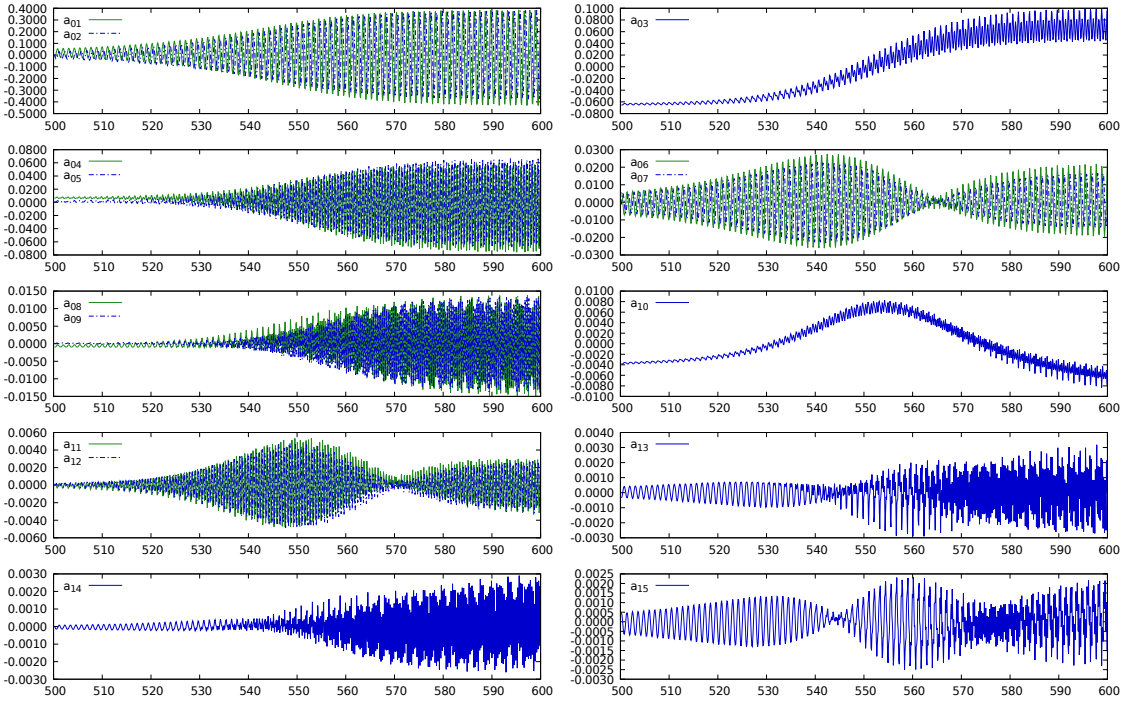


Fig. 8(b): Amplitude of POD modes and its FFT for $Re=10700$ obtained for the (257×257) grid for the point Q_2 in Fig. 2.

shown in Fig. 9(a), in pairwise fashion. One can clearly note that the FFT is dominated by

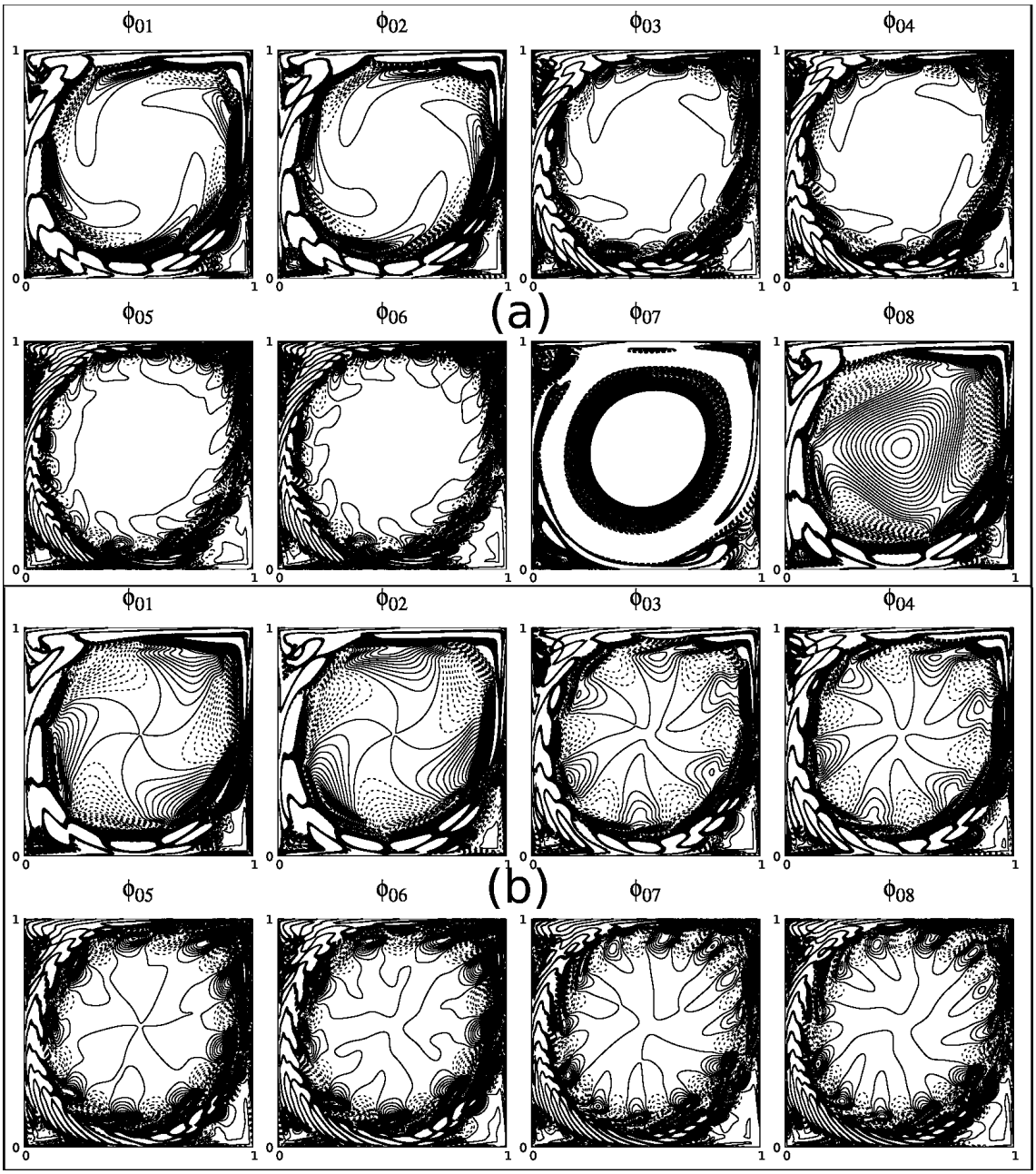


Fig. 9: Eigenfunctions of POD modes for $Re = 9800$ obtained with (257×257) grid during (a) $t = 500$ to 600 before and during (b) $t = 1900$ to 2000 after the secondary instability.

a single mode and amplitudes are time-shifted by quarter cycle. While there is a distinct secondary mode, but its amplitude is orders of magnitude smaller. The third and fourth modes' amplitude shows the peak which has a value that is twice of that noted for the first pair. However, this mode-pair also shows modulation in the time plane, which is due to the secondary peak shown in the FFT, which is the fundamental for the first and second modes' amplitude. In the same way, the fifth and sixth modes have the peak at thrice

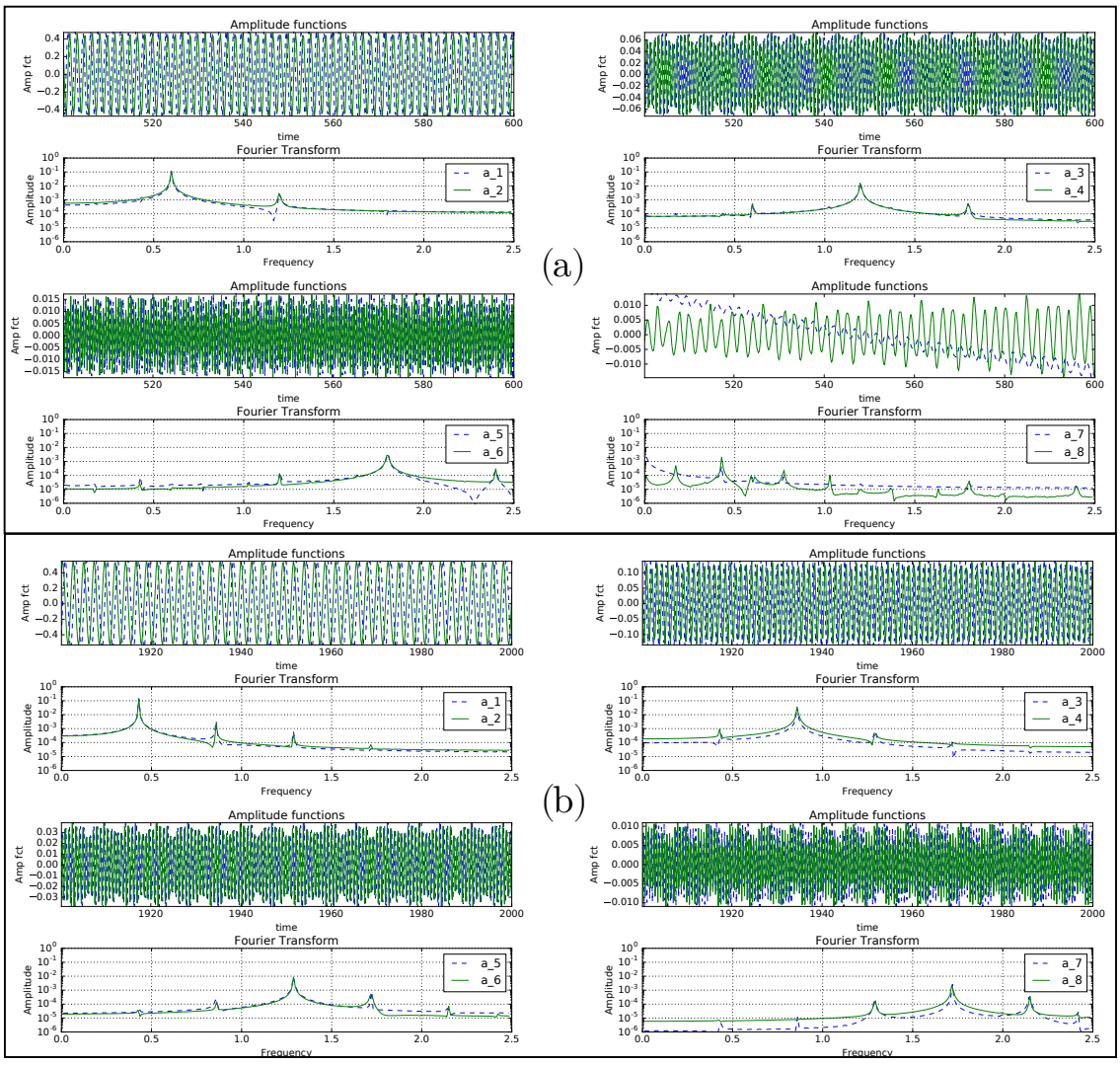


Fig. 10: Amplitude of POD modes and its DFT for $Re = 9800$ using (257×257) grid (a) before [$t = 500$ to 600] and (b) after [$t = 1900$ to 2000] the secondary instability, for the case of Fig. 9.

the value noted for the first pair. The seventh and eighth modes have no correlation, as noted in Fig. 10(a).

In Fig. 10(b), we note the amplitude functions corresponding to the eigenfunctions shown in Fig. 9(b), obtained during $t=1900$ and 2000, when one is in the final limit cycle stage. It is interesting to note that the action of the secondary instability is to shift the fundamental frequency for the first pair ($(f_0)_{before}=0.60$) to a lower value ($f_0=0.43$), as noted in the FFT plots. The second and third pair of amplitude functions have peaks at $2f_0$ and $3f_0$, respectively. The seventh and eighth modes are characterized by very high frequency fluctuations, and modulated at moderate frequencies, as a consequence one can categorize these as anomalous mode of second kind [41]. This phenomenon is explained by similar amplitudes of the leading peak ($4f_0$), with the next peak in amplitude ($5f_0$) that interact to create modulations. This pattern is visible for each final state, however, it is weaker for the finer grid in Fig. 3.

5 Conclusions

In the present research, we have used POD to characterize LDC flow for a range of Re for simulations performed using two grids (257×257) and (513×513) points. The numerical method is well established for similar exercise in Refs. [38,42], where very high accuracy combined compact scheme have been used. Although, the two grids produce different bifurcation sequences (in Fig. 2), the reason for this is explained in exciting the flow, as determined by the aliasing error (which reduces with grid refinement), while the wall vorticity increases with the refined mesh. As a consequence, the relative scaled amplitude of disturbance field is lower for the finer mesh, and that also explains why primary Hopf bifurcation is delayed for the refined grid. Furthermore, we show that despite difference in bifurcation sequences in the two grids, the qualitative similarity of flow fields are noted for points in the bifurcation diagram.

