Advanced model order reduction and data-driven technologies enabling physics-augmented digital twins

Victor Champaney

HESAM UNIVERSITÉ

Arts et Métiers Sciences et Technologies

ECOLE DOCTORALE SCIENCES ET METIERS DE L'INGENIEUR

Laboratoire Procédés et Ingénierie en Mécanique et Matériaux (PIMM)

Campus de Paris

THÈSE

Présentée par **Victor CHAMPANEY**

Soutenue le 15 septembre 2023

Pour obtenir le grade de **Docteur d'HESAM Université**

Préparée à **l'Ecole Nationale Supérieure d'Arts et Métiers**

Spécialité : Mécanique-matériaux

Réduction de modèle avancée et approches basées sur les données pour la construction de jumeaux numériques augmentés par la physique

Thèse dirigée par :

M. Francisco CHINESTA

et co-encadrée par :

M. Charbel FARHAT

		Jury
M. Yves REMOND	Professeur des universités, Université de Strasbourg	Président
M. Pierre-Alain BOUCARD	Professeur des universités, ENS Paris-Saclay	Rapporteur
M. Antonio HUERTA	Professeur des universités, Universitat Politecnica de Catalunya	Rapporteur
M. Francisco CHINESTA	Professeur des universités, Ecole nationale supérieure d'Arts et Métiers	Directeur de thèse
M. Charbel FARHAT	Professeur des universités, Stanford University	Co-encadrant
Mme Véronique VENTOS	Maître de conférences, Université Paris-Sud	Examinatrice
Mme Fatima DAIM	Docteur/Ingénieur, ESI Group	Invitée
Mme Véronique FAVIER	Professeur des universités, Ecole nationale supérieure d'Arts et Métiers	Invitée

THÈSE

This thesis is dedicated to
my parents and Maurine.

Remerciements

J'aimerais tout d'abord remercier Paco Chinesta pour toutes les belles opportunités qu'il m'a apportées, pour la confiance qu'il m'a accordée, pour la quantité incroyable de choses que j'ai apprises à son contact et pour tous les bons moments que nous avons passés ensemble. J'espère que nous pourrons continuer à collaborer encore longtemps et accomplir de belles choses. Je remercie également Charbel Farhat, bien que je n'aie pas pu aller en Californie, j'ai beaucoup apprécié et appris de nos échanges.

Merci à ESI Group de m'avoir fait confiance pour travailler au sein de la chaire et collaborer avec les équipes. Je remercie Anoop, Morgan, Satya, Krishna, Bhushan et tous les autres avec qui j'ai travaillé pendant ces trois années. Mention spéciale à Fatima et Mustapha, pour leur gentillesse infinie et tous les agréables moments et superbes projets que nous avons partagés.

Je remercie aussi l'équipe de CNRS@CREATE et du projet DesCartes de m'avoir accueilli à Singapour, que ce soit pour l'accueil chaleureux, l'aide administrative ou pour les travaux que nous avons menés ensemble. Merci particulièrement à Jean-Yves Marzin pour les échanges sur les PINNs et à Dominique Baillargeat pour sa confiance et les opportunités qu'il m'a offertes.

Je souhaite aussi remercier tous les professeurs avec qui j'ai pu collaborer et apprendre durant ma thèse, en particulier Amine Ammar, Elías Cueto, Chady Ghnatios, Xavier Kestelyn et Fodil Meraghni.

Merci à Nicolas et Simon, qui m'ont accompagné et guidé tout au long de ma thèse et avec lesquels j'ai accompli de nombreux projets. Merci à mes collègues et amis doctorants et post-docs qui m'ont motivé à avancer et avec qui nous avons eu des discussions enrichissantes tout au long de la thèse. Je pense particulièrement à Abel, Sevan, Sergio, Sebastian, et surtout à Angelo et Daniele pour tout ce que nous avons fait ensemble notamment dans les projets liés à ESI Group et DesCartes.

Je remercie par-dessus tout Maurine d'avoir partagé le quotidien de la thèse avec moi, et de m'avoir soutenu à chaque instant même pendant les moments les plus difficiles.

Enfin, merci à ma famille et mes amis d'avoir toujours cru en moi.

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to Paco Chinesta for the countless opportunities he has provided me with, the trust he has bestowed upon me, the immense knowledge I have gained from him, and the wonderful experiences we have shared. I hope to continue our collaboration for a long time and achieve remarkable accomplishments together.

I would also like to extend my appreciation to Charbel Farhat. Although I was unable to visit California, I deeply value and have learned from our exchanges.

My sincere thanks go to ESI Group for entrusting me to work within the Chair and collaborating with their exceptional teams. I am grateful to Anoop, Morgan, Satya, Krishna, Bhushan, and all the other individuals I have worked with over the past three years. I want to give a special mention to Fatima and Mustapha for their boundless kindness and the delightful moments and remarkable projects we have shared.

I also want to express my gratitude to the CNRS@CREATE team and the DesCartes project for warmly welcoming me to Singapore, providing administrative assistance, and collaborating on joint work. I extend my heartfelt thanks to Jean-Yves Marzin for our exchanges on PINNs and to Dominique Baillargeat for his trust and the opportunities he has offered me.

Furthermore, I would like to thank all the teachers with whom I had the privilege to collaborate and learn from during my thesis, particularly Amine Ammar, Elías Cueto, Chady Ghnatios, Xavier Kestelyn, and Fodil Meraghni.

I am grateful to Nicolas and Simon for their constant guidance and support throughout my thesis journey and for the numerous projects we accomplished together. I am also thankful to my doctoral and post-doctoral colleagues and friends who inspired me to keep pushing forward and engaged in enriching discussions throughout the thesis. I would like to mention Abel, Sevan, Sergio, Sebastian, and especially Angelo and Daniele for the valuable collaborations we had, particularly in projects related to ESI Group and DesCartes.

Above all, I want to express my deepest appreciation to Maurine for sharing the daily ups and downs of the thesis journey with me and for her support during even the most challenging times.

Lastly, I would like to thank my family and friends for always believing in me.

Advanced model order reduction and data-driven technologies enabling physics-augmented digital twins

Abstract

In the 20th century, engineering made remarkable strides in various fields, while other disciplines turned to data for diagnostic and prognostic purposes. Recognizing the potential of data and AI, engineering sciences have embraced these technologies to make better predictions, enhance performance, and gain a deeper understanding of complex systems. This has given rise to the digital twin paradigm.

The challenge lies in developing accurate models that can predict output based on input data. Choosing between physics-based and data-driven approaches in engineering can be difficult. Physics-based approaches offer advantages but are computationally intensive and struggle with large-scale systems and uncertainty. They often require the use of Model Order Reduction techniques [1–3] to become adequate with the real-time operation constraints. Meanwhile, data-driven approaches are promising when accurate models are unavailable, yet they face issues related to data costs [4–7], extrapolation risks [8, 9], and lack of explanations and certifications [10, 11].

Therefore, combining both approaches seems to be the optimal choice as it strikes a balance between their advantages and disadvantages. By integrating physics-based and data-driven perspectives, we can leverage the strengths of each approach to achieve better engineering outcomes.

One significant advantage of this alliance is the reduction in data requirements for model building. This reduction is achieved by leveraging known laws or focusing on simplifying the approximation of the gap between model and reality. Moreover, using physics-based models allows for explanations of fundamental aspects and resulting predictions, enabling certification of results and mitigating extrapolation issues.

Furthermore, the reduction in data requirements is not solely attributed to the use of physics but also to how physics can guide the selection of optimal locations and times for data collection. This is particularly evident in active learning [12], where incorporating existing physics-based knowledge can enhance its effectiveness.

By combining physics-based understanding with data-driven techniques, engineers can harness the power of both approaches, leading to more efficient and insightful engineering practices.

In this thesis, divided into three chapters, we delve into the tools, methodologies, and personal contributions that are crucial components in the construction of hybrid twins. Chapter 1 provides a review of methods enabling hybrid twins, including intrusive and non-intrusive Model Order Reduction (MOR) techniques and Physics-Informed Neural Networks (PINN) [13]. Chapter 2 explores advances in non-intrusive Model Order Reduction, focusing on regularization of separated representations, the novel ANOVA-PGD strategy based on ANOVA (Analysis of variance) [14, 15], reduced modeling of curves, and the use of Optimal Transport [16, 17] in reduced order modeling. Finally, Chapter 3 delves into data assimilation and hybrid modeling, discussing modeling systems from partial observations and presenting hybrid modeling approaches based on learning a source term and identifying properties.

By combining these chapters, this thesis contributes to the development, improvement, and application of methodologies that enable the construction of hybrid twins. These methodologies bridge the gap between data science and numerical simulation, addressing current industrial challenges. They offer high-fidelity and high-dimensional parametric models, accelerate physics-based models through MOR and data-driven techniques, tackle challenges in model construction under partial observability, and create hybrid models that combine physics-based principles with data. Through these contributions, this thesis aims to advance the field of engineering by leveraging the power of data and simulation to address complex real-world problems.

Réduction de modèle avancée et approches basées sur les données pour la construction de jumeaux numériques augmentés par la physique

Résumé

Au 20e siècle, l'ingénierie a fait des progrès remarquables dans divers domaines, tandis que d'autres disciplines se sont tournées vers les données à des fins de diagnostic et de pronostic. Reconnaissant le potentiel des données et de l'IA, les sciences de l'ingénieur ont adopté ces technologies pour faire de meilleures prédictions, améliorer les performances et mieux comprendre les systèmes complexes. C'est ainsi qu'est né le paradigme du jumeau numérique.

Le défi consiste à développer des modèles précis capables de prédire des résultats en fonction de données d'entrée. Dans le domaine de l'ingénierie, il peut être difficile de choisir entre les approches basées sur la physique et les approches basées sur les données. Les approches basées sur la physique offrent des avantages, mais elles sont gourmandes en calculs et ont du mal à gérer les systèmes à grande échelle et l'incertitude. Elles nécessitent souvent l'utilisation de techniques de réduction de modèle [1–3] pour devenir adéquates avec les contraintes de fonctionnement en temps réel. En parallèle, les approches basées sur les données sont prometteuses lorsque des modèles précis ne sont pas disponibles, mais elles sont confrontées à des problèmes liés au coût des données [4–7], aux risques d'extrapolation [8, 9], et au manque d'explicabilité et de certifications [10, 11].

Par conséquent, la combinaison des deux approches semble être le choix optimal, car elle permet de trouver un équilibre entre leurs avantages et leurs inconvénients. En intégrant les perspectives basées sur la physique et les données, nous pouvons tirer parti des forces de chaque approche pour obtenir de meilleurs résultats en matière d'ingénierie.

Un avantage significatif de cette alliance est la réduction des besoins en données pour la construction de modèles. Cette réduction est obtenue en exploitant des lois connues ou en se concentrant sur l'approximation de l'écart entre le modèle et la réalité. En outre, l'utilisation de modèles basés sur la physique permet d'expliquer les aspects fondamentaux et les prédictions qui en résultent, ce qui permet de certifier les résultats et d'atténuer les problèmes d'extrapolation.

En outre, la réduction des besoins en données n'est pas seulement une conséquence

de l'utilisation des lois physiques, mais aussi de la manière dont la physique peut guider la sélection des lieux et des instants optimaux pour la collecte des données. Ceci est particulièrement évident dans l'apprentissage actif [12], où l'incorporation de connaissances existantes basées sur la physique peut améliorer son efficacité.

En combinant la compréhension basée sur la physique avec des techniques basées sur les données, les ingénieurs peuvent exploiter la puissance des deux approches, ce qui conduit à des pratiques d'ingénierie plus efficaces et plus perspicaces.

Dans cette thèse, divisée en trois chapitres, nous nous penchons sur les outils, les méthodologies et les contributions personnelles qui sont des éléments cruciaux dans la construction de jumeaux hybrides. Le chapitre 1 passe en revue les méthodes permettant d'obtenir des jumeaux hybrides, notamment les techniques intrusives et non intrusives de réduction de modèle (MOR) et les réseaux neuronaux informés par la physique (PINN) [13]. Le chapitre 2 explore des évolutions en réduction de modèle non intrusive, en se concentrant sur la régularisation des représentations séparées, une nouvelle stratégie ANOVA-PGD basée sur l'ANOVA (analyse de la variance) [14,15], la modélisation réduite des courbes, et l'utilisation du transport optimal [16,17] dans la construction de modèles réduits. Enfin, le chapitre 3 traite de l'assimilation de données et de la modélisation hybride, en discutant de la modélisation de systèmes à partir d'observations partielles et en présentant des approches de modélisation hybride basées sur l'apprentissage d'un terme source et l'identification de propriétés.

En combinant ces chapitres, cette thèse contribue au développement, à l'amélioration et à l'application de méthodologies qui permettent la construction de jumeaux hybrides. Ces méthodologies comblent le fossé entre la science des données et la simulation numérique, en répondant aux défis industriels actuels. Elles offrent des modèles paramétriques de haute fidélité et de haute dimension, accélèrent les modèles basés sur la physique grâce à la réduction de modèle et l'apprentissage machine, s'attaquent aux défis de la construction de modèles sous observabilité partielle, et créent des modèles hybrides qui combinent des principes basés sur la physique avec des données. Grâce à ces contributions, cette thèse vise à faire progresser le domaine de l'ingénierie en tirant parti de la puissance des données et de la simulation pour résoudre des problèmes complexes du monde réel.

Résumé étendu

Contexte et motivations

Au 20e siècle, l'ingénierie s'est fortement appuyée sur des modèles pour concevoir des composants et des systèmes de manière efficace. Ces modèles utilisaient des équations algébriques ou des équations aux dérivées partielles pour décrire les relations entrée/sortie dans l'ingénierie. Alors que les modèles algébriques étaient faciles à manipuler et à résoudre, les modèles basés sur des équations étaient de plus grands défis, nécessitant des techniques de discrétisation telles que celle des éléments finis pour donner au problème une forme résoluble.

La discrétisation consiste à résoudre le problème en un nombre limité de points, aussi appelés nœuds, et d'instants, au lieu d'essayer de le résoudre pour un nombre infini de points et d'instants. Le nombre d'inconnues du problème augmente avec la dimension de l'espace, ce qui entraîne une croissance exponentielle de la complexité. L'utilisation des ordinateurs et calculateurs a rendu la discrétisation possible, mais les problèmes à grande échelle exigent toujours un temps de calcul important, nécessitant des jours, des semaines, voire des mois pour être résolus.

Ces méthodes ont permis l'avènement du paradigme de l'ingénierie basée sur la simulation [18, 19]. Ces techniques [20, 21] regroupent notamment la méthode des éléments finis [22, 23], la méthode des différences finies [24, 25] et la méthode des volumes finis [26, 27].

L'ingénierie basée sur la simulation requiert de multiples calculs pour explorer l'espace paramétrique et identifier les paramètres de conception optimaux. Toutefois, le coût des simulations et la dimension élevée de l'espace paramétrique restent des défis permanents. En outre, la qualité des modèles paramétriques en ingénierie est affectée par des facteurs tels que la taille du système analysé, le grand nombre et la variabilité des paramètres impliqués et l'incertitude inhérente à l'écart entre le modèle et la réalité.

Pour relever ces défis, les ingénieurs utilisent diverses techniques et stratégies pour réduire les sources d'incertitude et garantir des prédictions précises pour des conceptions fiables.

De l'ingénierie aux jumeaux numériques

Le 20e siècle a été le témoin de réalisations remarquables [28] dans le domaine de l'ingénierie, notamment de progrès dans l'aéronautique, l'exploration spatiale, les transports, l'énergie et les infrastructures civiles. Pendant ce temps, des domaines tels que le marketing, l'économie, les sciences sociales et la médecine se sont orientés vers l'utilisation des données à des fins de diagnostic et de pronostic, motivant de fortes améliorations technologiques liées à la science des données. C'est ainsi que l'ingénierie a connu trois évolutions majeures :

- L'extension de l'ingénierie à l'utilisation : Avec l'avènement de technologies telles que l'Internet des objets (Internet of Things - IoT) et l'abondance des données, les fastidieuses méthodes traditionnelles d'ingénierie en amont ont été en partie remplacées par des opérations en temps réel. L'intérêt est passé d'une ingénierie basée sur le produit à une ingénierie basée sur le service, accentuant les prévisions en continu et le maintien de la performance tout au long de la durée de vie d'un produit.

- L'ingénierie à l'ère des systèmes connectés : L'ingénierie a élargi son champ d'action des composants individuels aux systèmes interconnectés, ce qui a conduit à des concepts tels que les industries, les villes et les nations intelligentes. Cette évolution a marqué la quatrième révolution industrielle, dans laquelle les données se sont intégrées aux révolutions précédentes, qui étaient alimentées par la vapeur, l'électricité, l'électronique et l'automatisation.
- La prochaine révolution de l'ingénierie immersive centrée sur l'homme : La cinquième révolution, qui est déjà en marche, place l'homme au cœur de l'ingénierie, envisageant un métavers sensible à la physique. Les progrès réalisés dans le domaine de la vision par ordinateur, des dispositifs multimodaux et du traitement du langage naturel pourraient permettre de dépasser le test de Turing. Cependant, l'intégration de l'homme dans le système pose des défis (notamment d'interfaces homme-machine et de calcul temps réel) aux approches de modélisation traditionnelles.

En réponse à ces développements, les sciences de l'ingénieur se sont intéressées aux possibilités offertes par les données et l'IA, dans le but d'obtenir de meilleures prédictions, d'améliorer les performances et de mieux comprendre les systèmes complexes [29]. L'apprentissage automatique, en particulier, offre un processus d'apprentissage général dans lequel la relation entre les entrées et les sorties peut être établie à partir des données collectées, ce qui permet l'application en temps réel des modèles appris. Ces nouvelles pratiques se placent dans le paradigme du jumeau numérique.

Le principal défi consiste à obtenir des modèles de régression précis qui prédisent la sortie en fonction de l'entrée. Il existe différentes techniques, qui dépendent de facteurs tels que la quantité de données, le temps d'entraînement et la qualité des données. Le choix entre ces techniques permet de répondre à des problématiques telles que l'obtention de données [4, 5], le coût des données entraînant un besoin de travailler avec peu de données [6, 7], l'interprétabilité du modèle [10, 11] et l'extrapolation du modèle [8, 9]. Bien que la modélisation basée sur les données présente des avantages, de nombreux défis persistent, notamment la difficulté et le coût de la collecte de données, l'interprétabilité des modèles et l'extrapolation en dehors des conditions observées.

Pour relever ces défis, il faut examiner attentivement les stratégies de collecte de données, l'interprétabilité et l'utilisation appropriée des modèles dans leurs domaines de validité.

Entre la physique et les données : les modèles hybrides

Dans le domaine de l'ingénierie, le choix entre les approches fondées sur la physique et les approches fondées sur les données est difficile. Les approches fondées sur la physique offrent des avantages, mais elles sont gourmandes en calculs et ont du mal à gérer les systèmes à grande échelle et l'incertitude. Les approches fondées sur les données sont prometteuses en l'absence de modèles précis, mais elles se heurtent à des difficultés telles que le coût élevé des données, les risques d'extrapolation et le manque d'explications et de certifications.

La combinaison des deux approches [30, 31] semble donc être le choix optimal, puisqu'il offre un bon compromis entre les avantages et inconvénients de chacune. En combinant les approches fondées sur la physique et sur les données, nous pouvons exploiter les points forts de chaque approche et obtenir de meilleurs résultats en matière d'ingénierie.

Cette alliance présente l'avantage de réduire considérablement les besoins en données pour la construction de modèles. Cette réduction est obtenue en exploitant les lois de la physique connues, ou en se concentrant sur l'écart entre le modèle et la réalité, qui est censé être plus simple à approximer que le comportement réel. En outre, l'utilisation de modèles basés sur la physique permet d'expliquer les aspects fondamentaux du modèle et les prédictions qui en résultent, ce qui permet de certifier les résultats, et parfois même d'atténuer les problèmes d'extrapolation.

La réduction des besoins en données n'est pas seulement permise par l'utilisation des modèles physiques, mais aussi au fait que la physique peut informer le choix des emplacements et instants optimaux de collecte des données. Cela est particulièrement évident dans le cadre de l'apprentissage actif, que l'incorporation des connaissances existantes fondées sur la physique peut rendre plus efficace.

Pour illustrer cela, prenons l'exemple de la mesure de la température dans une ville. Au lieu de placer de nombreux thermomètres dans chaque rue et d'enregistrer la température en permanence, la connaissance de la physique et le bon sens nous indiquent que quelques thermomètres placés stratégiquement, par exemple un dans chaque quartier, et des mesures à quelques moments précis comme le matin, le midi, l'après-midi et la nuit suffisent généralement à décrire la distribution spatiale et l'évolution de la température.

En conclusion, il est bénéfique de tirer le meilleur parti des connaissances existantes en intégrant la compréhension de la physique et les approches basées sur les données dans les projets d'ingénierie.

Réduction de modèle : physique en temps réel

Les techniques visant à améliorer la résolution des modèles basés sur la physique afin d'obtenir des solutions en temps réel appartiennent au domaine de la réduction de modèle (Model Order Reduction - MOR) [1–3].

Il en existe deux déclinaisons :

- La première consiste à construire un modèle d'ordre réduit, dit intrusif, en appliquant des techniques de réduction de la dimensionnalité à la solution du modèle physique. Ce mécanisme est intégré au processus de résolution, et permet au solveur de fonctionner rapidement.

- La seconde solution consiste à construire un modèle de substitution, qui établit une relation directe entre la solution du modèle physique et les paramètres du modèle. Une fois le modèle de substitution construit, il permet de déduire la solution quasiment en temps réel lorsque les paramètres du modèle sont fournis.

Les modèles d'ordre réduit et les solutions paramétriques jouent un rôle crucial dans la réalisation de diverses tâches, notamment la simulation, l'optimisation, l'analyse inverse, le contrôle basé sur la simulation et la propagation d'incertitudes, tout en respectant des contraintes strictes en matière de temps réel. Ces solutions paramétriques offrent un moyen exceptionnellement efficace et sans précédent pour effectuer de l'ingénierie basée sur des modèles en temps réel.

Données et apprentissage

Les techniques de régression, qui permettent d'extraire des modèles des données, peuvent être appliquées à différents types de données. Parmi les types de données les plus courants, on peut citer :

- Listes (tableaux) : données organisées en tableaux, constitués de valeurs numériques continues ou discrètes, et parfois des caractéristiques catégorielles ;
- Images : images réelles en 2D ou 3D ou résultats de simulations numériques ;
- Formats tensoriels : images compressées ou données de champs spatiaux décomposées en produits tensoriels ;
- Graphes : caractéristiques de nœuds et d'arêtes représentées par une structure de graphe ;
- Courbes : courbes géométriques ou fonctions à valeur scalaires ;
- Séries temporelles : données ordonnées chronologiquement impliquant une causalité ;
- Nuages de points : densités ou distributions de probabilité.

Plusieurs techniques d'apprentissage automatique sont couramment utilisées pour la régression :

- Régressions polynomiales régularisées [32] : Elles sont utiles lorsque les données sont limitées et que la solution peut être exprimée à l'aide de bases polynomiales. Elles peuvent intégrer des connaissances existantes par la construction de variables adéquates.
- Réseaux neuronaux artificiels (NN) : Ils donnent de bons résultats lorsque les données sont abondantes et que les hyperparamètres sont bien réglés. Ils peuvent approximer n'importe quelle fonction [33], ce qui les rend extrêmement polyvalents.
- Auto-encodeurs (AE) [34] : Ils encodent les données dans un espace latent, en approximant la dimensionnalité intrinsèque et en supprimant les corrélations. D'autres techniques de réduction de la dimensionnalité telles que l'ACP [35], la kPCA [36], le LLE [37] et le tSNE [38] peuvent également être utilisées.
- Réseaux antagonistes génératifs (GAN) [39] : Ils génèrent des échantillons de données synthétiques permettant d'augmenter l'ensemble de données, ou de générer des échantillons réalistes.
- Réseaux neuronaux convolutifs (CNN) [40] : Ils sont bien adaptés à l'analyse d'images et aux tâches de reconnaissance et extraction de motifs. Les réseaux neuronaux en graphes (GNN) [41] étendent les CNN pour qu'ils fonctionnent sur des maillages ou des graphes non structurés.
- Réseaux neuronaux récurrents (RNN) [42] et long short-term memory (LSTM) [43] : Puissants pour traiter les données de séries temporelles. Ils peuvent capturer les dépendances temporelles grâce à des capacités de mémoire.

- Systèmes dynamiques : Diverses techniques sont utilisées pour modéliser les systèmes dynamiques, notamment les RNN, les LSTM, ResNet [44], NeuralODE [45], DMD [46], l'opérateur de Koopman [47], le calcul par réservoir [48] et les DeepONets [49].
- Réseaux neuronaux informés par la physique (PINN) [13], réseaux neuronaux informés par la thermodynamique (TINN) [50] et modèles hybrides : Ils intègrent des connaissances fondées sur la physique dans le processus d'apprentissage, garantissant ainsi la satisfaction des équations fondamentales et combinent des approches fondées sur la physique et des approches fondées sur les données.

Ces techniques offrent diverses options pour les tâches de régression, en fonction de la nature des données et des exigences de modélisation souhaitées.

Modélisation hybride

Dans le contexte de la modélisation hybride [51], il existe deux approches : l'enrichissement de la solution et l'enrichissement du modèle.

Dans le premier cas, la solution du modèle basée sur la physique, obtenue par des techniques spécifiques, est combinée avec une correction basée sur les données. Le modèle est calibré en ligne en identifiant les paramètres opérationnels qui minimisent la différence entre les prédictions du modèle et les mesures disponibles. La prédiction hybride est la combinaison de la solution calibrée basée sur la physique et de la correction basée sur les données. Cette procédure s'applique également aux problèmes transitoires.

Dans le second cas, l'objectif est d'améliorer la capacité prédictive d'un modèle nominal connu. Les mesures sont comparées aux prédictions du modèle nominal et un écart notable est observé. Pour y remédier, un terme d'enrichissement du modèle est introduit afin de mieux représenter les données collectées. Le modèle discret enrichi, ainsi que la prédiction associée, doivent satisfaire à la condition d'équilibre tout en représentant précisément les données. Une paramétrisation de la correction du modèle est choisie, ainsi qu'une technique de régularisation appropriée, pour calculer l'enrichissement du modèle et compléter les données de manière efficace.

Contenu du mémoire

Ce mémoire est divisée en trois chapitres : tout d'abord, quelques outils et méthodologies qui sont des éléments clés dans la construction de jumeaux hybrides sont décrits, puis les contributions personnelles dans le domaine de la réduction non intrusive de l'ordre des modèles sont présentées et enfin, les contributions personnelles à la modélisation hybride sont décrites.

Chapitre 1 : Revisite de méthodes permettant la construction de jumeaux hybrides

Section 1 : Méthodes de réduction de modèle intrusives et non-intrusives

Les problèmes d'ingénierie nécessitent souvent des simulations rapides et précises utilisant des modèles complexes et de grandes quantités de données. Bien que les dévelop-

pements du hardware aient permis d'améliorer la qualité des simulations, la résolution de problèmes en grande dimension reste difficile en raison de la complexité algorithmique et des contraintes de traitement en temps réel. Les techniques de réduction de modèle (MOR), qui peuvent être intrusives ou non-intrusives, ont attiré l'attention pour résoudre ces problèmes.

La décomposition orthogonale aux valeurs propres (POD) [52, 53] extrait les caractéristiques essentielles du système et crée une base d'approximation réduite, ce qui diminue considérablement le temps de calcul. La Proper Generalized Decomposition (PGD) [54, 55] est une méthode intrusive qui utilise la séparation de variables pour réduire la complexité des algorithmes de résolution. La décomposition orthogonale aux valeurs propres avec interpolation (PODI) [56, 57] est une méthode non intrusive qui construit des solutions paramètriques à partir de la base réduite sans nécessiter de modifier le solveur. La sparse PGD (sPGD) [58] construit une solution paramétrique à l'aide d'un ensemble peu dense de solutions calculées par un solveur d'équations aux dérivées partielles en utilisant la séparation de variables pour traiter les problèmes avec beaucoup de paramètres.

Dans l'ensemble, les techniques de réduction de modèle fournissent des solutions efficaces pour des problèmes d'ingénierie complexes nécessitant une application en temps réel.

Section 2 : Réseaux neuronaux informés par la physique

Les réseaux neuronaux informés par la physique (PINN) [13] sont apparus comme un cadre prometteur qui intègre dans les réseaux neuronaux les principes de la physique pour résoudre des problèmes scientifiques et d'ingénierie complexes. Contrairement à l'apprentissage automatique traditionnel, les PINN intègrent des connaissances préalables sur les lois physiques afin de permettre des prédictions précises et de saisir la physique sous-jacente, même avec un nombre limité de données annotées.

Par ailleurs, ces architectures sont un cadre propice aux stratégies hybrides. En effet, il est naturel d'ajouter dans la fonction de perte du réseau neuronal toutes les contraintes que doit respecter la solution. Ces contraintes peuvent provenir de lois physiques, souvent sous la forme d'équations aux dérivées partielles, mais aussi de données mesurées ou encore de mécanismes de régularisation. Pendant l'entraînement, la solution est guidée par le gradient de la fonction de perte vers un compromis entre toutes les contraintes.

Il est important de remarquer que combiner un modèle physique totalement déterminé avec des données expérimentales ne donne pas réellement un modèle hybride mais une solution hybride, qui est très utile dans de nombreux cas d'utilisation mais n'apporte pas de possibilité d'explicabilité ou de généralisation à des problèmes similaires.

Chapitre 2 : Progrès dans la réduction de modèle non intrusive

Section 1 : Regularisation de représentations séparées

Le défi de la régression avec des données limitées consiste à traiter les non-linéarités dans des contextes multiparamétriques, ce qui entraîne la malédiction de la dimensionnalité [59, 60]. Le principe du rasoir d'Occam [61, 62] suggère de privilégier les modèles les plus simples aux modèles les plus complexes, ce qui est possible grâce à la

favorisation de la parcimonie dans la régression [62, 63]. Cette section propose des méthodologies de régression robustes et efficaces pour les représentations séparées. Deux techniques sont présentées : la rs-PGD, qui combine la régularisation des normes L2 et L1, et la s^2-PGD, une méthode de régression doublement parcimonieuse utilisant la régularisation Lasso.

Ces deux techniques reposent initialement sur un mécanisme commun : l'ajout d'un terme de pénalisation lors de la définition des problèmes de minimisation. Suivant l'algorithme de la PGD et plus particulièrement de la sPGD, la solution recherchée est exprimée sous une forme séparée, une somme de produits de fonctions unidimensionnelles, puis chacune de ces fonctions est calculée en résolvant successivement un certain nombre de problèmes unidimensionnels. Dans la s^2-PGD (resp. rs-PGD), un terme pénalisant la norme L1 (resp. une combinaison des normes L1 et L2) est introduit afin de réduire le surapprentissage qui intervient lorsque peu de données sont disponibles.

Par ailleurs, la s^2-PGD ajoute à cette régularisation une étape de sélection des fonctions de base qui permettent une bonne représentation de la solution. Une fois les fonctions adéquates sélectionnées, la solution est recalculée sans pénalisation, mais dans un espace de dimension réduit mais adapté aux données qui permet donc d'éviter le surapprentissage tout en représentant bien la solution.

Les exemples proposés montrent que ces stratégies permettent une amélioration de la capacité prédictive de la sPGD.

Section 2 : ANOVA-PGD : Une stratégie originale

Les techniques présentées dans la section précédente soulèvent de nouveaux défis et de nouvelles questions. Dans le contexte de la s^2-PGD, le choix des fonctions de base a un impact significatif sur la qualité de la solution, contrairement à la sPGD, qui donne des résultats dépendants de l'espace engendré par la base mais indépendant du choix de la base en elle-même. En outre, les méthodes de régression basées sur la PGD utilisent actuellement des plans d'expériences parcimonieux qui n'exploitent pas de technique d'échantillonnage intelligent ; cela laisse de côté un potentiel inexploité. Ces observations motivent le développement d'une nouvelle technique appelée ANOVA-PGD, qui combine les approches de la section précédente avec la décomposition ANOVA (Analyse de la variance) [64–66] et sa version "low-cost" dite "ancrée" [67–69].