We note that the flow is better characterized by the bifurcation diagram (Fig. 2), rather than Re. The flow in the two grids will be similar when A_e^2 versus Re curves have identical slope, even if the Re are different. This is shown first by comparing the POD modes of the flow field for $Re=9700$ for the two grids, which is expected from similarity of Re and the slope of the bifurcation diagram at P_1 and P_2. The POD eigenmodes are shown in Fig. 3 and corresponding FFT amplitude functions are shown in Fig. 5 for P_1 and P_2. This is also supported by comparing two points Q_1 and Q_2 in Fig. 2, which correspond to $Re=10000$ using the (513×513) grid and $Re=10700$ for the (257×257) grid without excitation. The POD eigenfunctions and amplitudes together with FFT are shown in Figs. 6 and 7. These observations are strongly supported by the cumulative enstrophy plots in Fig. 4, for these four points, P_1, P_2, Q_1 and Q_2.

We also characterize the primary temporal instability without excitation (indicated by point O in Fig. 2) by POD analysis, showing eigenfunctions and amplitudes in Fig. 8, which shows clearly multi-periodic dynamics of the flow, with a single dominant funda-

mental frequency and its super-harmonics. Finally, we characterize the secondary instability indicated in Fig. 1 by showing POD eigenmodes and the corresponding amplitudes in Figs. 9 and 10, during $t = 500$ to 600 and then during $t = 1900$ and 2000. These time intervals correspond to before and after the secondary instability for $Re = 9800$, which has been identified in Fig. 1. We note that such secondary instability does not occur for all Reynolds number cases, but when it does occur, the effect is to change the fundamental frequency from a higher value (0.60) to a lower value (0.43). The eigenfunctions are also completely different, before and after the secondary instability.

This work reports the study of the LDC flow by DNS and resultant Hopf bifurcation patterns. The added understanding of this flow instability behaviour will allow us to build reduced order models relying on POD and the bifurcation diagram presented in Fig. 2. It will focus on ranges of parameters for different ROMs, as we have shown that the nature of the flow changes drastically through Hopf bifurcation process.

Acknowledgement

The authors acknowledge the support provided to the first author from the Raman-Charpak Fellowship by CEFIPRA which made his visit to HPCL, IIT Kanpur possible. This work reports partly the results obtained during the visit.

References

[1] Auteri F., Quartapelle L. and Vigevano L., Accurate ω-ψ spectra; solution of the singular driven cavity problem, *J. Comput. Phys.*, **180**, *597-615* (2002).
[2] Auteri F., Parolini N. and Quartapelle L., Numerical investigation on the stability of singular driven cavity flow, *J. Comput. Phys.*, **183**, *1-25* (2002).
[3] Azaiez, M., Ben Belgacem, F. Karhunen-Loèves truncation error for bivariate functions.*Computer Methods in Applied Mechanics and Engineering*, **290**, *57-72* (2015).
[4] Beckers M. and van Heijst G. J. F., The observation of a triangular vortex in a rotating fluid, *Fluid Dyn. Res.*, **22**, *265-279* (1998).
[5] Botella O. and Peyret R., Benchmark spectral results on the lid-driven cavity flow, *Comput. Fluids*, **24**, *421-433* (1998).
[6] Bruneau C. H. and Saad M., The 2D lid-driven cavity problem revisited, *Comput. Fluids*, **35***(3), 326-348* (2006).
[7] Carnevale G. F. and Kloosterziel R. C., Emergence and evolution of triangular vortices, *J. Fluid Mech.*, **259**, *305-331* (1994).
[8] Cazemier W., Verstappen R. W. C. P. and Veldman A. E. P., Proper orthogonal decomposition and low-dimensional models for driven cavity flows, *Physics Fluids*, **10***(7) 1685-1699* (1998).
[9] Cordier, L., Bergmann, M. Post-processing of experimental and numerical data: POD an overview. *Lecture notes at von Karman Institute for Fluid Dynamics, 146* (2003).
[10] Cullum J. K. and Willoughby R. A., *Lanczos Algorithms for Large Symmetric Eigenvalue Computations. Theory, Vol. I*, Birkhauser, Boston, USA (1985).

[11] Deane A. E., Kevrekidis I. G, Karniadakis, G. E. and Orszag, S. A., Low-dimensional models for complex geometry flow: Application to grooved channels and circular cylinders, *Phys. Fluids A*, **3**, 2337-2354 (1991).
[12] Drazin P. G. and Reid W. H., *Hydrodynamic Stability*, Cambridge Univ. Press, UK (1981).
[13] Eckhaus, W., *Studies in Nonlinear Stability Theory*, Springer, New York, USA (1965)
[14] Erturk E., Corke T. C., Gökcöl C., Numerical solutions of 2-D steady incompressible driven cavity flow at high Reynolds numbers, *Int. J. Num. Meth. Fluids,* **48***(7), 747-774* (2005).
[15] Fortin A., Jardak M., Gervais J. J. and Pierre R., Localization of Hopf bifurcations in fluid flow problems, *Int. J. Num. Meth. Fluids,* **24***(11), 1185-1210* (1997).
[16] Ghia U., Ghia K. N. and Shin C. T., High-Re solutions for incompressible flow using the NavierStokes equations and a multigrid method, *J. Comput. Phys.*, **48**, *387-411* (1982).
[17] Goodrich J. W., Gustafson K. and Halasi K., Hopf bifuraction in the driven cavity, *J. Comput. Phys.*, **90**, *219-261* (1990).
[18] Gustafson K. and Halasi K., Vortex dynamics of cavity flows, *J. Comput. Phys.,* **64**, *279-319* (1986).
[19] Holmes P., Lumley J. L. and Berkooz G., *Coherent Structures, Dynamical System and Symmetry*, Cambridge Univ. Press, U. K. (1996)
[20] Jansson T. R. N., Haspang M. P., Jensen K. H., Hersen P. and Bohr, T., Polygons on a rotating fluid surface, *Phys. Rev. Lett.*, **96**, *174502* (2006).
[21] Kosambi D. D., Statistics in function space, *Indian Math. Soc.*, **7**, 76-88 (1943)
[22] Lestandi L., Bhaumik S., Avatar G.R.K.C., Azaiez M, Sengupta T.K, Multiple Hopf bifurcations and flow dynamics inside a 2D singular lid driven cavity, *Comput. Fluids*, **166**, 86-103, (2018)
[23] Ma X. and Karniadakis G. E., A low-dimensional model for simulating three-dimensional cylinder flow, *J. Fluid Mech.*, **458**, 181-190 (2002)
[24] Noack B. R., Afanasiev K., Morzynski, M., Tadmor, G. and Thiele F., A hierarchy of low-dimensional models for the transient and post-transient cylinder wake, *J. Fluid Mech.*, **497**, 335-363 (2003)
[25] Osada T. and Iwatsu, R., Numerical simulation of unsteady driven cavity flow, *J. The Phys. Soc. Japan.* **80**, *094401* (2011).
[26] Poliashenko M., Aidun C. K., A direct method for computation of simple bifurcations. *J. Comput. Phys.,* **121***(2), 246-260* (1995).
[27] Rempfer D. and Fasel H. F., Evolution of three-dimensional coherent structures in a flat-plate boundary layer, *J. Fluid Mech.*, **260**, 351-375 (1994)
[28] Sahin M. and Owens R. G., A novel fully-implicit finite volume method applied to the lid-driven cavity problem. Part II. Linear stability analysis, *Int. J. Num. Meth. Fluids,* **42**, *79-88* (2003).
[29] Schreiber R. and Keller H. B., Driven cavity flows by efficient numerical techniques, *J. Comput. Phys.,* **49**, *310-333* (1983).
[30] Sengupta T. K., Instabilities of Flows and Transition to Turbulence, CRC Press, USA (2012).
[31] Sengupta T. K., High Accuracy Computing Methods: Fluid Flows and Wave Phenomena, Cambridge Univ. Press, USA (2013).
[32] Sengupta T. K., Bhaumik S. and Bhumkar Y. G., Nonlinear receptivity and instability studies by POD, *6th AIAA Theo. Fluid Mech. Conf., Honululu, Hawaii,USA*, 1-43 (2011)
[33] Sengupta T. K., De S. and Sarkar S., Vortex-induced instability of an incompressible wall-bounded shear layer, *J. Fluid Mech.*, **493**, 277-286 (2003).
[34] Sengupta T. K. and Dey S., Proper orthogonal decomposition of direct numerical simula-

tion data of by-pass transition, *Computers Struct.*, **82**, *2693-2703* (2004).
[35] Sengupta T. K., G. Ganeriwal and Dipankar, A., High accuracy compact schemes and Gibbs' phenomenon - Sengupta, T.K., Ganeriwal, G. and Dipankar, A.; J. Scientific Comput. vol. 21(3), pp 253-268 (2004)., **31**(5), 879-889 (1999).
[36] Sengupta, T. K. and Gulapalli, A., Enstrophy-based proper orthogonal decomposition of flow past rotating cylinder at super-critical rotating rate, *Phys. Fluids*, **28**(11), 114107 (2016)
[37] Sengupta T. K., Haider S. I., Parvathi M. K. and Pallavi G., Enstrophy-based proper orthogonal decomposition for reduced-order modeling of flow past a cylinder, *Phys. Rev. E,* **91**, *043303* (2015).
[38] Sengupta T. K., Lakshmanan V. and Vijay V. V. S. N., A new combined stable and dispersion relation preserving compact scheme for non-periodic problems, *J. Comput. Phys.,* **228**, *3048-3071* (2009).
[39] Sengupta T. K. and Nair M. T., Upwind schemes and large eddy simulation, *Int. J. Num. Meth. Fluids,* **31***(5), 879-889* (1999).
[40] Sengupta T. K., Singh N. and Suman V. K., Dynamical system approach to instability of flow past a circular cylinder, *J. Fluid Mech.*, **656**, *82-115* (2010).
[41] Sengupta T. K., Singh N. and Vijay V. V. S. N., Universal instability modes in internal and external flows, *Comput. Fluids,* **40**, *221-235* (2011).
[42] Sengupta T. K., Vijay V. V. S. N. and Bhaumik S, Further improvement and analysis of CCD scheme: Dissipation discretization and de-aliasing properties, *J. Comput. Phys.,* **228**, *6150-6168* (2009).
[43] Seydel R., *Practical Bifurcation and Stability Analysis from Equilibrium to Chaos*, Springer: Berlin (1994).
[44] Shen J., Hopf bifurcation of the unsteady regularized driven cavity flow , *J. Comput. Phys.,* **95**, *228* (1991).
[45] Sirovich L., Turbulence and dynamics of coherent structures. Part (I) Coherent structures, Part (II) Symmetries and transformations and Part (III) Dynamics and scaling, *Quart. J. Appl. Math.*, **45**(3), 561-590 (1987).
[46] Suman, V.K., Sengupta, T. K., Prasad, C. J. D., Mohan, K. S. and Sanwalia, D., Spectral analysis of finite difference schemes for convection diffusion equation, *Comput. Fluids, 150, 95-114* (2017).
[47] Van der Vorst H. A., Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems, *SIAM J. Sci. Stat. Comput.*, **13**, *631- 644* (1992).

RESEARCH

Reduced order model of flows by time-scaling interpolation of DNS data

Tapan K Sengupta[1], Lucas Lestandi[2*], S. I. Haider[1], Atchyut Gullapalli[1] and Mejdi Azaïez[3]

*Correspondence:
llestandi@u-bordeaux.fr
[2]University of Bordeaux, I2M UMR 5295, Bordeaux, France
Full list of author information is available at the end of the article
[†]Equal contributor

Abstract

A new reduced order model (ROM) is proposed here for reconstructing super-critical flow past circular cylinder and lid driven cavity (LDC) using time-scaling of vorticity data directly. The present approach is a significant improvement over instability-mode (developed from POD modes) based approach implemented in Sengupta *et al.* [**91**, 82-115, PRE (2015)], where governing Stuart-Landau-Eckhaus equations are solved. In the present method, we propose a novel ROM that uses relation between Strouhal number (St) and Reynolds number (Re). We provide a step by step approach for this new ROM for any Re and is a general procedure with vorticity data requiring very limited storage as well as being extremely fast. We emphasize on the scientific aspects of developing ROM by taking data from close proximity of the target Re to produce DNS-quality reconstruction, while the applied aspect is also shown. All the donor points need not be immediate neighbors and the reconstructed solution has equivalent relaxed accuracy. However, one would restrain the range where the flow behavior is coherent between donors. The reported work is a proof of concept utilizing the external and internal flow examples, and this can be extended for other flows characterized by appropriate Re-St data.

Keywords: sample; article; author

1 Introduction

High performance computing using DNS for complex flow problems provide insight into physical mechanism at prohibitive cost of data storage, as voluminous data are created to resolve small scales in both space and time. DNS of Navier-Stokes equation (NSE) to understand flow generates huge amount of data. The major challenges of big data are processing, storage, transfer and analysis etc. One of the motivations here is to replace time/ memory-intensive DNS for the model problems of flow past a circular cylinder and LDC. Similar attempts are recorded in [25, 29] and other references contained therein. Memory requirements of such instability mode-based ROM in [25] come down drastically, due to the requirement of storing only fewer coefficients of the SLE equations and initial conditions. Henceforth this reference will be called SHPG for brevity.

There are numerous efforts in developing ROM's, e.g. via Koopman modes, as in [5, 20]; dynamic mode decomposition in [21]; POD-based analysis of Reynolds-averaged Navier-Stokes (RANS) in [15, 16, 32]. In [3], authors reported low-dimensional model for 3D flow past a square cylinder using solutions of NSE obtained by a pseudo-spectral approach. However, even using thousands of snapshots, the reconstruction error was of the order of 30%, indicating an *exponential divergence between any model prediction and the actual solution outside the snapshot*

range. In [17], authors used fourth order finite difference scheme for spatial discretization of NSE in primitive variable formulation for time accurate simulation for POD analysis of the flow field. The time discretization used second order accurate, three-time level discretization method, which invokes a numerical extraneous mode. It was noted that with only four POD modes, the model without pressure term *gives rise to important amplitude errors which cannot be compensated by an increase in the number of modes*. In energy-based POD approaches, researchers calculate amplitude functions of POD representation by solving ODE's derived from NSE by simplifying nonlinear and pressure terms. Various techniques have been used in the attempt to reduce the resulting error, such as discrete empirical interpolation method (DEIM) in [1, 4]. Authors in [18] have also used an adaptive approach to construct ROM with respect to changes in parameters, by first identifying the parameters for which the error is high. Thereafter a surrogate model based on error-indicator was constructed to achieve a desired error tolerance in this work.

The flow governed by unsteady NSE presents the physical dispersion relation linking each length scale (wavenumber) with corresponding time scale (circular frequency). Thus, the ranges of time and length scales are important, even though a single St and Re are often used to describe the flow field. Multitude of length and time scales also are inherently noted in [13] via POD modes and multiple Hopf bifurcations for flow in LDC. The existence of such ranges assists in developing a ROM, when donor Re's are in the same range, where the target Re resides. If one takes one or two donor points far from the range where target Re resides, the presented ROM will provide a reconstructed solution, still with acceptable accuracy. These aspects of multiple Hopf bifurcations and existence of ranges of Re is highlighted in the present research, apart from developing an efficient ROM for this model problem.

For a vortex dominated flow, the time scale is defined as St ($= fD/U_\infty$), relating dominant physical frequency (f) with flow velocity, (U_∞) and the length scale (D). However the flow does not display a single frequency, as one notices several peaks for both flows in figure 1. The time series of the vorticity data at indicated locations are shown in the left hand side frames. While the flow past a circular cylinder displays a single dominant peaks with side bands in the spectrum (shown on the right hand side frames), the flow inside LDC clearly demonstrates multiple peaks. This property has been explored thoroughly for the LDC in [12] to explain the roles of multiple POD modes.

Specifically for flow past circular cylinders, an empirical relation of the type has been provided

$$\mathrm{St} = St^* + m/\sqrt{Re} \tag{1}$$

in [7] with experimental data, for variation of St with Re in the wide range of $47 < \mathrm{Re} < 2 \times 10^5$, with values of St^* and m being different, for different ranges of Re. Instead of using such an algebraic additive relationship, here we propose a power law relation and test it for the range: $55 \leq \mathrm{Re} \leq 200$, for the purpose of

demonstration. Consequently a relationship between Re and St will be proposed, in order to perform interpolation on the vorticity time series.

The existence of unique St for a fixed value of Re, as embodied in equation (1) implies that employing simple-minded interpolation strategies like Lagrange interpolation, will display unphysical wave-packets in reconstructed solution, as the time scales are function of Re at the target. This is clearly demonstrated in figure 3. The proposed ROM tackles this issue with the time scaling technique that is presented in this article.

The paper is formatted in the following manner. In the next section, governing equations employed for DNS and associated auxiliary conditions are described. In Section 3, the proposed time-scaling interpolation algorithm is presented. Time-scaled ROM of vorticity field is applied to two complex flows in Section 5. Summary and conclusions are provided in the last section.

2 Governing Equations and Numerical Methods

DNS of the 2D flow is carried out by solving NSE in stream function-vorticity formulation given by,

$$\nabla^2\psi = -\omega \tag{2}$$

$$\frac{\partial\omega}{\partial t} + (\vec{V}\cdot\nabla)\omega = \tfrac{1}{Re}\nabla^2\omega \tag{3}$$

where ω is the only non-zero, out-of-plane component of vorticity for the 2D problem considered. The velocity is related to the stream function as $\vec{V} = \nabla \times \vec{\Psi}$, where $\vec{\Psi} = [0\ 0\ \psi]^T$, with (D) and (U_∞) used as length and velocity scales for non-dimensionalization. Equations (2) and (3) are solved in an orthogonal curvilinear coordinates (ξ, η) and the governing equations in transformed plane are

$$\frac{\partial}{\partial\xi}\left(\frac{h_2}{h_1}\frac{\partial\psi}{\partial\xi}\right) + \frac{\partial}{\partial\eta}\left(\frac{h_1}{h_2}\frac{\partial\psi}{\partial\eta}\right) = -h_1h_2\omega \tag{4}$$

$$h_1h_2\frac{\partial\omega}{\partial t} + h_2u\frac{\partial\omega}{\partial\xi} + h_1v\frac{\partial\omega}{\partial\eta} = \frac{1}{Re}\left\{\frac{\partial}{\partial\xi}\left(\frac{h_2}{h_1}\frac{\partial\omega}{\partial\xi}\right) + \frac{\partial}{\partial\eta}\left(\frac{h_1}{h_2}\frac{\partial\omega}{\partial\eta}\right)\right\} \tag{5}$$

where h_1 and h_2 are the scale factors of the transformation given by: $h_1^2 = x_\xi^2 + y_\xi^2$ and $h_2^2 = x_\eta^2 + y_\eta^2$. The co-ordinate given by ξ is along azimuthal direction for the flow past the cylinder and along x-direction for flow inside LDC and the co-ordinate η is in the wall-normal direction for flow past the cylinder and along y-direction for the flow inside LDC. No-slip boundary condition is applied on the wall for both the flows via

$$\left(\frac{\partial\psi}{\partial\eta}\right)_{body} = 0 \quad \text{and} \quad \psi = constant$$

For the flow inside LDC, the corresponding conditions are given by the same equations, except along the lid, the right hand side of the first condition is given by U_∞. These conditions are used to solve equation (4) and to obtain the wall vorticity

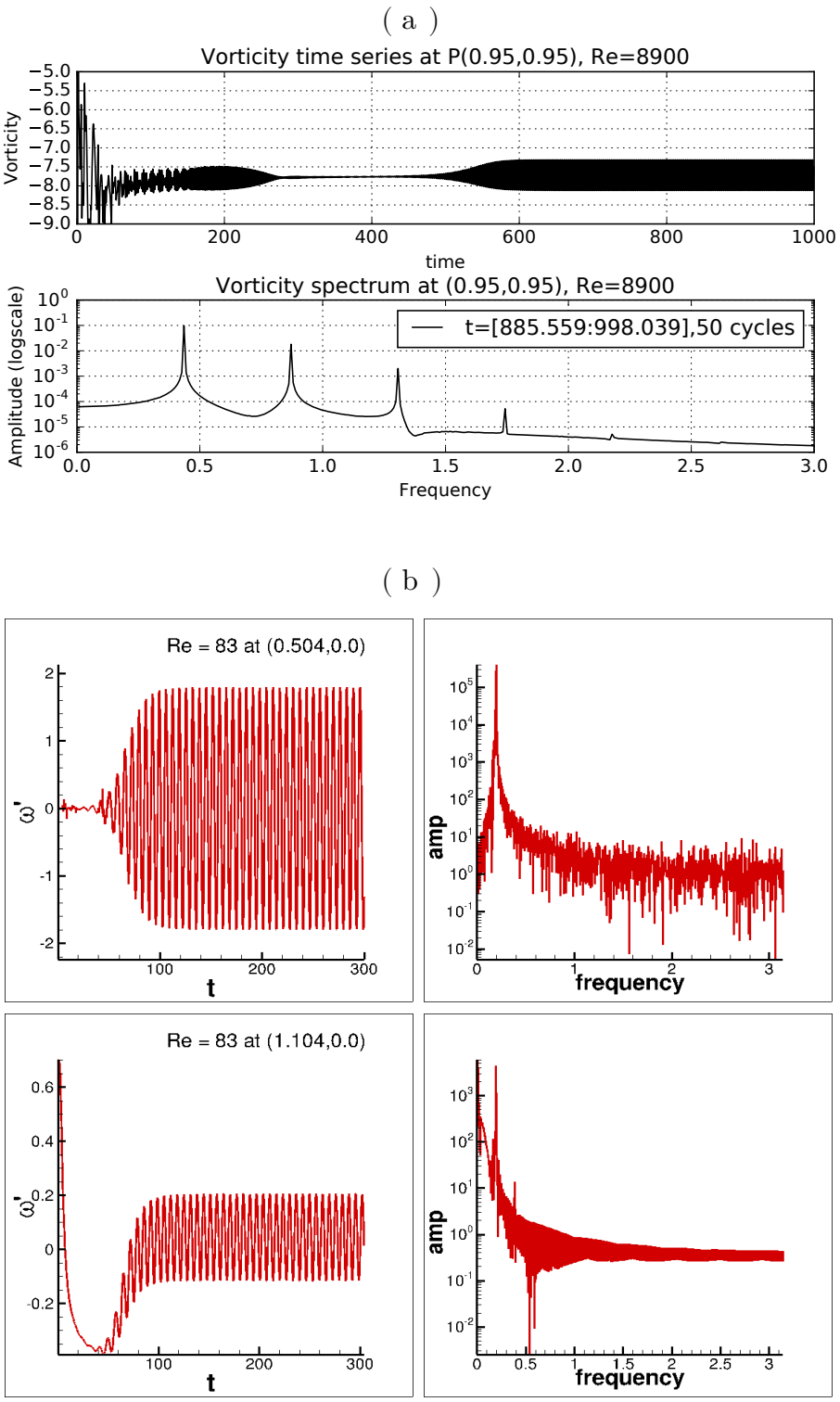


Figure 1 DNS time series and their associated FFT's are shown for (a) the flow inside a LDC and (b) the external flow past a cylinder, at indicated points in the flow.

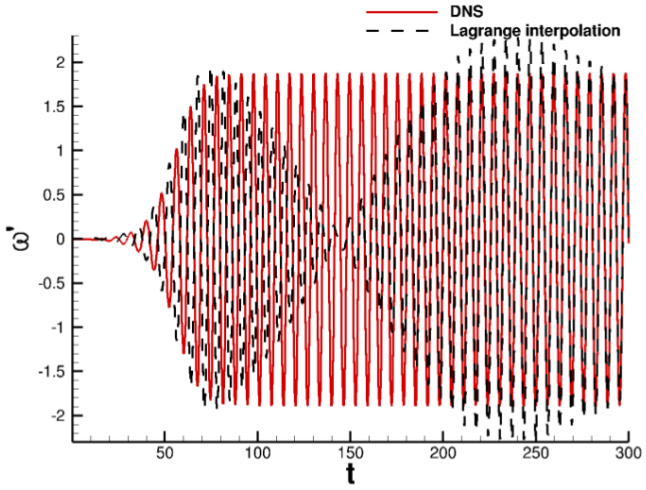


Figure 2 Direct Lagrange interpolation of DNS vorticity disturbance time series between Re causes wave packets in the cylinder wake at point (0.504,0.0).

ω_b, which in turn provides the wall boundary condition for equation (5). At the outer boundary of the domain for flow past cylinder, uniform flow boundary condition (Dirichlet) is provided at the inflow and a convective condition (Sommerfeld) is provided for the radial velocity at the outflow.

The convection terms of equation (5) are discretized using the high accuracy compact OUCS3 scheme for flow past the cylinder and the combined compact difference (CCD) scheme for the flow inside LDC, both of which provides near-spectral accuracy for non-periodic value of the convective acceleration terms, as explained in detail in [23]. A central differencing scheme is used to discretize the Laplacian operator of equations (4) and (5) for the circular cylinder and the CCD scheme is used for the flow inside LDC. An optimized four-stage, third-order Runge-Kutta (OCRK3) dispersion relation preserving method in [27] is used for time marching. Equation (4) is solved using Bi-CGSTAB method given in [35].