La décomposition ANOVA permet d'approximer une fonction de manière hiérarchique grâce à une base orthogonale qui contient des fonctions d'un nombre de variables croissant. A l'ordre 0, la fonction est approchée par une constante puis enrichie à l'ordre 1 par une somme de fonctions unidimensionnelles. Grâce à la stratégie ANOVA "ancrée", le calcul de ces fonctions nécessite un échantillonnage approprié, qui sépare la contribution de chacune des variables. Cet échantillonnage a la particularité de croître linéairement avec le nombre de paramètres impliqués, ce qui en fait une technique très intéressante pour des problèmes en grande dimension.

Par ailleurs, l'étude de ces contributions séparées des différents paramètres permet d'établir des bases construites à partir des données et adaptées à chaque paramètre. Ainsi, l'approximation à l'ordre 1 est combinée avec une technique basée sur la représentation séparée munie de ces bases adaptées afin de construire une solution paramétrique la plus fidèle possible, même dans des cas assez fortement non-linéaires et en très grande dimension.

Section 3 : Modèles réduits de courbes

La prédiction de courbes entières (notamment des fonctions scalaires du temps ou d'un paramètre) par des modèles paramétriques de substitution présente des difficultés. Le non-alignement des courbes, qui peut résulter de fréquences ou fenêtres d'échantillonnages différentes, et le décalage de points critiques dans les courbes présentant des caractéristiques communes (par exemple, la transition élastique-plastique en mécanique), font que les méthodes d'interpolation classiques ne fournissent pas des résultats cohérents du point de vue de la physique ; cela nécessite d'appliquer aux courbes un prétraitement approprié. Par ailleurs, lorsque des bifurcations se produisent dans l'espace paramétrique, le couplage avec des algorithmes de partitionnement (clustering) [70, 71] et de classification [72] améliore la précision du modèle de substitution.

En combinant des traitements des courbes basés sur l'alignement de caractéristiques remarquables avec une stratégie basée sur un clustering en amont et une classification en aval, il est montré que les modèles réduits permettant de prédire des courbes à partir de paramètres donnés sont largement améliorés. Ces méthodes ne sont pas totalement automatiques mais simples à mettre en œuvre pour améliorer drastiquement la qualité des résultats, notamment en s'assurant de la bonne représentation des caractéristiques de courbes qui sont chargées de sens physique.

Ensuite, la propagation d'incertitudes [73] dans ces modèles est étudiée ; supposant que les paramètres d'entrée du modèle sont connus de manière incertaine et modélisés par une loi de probabilité, un intervalle de confiance est calculé pour la courbe de sortie.

Section 4 : Réduction de modèle basée sur le transport optimal

La réduction de modèle dans des contextes tels que celui décrit dans la section précédente devient encore plus difficile lorsque les solutions sont des champs complets en plusieurs dimensions. Les bases réduites linéaires, couramment utilisées dans les algorithmes de réduction, peinent à capturer les déplacements continus des caractéristiques de la solution causés par des phénomènes tels que l'advection ou des chargements paramétriques localisés. Les limites des bases réduites proviennent de leur utilisation d'une métrique euclidienne, qui ne prend pas en compte le désalignement ou la localisation des caractéristiques. Pour y remédier, la théorie du transport optimal fournit des métriques conçues pour détecter les distances entre les caractéristiques remarquables.

Le transport optimal [16, 17] construit notamment une manière d'interpoler entre des fonctions en prenant en compte cette problématique de caractéristiques localisées. Cette interpolation est efficace quand seulement deux fonctions sont concernées, mais plus difficile quand un grand échantillonnage est concerné, particulièrement dans un vaste domaine paramétrique. Pour répondre à cette problématique, une stratégie reprenant l'échantillonnage de l'ANOVA "ancrée" et permettant de visiter l'espace paramétrique de proche en proche est proposée.

Cette combinaison entre l'ANOVA et le transport optimal permet d'agrandir le domaine d'application de la réduction de modèle. Cependant, les outils liés au transport optimal nécessitent des conditions strictes sur les fonctions étudiées, notamment qu'elles soient positives. Pour remédier à ce problème, une deuxième stratégie plus complète est proposée pour étendre encore un peu davantage ce domaine d'application.

Chapitre 3 : Assimilation de données et modélisation hybride

Section 1 : Modélisation à partir d'observations partielles

Cette section aborde la question de l'accessibilité des données dans la modélisation. Lorsque seule une partie d'un système peut être mesurée ou observée, des questions se posent concernant l'existence, la formulation et l'apprentissage d'un modèle qui relie les entrées observables aux sorties correspondantes, en tenant compte de l'influence des dynamiques cachées [74–77].

Dans un premier temps, la question est étudiée dans le cadre de formulations algébriques classiques. En utilisant la condensation statique (ou réduction de Guyan) [78, 79], les conditions selon lesquelles un modèle peut être reconstitué à partir d'observations partielles sont analysées. La même étude est conduite en régime dynamique en appliquant également cette condensation dans l'espace de Fourier.

Les idées développées dans le cadre algébrique simple sont ensuite utilisées pour résoudre un problème similaire mais non-linéaire à l'aide de méthodes d'apprentissage machine adaptées. En effet, les RNN et LSTM sont capables de modéliser la contribution des parties du système qui n'ont pas été observées, dans des conditions similaires à celles établies plus tôt.

Section 2 : Modélisation hybride par apprentissage de terme source

Cette section explore la combinaison de données et d'un modèle basé sur la physique pour créer un modèle hybride. Bien que l'approche PINN offre une solution simple en intégrant des équations physiques dans la fonction de perte, elle ne fournit qu'une solution hybride, et non un modèle hybride. La création d'un modèle hybride offre des avantages tels que la généralisation à d'autres conditions et potentiellement une interprétation physique des corrections du modèle.

Pour enrichir le modèle à partir des données, une inconnue est ajoutée au modèle sous la forme d'un terme source. Dans le cadre des méthodes classiques de discrétisation d'équations aux dérivées partielles, ce terme est appris à partir de l'approximation des opérateurs différentiels appliqués aux données. En revanche, cela n'est possible que lorsque les données sont abondantes puisqu'elles doivent être associées aux nœuds d'un maillage suffisamment fin pour approximer correctement les opérateurs.

Pour atténuer cette contrainte très forte, le problème est reformulé à l'aide d'un PINN. Dans cette nouvelle formulation, les dérivées sont calculées de manière exacte sans contrainte particulière, et la solution peut donc être obtenue à partir de données parcimonieuses.

Section 3 : Modélisation hybride par identification de propriétés

Dans le but d'enrichir les modèles physiques par le biais de la modélisation hybride, une approche alternative émerge : une approche qui suppose que le modèle lui-même est précis mais qui reconnaît l'incertitude entourant les propriétés physiques associées au système étudié. Cette incertitude peut résulter d'erreurs de modélisation ou d'approximations, de l'utilisation de représentations grossières ou de la présence de dommages ou de vieillissement dans le système. Cette section se penche sur ce dernier scénario, en

se concentrant sur le cas spécifique de l'endommagement localisé. Cet aspect permet notamment d'appliquer la méthode au contrôle de santé des structures [80].

En mettant à jour les paramètres physiques du modèle existant, il devient possible d'affiner ses capacités prédictives, en l'alignant plus étroitement sur le comportement observé du système endommagé. L'objectif est d'obtenir un modèle qui non seulement tient compte de la présence d'un dommage, mais qui prédit également avec précision son impact sur le comportement global du système.

En outre, cette méthodologie de correction des paramètres offre les moyens de mener une analyse approfondie du modèle mis à jour. En comparant les prédictions du modèle mis à jour avec les données mesurées, il devient possible d'identifier la nature et la localisation des dommages. Ces informations sont précieuses pour comprendre les causes sous-jacentes des écarts observés et fournissent des indications cruciales pour les stratégies de maintenance et de réparation.

La clé du succès de cette approche est l'utilisation de techniques de régularisation parcimonieuses, ce qui rend cette méthodologie originale. La régularisation parcimonieuse permet de sélectionner le scénario d'endommagement le plus probable à partir d'un éventail de possibilités pouvant expliquer les données de mesure. La méthodologie peut ainsi effectivement localiser les endroits où les dommages sont situés et caractériser la gravité et l'étendue des dommages dans le système.

Cette approche est développée pour deux types différents de méthodes numériques : la méthode des éléments finis dans le contexte des équations linéaires et les réseaux neuronaux informés par la physique.

Contents

Introduction

0.1 Context and Motivations

Engineering in the 20th century primarily relied on the utilization of models to effectively design components and systems. Generally, these models comprised mathematical operators that facilitated the transmission of input to its corresponding output. To provide a clearer explanation, let's consider a deformable solid that undergoes shape modification due to the application of a force on its surface. This deformation results in an internal mechanical condition within the solid, leading to reversible or irreversible changes in its geometry. Such changes are often observed in manufacturing processes like stamping or forging, among others. Hence, the mechanical model mentioned earlier can be perceived as the operator that deduces the final mechanical state in the structural system based on a given load.

There are two main typologies of models, referred later as `Type I` and `Type II`:

1. The first one (`Type I`) expresses the input/output relationship in engineering in an algebraic form, whether linear or nonlinear. For instance, consider the example of a rod experiencing tension. In this case, the applied stress on the rod is directly proportional to the resulting deformation, which is defined as the relative change in its length. This relationship remains linear for small deformations but can transition to a nonlinear relationship for larger deformations. Furthermore, for even larger deformations, it may become irreversible, leading to inelastic behavior.

   These algebraic models are easily manipulated and offer significant advantages in larger multi-component systems. Their algebraic nature greatly facilitates efficient manipulation and problem-solving.

2. The second type (`Type II`) involves an input/output relationship that gives rise to a mathematical problem involving derivatives in space and/or time. This situation arises when the solution at a particular point and time depends not only on the present conditions but also on the past history and the solution in all other points within the relevant domain. In such cases, models are expressed using more complex mathematical entities known as partial differential equations. These equations capture the interdependence of variables and their derivatives, enabling a comprehensive representation of the system dynamics.

Models of `Type II` generally pose greater challenges when it comes to finding efficient solutions. To illustrate this, let's consider a one-dimensional model, such as a rod, where we aim to determine the internal state at all points within it. Initially, we

encounter a difficulty: there is an infinite number of unknowns, corresponding to the numerous points along the rod.

However, if the problem can be solved analytically, this challenge can be overcome. In such cases, the solution takes the form of a closed mathematical expression, such as the displacement $u(x)$ in the example mentioned. This displacement at each point x in the rod can then be determined based on the forces $f(x')$ applied at each corresponding point x' in the rod, considering the given boundary conditions.

It is important to note that most models cannot be solved analytically due to various reasons. They are often defined in complex geometries, involve intricate boundary conditions, and may exhibit strong nonlinearity and coupling effects. As a result, discretization techniques have emerged as prominent tools in Simulation-Based Engineering (SBE) [18,19]. Examples of such techniques [20,21] include finite elements [22,23], finite differences [24,25], and finite volumes [26,27].

These discretization techniques aim to find solutions at a limited number of points, referred to as nodes, and specific time instances, rather than attempting to solve the problem for an infinite number of points and times. They effectively transform the continuous model into a discrete representation. Among these techniques, the finite element method has gained widespread popularity and is widely adopted in industrial practices today. It has proven to be a valuable approach for solving complex engineering problems.

However, the journey from the conceptual formalization of these discretization techniques to their widespread adoption in the industry has been quite extensive. Taking the previously mentioned one-dimensional problem as an example, the chosen discretization technique transforms the initial problem into a task of calculating the solution at only N points (referred to as nodes) distributed along the rod, forming a grid or mesh.

To illustrate, let us consider the case where N = 10. Once the solution is obtained at these 10 points, it can be interpolated to estimate the solution at any other location within the domain. However, in two dimensions (such as a square domain), the grid would contain 10x10 points that need to be calculated, and in three dimensions (e.g., a cube), it would involve 10x10x10 points. As a result, the algebraic problem associated with one-dimensional discretization has 10 unknowns, while two-dimensional and three-dimensional discretization problems have 100 and 1000 unknowns, respectively. The number of unknowns scales with the number of nodes in the grid or mesh, which increases exponentially with the dimension of the space where the problem is defined.

Even if an algebraic problem of size 10 can be solved relatively quickly by hand, the one involving the solution of a problem with 1000 unknowns could require a significant amount of time, potentially exceeding the lifespan of a human being.

Discretization, which is necessary for solving problems involving `Type II` models, became feasible thanks to the advent and contribution of computers. While computers can solve algebraic systems of equations quickly, problems with larger dimensions pose significant challenges. For example, solving an algebraic problem with 1000 unknowns may only take fractions of a second on a standard laptop. However, engineering problems are increasingly growing in size and complexity. They often involve hundreds of millions of unknowns and require many iterations due to nonlinearity or transient behavior (evolving solutions over time).

Even with the most powerful computational platforms available today, simulations of such large-scale problems still demand substantial time. They may take days,

weeks, or even months to complete. This extended computational time is necessary to accurately capture the intricate details of the system and obtain reliable results.

In summary, while most models can be solved with the aid of powerful computers, they often require a significant amount of time to perform the necessary computations. Engineering practices in the 20th century primarily focused on component design, and industries such as energy, automotive, and aerospace employed programs long enough to ensure the dialog between designers and simulation tools.

It is important to emphasize that the design process often necessitates multiple simulations to explore the parametric space and identify a quasi-optimal design. However, the cost associated with these simulations and the dimensionality of the parametric space, which can consist of hundreds of parameters, present ongoing challenges in achieving optimal certified designs even in modern times.

Another challenging aspect of Simulation-Based Engineering (SBE) relates to the quality of the models themselves. While models have evolved over centuries of scientific advancements, there are several factors that make it difficult to fully grasp and express the true behavior of a system:

- The size of the analyzed system in terms of space and time poses the first challenge. A model may be considered generally accurate, but it can exhibit noticeable local deviations. Similarly, a model that accurately represents the system at present may develop biases over time.
- Another difficulty lies in the large number of parameters involved in some models, making it challenging to calibrate and identify the appropriate values for these parameters.
- Additionally, even when using the correct model, the parameters themselves exhibit variability. For example, specimens made of the same material from the same supplier, produced by the same machine on the same day, and processed under the same nominal conditions can still exhibit slight differences in their mechanical properties. This leads to statistical distributions of model parameters, introducing uncertainty that propagates throughout the system.
- Lastly, there is always an inherent epistemic uncertainty or ignorance between the model and reality. Differences can exist at various scales in space and time, and these differences can potentially grow as time progresses.

In practice, engineers strive to mitigate the impact of these challenges to ensure accurate predictions and maintain the quality of their designs. Various techniques and strategies are employed to address these issues and minimize the uncertainties introduced by the models used in SBE.

0.1.1 From virtual to digital twins

The 20th century engineering was prolific, with many impressive successes [28]: aeronautics, space conquest, transport, energy, civil infrastructures, ...

In other domains such as marketing, economy, social sciences, and medicine, where traditional models were less established and had limited prognostic capabilities, researchers began shifting their focus towards the use of data for diagnosis and prognosis. This approach quickly led to significant achievements. However, in the field of engineering, three major new developments disrupted the status quo:

1. The emergence of online engineering: With the advent of technologies like the Internet of Things (IoT) and the availability of massive amounts of collected data, traditional offline engineering methods were pushed to operate in real-time. The focus shifted from product-based engineering to performance-based engineering. For example, in the aviation industry, companies became more interested in measuring hours of flight rather than solely focusing on aircraft engines. Engineering is now expected to provide continuous predictions and maintain performance throughout the entire lifespan of a product or system.

2. Engineering in the era of connected systems: Engineering, which previously concentrated on individual components or machines, had to adapt to address larger and interconnected systems. This shift led to the rise of concepts such as smart industries, smart cities, and smart nations. It marked the fourth industrial revolution, where data joined forces with previous revolutions driven by steam, electricity, and electronics/automation. Engineering expanded its scope to encompass complex systems of systems.

3. The upcoming human-centric immersive engineering revolution: The imminent fifth revolution will place humans at the core of engineering endeavors. It envisions a physics-aware metaverse, where advancements in computer vision, multimodal devices, and natural language processing (NLP) have the potential to defy the Turing test. However, integrating humans into the system poses challenges for traditional modeling approaches, as humans may escape or resist conventional modeling frameworks.

In response to these developments, engineering science turned its attention to the promising possibilities offered by data and Artificial Intelligence (AI). The expectations for leveraging these technologies in engineering are high, with the aim of achieving better predictions, improved performance, and enhanced understanding of complex systems [29].

The learning process in machine learning seems general and agnostic to the specific physics of the problem. Once input and output data (represented as $(p_i, u_i), i = 1, \ldots, n_s$, n_s being the number of available data) are collected, one can attempt to establish the relationship between the inputs (p) and outputs (u), denoted as $u = \mathcal{F}(p)$. By learning and establishing the functional relation $\mathcal{F}$ from the available data, the model can be applied to new input data to obtain the corresponding output. This allows for real-time application of the learned model to different scenarios.

The main challenge in this process is to obtain the regression model $u = \mathcal{F}(p)$ that accurately predicts the output u given the input p. There are various techniques available for constructing such regression models, and the choice depends on factors such as the quantity of available data (n_s), the time available for training (online versus offline), and the quality of the data.

In summary, if the quantity of interest (u) and the variables affecting it (p) are well identified, and if there is a sufficient quantity (n_s) and quality of data, there are numerous machine learning techniques capable of extracting the functional relationship (regression) $u = \mathcal{F}(p)$.

Imagine for a while the set of data: $(u_1 = 2, p_1 = 1)$, $(u_2 = 4, p_2 = 2)$, ... $(u_n = 2n, p_n = n)$, $n = 3, \ldots, n_s$. In that case the regression becomes linear and almost obvious: $u = 2p$, i.e. $\mathcal{F}(p) = 2p$.

While data-driven modeling has its advantages, there are still challenges and difficulties that persist in the process:

- Data collection: Determining which data to collect, at what scale, and when and where to collect it is crucial. The collected data should be relevant and contribute to the desired output. It is important not to discard any features that may be useful. Techniques exist to identify and remove irrelevant features and discover missing features, but collecting data can be costly and constrained by various factors such as available resources, technological limitations, regulations, and environmental considerations [4, 5].
- Cost of data: Data collection and processing can be expensive in engineering. Unlike in some domains where big data is readily available, engineering often faces limitations in data collection. The cost includes expenses related to sensing devices, instrumentation, data communication, data mining, processing, and addressing environmental concerns. The availability and cost of data can impact the feasibility and practicality of certain approaches [6, 7].
- Interpretability: While machine learning models can extract regressions and make predictions, explaining the learned model can be challenging. Understanding the underlying factors and relationships that the model has learned from the data may not be straightforward. It is easier to explain well-established scientific concepts like gravity, but explaining data-driven models may require additional effort and analysis [10, 11].
- Extrapolation and generalization: Models learned from data can be applied within the domain of validity of the collected data. When applying the learned model to situations or regions where data was not collected, extrapolation occurs. Extrapolation carries inherent risks as the model may not accurately predict outcomes in unobserved or extreme conditions. Care must be taken to understand the limitations and potential risks associated with extrapolating the model beyond its observed data range [8, 9].

Addressing these challenges requires careful consideration of data collection strategies, cost-effectiveness, interpretability of models, and the appropriate use of models within their valid domains.

0.1.2 In between models and data

As discussed earlier, the option that relies primarily on physics-based approaches has its advantages, but it also has drawbacks such as longer computing time and the need for significant computational resources. Additionally, it becomes challenging to handle large-scale systems with notable variability and uncertainty. On the contrary, the almost fully data-driven option appears promising, especially when accurate models are unavailable or lacking. However, this approach also faces difficulties when applied in engineering practices, including high data costs, risks associated with extrapolation, and challenges in providing explanations and certifications.

Perhaps the optimal choice lies in combining both approaches instead of choosing one over the other. Within this context, we will explore two possibilities:

1. Physics-informed learning [30],

2. Physics-augmented learning [31].

In both cases, we leverage the advantage that learning specific parts can be more cost-effective than learning the entire system. In the realm of physics, it is widely accepted that conservation laws are universal, and therefore, there is no need to relearn them from data. However, there are other phenomenological relationships that can benefit from a data-driven perspective, offering an opportunity for improvement.

Physics-informed learning

Let us consider a scenario where we are seeking to determine the output u at each point x of a domain Ω, i.e. $u(x)$, $x \in \Omega$. The regression model used to express $u(x)$ is expected to be complex and exhibit strong nonlinearity. Instead of employing the finite element method discussed earlier, we opt to approximate u using a nonlinear machine-learning-based regression, such as a neural network (NN).

However, according to established physics principles that we assume to be applicable here, $u(x)$ adheres to a state-of-the-art model defined by a partial differential equation in the form: $\mathcal{L}(u(x)) = f(x)$. In this equation, $\mathcal{L}(\cdot)$ represents a linear or nonlinear differential operator.

To address this, we turn to a physics-informed neural network (PINN), which seeks to find a neural network representation of u from x that approximates $u(x)$ while ensuring adherence to its governing equation. This is achieved by minimizing the residual $\mathcal{R} = \|\mathcal{L}(u(x)) - f(x)\|$, which enforces the verification of the partial differential equation during the construction of the regression for $u(x)$.

PINN can be seen as a type of collocation method, where the unknown field, $u(x)$, is approximated by a neural network, and the construction of the regression is based on minimizing the residual. This approach allows for easy assimilation of data in a straightforward and transparent manner.

Physics-augmented learning

The main concept here revolves around the assumption that the reference solution $u(x)$, obtained from a given loading term $f(x)$, can be decomposed into two contributions. The first contribution represents our existing knowledge, which originates from the physics-based model and is denoted as $u^P(x)$.

Subsequently, we calculate the discrepancy between the actual behavior and the prediction of the physics-based model, $u(x) - u^P(x)$. This discrepancy is commonly referred to as "ignorance."

To address this gap, machine learning techniques are employed to model this discrepancy, resulting in the data-driven model denoted as $u^D(x)$. Consequently, the fundamental approximation is expressed as: $u(x) = u^P(x) + u^D(x)$.

This relationship forms the basis of the hybrid modeling paradigm or the hybrid twin approach when applied to specific systems or assets.

Discussion

Both approaches, the physics-based model and the data-driven model, offer the advantage of significantly reducing the amount of data required for constructing the

models. This reduction is achieved either by leveraging the provided information or by focusing solely on modeling the discrepancy, which is assumed to be simpler to approximate than the actual behavior itself.

Furthermore, the use of the physics-based model enables us to explain the foundational aspects of the model and its resulting predictions, thereby alleviating extrapolation issues to some extent.

The reduction in data requirements is not solely due to the utilization of physics but also because physics informs us about the optimal locations and timing for data collection. This is particularly evident when considering active learning, which becomes highly effective when incorporating existing physics-based knowledge.

To illustrate this point, let us consider the example of temperature measurement in a city. To understand the temperature in a city, it is unnecessary to place thousands of thermometers on every street and record the temperature every millisecond. With knowledge of the physics that shapes our experience and common sense, we know that placing a few thermometers (e.g., one in each district) and measuring the temperature in the morning, at noon, in the afternoon, and at midnight would generally suffice.

In general, we should make the most of the existing knowledge available to us, incorporating both physics-based understanding and data-driven approaches.

0.1.3 Physics in real-time: Model Order Reduction

In this section we focus on the physics-based model, and tackle the methodologies to enhance its solution procedure to obtain real-time performance. Such procedures are known as Model Order Reduction (MOR) [1–3].

For this purpose, we consider two alternatives:

- The construction of an intrusive Reduced Order Model by applying dimensionality reduction techniques to the solution of the physics-based model, which subsequently enables the solver to operate very quickly.
- The construction of a surrogate, which refers to a direct relation between the solution of the physics-based model and the parameters involved in the model. Once this surrogate is built, it enables us to infer the solution almost in real-time by simply providing the model parameters.

Reduced order models and parametric solutions play a crucial role in enabling various tasks, such as simulation, optimization, inverse analysis, simulation-based control, and uncertainty propagation, all while operating under strict real-time constraints. These parametric solutions facilitate real-time engineering based on models in an exceptionally efficient and unprecedented manner.

To illustrate the concept, let's consider a simple parametric solution denoted as $u(x; p^1, p^2)$, where the unknown u depends on two parameters, p^1 and p^2, at each position x. It's important to note that while we are using two parameters for simplicity, the approach can be extended to include more parameters as needed.

The construction of the surrogate follows these steps:

1. A Design of Experiments (DoE) is defined, employing a sampling strategy. Several possibilities exist, such as Latin Hypercube [81], quadrature methods (e.g.,

Gauss-Lobatto-Chebyshev [82] or Smolyak [83], or approaches associated with active learning [12] (such as Gaussian processes [84] or Fisher information matrix [85]). The DoE provides a sampling of the parametric space: $(p_1^1, p_1^2), \ldots, (p_D^1, p_D^2)$.

2. Using the physics-based model and suitable software capable of solving it, the solution associated with each point in the DoE is calculated:
$u_1(x) = u(x; p_1^1, p_1^2), \ldots, u_D(x) = u(x; p_D^1, p_D^2)$.

3. Finally, a regression model $u(x; p^1, p^2)$ is constructed to approximate the relationship between the parameters and the solution. The regression model is constructed by minimizing the loss $\mathtt{L}$, which represents the approximation error. The loss, typically defined using the L^2-norm of the error, i.e. $\mathtt{L} = \sum_{i=1}^{D} \|u(x, p_i^1, p_i^2) - u_i(x)\|_2$, quantifies the discrepancy between the predicted surrogate solution and the true physics-based solution.

0.1.4 Data and learning

Regressions, which extract models from data, can take various forms depending on the nature of the data being manipulated. Some common types of data that regressions are applied to include:

- Lists (Tables): Data arranged in tables, which can consist of continuous or discrete numerical values, as well as categorical features.
- Images: Real 2D or 3D images or numerical simulation results.
- Tensor Formats: This pertains to compressed images or full-field data decomposeded in tensor product representation.
- Graphs: Nodal and edge features represented with a graph structure.
- Curves: Geometric curves or scalar-valued functions.
- Time-series, similar to curves, but concerning chronologically ordered data and implying causality.
- Points clouds: Densities or probability distributions.

Among the most usual machine learning techniques, we can mention:

- Regularized Polynomial Regressions [32]: These regressions are useful when data is scarce, and the solution can be expressed using polynomial bases. They can incorporate existing knowledge or engineered features.
- Artificial Neural Networks (NN): NNs perform well when there is a sufficient amount of data and well-tuned hyperparameters. The universal approximation theorem [33] states that NNs with an adequate number of neurons can approximate any function, making them versatile and powerful.
- Auto-Encoder (AE) [34]: AE is an NN architecture that maps data into a latent space, approximating the intrinsic dimensionality of the data. Modeling at the latent level removes linear and nonlinear correlations. Other dimensionality reduction techniques like PCA [35], kPCA [36], LLE [37], tSNE [38] based on manifold learning can also be used.

- Generative Adversarial Networks (GAN) [39]: GANs improve robustness by generating synthetic data samples that augment the dataset. They are particularly useful for data augmentation and generating realistic samples.

- Convolutional Neural Networks (CNN) [40]: CNNs are well-suited for image analysis and pattern recognition tasks. They excel at recognizing and extracting specific patterns from images. Graph Neural Networks (GNN) [41] extend CNNs to operate on unstructured meshes or graphs and are effective in learning physical behaviors.

- Recurrent Neural Networks (RNN) [42] and Long Short-Term Memory (LSTM) [43]: These techniques are powerful for handling time series data. RNNs and LSTMs capture temporal dependencies and have memory capabilities, making them suitable for sequential data analysis.

- Dynamical Systems: Various techniques are used for modeling dynamical systems. RNNs and LSTMs can learn state evolutions and associated observables. ResNet [44] or NeuralODE [45] can learn the forcing terms of dynamical systems. Techniques like Dynamic Mode Decomposition (DMD) [46] and the Koopman operator [47] are efficient for addressing nonlinear behaviors. Reservoir computing [48] and DeepONets [49] are emerging techniques in this field.

- Physics-Informed Neural Networks (PINN) [13], Thermodynamics-Informed Neural Networks (TINN) [50] and Hybrid Models: These techniques incorporate physics-based knowledge into the learning process. PINNs, TINNs, and SPNNs [86] ensure the satisfaction of governing equations. Hybrid models combine physics-based and data-driven approaches, often used in transfer learning scenarios.

Concurrently, learning modalities play a crucial role in machine learning. Here are some of the commonly used modalities:

- Supervised Learning [87]: In supervised learning, the model learns from labeled data, where the input samples are associated with corresponding target labels. The model aims to learn the mapping between inputs and outputs based on the provided labeled examples.

- Unsupervised Learning [88]: Unsupervised learning involves learning from unlabeled data. The model explores the inherent structure or patterns in the data without any specific target labels. Clustering, dimensionality reduction, and generative models are examples of unsupervised learning techniques.

- Self-supervised Learning [89]: Self-supervised learning is a variation of unsupervised learning where the model learns to predict certain aspects of the data from the data itself. It formulates learning tasks using the available unlabeled data, creating proxy supervisory signals.

- Semi-supervised Learning [90]: In semi-supervised learning, the model learns from a combination of labeled and unlabeled data. It leverages the limited labeled data and the additional unlabeled data to improve the learning process. This approach is beneficial when labeled data is expensive or time-consuming to obtain.

- Transfer Learning [91]: Transfer learning involves leveraging knowledge or models learned from one domain and applying it to another related domain. The pre-trained models on large-scale datasets can be fine-tuned or used as a feature extractor for a target task with limited data.

- Reinforcement Learning [92]: Reinforcement learning is used when an agent interacts with an environment and learns to take actions that maximize a reward signal. The agent explores the environment and learns through trial and error, receiving feedback in the form of rewards or penalties.

The choice of learning modality depends on factors such as the availability of labeled data, the complexity of the data structure, and the need for online or real-time learning. Each modality has its strengths and limitations, and the selection depends on the specific problem and context.

0.1.5 Hybridization

In the context of hybridization [51], there are two main routes that can be followed: enriching the solution or enriching the model and thus obtaining the enhanced solution.

Solution enrichment

The enriched or corrected solution is obtained by combining the physics-based model solution, $u^P(x, \boldsymbol{p})$ (computed by using the techniques discussed in Section 0.1.3), with the data-driven correction, $u^D(x, \boldsymbol{p})$ (learned by using one of the technologies introduced in Section 0.1.4). Here, $\boldsymbol{p}$ represents the vector of parameters associated with the model.

To calibrate the model online, as soon as data is collected, the operational parameters $\boldsymbol{p}^*$ are identified by minimizing the difference between the model predictions and the available measurements. This calibration process ensures that the model is aligned with the observed data and can accurately capture the behavior of the system.

Finally, and then the hybrid prediction yields the corrected or enriched solution from $u^P(x, \boldsymbol{p}^*) + u^D(x, \boldsymbol{p}^*)$.

The same procedure applies to transient problems, where the model parameters may vary over time. The online calibration is performed at each time step, and the data-driven model is particularized accordingly to provide an updated hybrid prediction.

Model enrichment

In the model enrichment route, the objective is to improve the predictive capability of a nominal model that is assumed to be known. The discrete form of the nominal model, for simplicity, can be represented as $\mathbf{MU} = \mathbf{F}$, where $\mathbf{M}$ is the matrix representing the system properties, $\mathbf{U}$ and $\mathbf{F}$ are the nodal vectors of displacements and forces, respectively.

In practice, measurements of displacements, denoted as $\tilde{\mathbf{U}}^{\mathtt{exp}}$, are collected at certain locations. When comparing these measured displacements with the predictions of the nominal model in the same locations, denoted as $\tilde{\mathbf{U}}$, a noticeable discrepancy, or gap, is observed. The norm of the difference between the measured and predicted

displacements $\|\tilde{\mathbf{U}}^{\mathtt{exp}} - \tilde{\mathbf{U}}\|$ remains larger than the acceptable error threshold ϵ, i.e. $\|\tilde{\mathbf{U}}^{\mathtt{exp}} - \tilde{\mathbf{U}}\| > \epsilon$.