These same methods have been used earlier for validating and computing the respective flows in [29], SHPG for flow over cylinder and in [26, 30, 31] for flow inside the LDC. Here the simulations are performed in a fine grid, with (1001×401) points in the ξ and η directions for the flow past circular cylinder, and (257×257) points are taken for the LDC problem.

3 Need for time scaling

The proposed ROM aims at interpolating vorticity fields at a target Re (Re_t) from precomputed DNS at different donor Re's. If Lagrange interpolation is used directly, then it will not work due to variation of St with Re. Even with close-by donor Reynolds numbers data, upon interpolation, will produce wave-packets for flow past a cylinder as shown in figure 3. In this figure, results are shown for Re = 83, as obtained by DNS of NSE (shown by solid lines) and that is obtained by Lagrange interpolation of NSE solution donor data obtained for Re = 78, 80, 86 and 90.

We have also noted in SHPG that the flow past a circular cylinder suffers multiple Hopf bifurcations (experimentally shown in [9, 34]) and in [30] for flow inside LDC

and flow over cylinder. Hence the accuracy of reconstruction naturally demands that the target and donor Re's should be in the same segments of figure 3, as the flow fields are dynamically similar. In figure 3, the equilibrium amplitude of disturbance vorticity are plotted as a function of Re for both the flows. The equilibrium amplitude refers to the value of the disturbance quantity, which settles down in a quasi-periodic manner, due to nonlinear saturation after the primary and secondary instabilities. Presence of multiple quadratic segments in figure 3, indicates multiple bifurcations originating at different Re's. Thus, it is imperative that one identifies the target Re in the same segment of donor Re's for DNS-quality reconstruction for flow past circular cylinder as in SHPG and for flow inside LDC in [12]. In each of these sectors of Re, the flow behaves similarly and the (St, Re)-relation is distinct. It is to be emphasized that the present sets of simulations are performed using highly accurate dispersion relation preserving numerical methods.

The physical frequency (f) varies slowly with Re and superposition of time-series of donor data causes beat phenomenon observed by superposition of waves of slightly different frequencies. Thus, the knowledge of variation of St with Re is imperative in scaling out f-dependence of donor data before Lagrange interpolation and this is one of the central aspects of the present work. After obtaining frequency-independent data at target Re, one can put back the correct f-dependence via its variation with Re at the target Reynolds number.

In figure 3(a), the range of Re from 8000 to 12000 for the LDC is subdivided according to the bifurcation sequence uncovered in [13] using a (257×257)-grid. For the purpose of interpolation, four ranges are defined with the first one given by: $\mathtt{R_I} = [8020 : 8660]$ that corresponds to externally excited range, which shows rapid variation of the amplitude, nearly culminating in a vertical fall at the onset of solution bifurcation. The used CCD scheme, for flow in LDC, has near-spectral accuracy, as explained in [26, 31], and the onset of unsteadiness is due to aliasing error predominant near the top right corner of LDC, while truncation, round-off and dispersion errors are negligibly small. To avoid the issue of lower numerical excitation in the present work, a pulsating vortex is placed (ω_s) at $x_0 = 0.015625$, and $y_0 = 0.984375$ whose spread is defined by the exponent α given in the following,

$$\omega_s = A_0[1 + \cos(\pi(r - r_0)/0.0221)]\sin(2\pi f_0 t) \quad \text{for } (r - r_0) \leq 0.0221$$

where in the presented results here we have taken $f_0 = 0.41$ for the single amplitude, $A_0 = 1.0$.

For the next two ranges, no explicit excitation is needed (i.e., $A_0 = 0$) to achieve a stable limit cycle. $\mathtt{R_{II}} = [8660 : 9350]$ and $\mathtt{R_{III}} = [9450 : 10600]$ are ranges for which the amplitude (A_e) follows a square root law, these are however different because of the peculiar behavior of the flow in the vicinity of Re = 9400, which indicates the onset of second Hopf bifurcation. Finally, $\mathtt{R_{IV}} = [10600 : 12000]$ is difficult for interpolation, as one can see two branches in this range, one of which is unstable (U-branch) with respect to any miniscule vortical excitation, as opposed to the stable one (S-branch). The flow past cylinder is also divided in ranges as shown in figure 3(b). The range of *Re* from 55 to 130 is subdivided according to the bifurcation sequences by: $55 \leq \text{Re} \leq 68$; $68 \leq \text{Re} \leq 78$; $78 \leq \text{Re} \leq 90$; $90 \leq \text{Re} \leq 100$ and

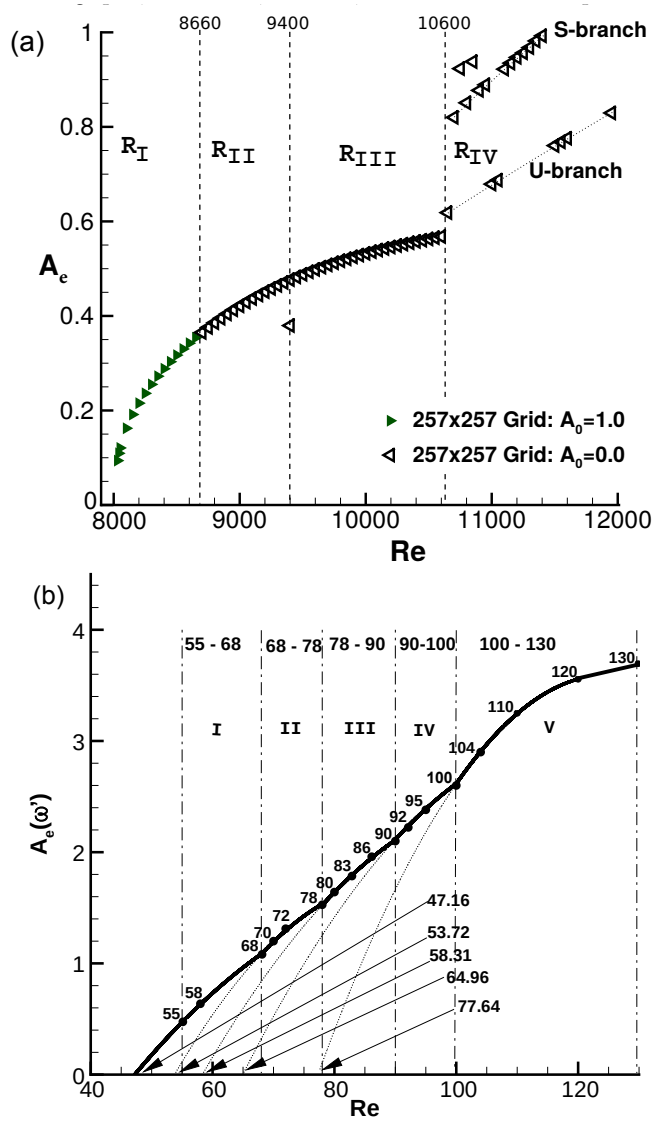


Figure 3 Variation of Equilibrium amplitude of disturbance vorticity with Re indicating the segments of Re with respect to bifurcation sequences for (a) flow in LDC and (b) for flow past a cylinder.

$100 \leq \mathrm{Re} \leq 130$. For example, to reconstruct solution for Re=83, we have used data in the range of $78 \leq Re \leq 90$ for the most accurate ROM.

3.1 Formulation and Modeling of ROM

In equation (1), a relation between St and Re is shown for a wide range, for the latter. In the proposed ROM here, we do not need DNS data for the target Re, as was the case in SHPG to train the ROM. This is a significant improvement over the previous approaches. One should scale out dependence of DNS data on f or St, for any Re, by a proposed power law scaling given below,

$$\frac{St(Re_s)}{St(Re_b)} = \left(\frac{Re_b}{Re_s}\right)^n \tag{6}$$

The exponent n will depend upon the segment of Re shown in figure 3, with Re_b denoting a base Reynolds number in each segment. Here in this equation, any donor Re is indicated as Re_s. Thus in a cluster of four donor Re's, one is identified as Re_b and the other three identified as Re_s. From equation (6) one identifies n, by the following,

$$n = \frac{\log(St(Re_s)/St(Re_b))}{\log(Re_b/Re_s)} \tag{7}$$

The scaling exponent n is a characteristic number of each segment and Re_b. In Table 1, we show five segments and the corresponding n, along with Re_b used in each range. For the flow past a circular cylinder, the value of n is obtained with the tolerance of ± 0.02 for all Re's in the respective segment. As discussed in [13], f is almost constant on each segment, so that we can set $n = 0$ for the LDC, individually in each segment. Having fixed n for any Re_s in the segment of choice, time-scaling is performed by the following,

$$t_s = t_b\left(\frac{Re_b}{Re_s}\right)^n + t_0(Re_b, Re_s) \tag{8}$$

To interpret equation (8), we plot the disturbance vorticity for the flow past a cylinder at a fixed location in the wake center-line ($x = 0.504$, $y = 0$), in figure 4. The same format of time scaling should apply to many other flows, including the same for the internal flow inside a LDC. It is noted that there exists a time-shift between the maximum of these two time series, shown as t_0 in the figure. Let us consider the time for Re_b as t_b, and then to apply the proposed time-scaling for the data for Re_s, we change the physical time of Re_s, by the expression given in equation (8). Consequently, the left hand side of equation (8) is the scaled time. After obtaining t_0, it is needed to collapse the two time series for Re_s and Re_b, so that the maximum for these two time series coincide. Thus having fixed the base Reynolds number in each windows of bifurcation sequences, we can obtain the time-scaled abscissa for each Re_s in that range.

The search for t_0 is performed in such a way that the phases of both Re_b and Re_s match accurately. One should note that the effects of t_0 are significant, despite the fact that it has a very small value. There are many ways to compute t_0, but accuracy must be very high in estimating it. A specific way is to view the time series in the spectral plane and using the imaginary part of FFT to be used as the accuracy parameter, as described in the next subsection.

3.2 Computing the initial time-shift (t_0)

The present method is both accurate and computationally cheap, since it relies on the fast Fourier transform (FFT) that is provided in the `numpy` library. A FFT is

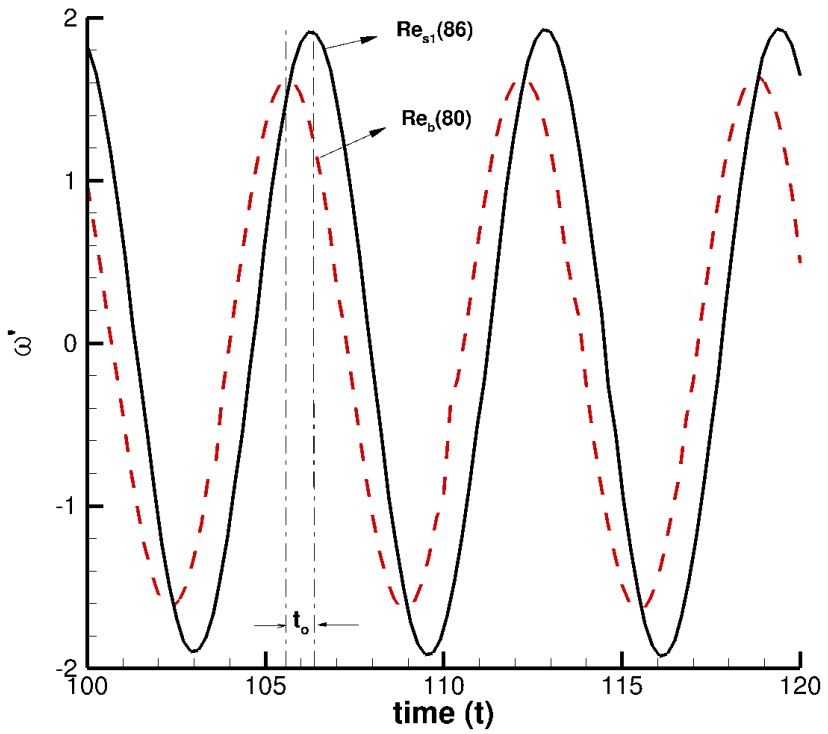


Figure 4 Variation of disturbance vorticity at a point (0.504, 0.0) with t_b and t_s for Re_b and Re_s, respectively, for the pair of $Re_b = 80$ and $Re_s = 86$ in the bifurcation sequence $78 \leq Re \leq 90$.

applied to the vorticity time series at one relevant space point. On one hand, for the LDC problem it has been shown in [13] that $(0.95, 0.95)$ point near the top right corner is relevant for monitoring the flow behavior. On the other hand for the flow past a circular cylinder, point $(0.504, 0.0)$ in the cylinder wake is adequate. For each sampled frequency, a complex value $(z(f) = Ae^{i\theta})$ is obtained consisting of the modulus (A), which corresponds to the amplitude and a phase (θ). Consequently, we can recover the phase associated with the leading frequency (L) for both signals θ_b and θ_s. Finally the time shift of signal s with respect to the signal b is given by

$$t_0 = \frac{\theta_b^L - \theta_s^L}{2\pi f^L} \tag{9}$$

Table 1 Scaling Constant and Base Re_b for Different Range of Re_s

Re Range	Scaling Constant (n)	Basic Re (Re_b)
$55 - 68$	-0.49 ± 0.02	60
$68 - 78$	-0.41 ± 0.02	72
$78 - 90$	-0.37 ± 0.02	80
$90 - 100$	-0.32 ± 0.02	95
$100 - 130$	-0.28 ± 0.02	110

Algorithm-1: Time-scaling algorithm for discrete signals

input: $\omega_b, Re_b, \omega_s, Re_s, t = \{t_i\}_{i=1}^N$
output: $\hat{\omega}_s$ `/* the time scaled signal.*/`

1. Perform FFT on both signals
2. Scale frequencies $\left(C = \left(\frac{Re_b}{Re_s}\right)^n\right)$
3. Evaluate $t_0(Re_b, Re_s) = \frac{\theta_s^L - \theta_b^L}{f_s^L 2\pi}$
4. New time $t_s = Ct + t_0$ is associated with ω_s
5. Interpolate the *time-scaled signal* $\hat{\omega}_s(t)$ from $\omega_s(t_s)$
`/* At this point, one can perform Lagrange interpolation between the donor points to the target Re to obtain` $\bar{\omega}^*$`.*/`

return $\hat{\omega}_s$

Here, f^L is the lead frequency in the amplitude spectrum for both the signals as t_0 is computed only after the frequency scaling has been performed, with θ as the angle of the complex value of the FFT associated with the lead frequency for signal b or s. This method yields reliable and accurate values of t_0, as the ROM accuracy will prove in the following sections.

3.3 Time-scaling ROM algorithm for discrete DNS data

In this subsection, a brief recap of the time shifting procedure for ROM building is given for the simple case of discrete signals $\omega_b(t_i)$ and $\omega_s(t_i)$ with $\{t_i\}_{i=1}^N$ indicating the time discretization. It can be directly applied to any space-time dependent field, with a reference signal chosen at a reference point. The ROM is then built as follow:

1 Perform the algorithm (Algorithm-1) on all signals, except the base donor signal, in order to scale their oscillations.
2 Perform Lagrange interpolation on the scaled donor signals at target Re_t for all discrete times t_i.

$$\bar{\omega}^\star(t_i) = \sum_{s \in \text{donors}} \hat{\omega}_s(t_i) l_s(Re_t) \tag{10}$$

where $\bar{\omega}^\star$ is the target signal and l_s are the Lagrange interpolation polynomials.
3 Scale-back $\bar{\omega}^\star$ to the physical time with $t^\star = \frac{t - t_0(Re_t)}{(Re_b/Re_t)^n}$.

The last step of the ROM is to scale back $\bar{\omega}^{\star(t)}$ to the physical time, $t^\star$. Indeed, the interpolation is performed at grid points for t, which is actually the time-scaled representation of the target vorticity field. Thus the scale-back operation is computed to associate $\bar{\omega}^\star$ with the scaled-back time $t^\star$. One should note that the final domain is cropped according to the information lost after each shift, despite this the discrete time points match the original discretization.

4 Time-shifting ROM applied to the LDC flow

As we have shown in [13], the main frequency of the LDC flow is nearly constant across large ranges of Re, as shown here in figure 5. Thus, the time-scaling procedures simplify to a time-shifting procedure with $n = 0$, resulting in $t_s = t - t_0$ for the donor and target points, which have the same frequency in figure 5.

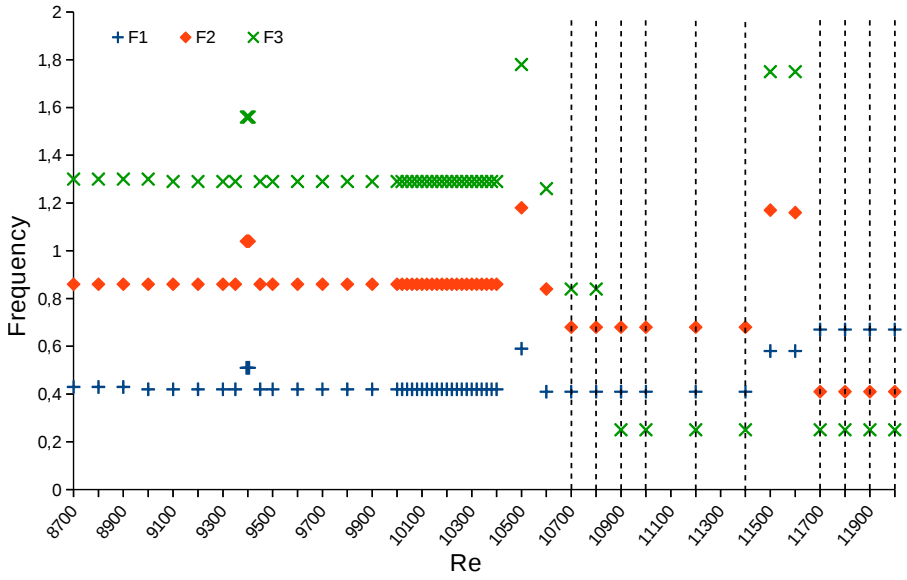


Figure 5 Frequency variation for Re=[8700,12000] for the first three leading frequencies of the vorticity time series at point $(0.95, 0.95)$ obtained for the last 50 periods. The dotted lines indicate the presence of multiple dominating peaks in the spectrum.

Following Algorithm-1 given above, we have obtained the vorticity field for Re = 10040, using the donor points at Re = 10000, 10020, 10060 and 10080. From the reconstructed ROM data, we have shown the vorticity time series in figure 6 for four representative points near four corners. Despite the change in the vorticity magnitude by two orders, the accuracy of reconstruction is excellent and match almost exactly.

In figure 7, the reconstructed vorticity contours inside the LDC is shown for Re = 10040, at the indicated time of $t = 1900.199$ by solid line, with the same donor data of Re's for the use in the ROM following Algorithm-1. The corresponding solution obtained by DNS of NSE-Solution for Re=10040 is shown in the same figure by dotted lines. It is readily observed that these exact and ROM solution overlap each other in the full domain with a relative RMS error of 7.1×10^{-4}.

The above exercise shows the special case of a flow, which is multi-periodic with respect to time, yet the predominant frequency remains constant over different ranges of Re, allowing one to use the special version of time scaling with power law exponent given by, $n = 0$ in equations (6) and (7). Thus, one needs to simply apply a time-shift and reconstruct by the methods described in Subsections 3.2 and 3.3.

Next, ROM is performed for Re = 9600, with the donor points at Re = 9350, 9500, 9800 and 10000. The choice of the second target Re for LDC is made on purpose, as the bifurcation diagram in figure 3(a) shows that the flow has discontinuity in equilibrium amplitude in the chosen donors the bounds of R_{III} for Re = 9400 and 10600. The interpolated vorticity time series are compared with direct simulation results, as shown in figure 8, at those same sampling points used in figure 6. Once again the match is excellent between interpolated results with DNS data with a very low RMS error of 5.6×10^{-4}.

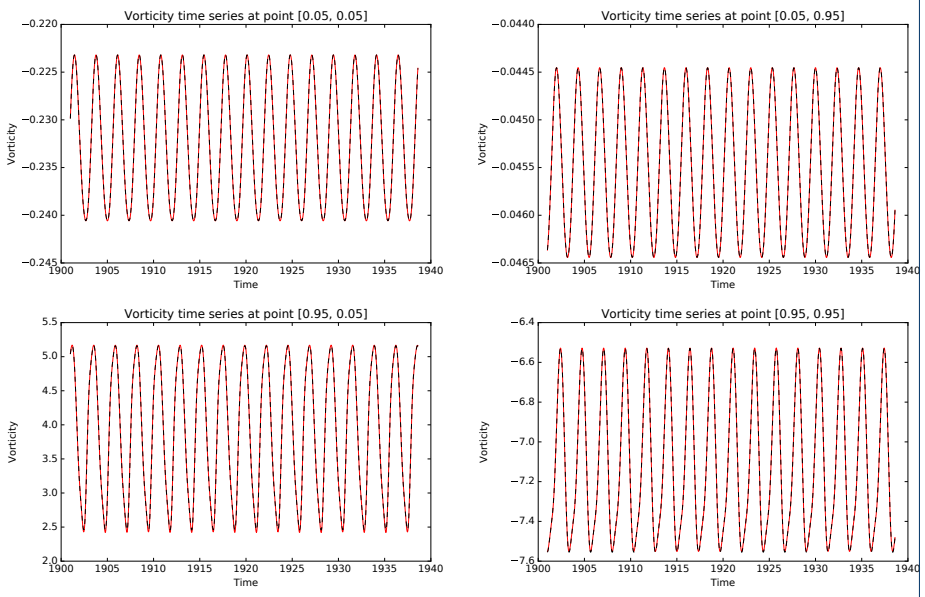


Figure 6 Reconstructed vorticity field at the points in the cavity: (a) the point near the bottom left corner, $x = 0.05$, $y = 0.05$, (b) the point near the top left corner, $x = 0.05$, $y = 0.95$, (c) the point near the bottom right corner, $x = 0.95$, $y = 0.05$ and (d) the point near the top right corner, $x = 0.95$, $y = 0.95$ for the target Reynolds number, $Re = 10040$ with donor points having Re as 10000, 10020, 10060 and 10080.

In figure 9, the interpolated vorticity contours for Re = 9600 are compared with those computed directly from NSE to show that interpolation works globally in the flow field and not merely at chosen sampling points. In this flow field, the power law exponent is zero and the strength of the interpolation is in obtaining the initial time shift (t_0) obtained using Algorithm-1, obtained from the FFT of the donor point vorticity with respect to the baseline Re chosen.

In the following, we study the case of flow past a circular cylinder to show the efficacy of the proposed time-scaling algorithm used here. For this flow also one notices presence of multiple time scales, but with a predominant frequency characterized by St, which follows the power law given by equation (6), with nonzero power law exponent, n.

5 Time-scaled ROM applied to the flow past a cylinder

All the time-scaled relation and corresponding power law exponent in equation (7), is applicable here for ROM with ω obtained by DNS. The time scaled interpolation of the ROM for disturbance vorticity for different combination of donor points, as indicated in Table 2, are obtained and root mean square (RMS) error with respect to DNS data are compiled in the table summed over all the points in the domain. Case I in the table corresponds to the case of donor points at Re = 78, 80, 86 and 90, which is noted as the most accurate based on RMS error for the ROM reconstruction for Re =83. When we choose the donors with Re = 55, 80, 86 and 130 for Case V in Table 2, the RMS error is again low, as compared to cases where only one donor point is taken from the same segment containing the target Re. As

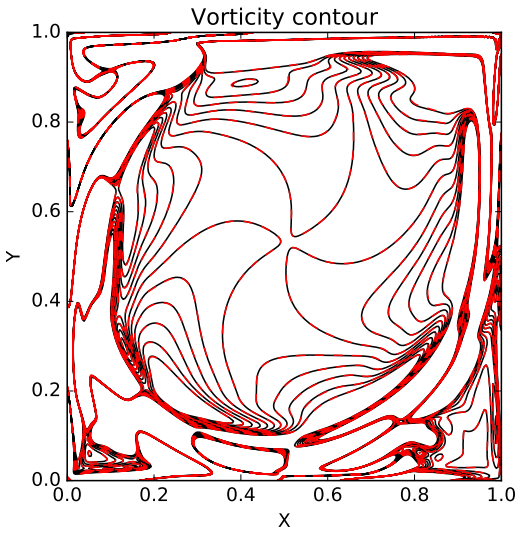


Figure 7 Disturbance vorticity contour plot for reconstructed vorticity (solid lines) and DNS vorticity (dotted lines) field at nondimensional time $t = 1900.199$ for target Re=10040 with donor points having Re $= 10000, 10020, 10060 and 10080$.

Table 2 RMS Error estimates of interpolation for Re = 83

Cases	Re of donor points	Error for interpolation using donor points
I	(78,80,86,90)	0.0434535949140671049
II	(72,80,86,90)	0.0438833300701889223
III	(68,80,86,90)	0.0445922677374889012
IV	(55,80,86,90)	0.0624577915198629291
V	(55,80,86,130)	0.140945940261735560
VI	(55,68,72,86)	1.3159752726807628345
VII	(55,68,72,130)	8.52240911220835436

has been noted before, for higher accuracy one must choose donor points from the same segment of target Re, as clearly shown in Table 2 in a quantitative manner.