To address this discrepancy, an enrichment matrix $\mathbf{M}^*$ is introduced. The goal is to find an enriched discrete model, represented as $\mathbf{M} + \mathbf{M}^*$, that can better represent the collected data. The corrected model, along with the associated displacement prediction, denoted as $\mathbf{U}+\mathbf{U}^*$, should satisfy the equilibrium condition $(\mathbf{M}+\mathbf{M}^*)(\mathbf{U}+\mathbf{U}^*) = F$ while accurately representing the collected data, i.e. $\|\tilde{\mathbf{U}}^{\mathtt{exp}} - (\tilde{\mathbf{U}}+\tilde{\mathbf{U}}^*)\| < \epsilon$.

To compute the model enrichment and perform data completion $\mathbf{U}+\mathbf{U}^*$, a parametrization of the model correction $\mathbf{M}^*$ is chosen, along with an appropriate regularization technique. This parametrization allows for the representation of the model correction in a computationally efficient manner.

0.2 Objectives of the thesis

The objective of the work performed in this thesis is to apply, develop, and investigate novel tools that enable the use of virtual, digital, and hybrid twins in order to tackle present-day industrial challenges. Specifically, the focus has been placed on exploring and expanding the Model Order Reduction (MOR) and Machine Learning (ML) frameworks and the combination of both. The contributions tackle three major difficulties in the context of Hybrid Twins:

- Performing model reduction and regression at the low-data limit.
- Employing adequate mathematical representations to empower model reduction.
- Collecting relevant measurements for data-driven and hybrid modelling.
- Using data to enrich physics-based models.

In particular, the different topics addressed concern:

- Contributions to the development of machine learning and non-intrusive reduced modeling techniques [93–95].
- Development of general MOR methods which address the localization of features [96, 97].
- Application of MOR and ML methods to industrial cases [98–100].
- Adaption of MOR techniques to solve difficult problems [101, 102].
- Development of tools to reduce component models in engineering applications [103, 104].
- Employment of adequate representations and metrics to apply machine learning to materials science [105, 106].
- Theoretical study on the feasibility of modeling from partial observations [107].
- Development of hybrid modeling techniques [108, 109].
- Application of hybrid modeling to model correction and damage identification [110, 111].
- Construction of Digital and Hybrid Twin methodologies [112, 113].

0.3 Content

This thesis is divided in three chapters: first, some tools and methodologies which are key components in the construction of Hybrid Twins are describes, then personal contributions in the field of non-intrusive Model Order Reduction are presented and finally, personal contributions to hybrid modelling are described.

0.3.1 Chapter 1: Review of methods enabling Hybrid Twins

Section 1: Intrusive and non-intrusive Model Order Reduction

Engineering problems often requires fast and accurate simulations using complex models and large amounts of data. Although the development of hardware has allowed to improve the quality of simulations, there are still challenges in solving high-dimensional problems due to numerical complexity and real-time processing limitations. Model Order Reduction (MOR) techniques, categorized as intrusive and non-intrusive methods, have gained attention to address these issues. Proper Orthogonal Decomposition (POD) [52, 53] extracts essential system characteristics and creates a reduced order model, significantly reducing computation time. Proper Generalized Decomposition (PGD) [54, 55] is an intrusive method which employs the separation of variables to reduce the complexity of the resolution algorithm. Proper Orthogonal Decomposition with Interpolation (PODI) [56, 57] is a non-intrusive methods that handle parametric without modifying the solver. The sparse Proper Generalized Decomposition (sPGD) [58] constructs a parametric solution using a sparse set of solutions computed by a PDE solver. Overall, MOR techniques provide efficient solutions for complex engineering problems in real-time applications.

Section 2: Physics-Informed Neural Networks

Physics-Informed Neural Networks (PINN) [13] have emerged as a promising framework that integrates neural networks with physics principles to solve complex scientific and engineering problems. Unlike traditional machine learning, PINNs incorporate prior knowledge of physical laws to enable accurate predictions and capture underlying physics, even with limited labeled data. This section introduces PINN, explaining how it combines governing equations and constraints with a neural network architecture.

0.3.2 Chapter 2: Advances in non-intrusive Model Order Reduction

Section 1: Regularization of separated representations

The challenge in regression with limited data is handling nonlinearities in multi-parametric settings, leading to the curse of dimensionality [59, 60]. Occam's razor [61, 62] principle suggests favoring simpler models over complex ones, achieved through sparsity promotion in regression [62, 63]. This section proposes robust and efficient regression methodologies for separated representation settings. Two techniques are presented: rs-PGD, combining L2 and L1 norm regularization, and s^2-PGD, a doubly sparse regression method utilizing Lasso regularization.

Section 2: ANOVA-PGD: A novel strategy

The previous section's workflow presents new challenges and questions. In the context of s^2-PGD, the choice of basis functions significantly impacts the solution's quality, unlike in sPGD. Additionally, PGD-based regression methods currently use sparse design of experiments without exploiting smart sampling techniques, leaving untapped potential. These observations motivate the development of a new technique called ANOVA-PGD, which combines the approaches from the previous section with the ANOVA (Analysis of variance) decomposition [64–66] strategy and its low-cost counterpart the anchored ANOVA [67–69]. ANOVA-PGD utilizes a hierarchical orthogonal basis and suitable sampling, scaling linearly with the number of parameters involved.

Section 3: Reduced Modelling of curves

Predicting whole curves from given input features in parametric surrogates presents challenges. Non-alignement in curves, which can result from different sampling or ending points, and shifted critical points in curves with common patterns (e.g., elastic-plastic transition in mechanics), make classical interpolation methods fail to provide physics-consistent results, requiring appropriate pre-processing. When bifurcations occur in the parametric space, coupling with clustering [70, 71] and classification [72] algorithms enhances the surrogate's accuracy. This section introduces methodologies to address these issues and create surrogates used for uncertainty quantification [73], providing statistical bounds for predicted curves.

Section 4: Reduced Modelling based on Optimal Transport

Reduced Order Modelling in contexts such as the one described in the previous section become even more challenging when dealing with full-field solutions in multiple dimensions. Linear reduced bases, commonly used in ROM algorithms, struggle to capture continuous displacements of solution features caused by phenomena like advection or localized parametric constraints. The limitations of reduced bases stem from their use of a Euclidean metric, which fails to consider misalignment or feature localization. To address this, Optimal Transport theory [16, 17] provides metrics designed for detecting distances between noteworthy features. This section introduces a methodology that combines the ANOVA framework and Optimal Transport to create parametric reduced models.

0.3.3 Chapter 3: Data assimilation and Hybrid Modelling

Section 1: Modeling systems from partial observations

This section addresses the issue of data accessibility in modelling. When only a portion of a system can be measured or observed, questions arise regarding the existence, formulation, and learning of a model that connects the observable inputs to the corresponding outputs, considering the influence of hidden dynamics [74–77]. The section revisits model construction in domains with partial observability, exploring both traditional algebraic formulations and Machine Learning approaches.

Section 2: Hybrid modeling by learning a source term

This section explores the combination of data and a physics-based model to create a hybrid model. While the PINN approach offers a simple solution by embedding physics equations in the loss function, it only provides a hybrid solution, not a hybrid model. Creating a hybrid model offers advantages such as versatility in different settings and the potential for physical interpretation of the model corrections. The section proposes a hybridization strategy established in the context of classical discretization methods and then improved within the PINN framework.

Section 3: Hybrid modeling by identifying properties

In the pursuit of refining physics-based models through hybrid modeling, an alternative approach arises: correcting uncertain physical parameters associated with the system. This correction enables the development of a model that accurately captures the behavior of a damaged system and identifies the nature and location of the damage. This section presents a novel methodology focused on parameter correction, refining the model's predictive capabilities and providing insights into the damage. Sparse regularization techniques are key to this approach, allowing for the selection of the most probable damage scenario and pinpointing its location and severity within the system.

0.4 Scientific contributions

0.4.1 Publications in international journals

1. S. Vermiglio, V. Champaney, A. Sancarlos, F. Daim, J. C. Kedzia, J. L. Duval, P. Diez, and F. Chinesta, *Parametric electromagnetic analysis of radar-based advanced driver assistant systems*, Sensors, 20 (2020), p. 5686 [101]

2. T. Loreau, V. Champaney, N. Hascoët, P. Mourgue, J.-L. Duval, and F. Chinesta, *Learning the parametric transfer function of unitary operations for real-time evaluation of manufacturing processes involving operations sequencing*, Applied Sciences, 11 (2021), p. 5146 [103]

3. A. Reille, V. Champaney, F. Daim, Y. Tourbier, N. Hascoet, D. Gonzalez, E. Cueto, J. L. Duval, and F. Chinesta, *Learning data-driven reduced elastic and inelastic models of spot-welded patches*, Mechanics & Industry, 22 (2021), p. 32 [104]

4. K. Derouiche, S. Garois, V. Champaney, M. Daoud, K. Traidi, and F. Chinesta, *Data-driven modeling for multiphysics parametrized problems-application to induction hardening process*, Metals, 11 (2021), p. 738 [98]

5. S. Torregrosa, V. Champaney, A. Ammar, V. Herbert, and F. Chinesta, *Surrogate parametric metamodel based on optimal transport*, Mathematics and Computers in Simulation, 194 (2022), pp. 36–63 [97]

6. V. Champaney, F. Chinesta, and E. Cueto, *Engineering empowered by physics-based and data-driven hybrid models: A methodological overview*, International Journal of Material Forming, 15 (2022), p. 31 [113]

7. T. Loreau, V. Champaney, N. Hascoet, J. Lambarri, M. Madarieta, I. Garmendia, and F. Chinesta, *Parametric analysis and machine learning-based parametric modeling of wire laser metal deposition induced porosity*, International Journal of Material Forming, 15 (2022), p. 33 [105]

8. V. Champaney, A. Pasquale, A. Ammar, and F. Chinesta, *Parametric curves metamodelling based on data clustering, data alignment, pod-based modes extraction and pgd-based nonlinear regressions*, Frontiers in Materials, 9 (2022), p. 363 [96]

9. D. Di Lorenzo, V. Champaney, C. Germoso, E. Cueto, and F. Chinesta, *Data completion, model correction and enrichment based on sparse identification and data assimilation*, Applied Sciences, 12 (2022), p. 7458 [110]

10. V. Champaney, V. J. Amores, S. Garois, L. Irastorza-Valera, C. Ghnatios, F. J. Montáns, E. Cueto, and F. Chinesta, *Modeling systems from partial observations*, Frontiers in Materials, 9 (2022) [107]

11. M. El Fallaki Idrissi, F. Praud, V. Champaney, F. Chinesta, and F. Meraghni, *Multiparametric modeling of composite materials based on non-intrusive pgd informed by multiscale analyses: Application for real-time stiffness prediction of woven composites*, Composite Structures, 302 (2022), p. 116228 [99]

12. S. Torregrosa, V. Champaney, A. Ammar, V. Herbert, and F. Chinesta, *Hybrid twins based on optimal transport*, Computers & Mathematics with Applications, 127 (2022), pp. 12–24 [109]

13. C. Ghnatios, V. Champaney, A. Pasquale, and F. Chinesta, *A regularized real-time integrator for data-driven control of heating channels*, Computation, 10 (2022), p. 176 [94]

14. A. Pasquale, V. Champaney, Y. Kim, N. Hascoët, A. Ammar, and F. Chinesta, *A parametric metamodel of the vehicle frontal structure accounting for material properties and strain-rate effect: application to full frontal rigid barrier crash test*, Heliyon, 8 (2022), p. e12397 [100]

15. M. Jacot, V. Champaney, F. Chinesta, and J. Cortial, *Parametric damage mechanics empowering structural health monitoring of 3d woven composites*, Sensors, 23 (2023), p. 1946 [102]

16. A. Sancarlos, V. Champaney, E. Cueto, and F. Chinesta, *Regularized regressions for parametric models based on separated representations*, Advanced Modeling and Simulation in Engineering Sciences, 10 (2023), p. 4 [93]

17. A. Runacher, M.-J. Kazemzadeh-Parsi, D. Di Lorenzo, V. Champaney, N. Hascoet, A. Ammar, and F. Chinesta, *Describing and modeling rough composites surfaces by using topological data analysis and fractional brownian motion*, Polymers, 15 (2023), p. 1449 [106]

18. C. Ghnatios, D. di Lorenzo, V. Champaney, E. Cueto, and F. Chinesta, *Optimal velocity planning based on the solution of the euler-lagrange equations with a neural network based velocity regression*, Discrete and Continuous Dynamical Systems-S, (2023), pp. 0–0 [95]

19. D. Di Lorenzo, V. Champaney, J. Marzin, C. Farhat, and F. Chinesta, *Physics informed and data-based augmented learning in structural health diagnosis*, Computer Methods in Applied Mechanics and Engineering, 414 (2023), p. 116186 [111]

0.4.2 Submitted articles

1. S. Torregrosa, V. Champaney, A. Ammar, V. Herbert and F. Chinesta. *Physics-based Active Learning for design space exploration and surrogate construction for multiparametric optimization.* Communications on Applied Mathematics and Computation.

2. M. Kazemzadeh-Parsi, A. Pasquale, D. Di Lorenzo, V. Champaney, A. Ammar and F. Chinesta. *NURBS-based shape parametrization enabling PGD-based space separability: methodology and application.* Finite Elements in Analysis & Design.

3. A. Schmid, A. Pasquale, C. Ellersdorfer, M. Ziane, M. Raffler, V. Champaney, F. Feist and F. Chinesta. *Application of PGD separation of space to create a reduced order model of a lithium-ion cell structure.* Frontiers in Materials.

4. S. Torregrosa, V. Champaney, A. Ammar, V. Herbert and F. Chinesta. *Predicting high-fidelity data from coarse-mesh Computational Fluid Dynamics corrected using Hybrid Twins based on Optimal Transport.* Mechanics & Industry.

5. A. Schmid, A. Pasquale, C. Ellersdorfer, V. Champaney, M. Raffler, S. Guevelou, S. Kizio, M. Ziane, F. Feist and F. Chinesta. *PGD based Meta Modelling of a Lithium-Ion Battery for Real Time Prediction.* Frontiers in Materials.

6. C. Ghnatios, X. Kestelyn, G. Denis, V. Champaney, and F. Chinesta. *Learning data-driven stable corrections of dynamical systems: Application to the simulation of the top-oil temperature evolution of a power transformer.* Energies.

7. M. Jacot, V. Champaney, S. Torregrosa, F. Chinesta and J. Cortial. *Empowering Optimal Transport matching algorithm for the construction of surrogate parametric metamodel.* Mechanics & Industry.

0.4.3 Conference proceedings

1. V. Champaney, A. Sancarlos, F. Chinesta, E. Cueto, D. González, I. Alfaro, S. Guevelou, J. L. Duval, A. Chambard, and P. Mourgue, *Hybrid twins-a highway towards a performance-based engineering. part i: Advanced model order reduction enabling real-time physics*, in ESAFORM 2021, Prof. Anne Marie Habraken, 2021 [112]

2. X. Kestelyn, G. Denis, V. Champaney, N. Hascoet, C. Ghnatios, and F. Chinesta, *Towards a hybrid twin for infrastructure asset management: Investigation on power transformer asset maintenance management*, in 2022 7th International Advanced Research Workshop on Transformers (ARWtr), IEEE, 2022, pp. 109–114 [108]

0.4.4 International conferences

1. V. Champaney, A. Sancarlos, F. Chinesta, E. Cueto, D. González, I. Alfaro, S. Guevelou, J. L. Duval, A. Chambard and P. Mourgue. *Hybrid Twins : A highway towards a performance-based engineering. Part I : Advanced Model Order Reduction enabling Real-Time Physics.* 24th International Conference on Material Forming, April 14-16, 2021.

2. F. Chinesta, E. Cueto, V. Champaney and J. L. Duval. *Material Hybrid Descriptions Combining Physics Based and Data-Driven Models.* 16th U.S. National Congress on Computational Mechanics (USNCCM16), July 25-29, 2021.

3. V. Champaney, A. Sancarlos, F. Chinesta, E. Cueto, J. L. Duval and A. Chambard. *Hybrid Twins for empowering performance-based engineering. Part I. Real-time physics-based models.* Mechanistic Machine Learning and Digital Twins for Computational Science, Engineering and Technology (MMLDT-CSET) 2021, September 26-29, 2021.

4. V. Champaney, A. Sancarlos, F. Chinesta, E. Cueto, J. L. Duval and A. Chambard. *Hybrid Twins for empowering performance-based engineering. Part II. Real-time data-driven models and hybridation.* Mechanistic Machine Learning and Digital Twins for Computational Science, Engineering and Technology (MMLDT-CSET) 2021, September 26-29, 2021.

5. A. Pasquale, V. Champaney, A. Ammar, F. Chinesta and J. L. Duval. *PGD based Time Multiscale.* Mechanistic Machine Learning and Digital Twins for Computational Science, Engineering and Technology (MMLDT-CSET) 2021, September 26-29, 2021.

6. S. Torregrosa, V. Champaney, F. Chinesta and V. Herbert. *Ensuring confort in vehicle cabin by combining advanced non-linear regressions and parametric optimal transport.* Mechanistic Machine Learning and Digital Twins for Computational Science, Engineering and Technology (MMLDT-CSET) 2021, September 26-29, 2021.

7. V. Champaney, N. Hascoet, J. Cotton, G. Hermand and F. Chinesta. *Towards a resilient and robust smart sensing empowering diagnosis and prognosis in large civil infrastructures.* ALERT geomaterials European Lab Network, September 28, 2021.

8. P. Kyvelou, L. Gardner, L. Zou, S. Lauria, C. Gonzales, F. Gilardi, O. Krystalakos, A. Ammar, V. Champaney and M. Megahed. *Predicting WAAM Material Properties via Machine Learning.* TMS 2022 Annual Meeting & Exhibition, February 27-March 3, 2022.

9. M. El Fallaki Idrissi, F. Praud, V. Champaney, F. Chinesta and F. Meraghni. *Non-intrusive PGD for real-time prediction of macroscopic stiffness properties of woven composites.* 11th European Solid Mechanics Conference, July 4-8, 2022.

10. M. Jacot, V. Champaney, F. Chinesta and M. Théobald. *Real-time structural health monitoring of aeronautical structures: A data-driven modeling approach for optimized sensors placement, detection and localization of damage.* 15th World Congress on Computational Mechanics (WCCM-XV), July 31-August 5, 2022.

11. C. Ghnatios, A. Sancarlos, V. Champaney, F. Chinesta, J. Mouterde, Y. Dupuis, F. Caire and J. Da Silva *A real-time data-driven modelling frawork for control and simulate the behavior of industrial controllers.* 15th World Congress on Computational Mechanics (WCCM-XV), July 31-August 5, 2022.

12. X. Kestelyn, G. Denis, V. Champaney, N. Hascoet, C. Ghnatios and F. Chinesta. *Towards a hybrid twin for infrastructure asset management: Investigation on power transformer asset maintenance management.* 7th International Advanced Research Workshop on Transformers (ARWtr2022), October 24-26, 2022.

13. V. Champaney, D. Di Lorenzo, A. Pasquale, A. Ammar, M. Kazemzadehparsi, D. Baillargeat, E. Cueto and F. Chinesta. *Hybrid modelling for decision making in critical systems.* IUTAM Symposium on Data-Driven Mechanics and Surrogate Modeling, October 25-28, 2022.

14. S. Torregrosa, V. Champaney, A. Ammar, V. Herbert and F. Chinesta. *Hybrid twins based on optimal transport.* IUTAM Symposium on Data-Driven Mechanics and Surrogate Modeling, October 25-28, 2022.

15. D. Di Lorenzo, A. Pasquale, V. Champaney, M. Kazemzadehparsi, A. Ammar, D. Baillargeat and F. Chinesta. *The importance of wind simulation to find in real time the optimal flight trajectory in drones operations.* 10th OpenFOAM Conference, November 8, 2022.

16. M. Jacot, V. Champaney, F. Chinesta and J. Cortial. *An Adaptive Sparse Proper Generalized Decomposition for Real-Time Structural Health Monitoring.* SIAM CSE23, February 26-March 3, 2023.

17. D. Di Lorenzo, V. Champaney, C. Germoso, E. Cueto and F. Chinesta. *Models correction based on sparse identification and data assimilation.* SIAM CSE23, February 26-March 3, 2023.

18. S. Torregrosa, V. Champaney, A. Ammar, V. Herbert and F. Chinesta. *Hybrid twins based on optimal transport.* Congrès SIA Simulation Numérique, April 5-6, 2023.

19. A. Schmid, M. Raffler, D. Feist, A. Pasquale, V. Champaney, M. Ziane and F. Chinesta. *Mechanical Characterization of Li-Ion Cells and the Calibration of Numerical Models Using Proper Generalized Decomposition.* IMECE2023, October 29-November 2, 2023.

1

Review of methods enabling Hybrid Twins

1.1 Intrusive and non-intrusive Model Order Reduction

Engineering often involves intricate models that demand swift and precise solutions, requiring processing of vast amounts of data. Meanwhile, from the requirements of their applications, these models must frequently be computed in real-time on deployed platforms. Despite the significant advancements made in modelling, numerical analysis, discretization techniques, and computer science over the past decades, there are still numerous problems in science and engineering that cannot be solved due to their numerical complexity or limitations imposed by specific needs such as real-time processing on current technological platforms. Very often, these limitations come from the necessity to solve high-dimensional problems on very fine discretizations of complex domains. Recently, Model Order Reduction has attracted a lot of interest because of its ability to address previously intractable problems in numerical simulation. Model Order Reduction techniques are often classified in two groups: intrusive and non-intrusive methods. In this chapter, a review of a selection of intrusive and non-intrusive Model Order Reduction techniques is presented. First, Proper Orthogonal Decomposition (POD) is a general method used to extract the most essential characteristics of a system's behavior and express them with a low-dimensional representation through a set of basis vectors. One of the most common uses of POD is to project the governing equations of the system onto the reduced-order subspace spanned by the extracted basis. This produces a reduced order model that can be resolved instead of the initial system, in a usually much lower amount of time. Section 1.1.2 presents the Proper Generalized Decomposition (PGD), a method making use of separated representations to circumvent the curse of dimensionality. A greedy enrichment procedure coupled with a fixed-point algorithm enable replacing a high-dimensional problem with a set of lower-dimensional problems solved iteratively. This strategy is especially useful when dealing with parametric problems or problems that can be separated in a multi-scale setting. Section 1.1.3 describes the Proper Orthogonal Decomposition with Interpolation (PODI), a non-intrusive technique based on a similar approach as POD, which

consists in extracting a reduced basis from collected data and performing an interpolation or regression to express the reduced coordinates as a function of the parameters of the problem. Unlike POD, this method does not require to modify the solver, hence it can be combined with any commercial software without needing access to the source code. Finally, section 1.1.4 introduces the sparse Proper Generalized Decomposition (sPGD), the non-intrusive counterpart of the PGD, which constructs a parametric solution from a set of solutions computed by a PDE solver based on a design of experiments. The full order solutions are obtained using any commercial software, and the PGD algorithm combined with one-dimensional approximation bases is then applied to build a parametric solution in which the separation of variables is applied to the parametric space.

1.1.1 Proper Orthogonal Decomposition

Proper Orthogonal Decomposition (POD) [52, 53] is a method which consists in reducing the complexity of a problem by using a low-dimensional representation of its state instead of the initial high-dimensional representation. It is the application of Principal Component Analysis (PCA) [35] to Model Order Reduction.

Let us consider a general system of N ordinary differential equations (ODE) which can in particular originate from the spatial discretization of a partial differential equation (PDE):

$$\boldsymbol{E}(\boldsymbol{p})\frac{d\boldsymbol{u}}{dt} + \boldsymbol{A}(\boldsymbol{p})\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{p})$$

where $\boldsymbol{u} : (t;\boldsymbol{p}) \mapsto \boldsymbol{u}(t;\boldsymbol{p}) \in \mathbb{R}^N$ is the state of the system being studied, $\boldsymbol{p} \in \mathcal{D} \subset \mathbb{R}^d$ is a vector of d parameters, $\boldsymbol{E}(\boldsymbol{p}) \in \mathbb{R}^{N\times N}$, $\boldsymbol{A}(\boldsymbol{p}) \in \mathbb{R}^{N\times N}$ and $\boldsymbol{f}(\boldsymbol{p}) \in \mathbb{R}^N$.

Using any method of integration, we can compute a set of n_s "snapshots" $\{\boldsymbol{u_1}, \boldsymbol{u_2}, ..., \boldsymbol{u_{n_s}}\}$ where $\boldsymbol{u_i} = \boldsymbol{u}(t_i;\boldsymbol{p_i})$ is the solution of the problem at $t = t_i$ and for the parameters $\boldsymbol{p} = \boldsymbol{p_i} \in \mathcal{D}$. The snapshots can also come from prior knowledge of the system or measurements.We can construct a matrix containing all the snapshots, the "snapshot matrix" $\boldsymbol{U} \in \mathbb{R}^{N\times n_s}$ which has $\boldsymbol{u_i}$ as its i-th column.

The Singular Value Decomposition (SVD) of $\boldsymbol{U}$ can be written $\boldsymbol{U} = \boldsymbol{L}\boldsymbol{\Sigma}\boldsymbol{R}^T$, $\boldsymbol{L} \in \mathbb{R}^{N\times N}$ and $\boldsymbol{R} \in \mathbb{R}^{n_s\times n_s}$ being respectively the left and right singular vectors and $\boldsymbol{\Sigma} \in \mathbb{R}^{N\times n_s}$ the matrix containing the singular values on its diagonal. The POD basis $\boldsymbol{\Phi} = (\boldsymbol{\phi_1}, ..., \boldsymbol{\phi_M})$ is defined as the $M \in \mathbb{N}$ left singular vectors of $\boldsymbol{U}$ which correspond to the M largest singular values. This is an orthogonal basis which gives a good approximation of the snapshots provided M is chosen large enough. In practice, M is chosen by considering the singular values since they give a good idea of how much "energy" or "information" each basis function represents.

The POD reduced model is then written:

$$\boldsymbol{E_M}(\boldsymbol{p})\frac{d\boldsymbol{u_M}}{dt} + \boldsymbol{A_M}(\boldsymbol{p})\boldsymbol{u_M} = \boldsymbol{f_M}(\boldsymbol{p})$$

where $\boldsymbol{E_M}(\boldsymbol{p}) = \boldsymbol{\Phi}^T\boldsymbol{E}(\boldsymbol{p})\boldsymbol{\Phi} \in \mathbb{R}^{M\times M}$, $\boldsymbol{A_M}(\boldsymbol{p}) = \boldsymbol{\Phi}^T\boldsymbol{A}(\boldsymbol{p})\boldsymbol{\Phi} \in \mathbb{R}^{M\times M}$ and $\boldsymbol{f_M} = \boldsymbol{\Phi}^T\boldsymbol{f} \in \mathbb{R}^M$. Computing the reduced solution $\boldsymbol{u_M}$ is much faster than computing $\boldsymbol{u}$ because $M << N$. The approximation of the solution is then given by $\boldsymbol{u} \approx \boldsymbol{\Phi}\boldsymbol{u_M}$.

One of the drawbacks encountered in POD is that it requires solving a large eigenvalue problem to obtain the reduced basis since the snapshot matrix has the size of the number of nodes or elements in the considered mesh, which can easily reach millions

in industrial applications. This cost can be alleviated with the so-called snapshot POD [114] by considering the fact that the number of snapshots is much smaller than the size of the discretization in space, hence solving for the eigenvalues of $\boldsymbol{U}^T\boldsymbol{U}$ is much faster than solving for those of $\boldsymbol{U}\boldsymbol{U}^T$.

In a non-linear setting, this approach is not as efficient as in linear cases, because the computation of non-linear terms involves recalculating the full solution at each time step, increment or iteration of the non-linear solver, which almost completely negates the positive impact of POD on the computational complexity. Different methods have been proposed in the literature to circumvent this issue, such as the Empirical Interpolation Method (EIM) [115,116] or hyper reduction [117,118].

1.1.2 Proper Generalized Decomposition

Proper Generalized Decomposition (PGD) [54,55] is a method which consists in computing a solution to a PDE in a reduced basis compared to an ordinary finite element basis. However, unlike POD, the reduced basis is not computed beforehand but at the same time as the solution. The numerical solution to a PDE using a finite element scheme can be written:

$$u(x,t) \approx \sum_{m=1}^{N} \alpha_m(t) w_m(x)$$

where the w_i are the basis functions, each related to one node of the mesh and the α_i are the unknown coefficients. Independently of the problem at hand, this approximation involves a separation of variables between time and space.

Inspired by this idea, we can consider a more general space-time separation:

$$u(x,t) \approx \sum_{m=1}^{M} X_m(x) T_m(t)$$

which is similar to the previous notation but this time both the X_i and T_i are unknown and will be computed by the solver. The number of terms in the sum M is also chosen by the solver based on a convergence criterion.

The solution is computed by iteratively solving a succession of one-dimensional problems. Before the M-th enrichment step, we have:

$$u^{M-1}(x,t) = \sum_{m=1}^{M-1} X_m(x) T_m(t)$$

The M-th term $X_M(x)T_M(t)$ is computed such that

$$u^{M}(x,t) = u^{M-1}(x,t) + X_M(x)T_M(t)$$

$X_M(x)$ and $T_M(t)$ are calculated at the same time: starting from an arbitrary initial guess $T_M^0(t)$, the alternating algorithm computes $X_M^j(x)$ from $T_M^{j-1}(t)$, and then $T_M^j(t)$ from $X_M^j(x)$ by injecting everything into the PDE and solving the 1D problem obtained, until the condition $\|X_M^j(x)T_M^j(t) - X_M^{j-1}(x)T_M^{j-1}(t)\| < \epsilon$ is reached.

The same principle can be translated to separated representations in higher dimension, for instance separation of the different dimensions of the physical space, the phase

space or the parametric space:

$$u(x,y,z,t) \approx \sum_{m=1}^{M} X_m(x)Y_m(y)Z_m(z)T_m(t)$$

$$u(r,\theta,t) \approx \sum_{m=1}^{M} R_m(r)\Theta_m(\theta)T_m(t)$$

$$u(x,\dot{x}) \approx \sum_{m=1}^{M} X_m(x)\tilde{X}_m(\dot{x})$$

$$u(x,t,p^1,..p^d) \approx \sum_{m=1}^{M} X_m(x)T_m(t)\prod_{k=1}^{d} f_m^k(p^k)$$

1.1.3 Proper Orthogonal Decomposition with interpolation

Proper Orthogonal Decomposition with Interpolation (PODI) [56, 57] is the non-intrusive counterpart of POD. The reduced basis is extracted through the same procedure as in intrusive POD, however the reduced coefficients for a new set of parameters are obtained by using an interpolation or regression technique instead of projecting the PDE and associated solution in the lower-dimensional space.

From the matrix of snapshots $\boldsymbol{U}$, the reduced basis $\boldsymbol{\Phi} = (\boldsymbol{\phi_1}, ..., \boldsymbol{\phi_r})$ is calculated using SVD or the so-called snapshot-POD as described in 1.1.1. The reduced coordinates are similarly collected by extracting the first M rows of matrix $\boldsymbol{\Sigma R}$, where the m-th column of this extracted matrix contains the reduced coordinates $\boldsymbol{\alpha_i}$ related to snapshot $\boldsymbol{u_i}$ which allow reconstructing the full solution in an approximated manner: $\boldsymbol{u_i} \approx \boldsymbol{\Phi\alpha_i}$.

Then, an interpolation or regression method is used to construct the function $\boldsymbol{\alpha}(\boldsymbol{p})$ such that for all i, $\boldsymbol{\alpha_i} \approx \boldsymbol{\alpha}(\boldsymbol{p_i})$. This can be performed with interpolation algorithms such as polynomial approximation, piecewise-polynomial interpolation [119], kriging [120], Radial Basis Function (RBF) interpolation [121] ... or regression algorithms such as Moving Least Squares (MLS) [57], Dynamic Mode Decomposition (DMD) [56], Neural Networks (NN) [122], ...

1.1.4 Sparse Proper Generalized Decomposition

The sparse Proper Generalized Decomposition (sPGD) [58] is a non-intrusive reduction method making use of the separation of variables to construct the solution of a parametric problem, which can be efficiently obtained in real time once the training phase is complete. The problem consists in constructing the parametric solution

$$\boldsymbol{u}(p^1,\ldots,p^d) : \Omega \subset \mathbb{R}^d \to \mathbb{R}^N, \tag{1.1.1}$$

that depends on d parameters p^k, $k = 1,\ldots,d$, belonging to the parametric space Ω, in which a sparse sample of n_s points and their corresponding solutions have been collected from high-fidelity simulations.