We draw the attention on error estimates provided in Table 2 for different combinations of donor Re's. It is evident from the table that the best result is obtained when all four donor points are in the same segment of target Re, as in Case I. In Cases II to IV, we have taken the lowest Re, farther to the left with increase in RMS error, with lowering of the smallest donor Re. But in Case V, the extreme Re's are chosen as 55 and 130, and yet the RMS error is acceptable, as two of the donor Re's belong to the segment of target Re. In contrast, for the Case VI, only a single donor Re belongs to the same segment, resulting in RMS error increasing almost ten folds as compared to the Case V. The worst case (Case VII) occurs in Table 2, when all the donor Re's are outside the target Re segment. This justifies the scientific basis of the adopted ROM keeping the various ranges of Re punctuated by various Hopf bifurcations shown in figure 3(b).

Role of t_0 is also investigated here for ω' (the disturbance vorticity field) and the variation of t_0 with the Re is shown in figure 10 in the subrange $55 \leq \text{Re} \leq 130$. Here, we obtain t_0 for the data sets of (Re$= 55, 80, 86, 130$) and (Re$= 78, 80, 86, 90$),

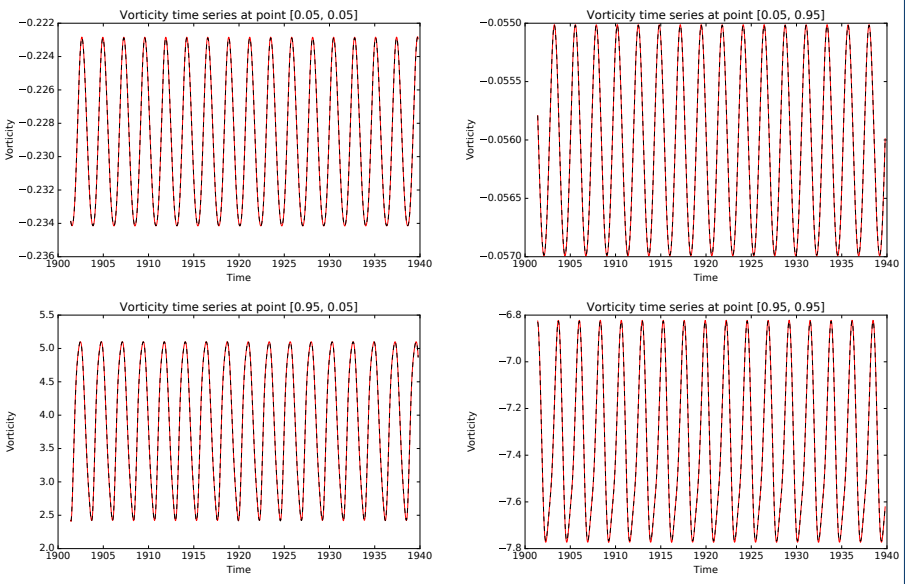


Figure 8 Reconstructed vorticity field at the same four points in LDC, as in figure 6 for a target Re = 9600 with donor points for Re = $\{9350, 9500, 9800$ and $10000\}$.

as indicated separately in the figure. Each of the discrete data are marked in the figure with Re and necessary time shifts in brackets, with $Re_b = 80$. It is noted that the finding of single t_0 is far easier and less time consuming for ω' for the present version of ROM, as compared to any method using POD or instability modes, which would require finding different t_0 for each retained modes.

In this method, ω' is reconstructed using the identical procedure of interpolation after time-scaling and initial time-shift, using equation 8 applied directly on ω obtained by DNS. Thus, this procedure even circumvents the need to use the time-consuming method of snapshots to obtain POD modes that is required for any POD based ROM e.g. POD-Galerkin, interpolated POD. Unlike the methods of solving SLE equations given in SHPG, proposed ROM in this paper requires storage of at most four DNS data sets in each segment for most accurate reconstruction. If one is willing to settle for lesser accuracy, then one can reduce the requirement of performing DNS for two Re only, in each segment of figure 3. Hence this ROM is not memory intensive and it is faster.

Figures 11(a) and (b) show the comparison between DNS and the time-scaled interpolated ω' at two different points for Re=70, located along the wake-center line at $(0.504, 0.0)$ and at $(1.014, 0.0)$, respectively. Excellent match with the DNS data even in the transient state proves the efficacy of the time-scaling interpolation technique applied to vorticity data. It is to be noted that despite the presence of a dominant St, the physical variables demonstrate multiple time-scales as discussed in the introduction and shown in figure 1.

The case for Re= 83 are shown in figures 11(c) and (d), which compare the disturbance vorticity at the same two locations with DNS data. Once again, the reconstructed ROM solution is indistinguishable from the corresponding DNS data. Thus,

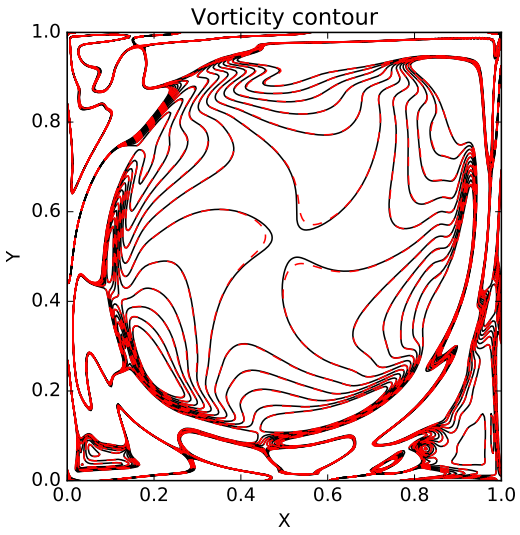


Figure 9 Disturbance vorticity contour plot for reconstructed vorticity (solid lines) and DNS vorticity (dotted line) field at nondimensional time $t = 1900.199$ for target Re = 9600 with donor points $\{9350, 9500, 9800 \text{ and } 10000\}$.

it is evident that spectrum with multiple peaks can be handled by the presented approach of time-scaling with initial time-shift, utilizing the power law between Re with St.

6 Summary and Conclusion

Here, we have proposed time-scaled ROM for reconstructing super-critical flow past circular cylinder and flow inside LDC using time-scaled Lagrange interpolation of vorticity data obtained by DNS for different donor data at Re's, largely located in the neighbourhood of the target Re. In performing the interpolation, a time-scaling is performed following equation (8) along with an initial time-shift, as a direct consequence of (St, Re)-relations given in equations (6) and (7).

The proposed method differ from the ROM based on instability modes in SHPG, with respect to speed, accuracy and generality of application. ROM Reconstruction at a target Re is of DNS-quality, if all the donor points belong in the same Re subrange, identified by multiple Hopf bifurcations in figure 3(a), for flow inside the LDC in the range $8700 \leq Re \leq 12000$ and in figure 3(b) for flow past a circular cylinder, in the range of $55 \leq Re \leq 130$ and in Table 1.

Data requirement of present ROM is at most for four Re's located in the same subrange. If one wants to perform ROM with only three Re's, then the reconstructed data are of slightly lower accuracy, but of very acceptable quality (not shown here). The present procedure provides scientific and applied basis of ROM, depending upon the number and location of donor points of target Re.

In instability based ROM in SHPG, one stores only the coefficients of SLE equations. However, one needs to obtain optimal initial conditions for the stiff SLE equations and is restricted to use of first five POD or three instability modes. This is due to difficulty in finding optimal initial conditions for SLE equations and only

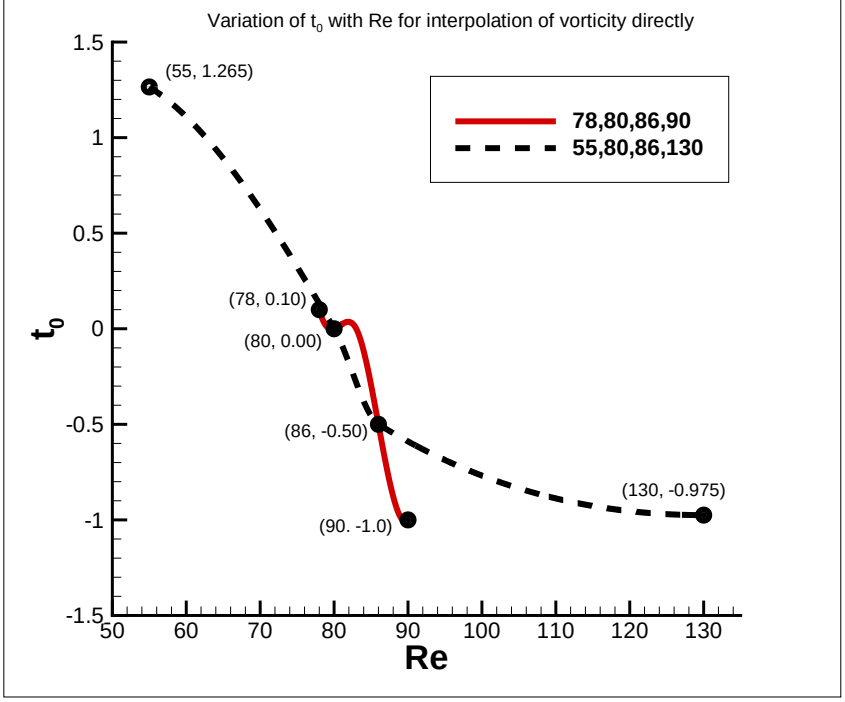


Figure 10 Variation of t_0 with Re_s for $Re_b = 80$ for Case I (solid line) and Case V (dashed line) of Table 2. Shown in parametric form are the pair of Reynolds number and corresponding optimal t_0.

three instability modes have been used in SHPG. In the present approach, one finds initial time-shift (t_0) for the donor vorticity data with respect to a base Reynolds number. This time shift can be obtained by FFT based approach as proposed here.

Present study opens the scope of data mining in computational fluid dynamics. DNS of NSE produces massive amount of data which can be used economically to predict flow behavior of dynamical systems dominated by single or multiple peaks in the spectrum. The proposed ROMs can be used at any arbitrary Re on demand, by the proposed ROM performed with limited number of DNS at neighbouring Re's. The novel procedure proposed here has been tested for the internal flow inside a LDC and an external flow over a circular cylinder, as proofs of concept.

Declaration

Competing Interests

None of the authors have any competing interests.

Acknowledgements

The authors acknowledge the support provided to the first author from the Raman-Charpak Fellowship by CEFIPRA which made his visit to HPCL, IIT Kanpur possible. This work reports partly the results obtained during the visit.

Author details

[1]High Performance Computing Laboratory, Department of Aerospace Engineering, I. I. T. Kanpur 208 016 Kanpur, India. [2]University of Bordeaux, I2M UMR 5295, Bordeaux, France. [3]Bordeaux Institut National Polytechnique, I2M UMR 5295, Bordeaux, France.

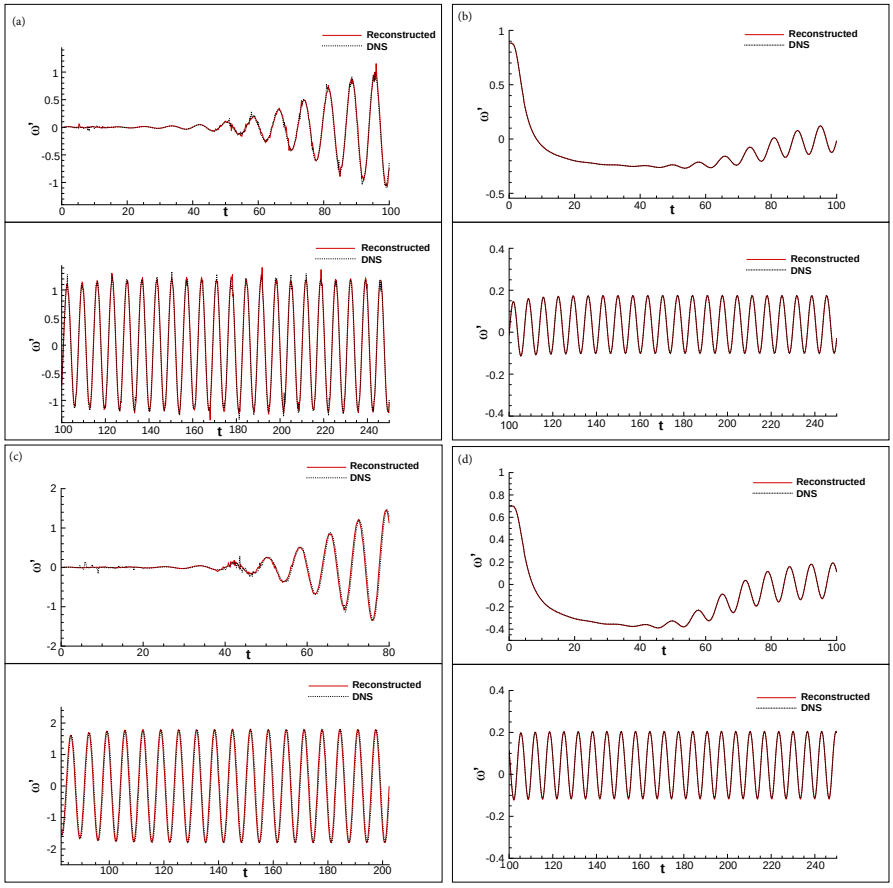


Figure 11 Reconstructed disturbance vorticity with time-scaling interpolation for (a) Re = 70 using Re = 68, 72, and 76 at $(0.504, 0.0)$, (b) at $(1.104, 0.0)$ and (c) Re = 83 using Re = 78, 80, 86 and 90 at $(0.504, 0.0)$ and (d) at $(1.104, 0.0)$. Within each subfigure, the top frame is for comparison at early times, while the bottom frame shows comparison at later times.