The sPGD expresses the function $\boldsymbol{u}$ from a low-rank separated representation

$$\boldsymbol{u}(p^1,\ldots,p^d) \approx \tilde{\boldsymbol{u}}^M(p^1,\ldots,p^d) = \sum_{m=1}^{M} \boldsymbol{u}_m \prod_{k=1}^{d} \psi_m^k(p^k), \tag{1.1.2}$$

constructed from greedy rank-one updates. In the previous expression $\tilde{\boldsymbol{u}}^M$ refers to the approximation, M the number of employed modes, $\boldsymbol{u}_m \in \mathbb{R}^N$ are vectors which can be read as spatial modes and ψ_m^k are the one-dimensional functions concerning mode m and dimension k.

Functions ψ_m^k, $m = 1, \ldots, M$ and $k = 1, \ldots, d$ are expressed from a standard approximation basis $\boldsymbol{N}_m^k$, via coefficients $\boldsymbol{a}_m^k$:

$$\psi_m^k(p^k) = \sum_{j=1}^{D} N_{j,m}^k(p^k) a_{j,m}^k = (\boldsymbol{N}_m^k)^T \boldsymbol{a}_m^k, \tag{1.1.3}$$

where D represents the number of degrees of freedom (nodes) of the chosen approximation and $\boldsymbol{N}_m^k$ is the vector collecting the shape functions.

In the context of usual regression the approximation $\tilde{\boldsymbol{u}}^M$ results from

$$\tilde{\boldsymbol{u}}^M = \arg\min_{\boldsymbol{u}^*} \|\boldsymbol{u} - \boldsymbol{u}^*\|_2^2 = \arg\min_{\boldsymbol{u}^*} \sum_{i=1}^{n_s} \|\boldsymbol{u}(\boldsymbol{p}_i) - \boldsymbol{u}^*(\boldsymbol{p}_i)\|^2, \tag{1.1.4}$$

where $\tilde{\boldsymbol{u}}^M$ takes the separated form of Eq.(1.1.2), n_s is the number of sampling points to train the model and $\boldsymbol{p}_i$ the vectors that contain the input data points of the training set. Notice that, to avoid overfitting, the number of basis functions D must be $D < n_s$.

The approximation coefficients of each one-dimensional function are computed by employing a greedy algorithm, such that, once the approximation up to order $M-1$ is known, the M-th order term reads

$$\tilde{\boldsymbol{u}}^M = \sum_{m=1}^{M-1} \boldsymbol{u}_m \prod_{k=1}^{d} \psi_m^k(p^k) + \boldsymbol{u}_M \prod_{k=1}^{d} \psi_M^k(p^k). \tag{1.1.5}$$

The computed function is expected to approximate $\boldsymbol{u}$ not only in the training set but in any point $\boldsymbol{p} \in \Omega$. The main issue is how to ally rich approximations and scarce available data, while avoiding overfitting. For that purpose a modal adaptivity strategy (MAS) was associated to the sPGD, however, it has been observed that the desired accuracy is not achieved before reaching overfitting or the algorithm stops too early when using MAS in some cases. This last issue implies a parametric solution composed of low order approximation functions, thus not getting an as rich as desired function. Some papers describing the just referred techniques are [58, 123].

In addition, in problems where just a few terms of the interpolation basis are present (that is, there are just some sparse non-zero elements in the interpolation basis to be determined), the strategy fails in recognizing the true model and therefore lacks accuracy.

This methodology is presented as a Model Order Reduction technique, but it can also be considered as a regression technique, and it will be used as such in the following chapters.

1.2 Physics-Informed Neural Networks

Physics-Informed Neural Networks (PINN) [13] have emerged as a promising framework that combines the power of neural networks with the fundamental principles of physics to address complex scientific and engineering problems. Traditional machine

learning techniques heavily rely on vast amounts of labeled data, which may not always be available in domains governed by physical laws. PINNs bridge this gap by incorporating prior knowledge of physical laws into the learning process, allowing for accurate predictions and capturing underlying physics even with limited data. This section provides a concise introduction to PINN, describing how the governing equations and constraints are combined with a neural network architecture to leverage data while ensuring physical consistency.

1.2.1 Neural Networks as PDE solvers

Consider a system described by a state $u(x,t)$ governed by a partial differential equation in the form:

$$\begin{cases} \mathcal{L}(u)(x,t) = f(\boldsymbol{p}(x,t)) & \forall (x,t) \in \Omega \times [0,T] \\ u(x,t) = f_{\mathbf{d}}(x,t) & \forall (x,t) \in \Gamma_{\mathbf{d}} \times [0,T] \\ \partial_{\mathbf{n}} u(x,t) = f_{\mathbf{n}}(x,t) & \forall (x,t) \in \Gamma_{\mathbf{n}} \times [0,T] \\ u(x,0) = u_{\mathbf{0}}(x) & \forall x \in \Omega \end{cases} \tag{1.2.1}$$

where $\mathcal{L}$ is a differential operator, f is the loading or source term, $\boldsymbol{p}(x,t)\mathbb{R}^d$ is the vector which contains the properties of the loading or source at point x and time t, Ω is the physical domain, $f_{\mathbf{d}}$ (resp. $f_{\mathbf{n}}$) is the Dirichlet (resp Neumann) boundary condition taking place on $\Gamma_{\mathbf{d}}$ (resp. $\Gamma_{\mathbf{n}}$) and $u_{\mathbf{0}}$ is the initial condition.

When no measurement data is available and all the requirements coming from the field of PDE analysis [124] are met, the problem is usually well-posed in the sense that it has a unique solution that satisfies completely all the constraints.

In the PINN methodology, the solution is approximated by a neural network $\hat{u}(x,t)$ which is trained by minimizing the loss function L_Φ:

$$\mathrm{L}_\Phi = \lambda_{\mathbf{\Omega}} \mathrm{L}_{\mathbf{\Omega}} + \lambda_{\mathbf{d}} \mathrm{L}_{\mathbf{d}} + \lambda_{\mathbf{n}} \mathrm{L}_{\mathbf{n}} + \lambda_{\mathbf{0}} \mathrm{L}_{\mathbf{0}} \tag{1.2.2}$$

where

$$\mathrm{L}_{\mathbf{\Omega}} = \frac{1}{N_{\mathbf{\Omega}}} \sum_{i=1}^{N_{\mathbf{\Omega}}} \left| \mathcal{L}(\hat{u})(x^i_{\mathbf{\Omega}}, t^i_{\mathbf{\Omega}}) - f(\boldsymbol{p}(x^i_{\mathbf{\Omega}}, t^i_{\mathbf{\Omega}})) \right|^2 \tag{1.2.3}$$

$$\mathrm{L}_{\mathbf{d}} = \frac{1}{N_{\mathbf{d}}} \sum_{i=1}^{N_{\mathbf{d}}} \left| \hat{u}(x^i_{\mathbf{d}}, t^i_{\mathbf{d}}) - f_{\mathbf{d}}(x^i_{\mathbf{d}}, t^i_{\mathbf{d}}) \right|^2 \tag{1.2.4}$$

$$\mathrm{L}_{\mathbf{n}} = \frac{1}{N_{\mathbf{n}}} \sum_{i=1}^{N_{\mathbf{n}}} \left| \partial_{\mathbf{n}} \hat{u}(x^i_{\mathbf{n}}, t^i_{\mathbf{n}}) - f_{\mathbf{d}}(x^i_{\mathbf{n}}, t^i_{\mathbf{n}}) \right|^2 \tag{1.2.5}$$

$$\mathrm{L}_{\mathbf{0}} = \frac{1}{N_{\mathbf{0}}} \sum_{i=1}^{N_{\mathbf{0}}} \left| \hat{u}(x^i_{\mathbf{0}}, 0) - u_{\mathbf{0}}(x^i_{\mathbf{0}}) \right|^2 \tag{1.2.6}$$

and $\lambda_{\mathbf{\Omega}}, \lambda_{\mathbf{d}}, \lambda_{\mathbf{n}}, \lambda_{\mathbf{0}} \in \mathbb{R}_+^*$ are the weights associated to each term. Finding appropriate values for these weights is far from an easy task, but several works have addressed this issue, including [125–127].

1.2.2 Neural Networks for hybrid solutions

PINNs are a very appealing tool when trying to combine physics and data. Indeed, a neural network can be built by adding one by one to its loss function all the different constraints it needs to fullfill, whether they come from physical knowledge in the form of equations and boundary conditions, or from measurement data.

In section 1.2.1, only the physics-based model is considered, and the solution obtained with this methodology is the physics-based solution. Now, consider that measurements have been performed on the real system and data has been collected in various locations and at different times: $u_{\mathbf{m}}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}), i = 1, \dots, N_{\mathbf{m}}$. To obtain the hybrid solution taking into account this additional information, the loss function becomes:

$$\mathrm{L}_{\mathcal{H}} = \lambda_{\mathbf{\Omega}}\mathrm{L}_{\mathbf{\Omega}} + \lambda_{\mathbf{d}}\mathrm{L}_{\mathbf{d}} + \lambda_{\mathbf{n}}\mathrm{L}_{\mathbf{n}} + \lambda_{\mathbf{0}}\mathrm{L}_{\mathbf{0}} + \lambda_{\mathbf{m}}\mathrm{L}_{\mathbf{m}} \tag{1.2.7}$$

where

$$\mathrm{L}_{\mathbf{m}} = \frac{1}{N_{\mathbf{m}}} \sum_{i=1}^{N_{\mathbf{m}}} \left| \hat{u}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}) - u_{\mathbf{m}}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}) \right|^2 \tag{1.2.8}$$

and $\lambda_{\mathbf{m}}$ is its associated weight.

Minimizing this loss yields a solution which is a compromise between the physics-based solution and the measurements. However, while this solution may be very useful in many cases, there is no possibility to learn from this what was missing in the modeling which could explain the discrepancy between the physics-based model and reality.

2

Advances in non-intrusive Model Order Reduction

Model Order Reduction (MOR) techniques involve expressing the solution of a problem, typically a partial differential equation (PDE), in a reduced basis with a strong physical or mathematical foundation. This basis is often obtained offline from the results of a high-fidelity solver, such as through the proper orthogonal decomposition (POD) or reduced basis method (RB) [2]. By using a reduced basis, the complexity of the solution scales with the size of this basis, which is typically much smaller than the multi-purpose approximation basis associated with the finite element method (FEM).

While a reduced basis may result in some loss of generality, it can greatly reduce computing time while maintaining accuracy as long as the problem solution falls within the space spanned by the reduced basis. However, accuracy may suffer if the desired solution cannot be accurately approximated by the reduced basis. To improve generality and accuracy, the Proper Generalized Decomposition (PGD) [128, 129] can construct the reduced basis and solve the problem simultaneously, although this approach can be more intrusive.

To address intrusiveness, non-intrusive procedures have been developed. These procedures construct the parametric solution of a parametric problem from a number of high-fidelity solutions performed offline. These solutions can be computationally expensive, as different choices of the model parameters are used to constitute the design of experiments (DoE).

There are various techniques to approximate solutions to parametric problems, including standard polynomial approximations on sparsely sampled domains. However, caution must be taken when using these methods. Orthogonal polynomial bases with Gauss-Lobatto points as the Design of Experiments (DoE) can produce very accurate approximations, but the number of samples required increases exponentially with the number of dimensions and polynomial degree. Using randomly sampled DoE or an overly complex approximation can result in overfitting. To avoid this, one option is to use a basis that avoids these spurious oscillations, such as kriging approximations [130], or to restrict the polynomial approximation to a low degree.

Another approach is to use Proper Orthogonal Decomposition with Interpolation

(PODI) [56, 57], where usual regressions are used to express the dependence of the modal coefficients on the parameters. Sparse Subspace Learning (SSL) [123] interpolates pre-computed solutions related to the DoE over the entire parametric space using a hierarchical approximation basis, which ensures the separated representation of the interpolated parametric solution. However, the volume of data required for SSL increases exponentially with the number of parameters involved, and the use of higher degree approximations with very little data increases the risk of overfitting.

To address this issue, a sparse PGD (sPGD) [58] was proposed, which uses a sparse sampling and adaptive approximation bases to limit over-oscillating behaviors. The sPGD is a nonlinear regression that uses the separation of variables, making it useful for multi-parametric settings. The Modal Adaptive Strategy (MAS) is used by the authors in [58] to keep the degree of the approximation basis to a minimum for the first PGD modes and increase it progressively for higher level modes. Other choices of the approximation bases, such as kriging, can also be used to limit overfitting. This method is described in 1.1.4.

Besides, Model Order Reduction often focuses on the prediction of temporal, spatial or spatio-temporal responses, which could be a lot more complex than scalar quantities of interest and which are often unsuited to "point by point" interpolation or interpolation in a Euclidean space. A classical example of such an issue in the context of MOR is the Kolmogorov barrier [131–133], a phenomenon related to the slow decay of the Kolmogorov n-width [134, 135] in advection-dominated problems, which greatly reduces the efficiency of classical MOR techniques.

Valuable strategies to circumvent these difficulties often rely on curve alignment operations [136, 137] or registration methods [138, 139] which act as a sort of pre-processing step on the data to map the solution to a reference domain where the computations are performed.

This chapter is divided into four sections which explore and attempt to give solutions to the aformentioned challenges. First, the use of separated representations in MOR is combined with regularization methodologies [140] to enhance the power of expression of the technique while reducing the potential overfitting. Subsequently, the ANOVA-PGD is presented, a novel strategy which makes use of the so-called ANOVA decomposition [15] to enable the representation of complex parametric behaviors in MOR. Afterwards, the prediction of curves and temporal responses is tackled through the use of feature alignment and an adapted representation of the data. Lastly, challenges related to advection and localization of features in two or three dimensions are dealt with through the use of Optimal Transport [16].

2.1 Regularization of separated representations

Regressions are widely utilized in the field of artificial intelligence in general, and specifically in supervised scientific machine learning [61, 62, 86] and the development of cognitive or hybrid digital twins [141–143], as well as in neuroscience. Regression serves as a crucial component in automatically constructing models that represent the physical reality around us, which is essential for enabling artificial intelligence to operate in the physical world [144, 145].

When implementing regression with limited data, the main challenge that arises is dealing with nonlinearities in multi-parametric settings. This leads to the curse of

dimensionality, where the number of degrees of freedom or sampling points required in the parametric space increases exponentially in order to achieve accurate results [146].

When constructing models, it is always preferable to keep them as simple as possible. In other words, it is better to have parsimonious models rather than complex ones. This principle, known as Occam's razor [61, 62], suggests that simpler explanations should be favored when explaining any physical phenomenon among the available options. In the literature, this is achieved by promoting sparsity in regression [62, 63]. To obtain parsimonious models that address sparsity, it is beneficial to perform regression by combining L2 and L1 norm regularization.

The objective of this section is to propose regression methodologies that are robust, general, efficient, and accurate, specifically designed for separated representation settings. To achieve this goal, two techniques are presented and analyzed. The first technique is called rs-PGD, and is based on an Elastic Net regularized formulation combining Ridge (L2 norm regularization) and Lasso (L1 norm regularization) regressions. The rs-PGD technique unites these regularizations with a rich approximation basis, and ensures specific solutions with smaller coefficients to prevent overfitting.

Subsequently, the section introduces the s^2-PGD technique, which is a doubly sparse regression method. This technique utilizes Lasso regularization which enforces the sparsest possible solution by selecting the most significant contributors in each dimension to the solution approximation.

Numerical experiments are then provided to show the performance of these two methods compared to the regular sPGD algorithm.

2.1.1 rs-PGD

For the sake of simplicity of representation but without loss of generality, let us continue by assuming that the unknown sought function u is scalar-valued,

$$u(p^1, \dots, p^d) : \mathcal{D} \subset \mathbb{R}^d \to \mathbb{R},$$

and that it is to be recovered from scarce data.

The goal is therefore to find a function $\tilde{u}^M$ which minimizes the distance to the function to be recovered:

$$\tilde{u}^M = \arg\min_{u^*} \sum_{i=1}^{n_s} |u(\boldsymbol{p}_i) - u^*(\boldsymbol{p}_i)|_2^2,$$

and that takes the separated form

$$\tilde{u}^M(p^1, \dots, p^d) = \sum_{m=1}^{M} \prod_{k=1}^{d} \psi_m^k(p^k) = \sum_{m=1}^{M} \prod_{k=1}^{d} (\boldsymbol{N}_m^k)^T \boldsymbol{a}_m^k, \tag{2.1.1}$$

where n_s is the number of sampling points employed to train the model (training set). Here, the superscript M is employed to highlight the rank of the sought function. How to determine the precise value of M will be detailed hereafter.

In the PGD framework, an greedy enrichment algorithm combined with iterative scheme based on an alternating direction strategy is usually used to solve the resulting non-linear problem. At enrichment step m, this strategy initializes randomly all the $\boldsymbol{a}_m^k$, and then proceeds to update them iteratively. At iteration l of the alternating direction strategy, $\boldsymbol{a}_m^l$ is updated by fixing all the $\boldsymbol{a}_m^k$ for $k \neq l$ and minimizing

the residual. The iterations proceed until reaching a fixed point according to a user-specified tolerance.

The following matrix equations define the systems needed to be solved to perform these updates:

$$\boldsymbol{M}_l \boldsymbol{a}_m^l = \boldsymbol{r}, \tag{2.1.2}$$

where:

$$\boldsymbol{r} = \begin{pmatrix} u(\boldsymbol{p}_1) - \tilde{u}^{m-1}(\boldsymbol{p}_1) \\ \vdots \\ u(\boldsymbol{p}_{n_s}) - \tilde{u}^{m-1}(\boldsymbol{p}_{n_s}) \end{pmatrix},$$

$$\boldsymbol{M}_l = \begin{pmatrix} \prod_{k \neq l} \left(\boldsymbol{N}_m^k(p_1^k)\right)^T \boldsymbol{a}_m^k \left(\boldsymbol{N}_m^l(p_1^l)\right)^T \\ \vdots \\ \prod_{k \neq l} \left(\boldsymbol{N}_m^k(p_{n_s}^k)\right)^T \boldsymbol{a}_m^k \left(\boldsymbol{N}_m^l(p_{n_s}^l)\right)^T \end{pmatrix}$$

In the context of sPGD, Eq.(2.1.2) is solved in the Ordinary Least Squares (OLS) setting:

$$\boldsymbol{a}_m^l = (\boldsymbol{M}_l^T \boldsymbol{M}_l)^{-1} \boldsymbol{M}_l \boldsymbol{r}, \tag{2.1.3}$$

The rs-PGD is based on putting a penalty term when solving (2.1.2) in order to reduce overfitting.

It is important to recognize that in the sPGD context, the problem of overfitting can easily arise when employing high-order approximations, which are exacerbated by separated representations. This is primarily due to the typical usage of unstructured low data during the model training process. Overfitting significantly hampers the model's performance when faced with new and unseen datasets. Hence, the concept of utilizing a penalty term aims at enhancing the model's capability to perform well on new samples, even if it results in an increase in bias or error within the training set for a specific set of basis functions.

Different regularizations can be chosen depending on the properties of the problem such as the Tikhonov regularization [147] or the Elastic Net regularization [148, 149].

In this work, the Elastic Net is employed, including in one of its extreme forms, the Ridge regularization (which is also a special case of Tikhonov regularization) and which is here presented in the first place.

For this purpose, Eq.(2.1.3) is rewritten:

$$\boldsymbol{a}_m^l = (\boldsymbol{M}_l^T \boldsymbol{M}_l - \lambda \boldsymbol{I})^{-1} \boldsymbol{M}_l \boldsymbol{r}, \tag{2.1.4}$$

where λ is the penalty factor and $\boldsymbol{I}$ is the identity matrix. In this case, all the dimensions are equally penalized but different penalty factors could be considered depending on the considered dimension.

The regularized problems associated to Eq.(2.1.4) is:

$$\boldsymbol{a}_m^l = \arg\min_{\boldsymbol{a}^*} \left\{ \|\boldsymbol{r} - \boldsymbol{M}_l \boldsymbol{a}^*\|_2^2 + \lambda \|\boldsymbol{a}^*\|_2^2 \right\}, \tag{2.1.5}$$

where the problem is divided in solving a ridge regression problem for each dimension when computing $\boldsymbol{a}_m^l$ during the alternating direction fixed point strategy.

While a null intercept term is assumed in the deduction of equations (2.1.4) and (2.1.5) it could easily be includedand treated as in standard ridge procedures when solving the corresponding linear regularized regression problem for each dimension during the alternating direction strategy.

Since one is generally looking for the mode with best predictive abilities in each enrichment, the proposed criterion to choose λ is to perform a k-fold cross-validation [150, 151] and select the value of λ that minimizes the cross-validated sum of squared residuals (or some other measure). It is also possible to use the "one-standard error" rule (heuristic) with cross-validation, in which the chosen model is the most penalized model which has an error that is no more than one standard error above the error of the best model. Such a rule acknowledges the fact that the tradeoff curve is estimated with error, and hence takes a conservative approach [152].

As the terminology used in this section shows, a regularization problem is formulated at each enrichment step. Thus, the best penalty factor is obtained at each updating stage, adapting the regularization whenever the approach is enriched. Other options have been investigated but numerical experiments suggest that this option is the most promising one.

If enough data is available, the split of the training set in two subgroups is equally a reasonable option to select λ and in addition, computationally less demanding. In this case, one subgroup is employed to construct the model and the other one to evaluate the predictive ability and then to select λ accordingly.

The Elastic Net regularization is obtained by adding an additional L1-norm regularization, from which Eq.2.1.5) becomes:

$$\boldsymbol{a}_m^l = \arg\min_{\boldsymbol{a}^*} \left\{ \|\boldsymbol{r} - \boldsymbol{M}_l \boldsymbol{a}^*\|_2^2 + \alpha\lambda_2 \|\boldsymbol{a}^*\|_2^2 + (1-\alpha)\lambda_1 \|\boldsymbol{a}^*\|_1 \right\}, \tag{2.1.6}$$

where $\alpha \in [0,1]$ and λ_2 and λ_1 are the penalty coefficients that affect the L2 and L1 norm penalization terms respectively. Once again, these coefficients could also be different for the different dimensions. The limit cases $\alpha = 0$ and $\alpha = 1$ result in the Ridge and Lasso regressions respectively.

2.1.2 s^2-PGD

For the sake of simplicity of representation but without loss of generality, let us continue by assuming the same scalar unknown function discussed in section 2.1.1.

In this case, the sought solution admits a sparse solution for a certain basis using the PGD separated form. The goal is therefore to identify the correct non-zero coefficients at each enrichment step in order to guide the approach to the correct separated representation.

Without a roadmap to select these nonzero coefficients, the traditional sPGD fails to capture the true relationship between the model's features as well as its final response. Furthermore, if high-order terms appear in the searched function, this issues become even worse leading to serious overfitting issues.

Let us consider the theory discussed in the previous section but now considering the L1 regularization with the aim of promoting sparsity in the coefficients of the solution.

The L1 regularization is convenient because the nonlinear problem can be solved using the PGD constructor, with an alternating direction fixed point strategy, where just a LASSO regression problem is considered in each dimension.

Therefore, the regression problems for the iterative scheme will be:

$$\boldsymbol{a}_m^l = \arg\min \boldsymbol{a}^* \Big\{ \|\boldsymbol{r} - \boldsymbol{M}_l \boldsymbol{a}^*\|_2^2 + \lambda \|\boldsymbol{a}^*\|_1^2 \Big\}, \tag{2.1.7}$$

that consists in solving a LASSO regression problem for each dimension when computing $\boldsymbol{a}_m^l$ within the alternate direction fixed point strategy. Moreover, as previously discussed, in the present case again, all the dimensions are equally penalized but different penalty factors could be used.

As the LASSO problem are iteratively solved in each dimension, each one-dimensional function has a sparse representation with respect to its base choosing the right penalty factor. Again, a null intercept term is assumed in the formulation, but the penalty term could be non zero in a more general setting.

In the case of looking for sparsity only in some of the dimensions, Eq.(2.1.7) only applies in the computation of the considered dimensions, whereas the other dimensions are treated by invoking the standard sPGD or its ridge regularization counterpart, *rs*-PGD, addressed in the previous section.

To determine λ, the discussion of the previous section still applies. However, the following considerations can also be applied in the case of the s^2-PGD:

- Before selecting a model according to the predictive criterion, a filter is considered taking only the models with a minimum sparsity criterion $\|\vec{a}_m^l\|_0 \leq \chi^{lim}$ for the dimensions in which the sparsity is desired. Note: $\|\bullet\|_0$ is defined by $\|\boldsymbol{x}\|_0 = \#\{k : x_k \neq 0\}$ and is refered to as "L0 norm", although it is actually not a norm.

- Once model selection is performed, the OLS methodology is employed with the detected non-zero elements to obtain the correct update. The reason of this step is that LASSO regression terms are in general not accurate, and so it may be necessary to de-bias the obtained values. This is because the LASSO shrinkage causes the estimates of the non-zero coefficients to be biased towards zero and in general they are not consistent [152, 153].

2.1.3 Numerical experiments

This section presents the results obtained from employing the aforementioned techniques in various scenarios. Initially, the reduction in error is demonstrated by comparing the *rs*-PGD approach with the classical procedure (sPGD). Then, the identification of sparsity and error reduction is showcased by contrasting the s^2-PGD method with the standard sparse procedure (sPGD).

Results for the *rs*-PGD approach

The following examples consider the utilization of Elastic Net Regularization. To accomplish this, the parameter α that cotronls the relative importance between Ridge and Lasso regression is employed. α is determined by executing the algorithm multiple times with various α values, and subsequently selecting the value that yields superior predictive performance.

In the first example, the method is applied to try to approximate the five-dimensional function

$$f(x_1, x_2, x_3, x_4, x_5) = (8x_1^3 - 6x_1 - 0.5x_2)^2 + (4x_3^3 - 3x_3 - 0.25x_4)^2 + 0.1(2x_5^2 - 1). \quad (2.1.8)$$

The objective is to reconstruct the aforementioned function within the domain $\mathcal{D} = [-0.51, 0.51]^5$. The training set consists of 160 points, and thus, only these points are utilized to construct the model using either the sPGD or the rs-PGD methodology. The Latin hypercube sampling (LHS) technique is employed to generate this set of data.

In contrast, a testing set comprising 54000 untrained points is considered to compare the results between the techniques when predicting unseen scenarios. This second set enables the evaluation of the predictive capability of both models once they are constructed.

Concerning the sPGD, a standard MAS is employed, reaching 4th degree polynomials. To measure the error of both methodologies in the testing set, the following error criterion is used:

$$\text{err}_{pgd} = \frac{\|\boldsymbol{f} - \boldsymbol{f}_{pgd}\|_2}{\|\boldsymbol{f}\|_2}; \qquad \text{err}_{rpgd} = \frac{\|\boldsymbol{f} - \boldsymbol{f}_{rpgd}\|_2}{\|\boldsymbol{f}\|_2}$$

where $\boldsymbol{f}$ is the vector containing the values of $f(x_1, x_2, x_3, x_4, x_5)$ in the testing set, $\boldsymbol{f}_{pgd}$ and $\boldsymbol{f}_{rpgd}$ are the vectors containing the prediction in the testing set of both methodologies (sPGD and rs-PGD, respectively).

After employing the discussed techniques in the above conditions, in this example the error is reduced by 52.38 % using the rs-PGD with $\alpha = 0.1$.

To perceive the improvements and the overfitting reduction, Figure 2.1 shows a plot of the original function $f(x_1, x_2, x_3 = 0, x_4 = 0, x_5 = 0.7071)$. It can be noticed that the rs-PGD corrects the shape of the function in the indicated areas in Fig. 2.1, improving the performance of the regression.

This improvement occurs over the whole five-dimensional domain. Another part of the domain is shown in Figure 2.2 that depicts $f(x_1, x_2, x_3 = -0.17069, x_4 = -0.17069, x_5 = -0.015517)$.

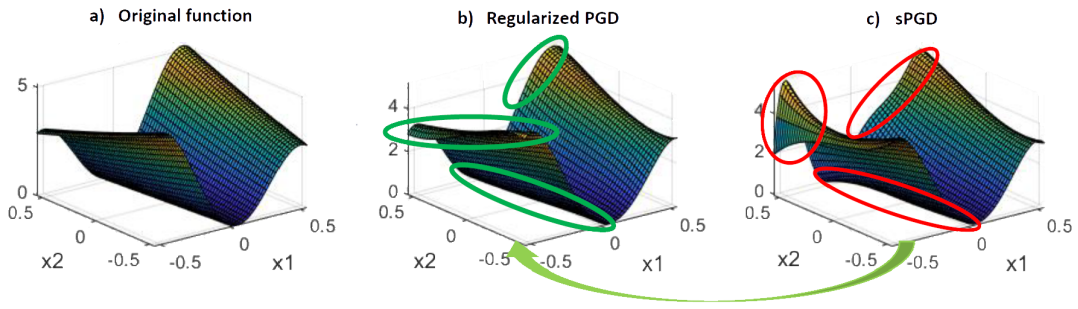


FIGURE 2.1: Comparing the reference Eq.2.1.8 and its associated sPGD and rs-PGD regressions, at points $(x_1, x_2, x_3 = 0, x_4 = 0, x_5 = 0.7071)$

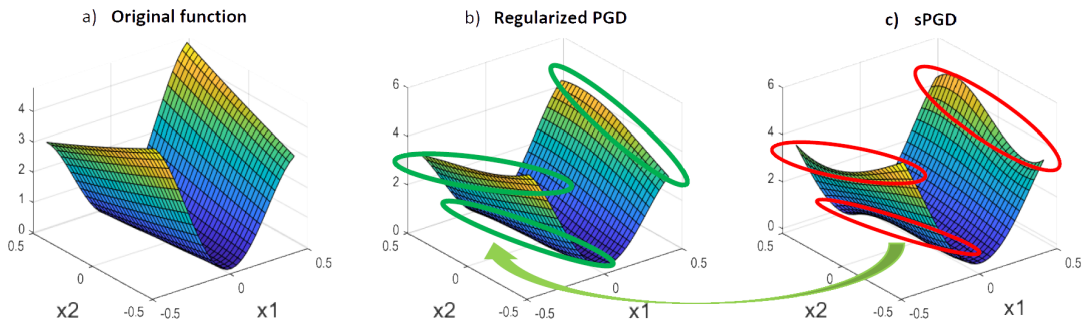


FIGURE 2.2: Comparing the reference Eq.2.1.8 and its associated sPGD and *rs*-PGD regressions, at points $(x_1, x_2, x_3 = -0.17069, x_4 = -0.17069, x_5 = -0.015517)$

As a second example, the following function is considered:

$$f(x_1,x_2,x_3,x_4,x_5) = \cos(x_1x_2)\Big[\big(\sin(2x_3)-3.14\big)\log(3x_4+1.5)\cos(x_5) + \exp(x_4)\cosh(x_3)\sinh(x_5)\Big], \tag{2.1.9}$$

by using the *rs*-PGD with polynomials. The above function is intended to be reconstructed in the domain $\mathcal{D} = [-1,1]^5$.

In this particular scenario, the training set is comprised of 200 points. Consequently, only these points are utilized to train the model, employing either the sPGD or the *rs*-PGD methodology. The Latin hypercube sampling method is employed to generate this specific dataset.

Additionally, a testing set consisting of 2000 untrained points is available to facilitate a comparison of the results when predicting unseen scenarios.

Concerning the sPGD, again a standard MAS is used up-to fourth-degree polynomials. An error reduction of about 47% is accomplished with $\alpha = 0.5$.

RESULTS FOR THE s^2-PGD APPROACH

In the first numerical example for s^2-PGD, the considered function is:

$$f(x_1,x_2,x_3) = (\sin(2x_1)-3.14)T_5(x_2) + \exp(x_3)\cosh(x_1), \tag{2.1.10}$$

by using a Chebyshev basis for the one-dimensional functions of the PGD. The above function is intended to be reconstructed in the domain $\mathcal{D} = [-1,1]^3$.

Furthermore, the training set is sampled using a sparse grid based on the Smolyak quadrature rule [83, 154] with a level 3, employing the Clenshaw-Curtis univariate quadrature rule. Consequently, only these points are used for constructing the model, employing either the sPGD or the s^2-PGD methodology. Figure 2.3 illustrates the mesh utilized for the training set.

On the other hand, a testing set consisting of 27000 untrained points is available to compare the results between techniques when predicting unseen scenarios. This second set enables the evaluation of the predictive ability of both models once they are constructed.

In this example, the conditions for employing the s^2-PGD are as follows: a basis that encompasses eighth-degree polynomials is selected for the sparse dimensions.

Additionally, a standard MAS-based s-PGD is employed, utilizing fourth-degree polynomials along the non-sparse dimensions. To detect sparsity, three simulations of the s^2-PGD are conducted, penalizing a different dimension at each iteration. Consequently, the model with the best predictive ability (outside the training set) is selected. As anticipated, the chosen model corresponds to the one obtained when penalizing the x_2 dimension.

Figure 2.4 presents the outcomes obtained from the standard sPGD methodology. In this case, it is evident that the predictions are poor, as this methodology fails to identify the desired sparse solutions. This is one of the challenges faced by the sPGD approach, which s^2-PGD aim to address.

Moreover, upon observing the sPGD solution, it is apparent that all possible elements are non-zero, indicating a failure in identifying sparsity.

Figure 2.5 showcases the results of the s2-PGD methodology. As observed, the predictions are nearly perfect. Upon examining the solution, it becomes apparent that the model correctly identifies four modes, representing four sums of the PGD decomposition.

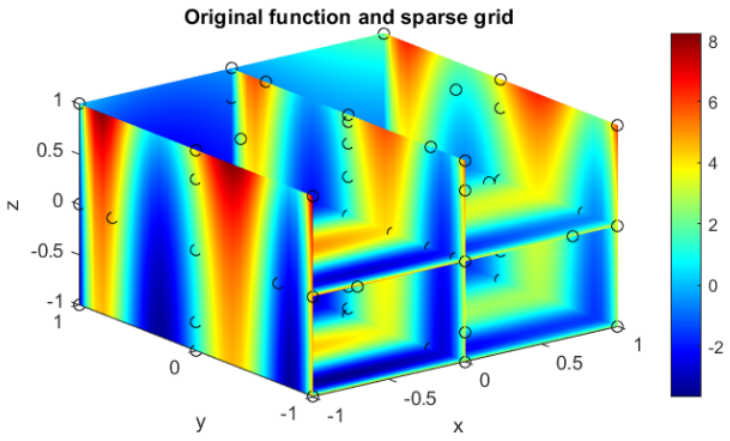


FIGURE 2.3: Plot of the original function and the training set (circles) used to construct the PGD models.

As a second test case, the approximation of the following function is tackled:

$$\begin{aligned} f(x_1,x_2,x_3,x_4,x_5) = &\big[T_5(x_1)+2T_1(x_1)\big]\big[T_2(x_2)+2T_4(x_2)\big] \\ &\Big[\big(\sin(2x_3)-3.14\big)\log(3x_4+1.5)\cos(x_5) \\ &+\exp(x_4)\cosh(x_3)\sinh(x_5)\Big] \end{aligned} \tag{2.1.11}$$

by using a Chebyshev approximation basis for the one-dimensional functions involved in the PGD constructor.

The objective is to reconstruct the given function within the domain $\mathcal{D}=[-1,1]^5$. The sampling for the training set contains 200 points. In addition, the Latin hypercube sampling is used to generate this random set of data.

The training set comprises 290 points, and the Latin hypercube sampling is utilized to generate this randomized dataset. Additionally, a testing set consisting of 2000

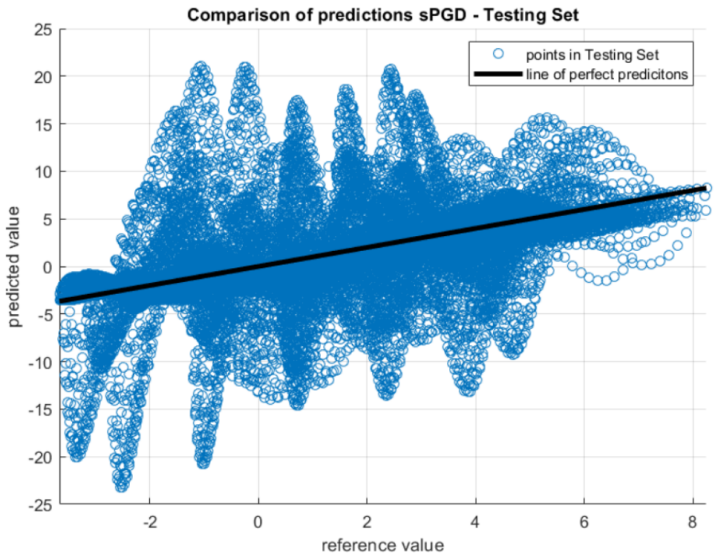


FIGURE 2.4: Problem defined in Eq.(2.1.10): Comparison of predicted sPGD values with the reference ones in the testing set (the black line represents a perfect prediction)

untrained points is available to compare the results between techniques when predicting unseen scenarios. As with previous examples, this second set will be employed to evaluate the predictive ability of both models once they are constructed.

Regarding the s^2-PGD methodology, a basis consisting of sixth-degree polynomials is chosen for the sparse dimensions. Moreover, a standard MAS (Model-Adapted Sparse) approach is utilized, employing up to fourth-degree polynomials in the non-sparse dimensions.

Figure 2.6 illustrates the outcomes obtained from the standard sPGD methodology. In this case, the predictions are unsatisfactory due to the incorrect identification of non-zero elements in the separated representation, leading to overfitting issues.

To identify sparsity, five separate simulations of the s^2-PGD are conducted, penalizing one dimension at a time. Consequently, the model with the best predictive ability (outside the training set) is selected. As expected, the chosen model corresponds to the one obtained when penalizing the x_1 dimension. This choice is made because it correctly identifies the non-zero terms for x_1 and x_2 when penalizing x_1 alone.

Figure 2.7 presents the results of the s^2-PGD methodology, showcasing excellent agreement between the real function and the proposed approach. Upon examining the s^2-PGD solution, it becomes apparent that the model has accurately identified the non-zero elements. Additionally, this PGD solution requires 104 modes, representing 104 sums of the PGD decomposition, which can be further compressed by invoking the PGD again [129].

Finally, the errors for the sPGD and s^2-PGD solutions are reported as $\text{err}_{pgd} = 46.39\%$ and $\text{err}_{s2pgd} = 2.4\%$ respectively.

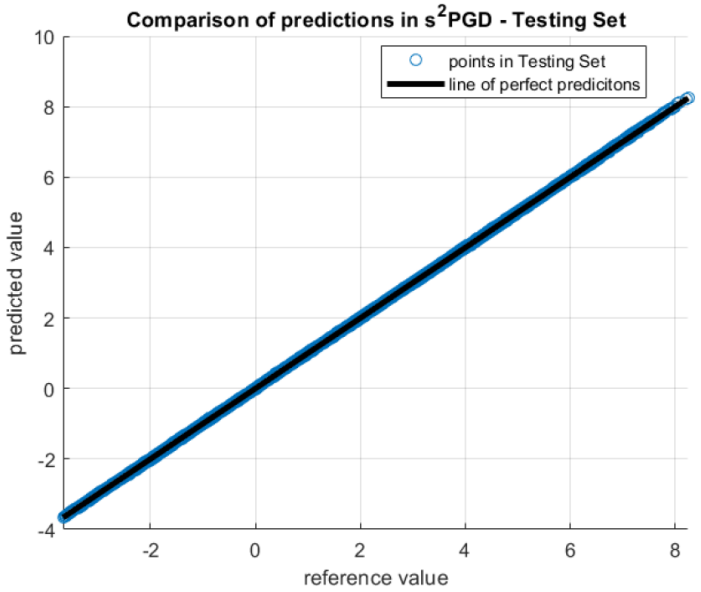


FIGURE 2.5: Problem defined in Eq.(2.1.10): Comparison of predicted s^2-PGD values with the reference ones in the testing set (the black line represents a perfect prediction)

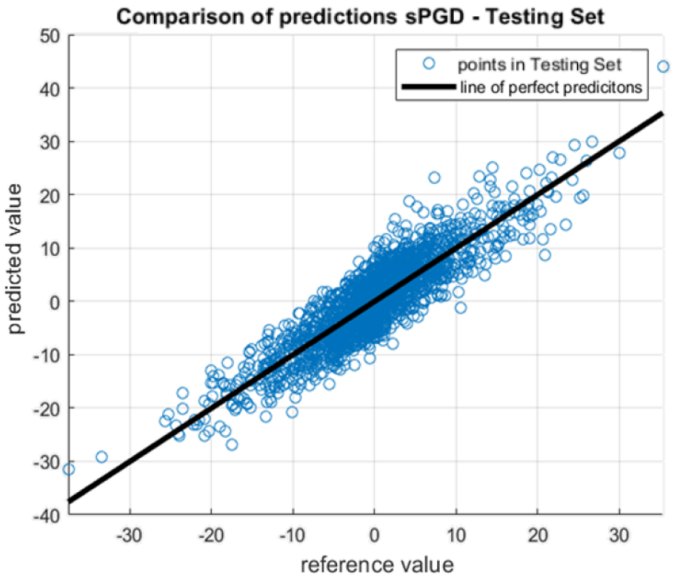


FIGURE 2.6: Problem defined in Eq. (2.1.11): Comparison of predicted sPGD values with the reference ones in the testing set (the black line represents a perfect prediction)

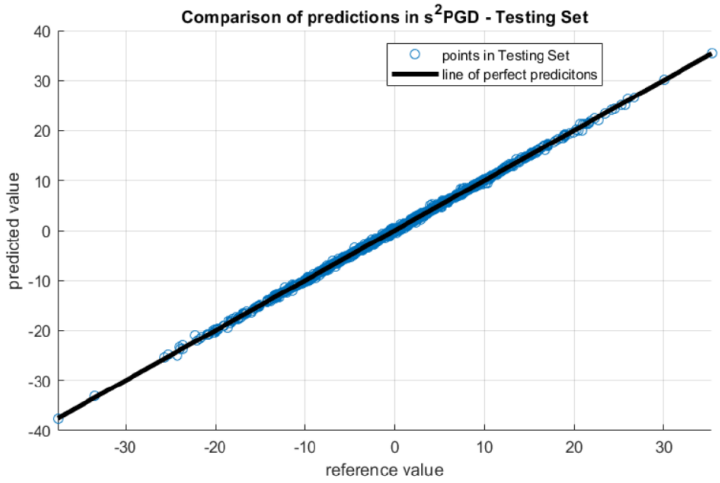


FIGURE 2.7: Problem defined in Eq. (2.1.11): Comparison of predicted s^2-PGD values with the reference ones in the testing set (the black line represents a perfect prediction)

2.2 ANOVA-PGD: A NOVEL STRATEGY

In section 2.1, it has been shown that applying regularization techniques to the sPGD is an appealing route to improve its prediction capabilities when dealing with difficult regression problems. However, the workflow thus created introduces new challenges and questions which should be investigated. For instance, in the context of s^2-PGD, the algorithm is looking to find a solution that has a sparse representation in the basis used to build the one-dimensional functions, therefore the choice of the basis becomes an important matter. Unlike in sPGD, where only the mathematical space spanned by the basis is important, in s^2-PGD, choosing the right basis functions has a substantial impact on the quality of the solution. Furthermore, the PGD-based regression methodologies are all employing sparse design of experiments and no work has been performed to try to make use of a smart sampling technique. Although this is convenient when working with randomly generated datasets or constrained samplings, the idea of trying to make the most of the sampling method is unexploited potential. These two remarks motivate the development of a new PGD-based regression or non-intrusive Model Order Reduction technique : the so-called ANOVA-PGD. This novel method enhances the techniques presented in sections 1.1.4 and 2.1 by combining them with a strategy based on the anchored ANOVA decomposition. It employs a hierarchical orthogonal basis associated to an adequate sampling, which has the particularity of scaling linearly with the number of parameters involved in the problem. This section is divided in three parts. To begin with, the ANOVA decomposition [64–66] and its low-cost approximation, the anchored ANOVA decomposition [67–69], are introduced. Then, two versions of the ANOVA-PGD strategy are presented, the first one for the prediction of scalar qunatity of interests and the second one for vector and full-field outputs in the context of non-intrusive MOR. Finally, numerical experiments are performed to examine the performance of this method compared to other Machine Learning and non-intrusive MOR methods.

2.2.1 ANOVA decomposition and anchored ANOVA

The ANOVA decomposition of a function $u(p^1,\dots,p^d) : \mathcal{D} = \mathcal{D}_1 \times \cdots \times \mathcal{D}_d \subset \mathbb{R}^d \to \mathbb{R}$, where the $\mathcal{D}_k$ are intervals of $\mathbb{R}$, is an orthogonal decomposition based on the analysis of variance [14, 15], a statistical model designed for data analysis. Thus, the function $u(\boldsymbol{p})$ can be written as a sum of orthogonal functions:

$$u(\boldsymbol{p}) = u_0 + \sum_{k=1}^{d} u_k(p^k) + \sum_{k_1=1}^{d}\sum_{k_2=k_1+1}^{d} u_{k_1,k_2}(p^{k_1},p^{k_2}) + \cdots + u_{1,2,\dots d}(p^1,p^2,\dots,p^d), \tag{2.2.1}$$

verifying

$$\mathbb{E}_k(u_{k_1,\dots,k_b}(p^{k_1},\dots,p^{k_b})) = 0, \tag{2.2.2}$$

where $\mathbb{E}_k$ refers to the expectation with respect to any coordinate k in the set $(k_1,\dots,k_b)$, $1 \le b \le d$. This property results in the orthogonality of functions involved in the previous decomposition.

To prove it, consider for example a simple 2D case with, $\boldsymbol{p} = (p^1,p^2)$, $u(\boldsymbol{p}) = u(p^1,p^2)$. Thus, with $\mathbb{E}_1(u_1(p^1)) = 0$, $\mathbb{E}_1(u_{1,2}(p^1,p^2)) = 0$ and $\mathbb{E}_2(u_{1,2}(p^1,p^2)) = 0$, it results $\mathbb{E}_{1,2}(u_{1,2}(p^1,p^2)u_1(p^1)) = \mathbb{E}_1[\mathbb{E}_2(u_{1,2}(p^1,p^2))\ u_1(p^1)] = 0$.

The number of functions involved in this decomposition (without considering the constant term) is 2^d-1, and they can be parametrized by the integer n, $n = 1,\dots,2^d-1$. The different functions involved in the ANOVA decomposition are linked to the conditional expectations according to:

$$\begin{cases} \mathbb{E}(u) = u_0 \\ \mathbb{E}(u|p^k) = u_k(p^k) + u_0 \\ \mathbb{E}(u|p^k,p^l) = u_{k,l}(p^k,p^l) + u_k(p^k) + u_l(p^l) + u_0 \\ \vdots \end{cases} \tag{2.2.3}$$

hence the functions involved in the ANOVA decomposition can be expressed from the expectations in the following manner:

$$\begin{cases} u_0 = \mathbb{E}(u) \\ u_k(p^k) = \mathbb{E}(u|p^k) - u_0 \\ u_{k,l}(p^k,p^l) = \mathbb{E}(u|p^k,p^l) - u_k(p^k) - u_l(p^l) - u_0 \\ \vdots \end{cases} \tag{2.2.4}$$

where $\mathbb{E}(u|p^k,p^l)$ refers to the integration with respect to all the variables except p^k and p^l.

Variance-based sensitivity analysis

The variance of u, $\mathtt{Var}(u)$, taking into account the orthogonality of the functions involved in the ANOVA decomposition, reads

$$\mathtt{Var}(u) = \sum_{n=1}^{2^d-1} \mathbb{E}\big(u_n^2\big) = \sum_{n=1}^{2^d-1} \mathtt{Var}(u_n), \tag{2.2.5}$$

that allows defining the so-called Sobol sensitivity coefficients $\mathcal{S}_n$

$$\mathcal{S}_n = \frac{\texttt{Var}(u_n)}{\texttt{Var}(u)}. \tag{2.2.6}$$

These coeficients allow to evaluate the relative importance of all the functions in the decomposition and give a sense of the sensitivity of the function to each of its parameters and to the interactions between any subset of the parameters.

By construction,

$$\sum_{n=1}^{2^d-1} \mathcal{S}_n = 1. \tag{2.2.7}$$

Anchored ANOVA

Computing all the terms in the ANOVA decomposition implies evaluating many integrals in a high-dimensional spaces, which becomes extremely expensive whenever the number of variables is higher than 2 or 3. To alleviate those costly computations, a common approximation is performed by introducing the so-called anchor point $\boldsymbol{c} \in \mathcal{D}$ and evaluating all the expectations with the Dirac measure $\delta_{\boldsymbol{c}}$. As a conquence, computing integrals is equivalent to particularizing the function in the anchor point, except for the coordinates which are not integrated upon. For instance:

$$\begin{cases} \mathbb{E}(u) = u(\boldsymbol{c}) \\ \mathbb{E}(u|p^k) = u(c^1, c^2, \dots, p^k, \dots, c^d) \\ \vdots \end{cases} \tag{2.2.8}$$

The point $\boldsymbol{c}$ can be any point of the domain. It is typically chosen as the center of the domain, but it is also very convenient to define it as the nominal point of the problem when it exists.

2.2.2 ANOVA-PGD

The principle of ANOVA-PGD is to combine the ANOVA decomposition with the PGD-based techniques described in section 2.1. First, the standard anchored ANOVA is used to evaluate the constant term and the one-dimensional functions depending on each parameter $u_k(p^k)$, $k = 1, \dots, d$, by using an adequate sampling, a sort of multidimensional cross centered at the anchor point $\boldsymbol{c}$. Since these problems are one-dimensional, the sampling covers the space quite well, and interpolation methods that fail in high-dimensional problems are well-suited here. Hence, for each parameter, $u_k(p^k)$ can be approximated with piecewise polynomials, splines, or gaussian processes [84, 155] for instance. Then, the residual is evaluated in sampling points which fall outside of the multidimensional cross, and the sPGD, rs-PGD, or s^2-PGD is applied to that residual, which contains the different interactions between the parameters. In that case, an enhanced sparse-sampling can be considered, trying to sample as well as possible the area near the borders of the parametric domain, since the center is already well handled by the ANOVA terms.

Design of experiments (DoE)

Since ANOVA-PGD requires calculating the expressions defined by the anchored ANOVA, a specific design of experiments is needed. Indeed, the solution needs to be evaluated in points which belong to a multidimensional cross $\mathcal{D}_{\boldsymbol{c}}$ centered in $\boldsymbol{c}$. This cross is constructed from a set of one-dimensional domains $\mathcal{D}_{\boldsymbol{c}}^k$ which are referred to as branches in the following manner:

$$\mathcal{D}_{\boldsymbol{c}}^k = \left\{ \boldsymbol{p} \in \mathcal{D}, \forall l \neq k, p^l = c^l \right\} \tag{2.2.9}$$

and

$$\mathcal{D}_{\boldsymbol{c}} = \bigcup_{k=1}^{d} \mathcal{D}_{\boldsymbol{c}}^k. \tag{2.2.10}$$

The multidimensional cross is represented in Fig. 2.8.

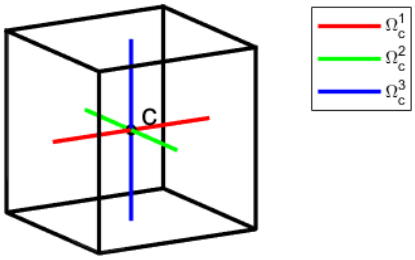


Figure 2.8: Multidimensional cross and its branches for $d = 3$

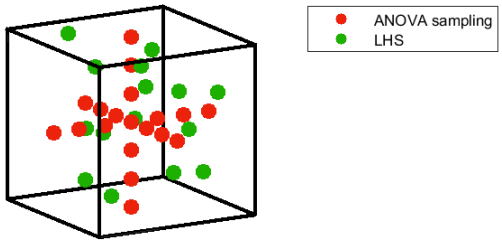


Figure 2.9: Example of a design of experiments for $d = 3$

The design of experiments is then constructed by combining the anchor point, a sampling of each branch and a Latin Hypercube Sampling. The combination of the anchor point and the samplings from all the branches, which is a discretization of $\mathcal{D}_{\boldsymbol{c}}$, is referred to as the ANOVA sampling. The n_k points which sample the branch related to parameter p^k are written $\boldsymbol{p}_{k,j}$, $j = 1, \ldots, n_k$ and are "sorted" such that $\forall (j_1, j_2), j_1 < j_2 \implies p_{k,j_1}^k < p_{k,j_2}^k$.

An example of such a design of experiments is shown in Fig. 2.9.

ANOVA-PGD for scalar quantities of interest

The ANOVA-PGD strategy is first detailed in its version aiming at predicting a scalar quantity of interest u which depends on d variables:

$$u(p^1, \ldots, p^d) : \mathcal{D} \subset \mathbb{R}^d \to \mathbb{R}.$$

It should be noted that the parametric space is required to be a hypercube: $\mathcal{D} = \bigtimes_{k=1}^{d} \mathcal{D}_k$.

Data for u is first collected according to the design of experiments strategy described above. The parameter values in the design of experiments are $\boldsymbol{p}_i$, $i = 1, \ldots, n_s$, among which are the adequate $\boldsymbol{p}_{k,j}$ and the anchor point, $\boldsymbol{p}_0 = \boldsymbol{c}$.

The solution is searched in the form:

$$\begin{cases} \tilde{u}(p^1,\dots,p^d) = u_{\text{ANOVA}}(p^1,\dots,p^d) + u_{\text{PGD}}(p^1,\dots,p^d) \\ u_{\text{ANOVA}}(p^1,\dots,p^d) = u_0 + \sum_{k=1}^{d} u_k(p^k) \\ u_{\text{PGD}}(p^1,\dots,p^d) = \sum_{m=1}^{M}\prod_{k=1}^{d} \psi_m^k(p^k) \end{cases} \tag{2.2.11}$$

The different functions are computed according to:

1. u_0 is obtained from $u(\boldsymbol{p}_0) = u(\boldsymbol{c})$.
2. Each function u_k is constructed by setting $u_k(p^k_{k,j}) = u(\boldsymbol{p}_{k,j}) - u_0$ for $j = 1,\dots,n_k$ and interpolating between the points using a spline or a gaussian process regression.
3. The residual is computed by removing the "ANOVA" term from all the available data: $\forall i = 1,\dots,n_s, r_i = u(\boldsymbol{p}_i) - u_{\text{ANOVA}}(\boldsymbol{p}_i)$. It can be noted that $\forall i = 1,\dots,n_s, \boldsymbol{p}_i \in \mathcal{D}_{\boldsymbol{c}} \implies r_i = 0$.
4. The function u_{PGD} is then obtained using a method described in 2.1 to predict the residual r.

Once these steps are completed, $\tilde{u}$ can be evaluated for any $\boldsymbol{p} \in \mathcal{D}$ to approximate u.

ANOVA-PGD for vector and full-field outputs

The function of interest is again a function defined in a hypercube of dimension d, the only difference being that it is now vector-valued.

$$\boldsymbol{u}(p^1,\dots,p^d) : \mathcal{D} \subset \mathbb{R}^d \to \mathbb{R}^N.$$

The same design of experiments is built and data is collected accordingly. The solution now takes the form:

$$\begin{cases} \tilde{\boldsymbol{u}}(p^1,\dots,p^d) = \boldsymbol{u}_{\text{ANOVA}}(p^1,\dots,p^d) + \boldsymbol{u}_{\text{PGD}}(p^1,\dots,p^d) \\ \boldsymbol{u}_{\text{ANOVA}}(p^1,\dots,p^d) = \boldsymbol{u}_0 + \sum_{k=1}^{d} \boldsymbol{u}_k(p^k) \\ \boldsymbol{u}_{\text{PGD}}(p^1,\dots,p^d) = \sum_{m=1}^{M} \boldsymbol{\phi}_m \prod_{k=1}^{d} \psi_m^k(p^k) \end{cases} \tag{2.2.12}$$

The workflow to construct the different functions is similar to the one introduced above, but adapted to the vector form of the solution. Moreover, two additional changes take into account some considerations which appear when dealing with vector outputs:

- the problem may be more complex than for a scalar output because of the presence of features in the vectors, which calls for caution: a selection criterion is introduced to decide whether or not the different terms defined in the ANOVA decomposition are relevant to predict the output;

- the increased volume of data available allows to extract information which should improve the performance of the algorithm: the one-dimensional setting empowered by the design of experiments yields N one-dimensional functions of each parameter, from which an adequate basis can be extracted and then used for the PGD method.

The construction of the functions thus follows the following steps:

1. $\boldsymbol{u}_0$ is obtained from $\boldsymbol{u}(\boldsymbol{p}_0) = \boldsymbol{u}(\boldsymbol{c})$.

2. One-dimensional functions $\hat{\boldsymbol{u}}_k$ are constructed by setting $\hat{\boldsymbol{u}}_k(p^k_{k,j}) = \boldsymbol{u}(\boldsymbol{p}_{k,j}) - \boldsymbol{u}_0$ for $j = 1, \dots, n_k$ and interpolating between the points using a spline or a gaussian process regression.

3. $\boldsymbol{u}_k$ is constructed by checking whether it has a relevant role in the prediction of the output:

$$\boldsymbol{u}_k = \begin{cases} \hat{\boldsymbol{u}}_k & \text{if } \sum_{i=1}^{n_s} \left\| \boldsymbol{u}(\boldsymbol{p}_i) - \hat{\boldsymbol{u}}_k(p_i^k) \right\| \leq \sum_{i=1}^{n_s} \|\boldsymbol{u}(\boldsymbol{p}_i)\| \\ \boldsymbol{0} & \text{otherwise.} \end{cases} \tag{2.2.13}$$

4. The residual is computed by removing the "ANOVA" term from all the available data: $\forall i = 1, \dots, n_s, \boldsymbol{r}_i = \boldsymbol{u}(\boldsymbol{p}_i) - \boldsymbol{u}_{\text{ANOVA}}(\boldsymbol{p}_i)$. It can be noted that $\forall i = 1, \dots, n_s, \boldsymbol{p}_i \in \mathcal{D}_{\boldsymbol{c}} \implies \boldsymbol{r}_i = \boldsymbol{0}$.

5. For each $k = 1, \dots, d$, the snapshot matrix $\boldsymbol{U}_k \in \mathbb{R}^{N \times n_k}$ is built by collecting the snapshots $\boldsymbol{u}(\boldsymbol{p}_{k,j})$ for $j = 1, \dots, n_k$. The Singular Value Decomposition is performed as described in 1.1.1. The D_k right singular vectors (columns of matrix $\boldsymbol{R}$) which correspond to the largest singular values are extracted. The number of vectors D_k is chosen according to an "energy" criterion on the singular values, as usual in POD. These vectors are interpolated in $\mathcal{D}_k$ in a similar fashion as the one-dimensional functions of the ANOVA decomposition, and form the basis $\boldsymbol{N}^k_m$ as presented in 1.1.4. Index m in $\boldsymbol{N}^k_m$ refers to the fact that, when using the Modal Adaptivity Strategy, some of the basis functions are set to 0 in early iterations.

6. The function $\boldsymbol{u}_{\text{PGD}}$ is then obtained using a method described in 2.1 to predict the residual $\boldsymbol{r}$ using the one-dimensional function bases $\boldsymbol{N}^k_m$ obtained in the previous step.

Using a basis extracted from the data in the PGD-based regression improves significantly the performance of the method when the searched functions can not be well described with a polynomial representation. A very usual example of this is when one of the parameters is a physical angle in the system; its associated responses are usually periodic functions that are very poorly decscribed with low-dimensional polynomials, especially if they are high-frequency functions. This behavior can be successfully extracted using the strategy detailed above, under the condition that enough points are used in the discretization of $\mathcal{D}^k_{\boldsymbol{c}}$. Therefore if a parameter p^k is expected to yield rich responses, its associated branch $\mathcal{D}^k_{\boldsymbol{c}}$ should be sampled with a finer discretization than for the other parameters. For instance geometry parameters tend to require richer functions than material properties such as conductivity, Young's modulus, ...

2.2.3 Numerical experiments

This section presents a few examples which aim at testing the performance of a few variants of the ANOVA-PGD algorithm, and comparing them with other regression methods.

Simple example

For the first example, a very simple case aiming at showcasing the power of ANOVA-based regression, a 2D function is considered:

$$f(x,y) = -2\cos(3x^{1.75}) + 10\log(y-0.6)^4 + 6\cos(x)(y-0.3y^2), \tag{2.2.14}$$

that perfectly fits the ANOVA structure, because it contains complex terms which depend on each coordinate independently, $2\cos(3x^{1.75})$ and $10\log(y-0.6)^4$ respectively, and then a term which is less complex coupling both coordinates, $6\cos(x)(y-0.3y^2)$.

When considering the ANOVA-based sampling consisting of the center point of the parametric domain acting as the anchor $\boldsymbol{c} = (x_c, y_c)$, 10 additional points on the first branch (of the form (x, y_c)) and 10 additional points on the second branch (of the form (x_c, y)), functions $f_x(x)$ and $f_y(y)$ are constructed using a cubic spline interpolation. Then, a standard 2D nonlinear regression using basis functions of the form $(x - x_c)^m(y - y_c)^n$, $m, n \geq 1$ (due to the low dimensionality of the treated problem the employ of separated representations is not needed) is employed to calculate the coupled term $f_{x,y}(x,y)$ using 4 extra sample points.

The constructed solution is depicted in Fig. 2.10 where it is compared with the exact solution as well as with the solution obtained by using the standard sPGD (with a Latin Hypercube Sampling containing 25 points), while Figs. 2.12 and 2.11 compare the predictions and the reference values. These results stress an excellent performance of the ANOVA-based regression. Among other things, it can be noted that the ANOVA is very helpful to capture the fast changing behavior near lower values of y due to the presence of the logarithm.