References

1. Barrault, M., Maday, Y., Nguyen, N. C. & Patera, A. T. (2004) An 'empirical interpolation' method: Application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, **339**(9), 667–672.
2. Bergmann, M., Cordier, L. & Brancher, J.-P. (2005) Optimal rotary control of the cylinder wake using proper orthogonal decomposition reduced-order model. *Phys. Fluids*, **17**, 097101.
3. Buffoni, M., Camarri, S., Iollo A. and Salvetti, M. V. (2006) Low-dimensional modelling of a confined three-dimensional wake flow. *J.Fluid Mech.*, **569**, 141.
4. Chaturantabut, S. (2011) Nonlinear model reduction via discrete empirical interpolation. Ph.D. Thesis *Rice Univ.*, Houston, Texas
5. Chen, K., Tu, J. H. & Rowley, C. (2012) Variants of dynamic mode decomposition: Boundary condition, Koopman and Fourier analyses. *J. Fluid Mech.*, **656**, 5.
6. Deane, A. E., Kevredekis, I. G., Karniadakis, G. E. & Orszag, S. A. (2002) Low-dimensional models for complex geometry flow: Application to grooved channels and circular cylinders. *Phys. Fluids A.*, **3**, 2337.
7. Fey, U., König, M. & Eckelmann, H. (1998) A new Strouhal-Reynolds-number relationship for the circular cylinder in the range 47<Re $< 2\times 10^5$ *Physics Fluids Lett.*, **10**, 1547.
8. Heiligenthal, S., Dahms, T., Yanchuk, S., Jingling, T., Flunkert, V., Kanter, I., Schöll, E. & Kinzel, W. (2011) Strong and weak chaos in nonlinear networks with time-delayed couplings *Phys. Rev. Lett.*, **107**(23), 234102.
9. Homann, F. (1936) Einfluss grosser zähigkeit bei strömung um zylinder. *Forsch. auf dem Gebiete des Ingenieurwesens*, **7**, 1–10.
10. Holmes, P., Lumley, J. L. & Berkooz, G. (1996) *Turbulence, Coherent Structures, Dynamical Systems and Symmetry.* (Cambridge University Press, U. K.
11. Glansdorff, P. & Prigogine, I. (2012) *Thermodynamic Theory of Structure, Stability and Fluctuation,* (CRC Press, USA).

12. Lestandi, L., Bhaumik, S. Sengupta, T. K., Avatar, G. R. K. C., & Azaiez, M. (2017) POD applied to numerical study of unstudy flow inside lid-driven cavity. *J. Math. Study*, (Under-review).
13. Lestandi, L., Bhaumik, S., Avatar, G. R. K. C., Azaiez, M. & Sengupta, T. K. (2017) Multiple Hopf bifurcations and flow dynamics inside a 2D singular lid driven cavity. *Comput. & Fluid*, (Under-review).
14. Ma, X. & Karniadakis, G. E. (2002) A low-dimensional model for simulating three-dimensional cylinder flow. *J. Fluid Mech.*, **458**, 181.
15. Morzynski, M., Afanasiev, K. & Thiele, F. (1999) Solution of the eigenvalue problems resulting from global nonparallel flow stability analysis. *Comput. Methods Appl. Mech. Engg.*, **169**, 161–176.
16. Noack, B. R., Afanasiev, K., Morzynski, M., Tadmor, G. & Thiele, F. (2003) A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.*, **497**, 335–363.
17. Noack, B. R., Papas, P. & Monkewitz, P. A. (2005) The need for a pressure-term representation in empirical Galerkin models of incompressible shear flows. *J. Fluid Mech.*, **523**, 339–365.
18. Paul-Dubois-Taine, A. & Amsallem, D. (2015) An adaptive and efficient greedy procedure for the optimal training of parametric reduced-order models. *Int. J. Num. Methods Engg.*, **102**, 1262.
19. Rowley, C., Colonius, T. & Murray, R. M. (2004) Model reduction for compressible flows using POD and Galerkin projection. *Physica D.*, **189**, 115.
20. Rowley, C., Mezić, I., Bagheri, S., Schlatter, P. & Henningson, D. S. (2009) Spectral analysis of nonlinear flows. *J. Fluid Mech.*, **641**, 1.
21. Schmid, P. J. (2010) Dynamic mode decomposition of numerical and experimental data. *J. Nonlinear Sci.*, **22**, 887.
22. Sengupta, T. K. (2012) *Instabilities of Flows and Transition to Turbulence.* (CRC Press, USA).
23. Sengupta, T. K. (2013) *High Accuracy Computing Methods: Fluid Flows and Wave Phenomena.* (Cambridge University Press, USA).
24. Sengupta, T. K. & Dey, S. (2004) Proper orthogonal decomposition of direct numerical simulation data of by-pass transition. *Computers & Structures*, **82**, 2693–2703.
25. Sengupta, T. K., Haider, S. I., Parvathi, M. K. & Gumma, P. (2015) Enstrophy-based proper orthogonal decomposition for reduced-order modeling of flow a past cylinder. *Physical Review E.*, **91**(4), 043303.
26. Sengupta, T. K., Lakshmanan, V. & Vijay, V. V. S. N. (2009) A new combined stable and dispersion relation preserving compact scheme for non-periodic problems. *J. Comput. Phys.*, **228**, 3048–3071.
27. Sengupta, T. K., Rajpoot, M. K. & Bhumkar, Y. G. (2011) Space-time discretizing optimal DRP schemes for flow and wave propagation problems. *Computers & Fluids*, **47(1)**, 144–154.
28. Sengupta, T. K., Singh, H., Bhaumik, S. & Chowdhury, R. Roy (2013) Diffusion in inhomogeneous flows: Unique equilibrium state in an internal flow. *Computers and Fluids*, **88**, 440–451.
29. Sengupta, T. K., Singh, N. & Suman, V. K. (2010) Dynamical system approach to instability of flow past a circular cylinder. *J. Fluid Mechanics*, **656**, 82–115.
30. Sengupta, T. K., Singh, N. & Vijay, V. V. S. N. (2011) Universal instability modes in internal and external flows. *Comput. Fluids*, **40**, 221–235.
31. Sengupta, T. K., Vijay, V. V. S. N. & Bhaumik, S. (2009) Further improvement and analysis of CCD scheme: Dissipation discretization and de-aliasing properties. *J. Comput. Phys.*, **228**, 6150–6168.
32. Siegel, S. G., Seidel, J., Fagley, C., Luchtenberg, D. M., Cohen, K. & McLaughlin, T. (2008) Low dimensional modelling of a transient cylinder wake using double proper orthogonal decomposition. *J. Fluid Mech.*, **28**, 1182.
33. Sirovich, L. (1987) Turbulence and the dynamics of coherent structures. Part (I) coherent structures, part (II) symmetries and trans-formations and part (III) dynamics and scaling. *Quart. J. Appl. Math.*, **45**(3), 561–590.
34. Strykowski, P. J. (1986) The control of absolutely and convectively unstable shear flows. *PhD dissertation, Yale University.*
35. Van der Vorst, H. A. (1992) Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems. *SIAM J. Sci. Stat. Comput.*, **12**, 631–644.
36. Weller, J., Lombradi, E., Bergmann, M. & Iollo, A. (2010) Numerical methods for low-order modeling of fluid flows based on POD. *Int. J. Num. Methods Fluids*, **63**, 249.
37. Williamson, C. H. K. (1989) Oblique and parallel modes of vortex shedding in the wake of a circular cylinder at low Reynold numbers. *J. Fluid Mech.*, **206**, 579.

List of Figures

1 Scientific computing workflow enriched with tensor reduction and reduced order modeling . . . 3
2 Separated approximation of an order 3 tensor. . . . 11

1.1 Synthetic view of the procedures described in chapter 1 for model order reduction of bivariate PDEs. The vertical arrows describe the work flow of these techniques and the dotted lines highlight the conceptual differences between them. . . . 14
1.2 Singular Value Decomposition two configurations . . . 15
1.3 Rank k truncated-SVD for both configurations, the shadowed part is dropped upon truncation. $k \leq n,\ k \leq m$. . . . 16
1.4 Decomposition modes of f_3. . . . 34
1.5 Approximation error (L^2 or Frobenius norm) for bivariate methods . . . 35
1.6 A 4000×3000 pixels picture of Singapore Gardens by the Bay compressed through SVD as compared with JPEG compression. . . . 37
1.7 Singular values of `"singapore.tiff"` . . . 38
1.8 Schematic view of the LDC . . . 39
1.9 Vorticity contour of LDC DNS at $Re = 9800$, $t = 1900.2$. The vorticity contour lines are "centered" to $\omega_c = \omega(0.5, 0.5) \approx -1.84$ to emphasize the orbiting triangular structures in the middle, . . . 40
1.10 POD approximation error decay with the number of modes. . . . 41
1.11 The first 8 time modes obtained through POD of the vorticity disturbance field ω' of the LDC for $t \in [1900 : 1940]$. The norm of the couples $\{\ \phi_i, a_i\}$ is stored in a_i which is why they tend toward zero. . . . 42
1.12 The first 8 spatial modes contour obtained through POD of the vorticity disturbance field ω' of the LDC for $t \in [1900 : 1940]$. . . . 42

2.1 A third order tensor with $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$. . . . 48
2.2 The fibres of a third order tensor. . . . 49
2.3 Mode one matricization of third order tensor with $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$. . . . 50
2.4 Binary dimension partition tree of $D = \{1, ..., 5\}$ . . . 55
2.5 Tree representation of the HT format of $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$. The size of the matrices and tensors are inside blue braces. . . . 56
2.6 A graph representation of TT (left) and HT (right) format highlighting their similarities and differences. . . . 58
2.7 "Recursive" dimension tree associated with the extended tensor train of a 5th order tensor . . . 58

2.8 Tree representation of the arrays associated with a 5th order extended TT tensor. Dimensions are given according to HT labeling, first and last dimensions of the transfer tensor correspond to the TT ranks while the middle ones k_j is the "dimension" rank, expected to match the rank of the vector space spanned by the core tensors. 60
2.9 CP decomposition of third order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times J \times K}$. 62
2.10 Tucker Decomposition of a third order array $\boldsymbol{\mathcal{T}}$ 65
2.11 *Truncated* Tucker Decomposition of a third order array $\boldsymbol{\mathcal{X}}$ 65
2.12 A visual of the truncated HOSVD, it shows that the approximation $\breve{\boldsymbol{\mathcal{A}}}$ is defined simultaneously for each direction through least square approximation. Figure from [VVM12]. 69
2.13 A visual of the *sequentially* truncated HOSVD with the same axis as figure 2.12. Here each approximated tensor $\hat{\boldsymbol{\mathcal{S}}}^i$ is performed on one dimension then a new tensor $\hat{\boldsymbol{\mathcal{S}}}^{i+1}$ is processed on the next direction. Processing order is (3,2,1). Figure from [VVM12]. 69

3.1 Visual RPOD rank for 3 parameter function. Blue columns correspond to the coordinates (m, k) where $\tilde{\sigma}_k^m$ is defined while gray crossed areas correspond to coordinates where $\tilde{\sigma}_k^m$ is not defined (not computed). 83
3.2 Example of *a Recursive POD graph* of $f(x_1, x_2, x_3)$ 84
3.3 Visual RPOD rank for 4 parameter function. Red columns correspond to the coordinates (m_1, m_2) where σ_{m_1,m_2} are defined while gray crossed areas correspond to coordinates where they are not defined (not computed) i.e. arbitrarily set to 0. The third dimension (Blue bars) represent defined $\sigma_{m_1=1,m_2,m_3}$, $m_1 \geq 2$ was omitted to simplify the illustration however the same could have been drawn. 86
3.4 RPOD graph obtained for $f(x_1, x_2, x_3, x_4) = \sin(\sqrt{\sum_i x_i^2})$ with a POD cutoff value of 5×10^{-3}. The width of the edges represents their weight. This figure is Zoomable on PDF version. 88