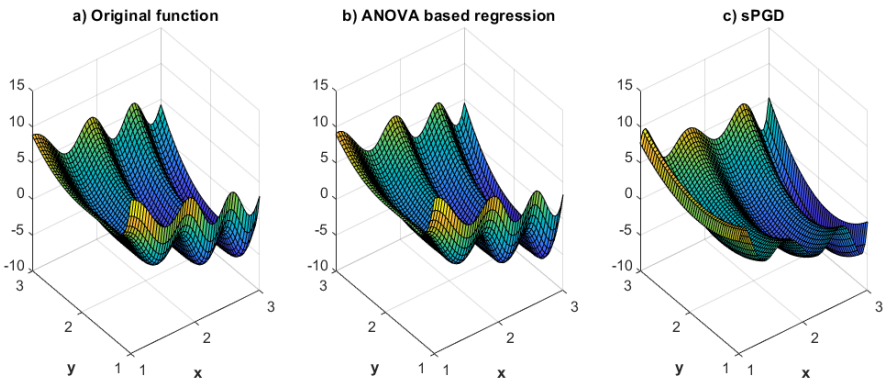


Figure 2.10: Comparing sPGD and ANOVA-PGD regressions

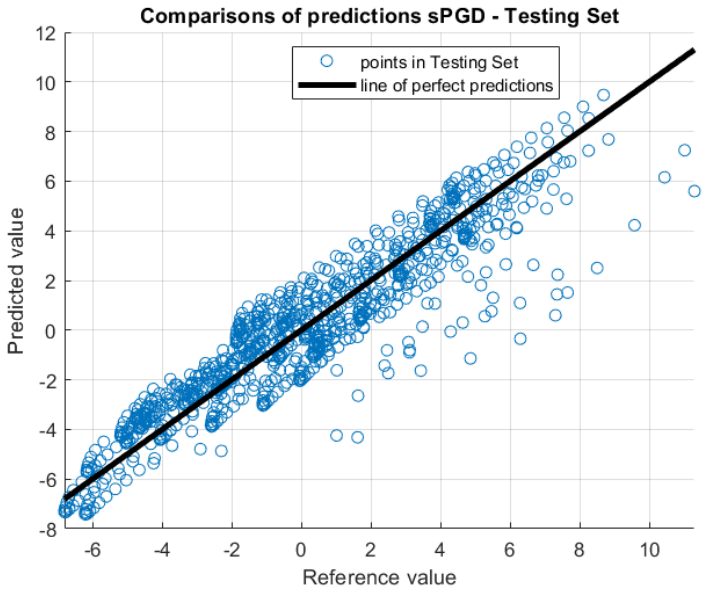


FIGURE 2.11: Problem defined in Eq. (2.2.14): Comparison of predicted sPGD values with the reference ones in the testing set (the black line represents a perfect prediction)

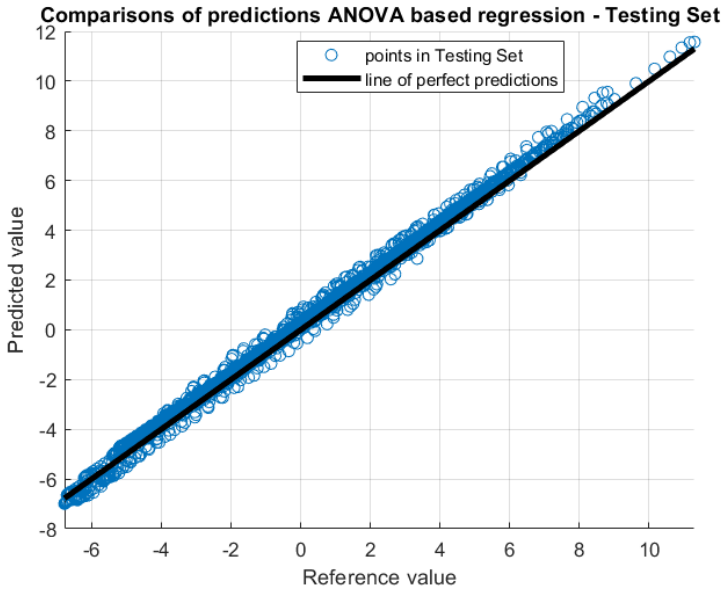


FIGURE 2.12: Problem defined in Eq. (2.2.14): Comparison of predicted ANOVA-PGD values with the reference ones in the testing set (the black line represents a perfect prediction)

Scalar-values function of 100 parameters

In the second example, the ANOVA-PGD is tested on a much more complex problem. The function of interest is $f : \mathcal{D} = [0,1]^d \to \mathbb{R}$ with $d = 100$ and it is defined by:

$$f(\boldsymbol{x}) = \sum_{k=1}^{d} \cos(\omega_k^c x_k + \phi_k^c) + \sin(\omega_k^s x_k + \phi_k^s) + \sum_{k=1}^{d-1} \frac{\cos(\omega_k^c x_k + \phi_k^c)\sin(\omega_{k+1}^s x_{k+1} + \phi_{k+1}^s)}{2} \tag{2.2.15}$$

where $\omega_k^c, \omega_k^s \in [0, 3\pi]$ and $\phi_k^c, \phi_k^s \in [0, 2\pi]$ are chosen randomly.

The design of experiments is constructed by setting the center point as the anchor point and by selecting 6 sample points on each branch (not including the anchor point). A Latin Hypercube Sampling containing 399 points is used for the PGD term, which is solved with the standard sPGD. The total number of samples is therefore 1000.

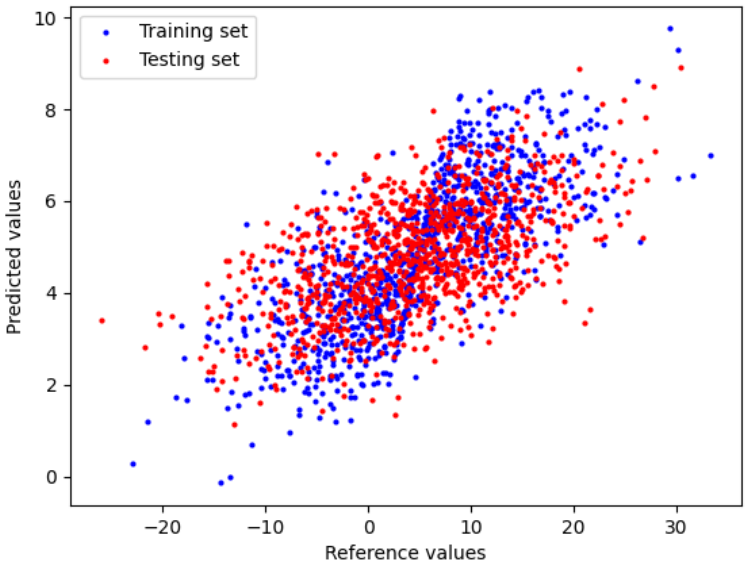


Figure 2.13: Problem defined in Eq. (2.2.15): Comparison of predicted SVR values with the reference ones in the training and testing sets

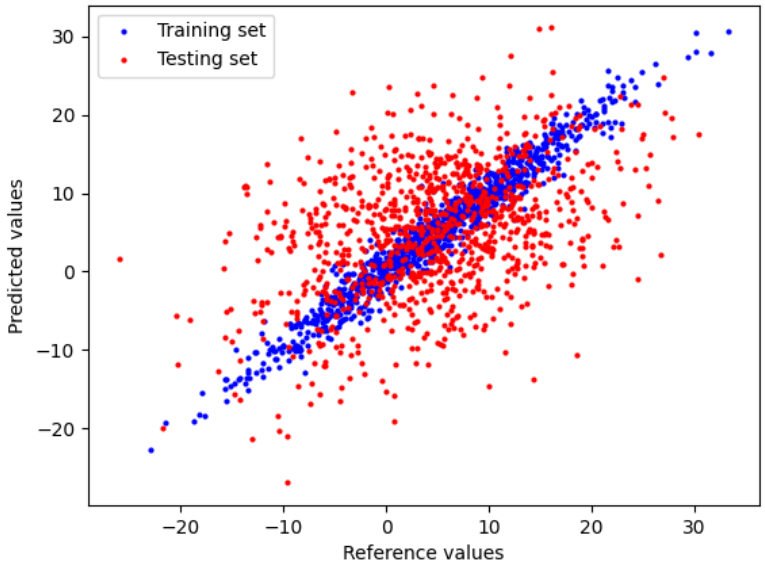


FIGURE 2.14: Problem defined in Eq. (2.2.15): Comparison of predicted Neural Network values with the reference ones in the training and testing sets

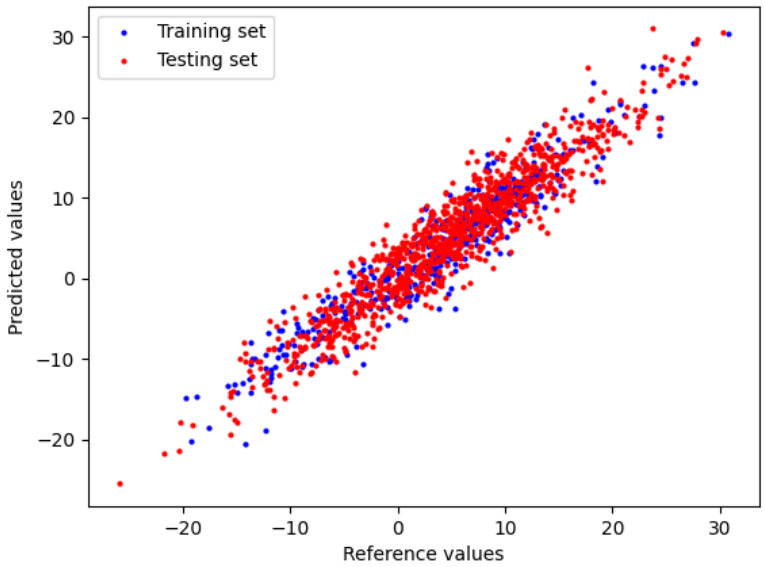


FIGURE 2.15: Problem defined in Eq. (2.2.15): Comparison of predicted ANOVA-PGD values with the reference ones in the training and testing sets

For comparison, the regression is also performed using a SVR with RBF kernel [156–158] and a fully-connected neural network with 4 layers of 50 neurons and ELU activation function [159].

The results obtained using these three methods are shown in Figs. 2.13, 2.14 and 2.15.

Diameter $[mm]$	Orientation $[°]$	Initial velocity $[m \cdot s^{-1}]$	Mass $[kg]$
$[20, 40]$	$[0, 180]$	$[0.5, 3]$	$[10, 120]$

TABLE 2.1: Parametric ranges

This problem can be considered to be extremely challenging because of the extreme dimension of the parametric space combined with nonlinear functions, which is why none of these methods produce results which are close to perfect. However, by exploiting the idea of dealing with each parameter one by one, the ANOVA strategy is able to extract valuable information from the data and clearly outperforms the other algorithms.

CYLINDRICAL INDENTATION TESTING OF A LITHIUM-ION CELL

This example adresses the construction of a parametric surrogate model for the 3D displacement field of a lithium-ion cell in a cylindrical indentation test. The cell and impactor are illustrated in Fig. 2.16.

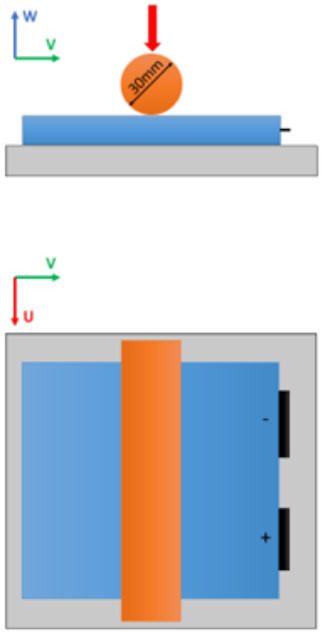


FIGURE 2.16: Cylindrical indentation test of a lithium-ion cell

The impactor which comes into contact with the cell is parametrized with four characteristics: diameter, orientation, initial velocity and mass. The parameters and their range of study are shown in table 2.1.

100 simulations are performed according to a design of experiments fitting the requirements of ANOVA-PGD using the commercial software LS-DYNA. The model is constructed using the version of ANOVA-PGD in the beta version of the software ESI ADMORE, which was implemented during this thesis.

The results are shown in Fig. 2.17.

The model is able to predict quite well the solutions. However due to the localization of the impact related to the orientation of the impactor, a bit of noise is present in

the model. The next sections address this type of issue.

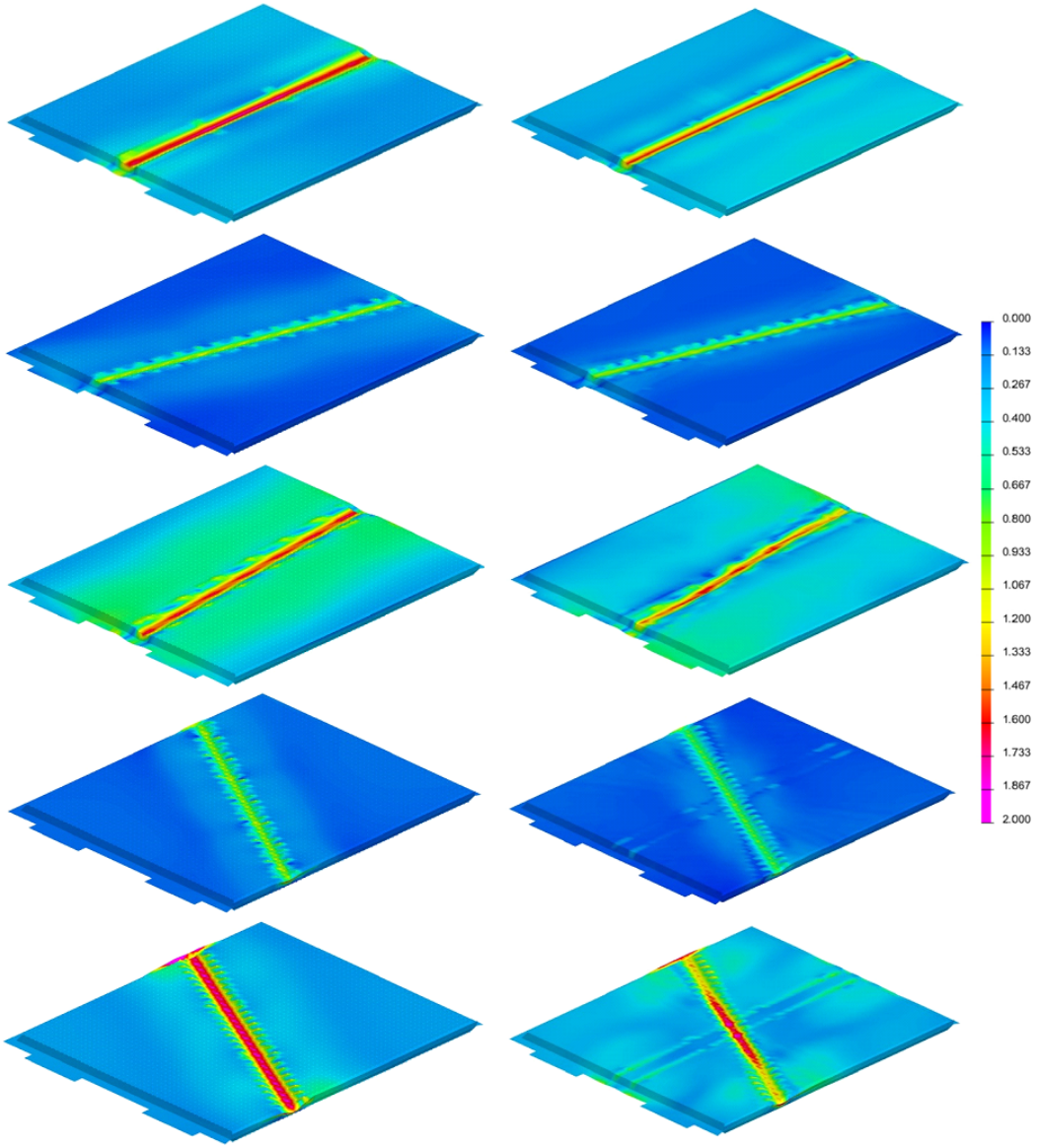


FIGURE 2.17: Comparison between the solutions (displacement norm in mm) obtained with finite elements using LS-DYNA (left) and the surrogate model (right)

2.3 Reduced Modelling of curves

In the context of parametric surrogates, several nontrivial issues arise when a whole curve needs be predicted from given input features. For instance, different sampling or ending points lead to non-aligned curves. This also happens when the curves exhibit a common pattern characterized by critical points at shifted locations (e.g., in mechanics, the elastic-plastic transition or the rupture point for a material). In such cases, classical interpolation methods fail to give physics-consistent results and appropriate pre-processing steps are required. Moreover, when bifurcations occur into the parametric space, to enhance the accuracy of the surrogate, a coupling with clustering and

classification algorithms is needed. In this section, several methodologies are proposed to overcome these issues. The surrogates thus created are then exploited to quantify and propagate uncertainty, furnishing parametric stastistical bounds for the predicted curves. The procedures are exemplified over problems in Computational Mechanics.

2.3.1 Data alignment and uncertainty propagation

This section presents the curve parameterization based on data alignment to obtain an accurate physics-informed interpolation. The procedure is applied to an example to study the mechanical response of parametric materials loaded in tension.

The problem considered is a parametric study over dog bone tensile test samples, as sketched in Fig. 2.18. It aims at studying the influence of the 3 parameters (n, K, ε_0) characterizing the Krupkowski hardening law (also known as *Swift hardening law*), widely used in FEM software

$$\sigma = K(\varepsilon + \varepsilon_0)^n,$$

linking True Strength and True Strain. ε denotes the effective plastic strain, ε_0 the offset strain, n the strain hardening exponent and K the material constant.

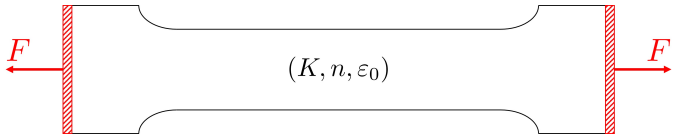


Figure 2.18: Parametric dog bone specimen loaded in tension.

Fig. 2.19 shows two patterns of the Force-Displacement curve, obtained for two different choices of the Krupkowski parameters (blue and orange lines). A classical "euclidian" interpolation of these two patterns would result in the non-physical black dashed pattern.

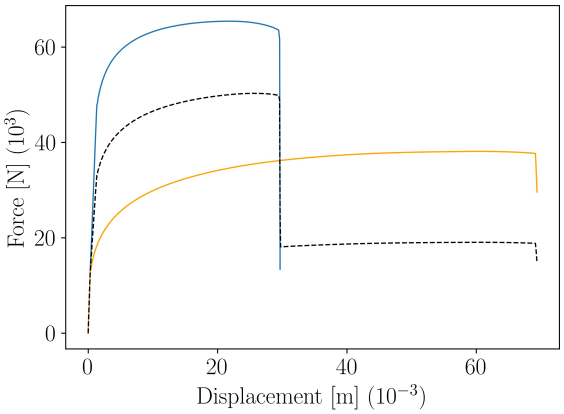


Figure 2.19: Main issue encountered when using standard interpolations on non-aligned curves (the black dashed line represents the interpolation between the two colored lines).

In what follows, a procedure is proposed in order to overcome such spurious effects, based on a curve alignment prior to interpolate. The method is illustrated over the

$K\ [MPa]$	n	ε_0
$[400, 700]$	$[0.1, 0.3]$	$[0.5, 3] \cdot 10^{-3}$

TABLE 2.2: Parametric ranges

Force-Displacement curves. However, for the sake of generality, such curves are referred to as generic functions $g(x)$, presenting two characteristic behaviors in the so-called primary and secondary zones. In the specific case of Force-Displacement, the primary zone is the elastic response of the material, up to the yield point x_E. The secondary zone is the post yield behaviour up to the failure point x_F, as illustrated in Fig. 2.20. x_E is referred to as the "transition point" and x_F as the "end point", related to the specimen fracture.

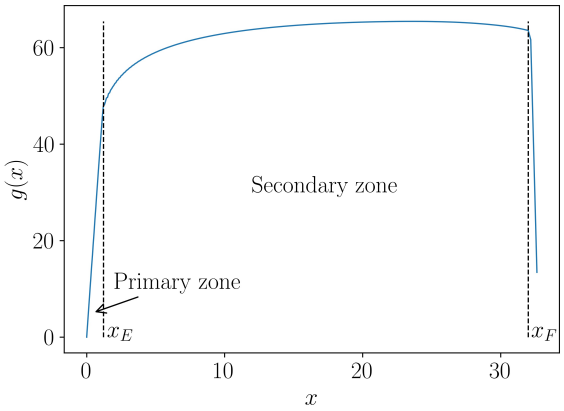


FIGURE 2.20: Behavior zones, transition and end points, for one function $g(x)$.

It is assumed that the behaviors in the primary and secondary zone, $g^1(x)$ and $g^2(x)$ respectively, and the transition and end points, x_E and x_F respectively, depend on a series of parameters grouped in vector $\boldsymbol{p}$, i.e. $g^1(x;\boldsymbol{p}) \equiv g(x \in [0, x_E];\boldsymbol{p})$, $g^2(x;\boldsymbol{p}) \equiv g(x \in [x_E, x_F];\boldsymbol{p})$, $x_E(\boldsymbol{p})$ and $x_F(\boldsymbol{p})$. Indeed, when considering different choices of the model parameter $\boldsymbol{p}_i = (K_i, n_i, \varepsilon_{0,i})$, $i = 1, \ldots, n_s$, one obtains a set of curves, as the ones shown in Fig. 2.21, for instance. Such curves correspond to a sparse DoE (Latin Hypercube) of 20 points in the 3-dimensional parametric space $\mathcal{D} = I_K \times I_n \times I_{\varepsilon_0}$, considering the parameters bounds specified in Table 2.2. Numerical simulations have been carried out with VPS simulation software from ESI Group. The variable x corresponds to the displacement in mm, while the function $g(x)$ to the force in kN.

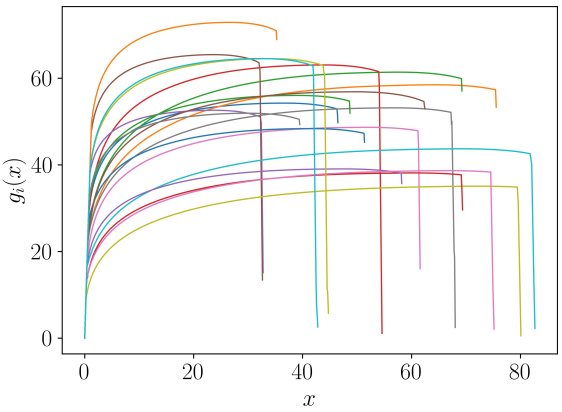


FIGURE 2.21: Curves $g(x;\boldsymbol{p}_i)$ related to different choices of the model features $\boldsymbol{p}_i = (K_i, n_i, \varepsilon_{0,i})$, $i = 1, \ldots, n_s$.

Once the transition and end points of each curve have been determined, the curves can be rediscretized over the same number of points (through a standard piecewise linear interpolation, for instance). To align them, a dimensionless coordinate is defined in each zone, y in the primary zone, $x \in [0, x_E]$, and z in the secondary zone, $x \in [x_E, x_F]$, defined through the change of variable

$$y = \frac{x}{x_E}, \ y \in [0,1] \text{ and } x \in [0, x_E], \tag{2.3.1}$$

and

$$z = \frac{x - x_E}{x_F - x_E}, \ z \in [0,1] \text{ and } x \in [x_E, x_F], \tag{2.3.2}$$

expressions that hold for each curve $g(x;\boldsymbol{p}_i)$, $i = 1, \ldots, n_s$, with

$$y = \frac{x}{x_E^i}, \ y \in [0,1] \text{ and } x \in [0, x_E^i], \tag{2.3.3}$$

and

$$z = \frac{x - x_E^i}{x_F^i - x_E^i}, \ z \in [0,1] \text{ and } x \in [x_E^i, x_F^i]. \tag{2.3.4}$$

Fig. 2.22 depicts functions $g_i^1(y) \equiv g^1(y;\boldsymbol{p}_i)$ and $g_i^2(z) \equiv g^2(z;\boldsymbol{p}_i)$.

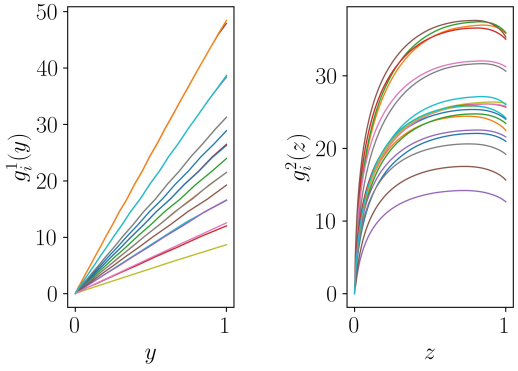


FIGURE 2.22: Functions $g_i^1(y) \equiv g^1(y;\boldsymbol{p}_i)$ (left) and $g_i^2(z) \equiv g^2(z;\boldsymbol{p}_i)$ (right), for $i = 1, \ldots, n_s$.

Actually, this procedure amounts at performing an alignment based on a dilatation of the curves in the first and secondary zone, as shown in Fig. 2.23. In such case, we can express the aligned curves as functions of $\tilde{x} \in [0, 2]$.

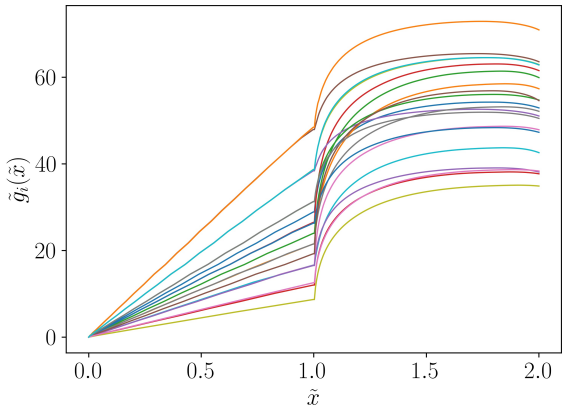


FIGURE 2.23: Functions $\tilde{g}_i(\tilde{x})$, for $i = 1, \ldots, n_s$, obtained after dilatation.

Once the curves have been aligned, the nonlinear regressor presented in section 1.1.4 can be invoked to build the parametric metamodel of the curve. This can be done separately in each zone or over the whole newly defined coordinate $\tilde{x}$. However, before proceeding with the regression, an ulterior parametrization via the Proper Orthogonal Decomposition is addressed to achieve a further Model Reduction.

2.3.2 POD MODES EXTRACTION

In order to extract the most significant modes able to describe these functions, the POD can be applied in each group of curves in Fig. 2.22. This amounts to build the snapshot matrix within each group and perform a truncated SVD. In the case that serves here to illustrate the procedure, a single mode suffices to describe the almost linear functions in the primary zone, that will be noted by $\xi_1(y)$, whereas in the secondary zone two functions are needed, $\phi_1(z)$ and $\phi_2(z)$.

Thus, any function $g_i^1(y)$ can be expressed $\forall i$ as

$$g_i^1(y) = \alpha_1^i \xi_1(y), \tag{2.3.5}$$

whereas functions $g_i^2(z)$, $\forall i$, read

$$g_i^2(z) = \beta_1^i \phi_1(z) + \beta_2^i \phi_2(z). \tag{2.3.6}$$

The α and β coefficients can be easily computed by simple projection, i.e.

$$\int_0^1 g_i^1(y)\xi_1(y)\ \mathrm{d}y = \alpha_1^i, \tag{2.3.7}$$

where the normality of $\xi_1(y)$ was used. In the same way, and taking into account the orthonormality of functions $\phi_1(z)$ and $\phi_2(z)$,

$$\int_0^1 g_i^2(z)\phi_1(z)\ \mathrm{d}z\ = \beta_1^i, \tag{2.3.8}$$

and

$$\int_0^1 g_i^2(z)\phi_2(z)\ \mathrm{d}z\ = \beta_2^i. \tag{2.3.9}$$

Thus, for each curve $g_i(x)$ can be extracted its five main descriptors: x_E^i, x_F^i, α_1^i, β_1^i and β_2^i, all of them related to the features grouped in vector $\boldsymbol{p}_i$.

Now, each of these descriptors can be expressed parametrically, $x_E(\boldsymbol{p})$, $x_F(\boldsymbol{p})$, $\alpha_1(\boldsymbol{p})$, $\beta_1(\boldsymbol{p})$ and $\beta_2(\boldsymbol{p})$, by using any regression technique. In this work, the regression used is the sPGD as described in 1.1.4 in the special case of scalar output.

2.3.3 Curves reconstruction

When considering a choice of the parameters $\boldsymbol{p}$, the curves descriptors are extracted from the regressions $x_E(\boldsymbol{p})$, $x_F(\boldsymbol{p})$, $\alpha_1(\boldsymbol{p})$, $\beta_1(\boldsymbol{p})$ and $\beta_2(\boldsymbol{p})$, the dimensionless coordinates defining both zones are calculated from

$$y = \frac{x}{x_E(\boldsymbol{p})} \rightarrow x = y\ x_E(\boldsymbol{p}), \tag{2.3.10}$$

and

$$z = \frac{x - x_E(\boldsymbol{p})}{x_F(\boldsymbol{p}) - x_E(\boldsymbol{p})} \rightarrow x = x_E(\boldsymbol{p}) + z\ (x_F(\boldsymbol{p}) - x_E(\boldsymbol{p})), \tag{2.3.11}$$

and, finally, the curve in each zone reconstructed according to

$$g^1(y;\boldsymbol{p}) = \alpha_1(\boldsymbol{p})\xi_1(y), \tag{2.3.12}$$

and

$$g^2(z;\boldsymbol{p}) = \beta_1(\boldsymbol{p})\phi_1(z) + \beta_2(\boldsymbol{p})\phi_2(z), \tag{2.3.13}$$

from which the curve $g(x;\boldsymbol{p})$ can be straightforward obtained via

$$g(x;\boldsymbol{p}) = \begin{cases} \alpha_1(\boldsymbol{p})\xi_1\left(\dfrac{x}{x_E(\boldsymbol{p})}\right), & x \in [0, x_E(\boldsymbol{p})] \\ \beta_1(\boldsymbol{p})\phi_1\left(\dfrac{x - x_E(\boldsymbol{p})}{x_F(\boldsymbol{p}) - x_E(\boldsymbol{p})}\right) + \beta_2(\boldsymbol{p})\phi_2\left(\dfrac{x - x_E(\boldsymbol{p})}{x_F(\boldsymbol{p}) - x_E(\boldsymbol{p})}\right), & x \in [x_E(\boldsymbol{p}), x_F(\boldsymbol{p})]. \end{cases} \tag{2.3.14}$$

To build the parametric metamodel, 17 curves have been used to train the sPGD regressor, while the remaining 3 for testing. Fig. 2.24 shows the resulting predictions over 3 training points and test points.

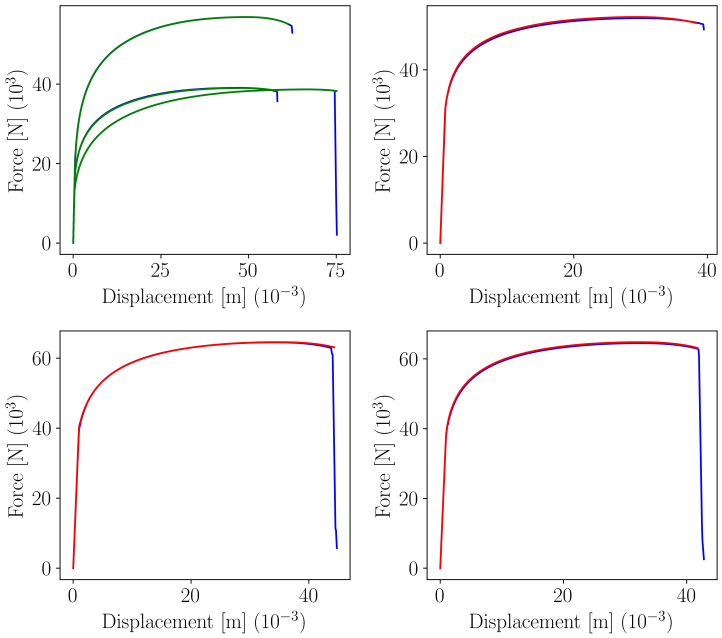


FIGURE 2.24: sPGD predictions (green line for training, red for testing) versus true curve (blue line).

2.3.4 Real-time calibration

Now, given an experimental curve $g(x)$, its parameters are extracted according to

- x_E from the point at which the change of behavior occurs (for instance, computing the function derivatives by means of finite differences);
- x_F is the terminal point;
- α_1 follows from $y = \frac{x}{x_E}$ and $\int_0^1 g(y)\xi_1(y)\,\mathrm{d}y = \alpha_1$;
- β_1 follows from $z = \frac{x-x_E}{x_F-x_E}$ and $\int_0^1 g(z)\phi_1(z)\,\mathrm{d}z = \beta_1$;
- β_2 follows from $z = \frac{x-x_E}{x_F-x_E}$ and $\int_0^1 g(z)\phi_2(z)\,\mathrm{d}z = \beta_2$.

Then, from the regression models $x_E(\boldsymbol{p})$, $x_F(\boldsymbol{p})$, $x_1(\boldsymbol{p})$, $\beta_1(\boldsymbol{p})$ and $\beta_2(\boldsymbol{p})$, the inverse problem is solved to extract the associated parameters, $\boldsymbol{p}$.

2.3.5 Statistical model derived by parametric curves

With the previously built surrogate model, the curve related to any possible value of $\boldsymbol{p}$ can be computed in real-time, i.e. $g(x;\boldsymbol{p})$. In this section, this surrogate will be employed for uncertainty quantification.

Each feature p^k in vector $\boldsymbol{p}$ is assumed characterized by a Gaussian distribution defined by its mean value μ_k and its variance σ_k^2, that is $p^k \sim \mathcal{N}(\mu_k, \sigma_k^2)$. Assuming all p^k being independent, we get

$$\boldsymbol{p} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad \boldsymbol{\mu} = (\mu_k)_{k=1}^d, \ \boldsymbol{\Sigma} = \mathrm{diag}(\boldsymbol{\sigma}), \ \boldsymbol{\sigma} = (\sigma_k^2)_{k=1}^d,$$

where diag(•) is the diagonal matrix of diagonal •.