4.1 Dynamic HTML documentaion of the `Fortran low order approximation library`. 98
4.2 Dynamic HTML documentaion of the `pydecomp` build with Sphinx. 99
4.3 Decomposition of 3 test functions with $d = 3$ on a 32^3 grid with 5 discretization methods, using L^2 integration and norm. 101
4.4 Decomposition of f_s on a 40^5 grid with L^2 and l^2 scalar products, decomposition error in their relative norm 102
4.5 f_2 decomposition with $d = 3$ to 5 on a 32^d grid with three decomposition methods, using L^2 integration and norm. 104
4.6 Decomposition of synthetic functions f_2 and f_s for unbalanced grid refinements. 105
4.7 Lid Driven Cavity Simulation within the stable limit cycle time range, see 5, input tensor is of shape $6 \times 201 \times 66049$. $t = 1900$ to 1940 with a stepping of 0.2, space is a 257×257 grid that can be **vectorized** (solid lines) i.e. taken as a long vector of size 66049. Space treated as 2 dimension is referred as **reshaped** (dashed lines). Reynolds is a parameter dimension with $Re \in [10000, 10100]$ and a stepping of 20. 108

4.8 Vorticity field of the lid driven cavity at Re=10000, t=1900s decomposition is reconstructed compared with 1% relative error in Frobenius norm i.e. rank=(10,10,3) and compared to original dataset. Isolines are plotted as well as colormap, they are exponentially spaced from the center of the square value, solid is superior to $\omega(C)$ while dashed lines are inferior. This is to make comparison with centered data. 110
4.9 Time and Reynolds modes of lid driven cavity during limit cycle ($t \in [1900, 1905]$) in for $Re \in [10000, 10100]$. 111
4.10 Visualization of 4 snapshots of the density field at the 21^{st} tabulated wavelength. Data kindly provided by C. Pradère (I2M Bordeaux). 112
4.11 Decomposition of experimental data kindly provided by C. Pradère (I2M Bordeaux). The density is given as a function of time, wavelength and space 112
4.12 First 5 modes study for droplets evaporation experiment obtained through ST-HOSVD. 113
4.13 Synoptic view of reconstructed decomposition, tolerance, $\varepsilon = 10^{-2}$, wavelength 21. Each line represent a different dataset, namely: original, ST-HOSVD reshape, ST-HOSVD vectorized, difference ST-HOSVD reshape, difference ST-HOSVD vectorized. 114
4.14 Breaking wave simulation computed with notus CFD code, wave height of 9cm and length of 10cm. The wave is going rightward from the initial state (left frame), crosses the periodic boundary (top right), breaks at $t \approx 4500$ follows to an unphysical chaotic state. Pink lines represent the water/air interface, arrows size are proportional to the velocity amplitude and the colormap accounts for kinetic energy. 115
4.15 Compression of breaking wave simulation data from `notus`. Parameters : 5 output variables, 3 wave heights, $n_t = 201$, $n_x = 256$, $n_x = 256$. Top frames are decomposition with output variable taken as an additional dimension with $n = 5$, bottom frames is the same dataset but each variable is seen as a seperate scalar decomposition problem. 117
4.16 The first vorticity modes for separated variables layout. 119
4.17 Levelset 50 of the reconstructed density field at 4 time steps (same as Fig. 4.14) with the difference field between the original and reconstructed data. 120
4.18 Reconstructed vorticity and DNS vorticity field for target Re=10040 with donor points at Re = 10000, 10020, 10060, 10080, 100000 at $t = 1900.199$. . 123

5.1 Schematic view of the LDC 134
5.2 The vorticity time series at the sampling point ($x = 0.95, y = 0.95$) obtained for $Re = 8800$ with vorticity contour plots shown at the indicated time instants. Solution of Navier-Stokes equation is obtained using (257×257) grid. 136
5.3 The vorticity time series for a point located at $x = 0.95$, $y = 0.95$, near top right corner for the displayed Reynolds numbers, obtained from solution of unsteady Navier- Stokes equation. 137
5.4 Vorticity contour is shown at different time instants from start till attainment of limit cycle for $Re = 10300$. Time series of the vorticity at point $(0.95, 0.95)$ is shown in the center. 138
5.5 Multiple Hopf-bifurcation shown with respect to the vorticity time series data shown for Fig. 5.2. All the simulated Reynolds numbers data are used to plot the amplitude of the final stable limit cycle data against Reynolds number. 139

5.6 The frequency spectrum of the vorticity time series shown for all the simulated Reynolds numbers, for the solution obtained from unsteady Navier-Stokes equation and the data are for $x = 0.95$ and $y = 0.95$. 141
5.7 Numerical sensitivity of the singular LDC problem. 141
5.8 Variation of the equilibrium amplitude (A_e) with Reynolds number (Re) for the two grids, with (257×257) and (513×513) points. Note the points (P_1, P_2) and (Q_1, Q_2) have similar dynamics, as shown later. Additional points O and S represent the onset of unsteadiness ($Re = 8670$) and secondary instability ($Re = 9800$) of the flow field computed using (257×257) grid points. 144
5.9 Cumulative enstrophy plots for the two grids shown for the indicated Re. . 145
5.10 Eigenfunctions of POD modes for $Re = 9700$ at points (P_1, P_2) in Fig. 5.8. $(\varphi_m)_m$ isolines are plotted in the $[-0.5, 0.5]$ range with 0.01 spacing. Solid lines are positive values, while dashed lines are negative value contour. . . 146
5.11 Amplitude of POD modes and its DFT for $Re = 9700$ for P1 and P2. . . . 147
5.12 Eigenfunctions of POD modes at points Q_1 (Re=10000) and Q_2 (Re=10700).$(\varphi_m)_m$ isolines are plotted in the $[-0.5, 0.5]$ range with 0.01 spacing. Solid lines are positive values, while dashed lines are negative value contour. 148
5.13 Amplitude of POD modes and its DFT for Q1 (Re=10000) and Q2 (Re=10700).149
5.14 Eigenfunctions of POD modes for $Re = 8670$ obtained with (257×257) grid for the point O in Fig. 5.8 during the linear instability stage. 151
5.15 Amplitude functions for point 'O' linear instability grows. They are regrouped by pairs for regular modes while anomalous modes are shown alone.152
5.16 Time series for $Re = 9800$ obtained with (257×257) grid. 152
5.17 Eigenfunctions of POD modes for $Re = 9800$ obtained with (257×257) grid during (a) $t = 500$ to 600 before and during (b) $t = 1900$ to 2000 after the secondary instability. 153
5.18 Amplitude of POD modes and its DFT for $Re = 9800$ using (257×257) grid (a) before [$t = 500$ to 600] and (b) after [$t = 1900$ to 2000] the secondary instability, for the case of Fig. 5.17 154

6.1 DNS time series and their associated DFT's are shown for (a) the flow inside a LDC and (b) the external flow past a cylinder, at indicated points in the flow. In (b), left frames are time series at different wake points and right frames are the associated spectra. 159
6.2 Vorticity field plot for the flow past a circular cylinder showing von Karman street at Re=75. Courtesy of T.K. Sengupta [SSS10]. 160
6.3 Direct Lagrange interpolation of DNS vorticity disturbance time series between Re causes wave packets in the cylinder wake at point (0.504,0.0). . . 161
6.4 Variation of Equilibrium amplitude of disturbance vorticity with Re indicating the segments of Re with respect to bifurcation sequences for (a) flow in LDC and (b) for flow past a cylinder. 162
6.5 Schematic view of the time-scaling algoritmm (Algorithm 13). The black line is the base signal ω_s while the red line is another donor signal ω_s that is transformed by the algorihtm. 164
6.6 Variation of disturbance vorticity at a point (0.504, 0.0) with t_b and t_s for Re_b and Re_s, respectively, for the pair of $Re_b = 80$ and $Re_s = 86$ in the bifurcation sequence $78 \leq Re \leq 90$. 164
6.7 Schematic view of the time scaling interpolation method. 165

6.8 Frequency variation for Re=[8700,12000] for the first three leading frequencies of the vorticity time series at point $(0.95, 0.95)$ obtained for the last 50 periods. The dotted lines indicate the presence of multiple dominating peaks in the spectrum. 166
6.9 Reconstructed vorticity (solid lines) and DNS vorticity (dotted lines) field for target Re=10040 with donor points at Re = 10000, 10020, 10060 and 10080. 167
6.10 Reconstructed vorticity (solid lines) and DNS vorticity (dotted line) field for target Re = 9600 with donor points $\{9350, 9500, 9800$ and $10000\}$. . . . 168
6.11 Variation of t_0 with Re_s for $Re_b = 80$ for Case I (solid line) and Case V (dashed line) of Table 6.2. Shown in parametric form are the pair of Reynolds number and corresponding optimal t_0. 170
6.12 Reconstructed disturbance vorticity with time-scaling interpolation for (a) Re = 70 using Re = 68, 72, and 76 at $(0.504, 0.0)$, (b) at $(1.104, 0.0)$ and (c) Re = 83 using Re = 78, 80, 86 and 90 at $(0.504, 0.0)$ and (d) at $(1.104, 0.0)$. Within each subfigure, the top frame is for comparison at early times, while the bottom frame shows comparison at later times. 171
6.13 Time scaling applied to first time mode, target Re=10040 donor points at Re = 10000, 10020, 10060 and 10080. No back scaling applied. 172
6.14 Lagrange spatial interpolation of target Re=10040 donor points at Re = 10000, 10020, 10060 and 10080. 173

List of Tables

1.1 Numerical orthonormality of the snapshot POD basis for f3 35
1.2 Comparison of POD, PGD and SVD for a target error of $\epsilon = 10^{-6}$. 36
1.3 Compression rate using SVD on 4000×3000 pixels grayscale image. Where CR is the compression rate and the error is computed with Frobenius norm. 38

2.1 Synoptic table of tensor formats 60

3.1 Operation count at each step of the RPOD algorithm. 89

4.1 CPU times on f_v for $n = 40$, $d = 5$ with a tolerance of $\epsilon = 10^{-12}$ 103
4.2 LDC decomposition ranks with the same prescribed cutoff value $\epsilon = 10^{-4}$ (last point in Fig. 4.7b). 109
4.3 Breaking ST-HOSVD ranks with the same prescribed cutoff value $\epsilon = 10^{-3}$ (last point in Fig. 4.15d). 118
4.4 Reconstruction RMS error of the interpolated ROM as 122

5.1 Coefficients of linear regression equation of the form : $A_e^2 = aRe + b$ with regression correlation coefficient R. 139

6.1 Scaling constant and base Re_b for different ranges of Re_s 163
6.2 RMS Error estimates of interpolation for Re = 83 169

Résumé

L'explosion de la puissance de calcul des ordinateurs ne répond à tous les besoins de la communauté scientifique. En particulier, en ingénierie des fluides et structures, la conception demande des simulations toujours plus précises et nombreuses. Cela génère des quantités de données très importante qui sont largement supérieures au téraoctet. Ainsi on se propose d'explorer de nouvelles techniques de compression de données adaptées à la physique pour représentée de façon approchée les données. Ces méthodes de séparation des dimensions d'un problème permettent une réduction de la taille des données de plusieurs ordres de grandeur pour une erreur inférieur à 1%. De plus elles apportent une nouvelle analyse de la physique qui permet de construire des modèles dit d'ordre réduit. Ces modèles sont très économes en temps de calcul (quelques secondes) en échange d'une faible erreur. Un nouveau modèle d'interpolation est présenté ici pour des cas tests de la mécanique des fluides.

Mots-clés : Réduction de données, réduction de modèle, MOR, POD, Cavitée entrainée, HOSVD, Tensor train, tenseurs, formats tensoriels, approximation de tenseurs, interpolation physique, approximation de rang faible.

Abstract

Rocketing computing power is not sufficient to fulfill the scientific community needs. In particular, design in fluid and structures engineering field requires ever more numerous and precise numerical simulations. It generates colossal amounts of data that largely exceeds terabytes. Thus, we propose to explore new data compression techniques suited for physics in order to provide approximated representations of the data. These problem dimension separation methods enable reducing the data size by orders of magnitude while keeping the error below 1%. Moreover they provide a new physics analysis tool that allow construction of reduced order models. They offer very low computing time (a few seconds) by slightly reducing accuracy. A new interpolation model is presented for fluid dynamics test cases.

Key words: Data reduction, Model Reduction, MOR, POD, lid driven cavity, Low rank approximation, tensors, HOSVD, Tensor train, tensor formats, tensor approximation, physics interpolation, time-scaling.