The aim is to link the sensitivity over the input features with the one over the output curve. This means computing some estimators of the average M and the variance Σ of the curve descriptors for different choices of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$, and from them, by using the sparse PGD presented in Subsection 1.1.4, build the set of statistical surrogates:

$$\begin{cases} \mathcal{S}_{g(x;\boldsymbol{p})} : (\boldsymbol{\mu}, \boldsymbol{\sigma}) \to \left(\bar{M}_{g(x;\boldsymbol{p})}, \bar{\Sigma}_{g(x;\boldsymbol{p})}\right), \\ \mathcal{S}_{\mathcal{O}(\boldsymbol{p})} : (\boldsymbol{\mu}, \boldsymbol{\sigma}) \to \left(\bar{M}_{\mathcal{O}(\boldsymbol{p})}, \bar{\Sigma}_{\mathcal{O}(\boldsymbol{p})}\right). \end{cases} \tag{2.3.15}$$

where $\mathcal{O}(\boldsymbol{p})$ denotes any quantity of interest (QoI) involved in the curves parametrization (i.e., an output depending on the input parameters; e.g., x_E, x_F, α_1, β_1 and β_2 in the example presented before) and $\bar{M}$ and $\bar{\Sigma}$ the corresponding estimators for mean and variance, respectively. This allows calculating the envelopes, for a given confidence, of the curves, as sketched in Fig. 2.25.

To build the surrogate (2.3.15), for instance for the curve descriptor $\mathcal{O}(\boldsymbol{p})$, a training dataset of N_s points shall be generated:

$$\{(\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j), (\bar{M}_{\mathcal{O}(\boldsymbol{p}_j)}, \bar{\Sigma}_{\mathcal{O}(\boldsymbol{p}_j)})\}_{j=1}^{N_s}.$$

This can be achieved by means of a Monte Carlo sampling, which gives the estimators of mean and variance for the curves $g(x; \boldsymbol{p}_j(\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j))$, and of any descriptor $\mathcal{O}(\boldsymbol{p}_j)$, for $j = 1, \ldots, N_s$.

The whole procedure is summarized in Algorithm 1.

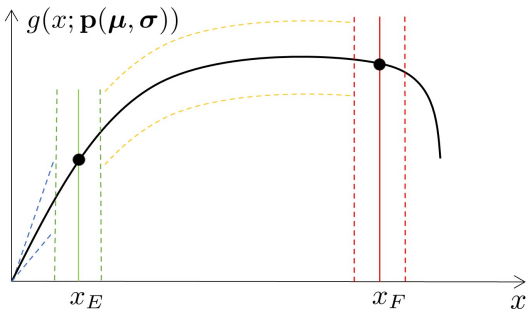


FIGURE 2.25: Sketch of curve envelopes.

Fig. 2.26 shows the parametric curve and its statistical sensing, for a given choice of the input features distribution parameters. Confidence Intervals have been computed using Algorithm 1, for the curve and the rupture point.

2.3.6 Data alignment and data clustering

This section focuses on the study of crack propagation in notched specimens loaded in tension, whose geometry is sketched in Fig. 2.27. The test piece has a V-shaped notch defect which is always in the same location (bottom-middle). On the other side of the test piece there is a half-circle groove. The goal is to predict the crack propagation from the defect in different configurations (different location S and radius R of the groove and test piece thickness h). Depending on the location of the groove, the crack will propagate differently from the defect, sometimes towards the groove and in other conditions straight towards the other side of the specimen.

Algorithm 1 Statistical sensing based on parametric curves

Input:

1. $f^X(\boldsymbol{p})$, $\boldsymbol{p} = (p^1, \ldots, p^d)$: curves regression model;
2. N_s: number of training points for the statistical surrogate model $\mathcal{S}_{\mathcal{O}(\boldsymbol{p})}$;
3. N_{MC}: number of Monte Carlo sampling points.

Output:

$(M_{\mathcal{O}(\boldsymbol{p})}, \Sigma_{\mathcal{O}(\boldsymbol{p})})$: regression model for mean and variance of curve descriptor $\mathcal{O}(\boldsymbol{p})$.

1: **for** $j = 1, \ldots, N_s$ **do**

2: Randomly sample (e.g., LHS) the model features means and variances

$$(\boldsymbol{\mu}_j, \boldsymbol{\sigma}_j), \quad \boldsymbol{\mu}_j = (\mu_{j,k})_{k=1}^d, \ \boldsymbol{\sigma}_j = (\sigma_{j,k}^2)_{k=1}^d.$$

3: Perform a Monte Carlo sampling of the curves statistical descriptor $\mathcal{O}(\boldsymbol{p})$:

4: 1. generate a population of N_{MC} vectors of features $\boldsymbol{p}_j = (p_j^k)_{k=1}^d$, by sampling N_{MC} points from

$$p_j^k \sim \mathcal{N}(\mu_{j,k}, \sigma_{j,k}^2), \ k = 1, \ldots, d;$$

5: 2. generate the population of the corresponding N_{MC} curves (and of any QoI involved in their parametrization), by using the curves surrogate $f^X(\boldsymbol{p})$, that is,

$$g(x; \boldsymbol{p}_{j,l}), \mathcal{O}(\boldsymbol{p}_{j,l}) = f^X(\boldsymbol{p}_{j,l}), \ l = 1, \ldots, N_{\mathrm{MC}};$$

6: 3. compute the population mean and variance to obtain the corresponding Monte Carlo estimators for the curve $g(x; \boldsymbol{p}_j)$ and its descriptor $\mathcal{O}(\boldsymbol{p}_j)$:

$$(\bar{M}_{g(x;\boldsymbol{p}_j)}, \bar{\Sigma}_{g(x;\boldsymbol{p}_j)}), \quad (\bar{M}_{\mathcal{O}(\boldsymbol{p}_j)}, \bar{\Sigma}_{\mathcal{O}(\boldsymbol{p}_j)}).$$

7: **end for**

8: Using the previously built population $\{(\bar{M}_{\mathcal{O}(\boldsymbol{p}_j)}, \bar{\Sigma}_{\mathcal{O}(\boldsymbol{p}_j)})\}_{j=1}^{N_s}$, train a regression model for the statistical sensing of $\mathcal{O}(\boldsymbol{p})$ involved in the curve parametrization:

$$\mathcal{S}_{\mathcal{O}(\boldsymbol{p})} : (\boldsymbol{\mu}, \boldsymbol{\sigma}) \to \big(\bar{M}_{\mathcal{O}(\boldsymbol{p})}, \bar{\Sigma}_{\mathcal{O}(\boldsymbol{p})}\big).$$

Same procedure holds for the whole curve $g(x; \boldsymbol{p})$.

9: Given a new couple $(\boldsymbol{\mu}^*, \boldsymbol{\sigma}^*)$ of model features means and variances, corresponding to the features $\boldsymbol{p}^*$, one can obtain a Confidence Interval –CI– at a given confidence level for the output. For instance, at level 0.95, one can build a CI for the curve $g(x; \boldsymbol{p}^*)$:

$$g(x; \boldsymbol{p}^*) \in \left[\bar{M}_{g(x;\boldsymbol{p}^*)} - 2\sqrt{\bar{\Sigma}_{g(x;\boldsymbol{p}^*)}}, \bar{M}_{g(x;\boldsymbol{p}^*)} + 2\sqrt{\bar{\Sigma}_{g(x;\boldsymbol{p}^*)}}\right].$$

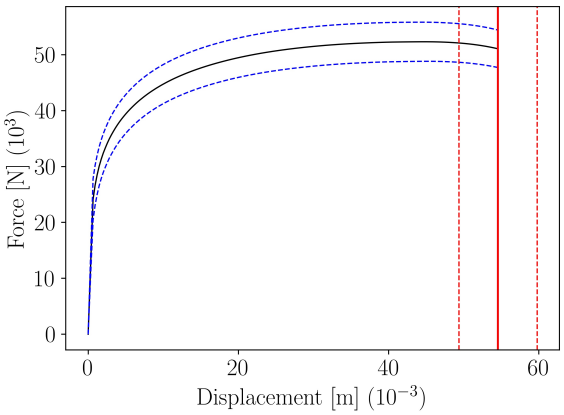


FIGURE 2.26: Confidence Interval of level 0.95 for the parametric Force-Displacement curve and for the rupture point, for a given choice of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$.

R [mm]	S [mm]	h [mm]
$[3, 8]$	$[0, 25]$	$[0.8, 1.6]$

TABLE 2.3: Parametric ranges.

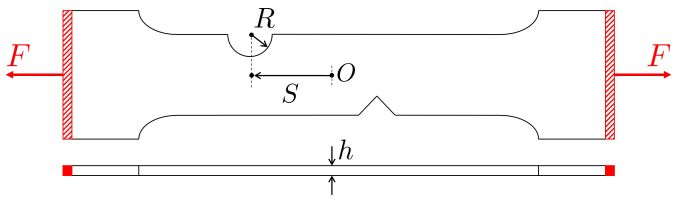


FIGURE 2.27: Parametric notched dog bone specimen loaded in tension (top and side views).

The design of experiments used in this study is a Latin Hypercube of 50 points in the 3-dimensional parametric space $\mathcal{D} = I_R \times I_S \times I_h$, with the parameter bounds specified in Table 2.3. Numerical simulations (carried out in VPS software from ESI Group) employ an Explicit Analysis and the EWK rupture model [160], using a mesh of 1096218 solid elements.

The focus is placed on the prediction of the Force-Displacement curves plotted in Fig. 2.28, which are considered as the generic functions $g(x)$, following the same notation as in section 2.3.1.

It can be observed that all the curves present a similar pattern in the first zone, monotonically increasing, while the response appears much different in the secondary zone. A first pre-processing step consists in splitting the zones as illustrated in Fig. 2.29, where x_M denotes the point where the curve reaches its maximum value, while x_F is its endpoint.

Cutting the curves yields the two groups of functions plotted in Fig. 2.30, which are of course not aligned. However, they can be expressed as functions of normalized coordinates y and z, respectively, and aligned following the dilatation procedure discussed in section 2.3.1.

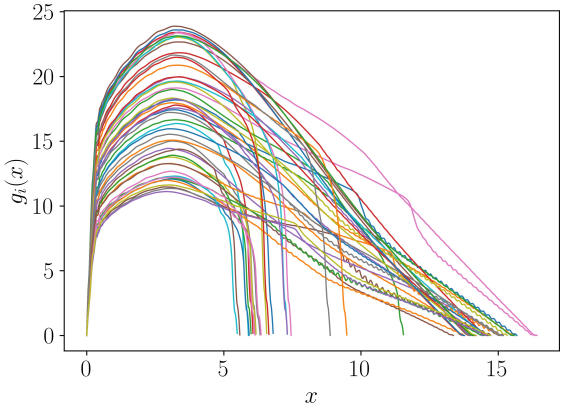


FIGURE 2.28: Curves $g_i(x) = g(x; \boldsymbol{p}_i)$ related to different choices of the model features $\boldsymbol{p}_i = (R_i, S_i, h_i),\ i = 1, \ldots, n_s$.

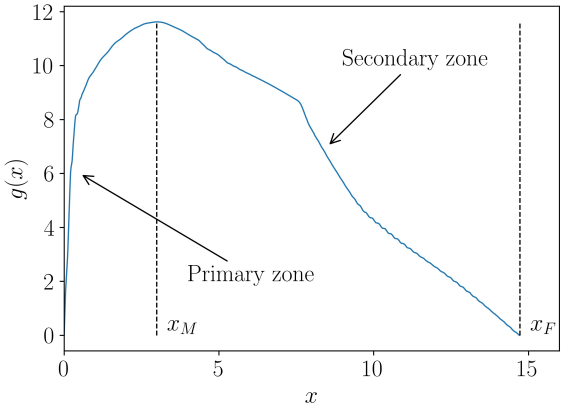


FIGURE 2.29: Behavior zones, transition and end points, for one function $g(x)$.

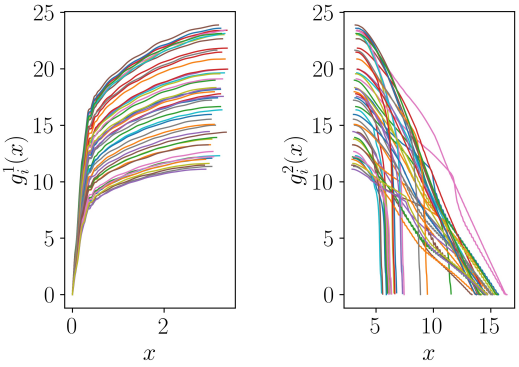


FIGURE 2.30: Functions $g_i^1(x) \equiv g^1(x; \boldsymbol{p}_i)$ (left) and $g_i^2(x) \equiv g^2(x; \boldsymbol{p}_i)$ (right), with $\boldsymbol{p}_i = (R_i, S_i, h_i)$, for $i = 1, \ldots, n_s$.

Once the alignment has been performed, using the usual nonlinear regression techniques and same notations as in section 2.3.1, two regression models, one for each

group, can be established:

$$\begin{cases} g^1(x;\boldsymbol{p}) \coloneqq g(x \in [0, x_M(\boldsymbol{p})]) = f_1^X(\boldsymbol{p}) \\ g^2(x;\boldsymbol{p}) \coloneqq g(x \in (x_M(\boldsymbol{p}), x_F(\boldsymbol{p})]) = f_2^X(\boldsymbol{p}). \end{cases} \tag{2.3.16}$$

In Eq. (2.3.16), for the sake of clarity, x_M and x_F are specified since these points are involved into the parametrization of the functions $g^1(x)$ and $g^2(x)$, respectively, and thus expressed parametrically.

As previously pointed out, the second group of functions $g_i^2(x)$, for $i = 1, \ldots, n_s$, presents really different shapes depending on the features $\boldsymbol{p}_i$. When bifurcations occur in the parametric space, the system responses related to two choices of the model parameters can be completely different. In such cases, a standard nonlinear regression over the full space can lead to inaccurate and nonphysical solutions. To enhance the accuracy of the model $f_2^X(\boldsymbol{p})$, a more valuable route consists in exploring the parametric space prior to interpolation. This can be done via a clustering of the system responses. Once the clusters have been established, several regression submodels can be built, minimizing the risk of mixing spurious effects coming from other clusters.

2.3.7 Clustering

To exemplify the bifurcation problem in the parametric space, two different configurations of the model parameters are considered, resulting into the specimens shown in Fig. 2.31.

Figure 2.31: Two different parameters configurations. Top: $R = 7.59$, $S = 18.23$, $h = 0.84$; bottom: $R = 3.75$, $S = 5.58$, $h = 1.51$ (all dimensions are provided in mm). The red zone is the part subject to rigid body constraints.

Fig. 2.32 shows four snapshots of the displacement field related to the specimens in Fig. 2.31, under axial tensile loading. The crack propagation follows two completely different patterns, drastically influencing the Force-Displacement curve, as shown in Fig. 2.33.

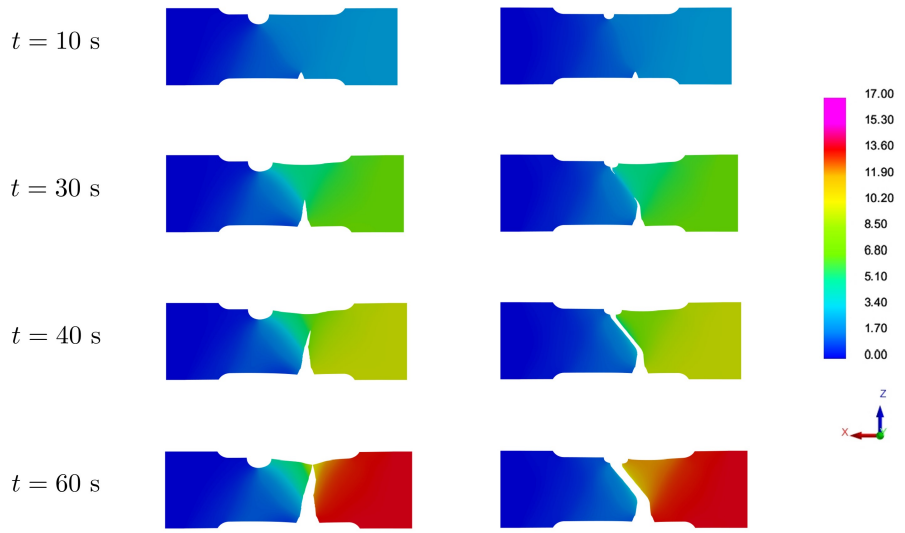


FIGURE 2.32: Bifurcation in the parametric space causing completely different crack propagation dynamics.

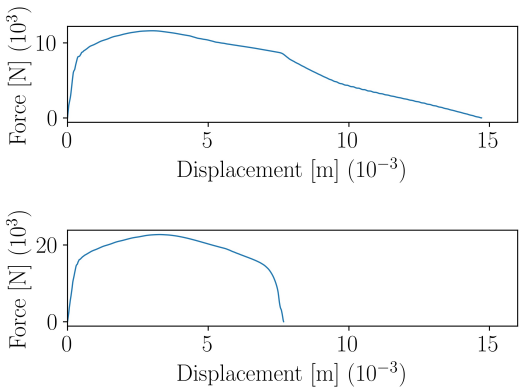


FIGURE 2.33: Force-Displacement curves corresponding to the two parameters configurations in Fig. 2.31.

The clustering step can be performed automatically by using a hierarchical clustering [70, 71] based on the curves shape or on the location of damaged elements into the finite element mesh. Once the clusters $\mathcal{C}_1$ and $\mathcal{C}_2$ have been established, two regression submodels can be trained, one for each cluster, and Eq. (2.3.16) becomes

$$\begin{cases} g^1(x;\boldsymbol{p}) = f_1^X(\boldsymbol{p}) \\ g^2(x;\boldsymbol{p}) = \begin{cases} f_{2,1}^X(\boldsymbol{p}) & \text{for } \mathcal{C}_1 \\ f_{2,2}^X(\boldsymbol{p}) & \text{for } \mathcal{C}_2. \end{cases} \end{cases} \tag{2.3.17}$$

Fig. 2.34 shows the functions in the secondary zone after the clustering.

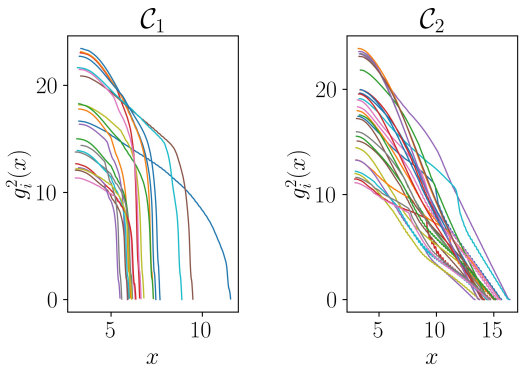


FIGURE 2.34: Functions $g_i^2(x)$ of Fig. 2.30 (right) after clustering, for $i = 1, \dots, n_s$.

In particular, one can remark that fracture occurs early on for tests belonging to cluster $\mathcal{C}_1$ and the final part of the curve is characterized by a steep slope. On the contrary, tests belonging to cluster $\mathcal{C}_2$ have an endpoint displacement around 15 mm and present a shallow slope. The clustering allows to avoid averaging such different dynamics, clearly enhancing the quality of the regressor.

2.3.8 CURVES RECONSTRUCTION AND CLASSIFICATION

For a newly defined choice of model features $\boldsymbol{p}^*$, the curve $g(x; \boldsymbol{p}^*)$ is obtained via

$$g(x; \boldsymbol{p}^*) = \begin{cases} g^1(x; \boldsymbol{p}^*), & 0 \leq x \leq x_M(\boldsymbol{p}^*) \\ g^2(x; \boldsymbol{p}^*), & x_M(\boldsymbol{p}^*) < x \leq x_F(\boldsymbol{p}^*), \end{cases} \tag{2.3.18}$$

where g^1 and g^2 are obtained through Eq. (2.3.17).

The training of the regression models has been performed using 40 points of the DoE, while the remaining 10 have been used for testing. Moreover, a Support Vector Machine classifier [161] (a Random Forest classifier [162] could also be used, for instance) has been trained to select the best regression submodel to predict $g^2(x; \boldsymbol{p}^*)$. This classifier has shown perfect accuracy, as shown by the Confusion Matrices in Fig. 2.35. Moreover, Fig. 2.36 shows the separating surface and classified points in the 3-dimensional parametric space.

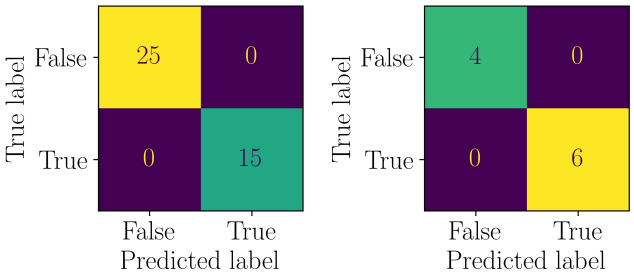


FIGURE 2.35: Confusion Matrices for the SVM classifier (left: training data, right: test data).

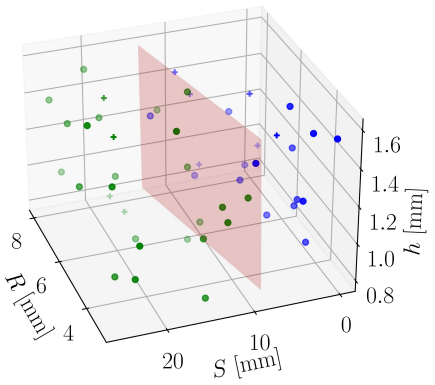


FIGURE 2.36: Parametric space and classified points (marker + is used for test points). The red plane is the separation surface.

Figs. 2.37 and 2.38 represent the plots of predictions for train and test, respectively, for 4 data points.

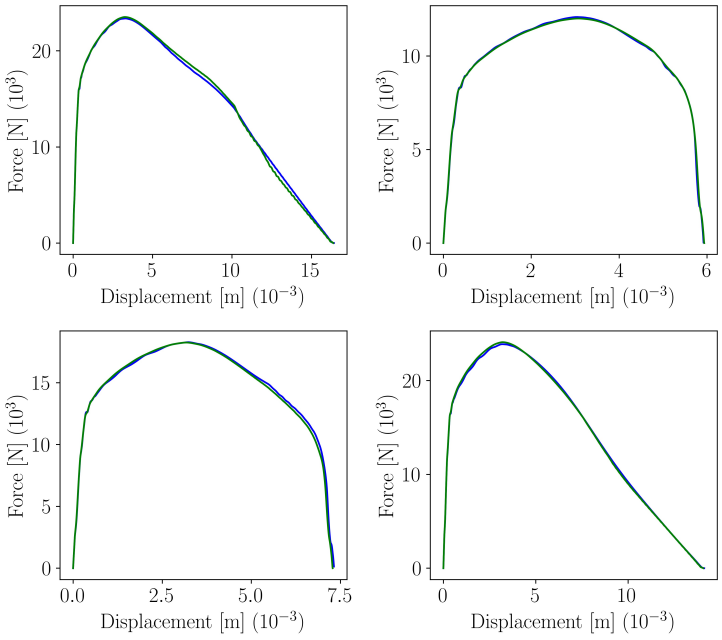


FIGURE 2.37: sPGD predictions (green line) versus true curve (blue line) for training data.

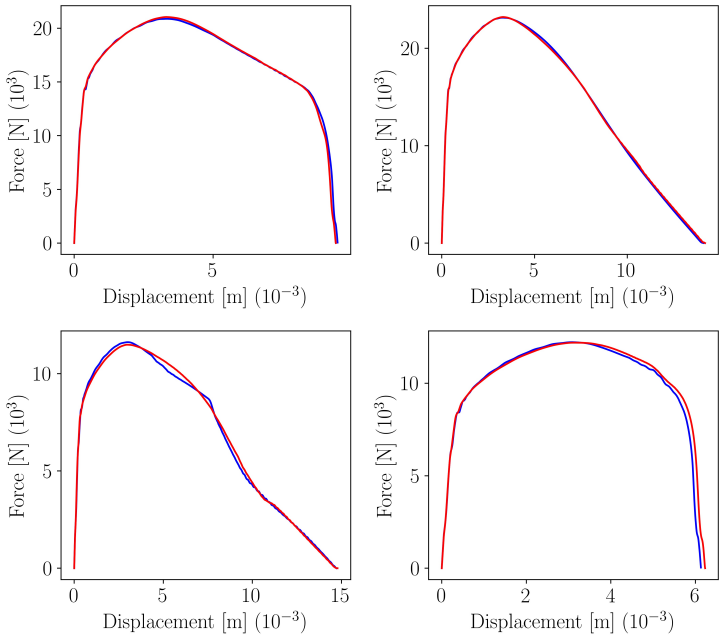


FIGURE 2.38: sPGD predictions (red line) versus true curve (blue line) for test data.

2.4 Reduced Modelling based on Optimal Transport

The type of problems faced in section 2.3 can be very challenging when trying to make reduced models of one-dimensional entities such as curves, but become extremely troublesome when the model concerns full-field solutions in two, three or more dimensions.

One of the usual difficulty which hinders the application of Reduced Order Modelling is the presence of features in the solution which move with time or as a result of a change in the parameters. The most popular and efficient ROM algorithms employ linear reduced bases, which are very powerful in many cases, but are not able to capture a continuous displacement of solution features resulting of phenomena such as advection or localized parametric constraints (e.g. loadings, impacts, sources, cracks, localized damage, ...). The adequacy of a reduced basis with a problem is studied within the theory of Kolmogorov n-width [134, 135], which quantifies the ability of a basis to represent faithfully the solutions depending on its dimension. Moving features and advection result in a slow evolution of the n-width with respect to the dimension of the reduced basis, a phenomenon known in MOR as the Kolmogorov barrier [131–133].

The ineffeciency of reduced bases in the aforementioned situations is related to their use of a Euclidean metric. This metric is not adequate for problems regarding misalignement or feature localization, because it only considers point by point difference which never allows to detect information such as the distance between noteworthy features. However, there exists a collection of metrics which are designed for this precise objective, the ones derived from the theory of Optimal Transport [16, 17].

Optimal Transport is witnessing a renewed interest because of its benefits for data science and machine learning which can rely on a strong mathematical background [163], wide array of formulations [164–166], as well as solid foundations for computational strategies [167–169] and efficient implementations [170].

The theory of Optimal Transport introduces the notion of spaces of probability measures, the Wasserstein spaces [171]. Although these spaces are often too restrictive to work with solutions of PDEs, which live in much larger spaces, they offer an interpolation framework [172,173] that is an attractive alternative to standard interpolation in Euclidean spaces. Besides the interpolation between two objects, the concept of barycenter in the Wasserstein space enables interpolation from multiple objects simultaneously, which could be a valuable route for regression and model reduction. However, the high computational cost and inadequacy of barycenters in high dimension limit their applicability in many cases.

To apply Optimal Transport theory to regression, a very appealing strategy consists in using Linearized Optimal Transport [174,175] to translate objects from the Wasserstein space to the tangent space, and perform the usual operations of machine learning in this Euclidian space, before mapping the results back to the Wassertsein space. This workflow introduces two problems, the challenges and cost related to computing the Exponential and Logarithmic maps [175], and the risks of operating away from the initial point [176].

This section presents a methodology combining the framework of ANOVA described in section 2.2 and Optimal Transport to create parametric reduced models. The theory of Optimal Transport is first briefly introduced, then the ANOVA-empowered OT regression strategy is described before it is applied to a numerical example.

2.4.1 Short introduction to Optimal Transport

This section introduces the basics of Optimal Transport and a few technical aspects relevant to describe the methodology which will be described later. For further insight and extensive information on the theory and applications of Optimal Transport, the interested reader can refer to [163,167,168].

Measure couplings and transport maps

Considering two measures μ and ν on two metric spaces $\mathcal{X}$ and $\mathcal{Y}$, coupling of μ and ν means to construct a measure π on $\mathcal{X} \times \mathcal{Y}$ such that π admits μ and ν as marginals on $\mathcal{X}$ and $\mathcal{Y}$ respectively.

The most trivial coupling is the measure $\mu \otimes \nu$ which is the probability law of (X, Y), where X and Y are two independent random variables such that μ is the law of X and ν is the law of Y. The opposite extreme case is when there exists a measurable function $T : \mathcal{X} \mapsto \mathcal{Y}$ such that $Y = T(X)$. In that case, π is called a deterministic coupling of μ and ν, and T is called a transport map. Deterministic couplings between μ and ν do not always exist, especially in discrete problems.

The objective of Optimal Transport is to find a coupling between μ and ν that is optimal in the sense that it minimizes a functional $J_{\mu,\nu} : \Pi_{\mu,\nu} \mapsto \mathbb{R}_+$ (where $\Pi_{\mu,\nu}$ is the set of couplings between μ and ν) which is defined as the cost of transportation of a coupling π between μ and ν.

In many cases, the optimal coupling is deterministic, but this not always the case which is not surprising since deterministic couplings do not always exist. In the methodology proposed in section 2.4.2, when the optimal coupling is not deterministic, the transport map will be replaced by an approximation thanks to the notion of

barycentric projection: $T = \mathbb{E}(\pi|x)$. This can also be expressed in the following way:

$$\forall z \in \mathcal{X}, T(z) = \int_{\mathcal{X}\times\mathcal{Y}} y\delta_z(x)\mathrm{d}\pi(x,y) \tag{2.4.1}$$

Kantorovitch problem

Introducing a cost function $c : \mathcal{X} \times \mathcal{Y} \mapsto \mathbb{R}_+$, the formulation introduced by Kantorovitch to define the optimal coupling is the following:

$$\pi^* = \underset{\pi\in\Pi_{\mu,\nu}}{\arg\min} \int_{\mathcal{X}\times\mathcal{Y}} c(x,y)\mathrm{d}\pi(x,y) \tag{2.4.2}$$

In the problems tackled in section 2.4.2, $\mathcal{X}$ and $\mathcal{Y}$ are both subsets of $\mathbb{R}^2$ or $\mathbb{R}^3$ and c is the Euclidean distance such that $c(x,y) = \|x-y\|^2$.

Furthermore, this formulation defines a metric, the p-Wasserstein distance W_p [177], which in turn allows to construct the p-Wassertein spaces of measures when μ and ν operate on the same space with the right properties:

$$W_p(\mu,\nu) = \left(\inf_{\pi\in\Pi_{\mu,\nu}} \int_{\mathcal{X}\times\mathcal{Y}} d(x,y)^p \mathrm{d}\pi(x,y)\right)^{\frac{1}{p}} \tag{2.4.3}$$

In order to solve Eq. 2.4.2, μ and ν and π are discretized and represented in the following manner:

$$\mu = \sum_{i=1}^{m} \alpha_i \delta_{x_i} \tag{2.4.4}$$

$$\nu = \sum_{j=1}^{n} \beta_j \delta_{y_j} \tag{2.4.5}$$

$$\pi = \sum_{i=1}^{m}\sum_{j=1}^{n} \gamma_{ij}\delta_{(x_i,y_j)} \tag{2.4.6}$$

Now, by defining $\mathbf{C} = (c(x_i,y_j))_{i,j}$ and $\mathbf{\Pi} = (\gamma_{ij})_{i,j}$, the optimal coupling π^* can be obtained by solving the linear program:

$$\begin{aligned} \min_{\mathbf{\Pi}} \quad & \langle \mathbf{\Pi}, \mathbf{C}\rangle \\ \text{s.t.} \quad & \mathbf{\Pi}\mathbf{1} = \boldsymbol{\alpha} \\ & \mathbf{\Pi}^T\mathbf{1} = \boldsymbol{\beta} \\ & \mathbf{\Pi} \geq \mathbf{0} \end{aligned} \tag{2.4.7}$$

where $\langle \mathbf{\Pi}, \mathbf{C}\rangle = \sum_{i=1}^{m}\sum_{j=1}^{n} \gamma_{ij} c(x_i,y_j)$.

An approximation of the optimal coupling can be obtained faster with the use of Entropic Optimal Transport [178] but it was not necessary in this work.

Linearized Optimal Transport

In order to combine the theory of Optimal Transport with linear methods and interpolation or regression in vector spaces, the Linearized Optimal Transport (LOT) [179, 180] is a very valuable tool.

Thanks to the theory of differential geometry, it is possible to locally linearize the 2-Wasserstein space and thus to work on the tangent space at a certain point μ_0. Any measure can be embedded into the tangent space at μ_0 using the logarithmic map Log_{μ_0}. Computing it requires finding the optimal coupling between μ_0 and μ and its associated transport map T. The logarithmic map is then computed as:

$$\mathrm{Log}_{\mu_0}(\mu) = T - \mathrm{Id} \tag{2.4.8}$$

The inverse operation, the exponential map Exp_{μ_0} computes a measure from an embedding v of the tangent space in the following manner:

$$\mathrm{Exp}_{\mu_0}(v) = (\mathrm{Id} + v)_\sharp \mu_0 \tag{2.4.9}$$

where $T_\sharp \mu = \mu \circ T^{-1}$ is the pushforward measure defined such that, considering a suitable σ-algebra Σ:

$$\forall A \in \Sigma, T_\sharp \mu(A) = \mu(T^{-1}(A)). \tag{2.4.10}$$

In practice, it is convenient to represent a discretized measure as a weighted point cloud, upon which the pushforward operation results in moving each point from x to $T(x)$. It can be noted that the continuous interpolation between two measures defined by operating in the tangent space consists in moving each point along the trajectory $(1-t)x + tT(x)$ for $t \in [0, 1]$.

2.4.2 ANOVA-based Optimal Transport methodology

This section presents a strategy which combines the ANOVA-PGD described in section 2.2 with the aforementioned tools to build non-intrusive parametric reduced models. To this aim, in what follows, the PDE solutions u will be transformed into measures by applying a few transformations. At first, the methodology will be described for positive solutions with localized support, and then it will be adapted to a more general setting.

The motivation for this work is that the natural interpolation in the tangent space allows to move localized features of the solutions in the parametric domain. This is a very interesting ability in contrast with the interpolations performed in the vector space in which the solutions originally live, which do not move features around the space and are only able to increase or decrease their magnitude. The difference between the two approaches is illustrated in Fig. 2.39.

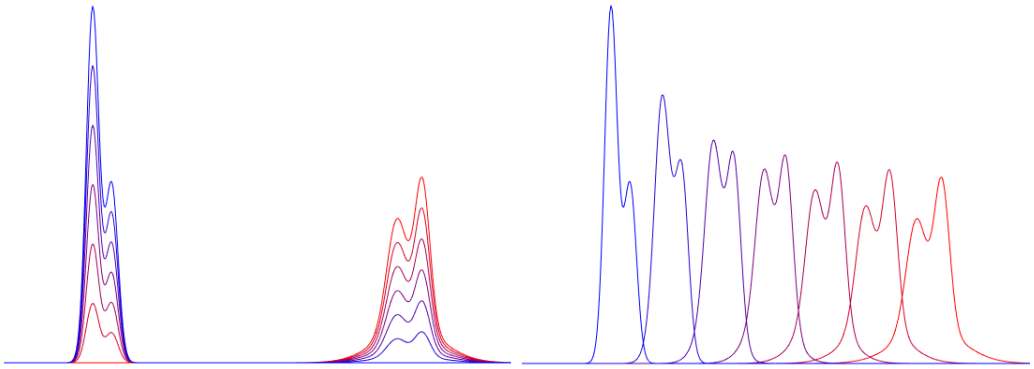


FIGURE 2.39: Different configurations (shades of purple) of the interpolation between the blue and red curves in the Euclidean space (left) and using Optimal Transport (right).

The problem considered in this section is the construction of the parametric solution $u : (\boldsymbol{x}; \boldsymbol{p}) \in \Omega \times \mathcal{D} \mapsto \mathbb{R}$ of a parametric partial differential equation, where Ω is a compact convex subset of $\mathbb{R}^2$ (for the sake of simplicity in this example but it could also be $\mathbb{R}^3$) and $\mathcal{D} \subset \mathbb{R}^d$. Some of the notations are those defined in section 2.2.2.

ANOVA-based Optimal Transport for positive localized solutions

In this first instance of the method, only positive solutions with localized support are considered: $\forall (\boldsymbol{x}; \boldsymbol{p}) \in \Omega \times \mathcal{D}, u(\boldsymbol{x}; \boldsymbol{p}) > 0$ and $\forall \boldsymbol{p} \in \mathcal{D}, \text{supp}(u(\bullet; \boldsymbol{p})) \subsetneq \Omega$ such that $|\text{supp}(u(\bullet; \boldsymbol{p}))| \ll |\Omega|$.

The solutions corresponding to a design of experiments such as the one described in section 2.2.2 are computed using an existing solver. The discretized counterpart of $u(\bullet; \boldsymbol{p}_i), i = 1 \dots n_s$ is denoted by $\boldsymbol{u}(\boldsymbol{p}_i) \in \mathbb{R}^N$, a vector which contains the nodal values of u and the nodal coordinates are stored in matrix $\boldsymbol{X} \in \mathbb{R}^{N \times 2}$. To transform them into measures suitable for the theory described in section 2.4.1, only the M_i points inside the support of u are considered and the solutions are normalized so that their integral over Ω is 1. $\boldsymbol{X}_i \in \mathbb{R}^{M_i \times 2}$ contains the points in the support of $u(\bullet; \boldsymbol{p}_i)$ and $\boldsymbol{v}_i$ the corresponding nodal values. Then, the discretized measure is represented by vector $\boldsymbol{\mu_i} \in \mathbb{R}^{M_i}$ given by:

$$\boldsymbol{\mu}_i = \frac{\boldsymbol{v}_i}{\int_\Omega u(\boldsymbol{x}; \boldsymbol{p}_i) \mathrm{d}\boldsymbol{x}} \tag{2.4.11}$$

and corresponding coordinates $\boldsymbol{X}_i$.

The idea is to compute the transport maps from the center point $\boldsymbol{c}$ and then create two regressions using the ANOVA-PGD method: one for the transport maps and the other one for the values $\int_\Omega u(\boldsymbol{x}; \boldsymbol{p}_i) \mathrm{d}\boldsymbol{x}$ in order to recover the unnormalized field.

The main issue with this strategy is related to the risks of operating far away from the tangent point. The transport map between solutions which are really dissimilar can produce unwanted behaviors. However, the particular structure of the ANOVA

sampling offers an interesting way to circumvent this issue based on the following example.

Suppose a problem with one parameter with $\mathcal{D} = [0, 2]$ and a 3 sample points and associated measures $(p_1 = 0, \mu_1), (p_2 = 1, \mu_2), (p_3 = 2, \mu_3)$. For the sake of the example, the chosen reference point is μ_1. Then, to take into account that p_2 lies between p_1 and p_3, the transport map from μ_1 to μ_3, denoted here as $T_{1\to3}$, is computed as:

$$T_{1\to3} = T_{2\to3} \circ T_{1\to2}. \tag{2.4.12}$$

This is done by solving the transport problem between μ_1 and μ_2 and then the one between μ_2 and μ_3, but never the one between μ_1 and μ_3.

The same principle is applied in multiple dimensions in the ANOVA sampling. The couplings and transport maps are computed step by step then composed, for each half branch starting from center point $\boldsymbol{c}$. An example for two parameters showing with arrows the coupling steps is shown in Fig. 2.40.

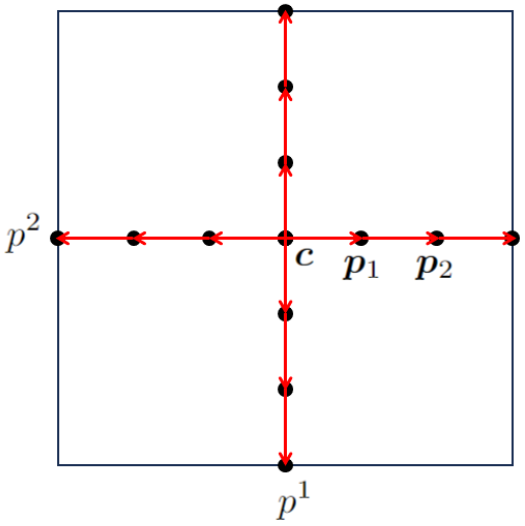


FIGURE 2.40: Example in two dimensions showing the coupling route followed in the ANOVA sampling. Each red arrow represents a step of measure coupling, from which the transport map is extracted and then composed with the previous ones to obtain a global transport map from point $\boldsymbol{c}$.

On the half branch associated with the positive values of p^1 (as shown in Fig. 2.40 with points $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$), the coupling algorithm is the following:

1. Starting from $\boldsymbol{p}_0 = \boldsymbol{c}$, the matrix $\boldsymbol{\Pi}_{0\to1}$ coupling μ_0 and μ_1 is obtained by solving the linear program 2.4.7 with $\boldsymbol{\alpha} = \boldsymbol{\mu}_0$, $\boldsymbol{\beta} = \boldsymbol{\mu}_1$ and matrix $\mathbf{C}$ computed from $\boldsymbol{X}_0$ and $\boldsymbol{X}_1$.

2. The transport map is computed by barycentric projection: $\forall j = 1, \ldots, M_0, T_1(\boldsymbol{X}_0^j) = T_{0\to1}(\boldsymbol{X}_0^j) = (\boldsymbol{\Pi}_{0\to1}\boldsymbol{X}_1)^j/\boldsymbol{\mu}_0^j$ where exponent j refers to the extraction of the j-th line of the matrix. The discretized version of measure μ_1 can now both be represented by $(\boldsymbol{X}_1, \boldsymbol{\mu}_1)$ and $(T_1(\boldsymbol{X}_0), \boldsymbol{\mu}_0)$ where $T_1(\boldsymbol{X}_0) = (T_1(\boldsymbol{X}_0^j))_{j=1,\ldots,M_0}$. The latter is preferred because it allows to compute easily the composition of transport maps.

3. The matrix $\boldsymbol{\Pi}_{1\to 2}$ coupling μ_1 and μ_2 is obtained by solving the linear program 2.4.7 with $\boldsymbol{\alpha} = \boldsymbol{\mu}_0$, $\boldsymbol{\beta} = \boldsymbol{\mu}_2$ and matrix $\mathbf{C}$ computed from $T_1(\boldsymbol{X}_0)$ and $\boldsymbol{X}_2$.

4. The transport map is computed by barycentric projection: $\forall j = 1, \dots, M_0, T_2(\boldsymbol{X}_0^j) = T_{1\to 2}(T_{0\to 1}(\boldsymbol{X}_0^j)) = (\boldsymbol{\Pi}_{1\to 2}\boldsymbol{X}_2)^j / \boldsymbol{\mu}_0^j$. Measure μ_2 is now represented by $(T_2(\boldsymbol{X}_0), \boldsymbol{\mu}_0)$.

5. Steps 3 and 4 are repeated until reaching the end of the branch.

Remark: in order to have a more general reference representation, it is interesting to discretize μ_0 with a decomposition similar to the ones performed in Smoothed Particle Hydrodynamics (SPH) [181]. Instead of computing μ_0 from Eq. 2.4.11, this consists in choosing a value of M_0, setting $\boldsymbol{\mu}_0^j = \frac{1}{M_0}$ for $j = 1, \dots, M_0$ and then constructing matrix $\boldsymbol{X}_0$ by solving:

$$\boldsymbol{X}_0 = \arg\min_{\boldsymbol{X}} \int_\Omega \left\| \frac{1}{M_0} \sum_{j=1}^{M_0} K(\|\boldsymbol{X}^j - \boldsymbol{x}\|) - \frac{u(\boldsymbol{x}; \boldsymbol{p}_0)}{\int_\Omega u(\boldsymbol{x}; \boldsymbol{p}_0) \mathrm{d}\boldsymbol{x}} \right\| \mathrm{d}\boldsymbol{x} \tag{2.4.13}$$

where K is a suitable kernel function. △

$T_i(\boldsymbol{X}_0)$ is stored in matrix $\boldsymbol{T}_i \in \mathbb{R}^{M_0 \times 2}$ for all the i related to the ANOVA sampling (note that $T_0 = \mathrm{Id}$). Then, the function $\boldsymbol{T}_{\mathrm{ANOVA}} : \mathcal{D} \mapsto \mathbb{R}^{M_0 \times 2}$ (similar to $\boldsymbol{u}_{\mathrm{ANOVA}}$ from Eq. 2.2.12) is constructed.

Note: In section 2.2.2, the methodology is defined for vector-valued outputs. For a matrix, the same methodology can be applied by reshaping it into a vector. △

For any $\boldsymbol{p} \in \mathcal{D}$, $\boldsymbol{T}_{\mathrm{ANOVA}}(\boldsymbol{p})$ is an approximation of the mapping $T_{\boldsymbol{p}}$, which transports μ_0 into $\mu_{\boldsymbol{p}}$, evaluated in the values of $\boldsymbol{X}_0$.

Now, for all the $\boldsymbol{p}_i$ in the design of experiments which do not belong to the ANOVA sampling:

1. $\boldsymbol{T}_{\mathrm{ANOVA}}(\boldsymbol{p}_i)$ is evaluated, allowing to define μ_i^{ANOVA} discretized by $(\boldsymbol{T}_{\mathrm{ANOVA}}(\boldsymbol{p}_i), \boldsymbol{\mu}_0)$ as a first approximation of μ_i.

2. The matrix $\boldsymbol{\Pi}_{i\to i}$ coupling μ_i^{ANOVA} and μ_i is obtained by solving the linear program 2.4.7 with $\boldsymbol{\alpha} = \boldsymbol{\mu}_0$, $\boldsymbol{\beta} = \boldsymbol{\mu}_i$ and matrix $\mathbf{C}$ computed from $\boldsymbol{T}_{\mathrm{ANOVA}}(\boldsymbol{p}_i)$ and $\boldsymbol{X}_i$.

3. The transport map is computed by barycentric projection: $\forall j = 1, \dots, M_0, T_i(\boldsymbol{X}_0^j) = T_{i\to i}(T_{0\to 1}(\boldsymbol{X}_0^j)) = (\boldsymbol{\Pi}_{i\to i}\boldsymbol{X}_i)^j / \boldsymbol{\mu}_0^j$. Measure μ_i is now represented by $(T_i(\boldsymbol{X}_0), \boldsymbol{\mu}_0)$.

Once the $(T_i(\boldsymbol{X}_0)$ have been computed for $i = 1, \dots, n_s$, the function $\boldsymbol{T}_{\mathrm{PGD}}$ (similar to $\boldsymbol{u}_{\mathrm{PGD}}$ from Eq. 2.2.12) can be built, and therefore the model $\tilde{\boldsymbol{T}} : \mathcal{D} \mapsto \mathbb{R}^{M_0 \times 2}$ is complete.

Similarly, a scalar ANOVA-PGD model $\tilde{u} : \mathcal{D} \mapsto \mathbb{R}$ relating $\boldsymbol{p}_i$ to $\int_\Omega u(\boldsymbol{x}; \boldsymbol{p}_i) \mathrm{d}\boldsymbol{x}$ is constructed.

Now, for any choice of the parameters $\boldsymbol{p} \in \mathcal{D}$, $u(\bullet, \boldsymbol{p})$ can be approximated by:

$$\forall \boldsymbol{x} \in \Omega, u(\boldsymbol{x}, \boldsymbol{p}) = \frac{\tilde{u}(\boldsymbol{p})}{M_0} \sum_{j=1}^{M_0} K(\|\tilde{\boldsymbol{T}}(\boldsymbol{p})^j - \boldsymbol{x}\|). \tag{2.4.14}$$

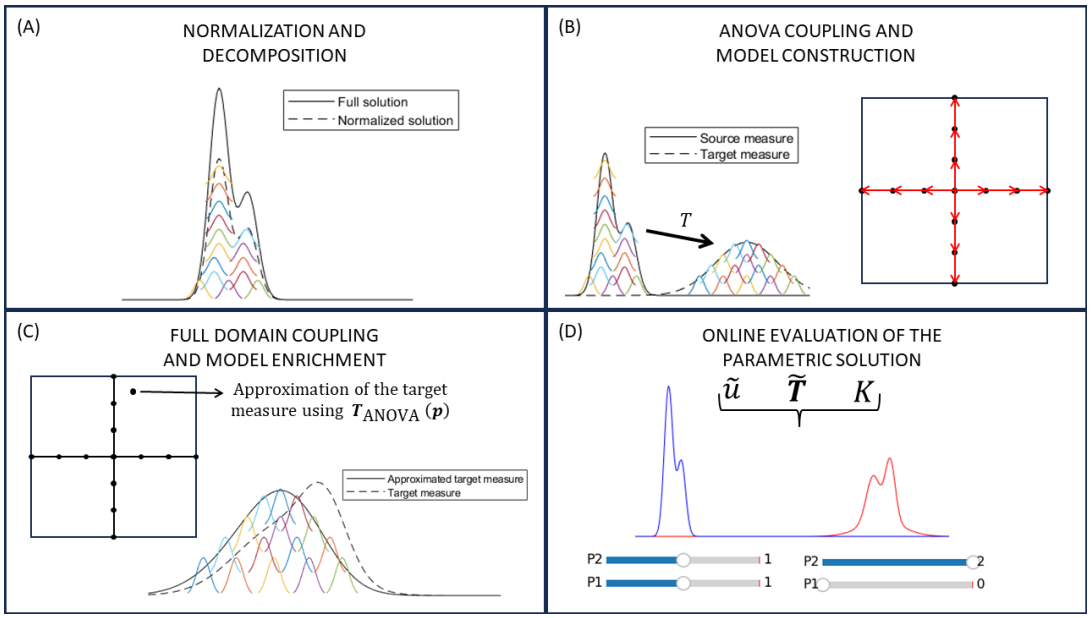


FIGURE 2.41: The worklow follows these four steps: (A) The solutions are normalized and the reference solution is decomposed in SPH fashion (B) Couplings and transport maps are obtained following the arrows to visit the entire ANOVA sampling (C) The measures outside of the ANOVA sampling are approximated using the ANOVA model and corrected using the solutions, thus allowing to enrich the model (D) The model can be evaluated in real-time to approximate the solution for any values of the parameters.

To recap, the complete workflow is the following:

1. For $i = 1, \dots, n_s$, the solutions $u(\bullet, \boldsymbol{p})$ are discretized and normalized, in order to obtain coordinates $\boldsymbol{X}_i$ and weights $\boldsymbol{\mu}_i$, which together form a discrete representation of measure μ_i.

2. Solution $u(\bullet, \boldsymbol{p}_0)$ is normalized, and discretized according to a decomposition similar as the ones performed for SPH, to obtain $(\boldsymbol{X}_0, \boldsymbol{\mu}_0)$

3. Now, following the coupling route described in Fig. 2.40, the transport maps which transform μ_0 into μ_i are obtained for all i such that $\boldsymbol{p}_i \in \mathcal{D}_{\boldsymbol{c}}$, using optimal couplings and barycentric projection.

4. A first model of the parametric transport map is created using the so-called "ANOVA terms" of the ANOVA-PGD.

5. This model is used to evaluate an approximation of the measure for all the i such that $\boldsymbol{p}_i \notin \mathcal{D}_{\boldsymbol{c}}$. This estimated measure is coupled to the discrete representation of μ_i, thus allowing to compute the transport map T_i thanks to barycentric projection.

6. These new transport maps allow to complete the construction of the ANOVA-PGD model.

7. A model of the magnitude of the solution is also constructed with the ANOVA-PGD.

8. These two models can be combined via Eq. 2.4.14 to approximate the parametric solution u.

This workflow is also illustrated in Fig. 2.41.

ANOVA-based Optimal Transport for more general solutions

In the previous part, it was assumed that the solutions u had a localized support. This was not explicitly necessary, but it is important for the quality of the solution for two reasons:

- Approximating a constant in the whole domain with a decomposition in the spirit of SPH is extremely expensive and inefficient.
- If the localized features are small with respect to the global part, they will be erased by the approximations and error of the model, therefore using Optimal Transport becomes useless.

It was also assumed that the solutions were positive, because the conservation of mass is at the heart of Optimal Transport, which in the general case does not allow negative-valued measures. However, some works in the literature study such an extension [182, 183], and although they are more interested in calculating the distance between objects, they can inspire ways to obtain transport maps.

The following proposes a revised version of the method to enable applying to more general solutions, by attempting to remove the two aforementioned constraints.

In order to deal with global solutions, such that $\text{supp}(u) = \Omega$, the idea is to separate the solutions into two contributions, one that is well represented by global modes and which should be addressed directly using a method such as the ANOVA-PGD, and one which has a localized support and which can be dealt with using Optimal Transport. Therefore, the solutions read:

$$\forall(\boldsymbol{x}, \boldsymbol{p}) \in \Omega \times \mathcal{D}, u(\boldsymbol{x}; \boldsymbol{p}) = u_{\text{glob}}(\boldsymbol{x}; \boldsymbol{p}) + u_{\text{loc}}(\boldsymbol{x}; \boldsymbol{p}). \tag{2.4.15}$$

In the simple case, $u_{\text{glob}}(\boldsymbol{x}; \boldsymbol{p}) = u_{\text{glob}}(\boldsymbol{x})$ does not depend on the parameters. In this case, u_{glob} can be extracted as the median of all the solutions. Indeed, the part of the solutions corresponding to u_{loc} are outliers in this context, and can be filtered out very efficiently by the median.

This idea can be extended to solve the problem when u_{glob} has parametric dependencies. u_{glob} is searched in the separated form

$$u_{\text{glob}}(\boldsymbol{x}; \boldsymbol{p}) = \sum_{k=1}^{m} \Xi_k(\boldsymbol{x}) \Psi_k(\boldsymbol{p}) \tag{2.4.16}$$

using the following methodology, derived from the PGD and based on the filtering of sparse outliers:

1. The terms of the sum are constructed successively in a greedy manner until convergence.
2. Ψ_k is initialized randomly, and a first approximation of Ξ_k is obtained by solving:

$$\min_{\Xi} \|\Xi \Psi_k - u\|_1. \tag{2.4.17}$$

3. From Ξ_k, Ψ_k is updated by solving:

$$\min_{\Psi} \|\Xi_k \Psi - u\|_1. \tag{2.4.18}$$

4. From Ξ_k, Ψ_k is updated by solving problem 2.4.17.
5. Steps 3 and 4 are repeated until convergence.
6. If Ψ_k or Ξ_k is too sparse (usually due to unlucky initialization of Ψ_k), the term is rejected and another initialization is attempted at step 2.
7. After a few successive rejections, the algorithm stops.

Numerical experiments have shown that this algorithm yields a good results when the solutions fullfill the requirements of the decomposition, which is nevertheless not guaranteed.

$u_{\text{loc}} = u - u_{\text{glob}}$ is therefore a good candidate for the Optimal Transport strategy previously described because it has a localized support. However, in many cases, and even if u itself is positive, u_{loc} has negative values. In order to be able to apply the ANOVA-based Optimal Transport strategy to negative-valued functions, the following describes how to perform couplings and compute transport plans between signed measures.

Once again, a solution $u(\bullet, \boldsymbol{p})$ is normalized by dividing it by its sum $\int_\Omega u(\bullet, \boldsymbol{p})$, which allows to keep the idea of mass conservation, even if negative mass is now introduced. For this reason, the integral of u needs to have always the same sign, which is a special requirement for u_{loc} without which divisions by 0 could occur.

Once the normalization is done, and the zero values are removed, the signed measure μ is divided into two parts, the positive part $\mu^+ > 0$ and the negative part $\mu^- > 0$ such that $\mu = \mu^+ - \mu^-$. There are infinitely many ways to obtain such a decomposition. The one chosen is the trivial one, the so-called Jordan decomposition [184], which is the only one in which μ^+ and μ^- have disjoint supports.

When coupling two signed measures, mass can be coupled according to three mechanisms:

- Positive mass of the first measure can be coupled to positive mass of the second measure.
- Negative mass of the first measure can be coupled to negative mass of the second measure.
- Positive mass of the "largest" measure, that is the one such that $\frac{\int |u|}{\int u}$ is the largest, can be coupled to negative mass of itself.

The coupling is thus divided in three steps which correspond to these three mechanisms:

1. Partial Optimal Transport [185] is used to cancel out the excess of positive and negative mass of the "largest" measure.
2. Regular Optimal Transport is used to couple the remaining positive mass with the positive mass of the other distribution.

3. Regular Optimal Transport is used to couple the remaining negative mass with the negative mass of the other distribution.

By applying the barycentric projection, two transport maps are obtained, T^+ which transports the positive mass and T^- which transports the negative mass.

The remaining negative and positive mass from the largest measure is added to the other measure in a way that it cancels itself out: for a unit of positive mass in position x^+ coupled with a negative unit of mass in position x^-, T^+ and T^- (or their inverse depending on if the largest measure is the source or target) are extrapolated to estimate $T^+(x^+)$ and $T^-(x^-)$, and the same unit of mass is added in both the positive and negative parts of the other signed measure, in position $\frac{T^+(x^+)+T^-(x^-)}{2}$.

Remark: For the same reason that transport maps do not always exist, the inverse of the transport maps obtained by barycentric projection may not exist, but they can be approximated by using barycentric projection in the opposite way. △

This strategy, which allows to couple together signed measures and to compute the transport maps between them, can be introduced in the complete workflow discussed previously, hence making it possible to apply it for more general types of solutions. However, many special cases remain difficult to address, and require further investigation.

2.4.3 Numerical experiment

The test case proposed to evaluate the method is the construction of a parametric solution of a parametric wave equation, formulated as follows:

$$\begin{cases} \Delta u(\boldsymbol{x},t) = \dfrac{\partial^2 u}{\partial t^2}(\boldsymbol{x},t) & \forall (\boldsymbol{x},t) \in \Omega \times [0,T] \\ \dfrac{\partial u}{\partial n}(\boldsymbol{x},t) = 0 & \forall (\boldsymbol{x},t) \in \partial\Omega \times [0,T] \\ u(\boldsymbol{x},0) = u_0(\boldsymbol{x};\boldsymbol{p}) & \forall \boldsymbol{x} \in \Omega \end{cases} \tag{2.4.19}$$

where $\Omega = [0,4] \times [0,4]$, $T = 0.5$ and $u_0(\boldsymbol{x};\boldsymbol{p}) = \sum_{k=1}^{3} e^{-(x^1-p^k)^2-(x^2-k)^2}$.

The problem is solved for 15 sets of parameters chosen following the design of experiments prescribed for ANOVA-PGD using Lagrange P1 finite elements. The solutions of this equation have a small support therefore no particular preprocessing is required, however they contain negative values so the coupling procedure described in the previous section is necessary. This problem is quite unsolvable with linear reduced bases because the location of the features depend strongly on the value of the parameters.

The parametric solution built is the 2D field function of time and of the parameters, which make a total of 4 parameters.

Results for four different sets of parameters are shown and compared to the equivalent solution obtained using finite elements in Figs. 2.42, 2.43, 2.44 and 2.45. The results that are shown correspond to the test set, they were not used to build the model.

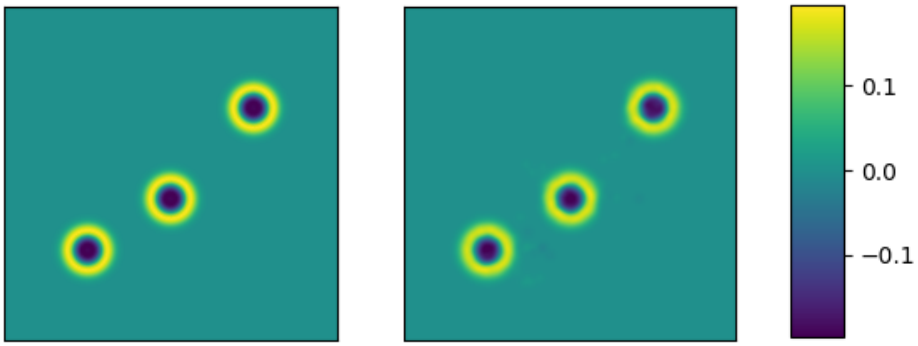


FIGURE 2.42: Solution of the problem for $t = 0.2$ and $\boldsymbol{p} = (2.9, 2.3, 1.2$ using finite elements (left) and the surrogate model based on Optimal Transport (right).

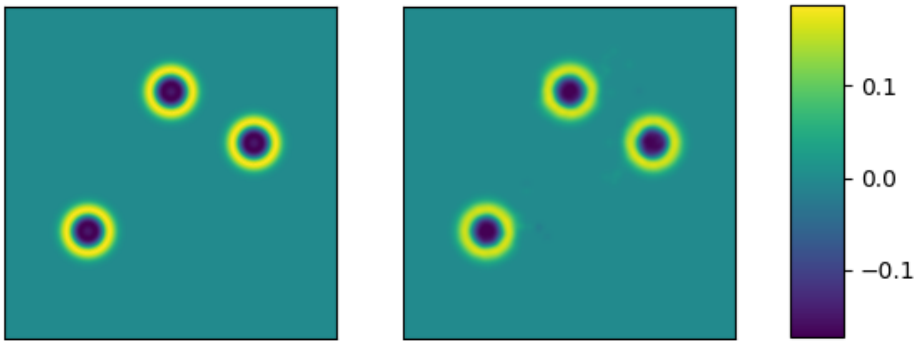


FIGURE 2.43: Solution of the problem for $t = 0.22$ and $\boldsymbol{p} = (2.7, 1, 1.7)$ using finite elements (left) and the surrogate model based on Optimal Transport (right).

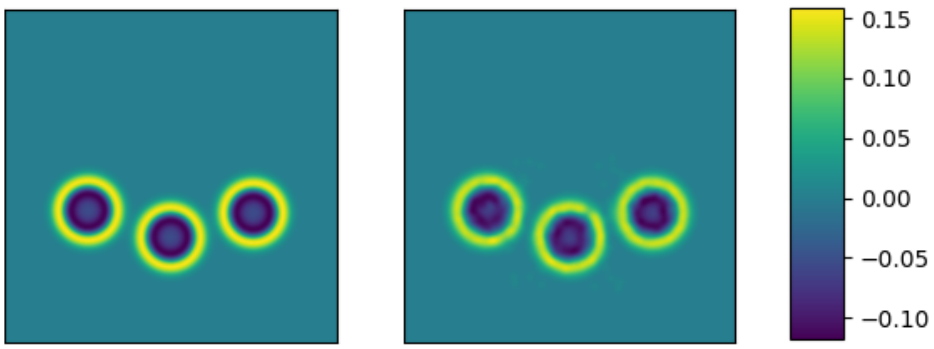


FIGURE 2.44: Solution of the problem for $t = 0.32$ and $\boldsymbol{p} = (2.4, 2.7, 2.4)$ using finite elements (left) and the surrogate model based on Optimal Transport (right).

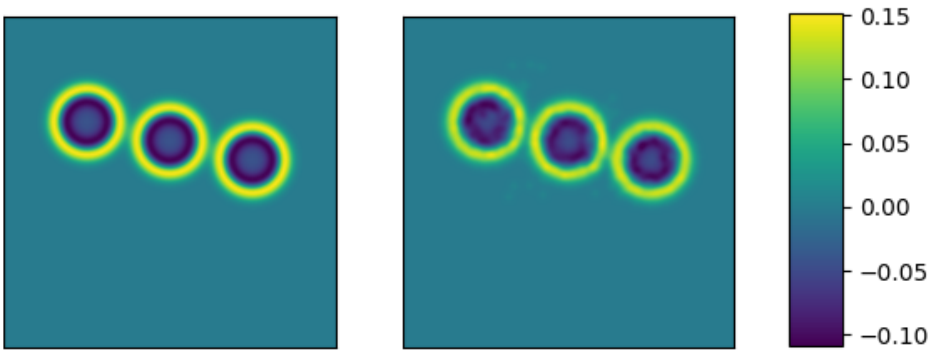


FIGURE 2.45: Solution of the problem for $t = 0.36$ and $\boldsymbol{p} = (1.3, 1.5, 1.7)$ using finite elements (left) and the surrogate model based on Optimal Transport (right).

The surrogate model is able to predict quite well the solution from a low amount of data. Indeed, to obtain such localization of the features with linear reduced bases would require a very large amount of data. This shows the power of Optimal Transport to deal with such problems.

3

Data assimilation and Hybrid Modelling

In the realm of scientific modeling, the application of physics-based models has long been a cornerstone of industrial practices. These models, often formulated using partial differential equations, provide a valuable framework for understanding and predicting the behavior of complex systems. However, despite their widespread use, these physics-based models are not without their limitations. Discrepancies between model predictions and real-world measurements are frequently observed, indicating the presence of inherent imperfections within the models.

The recognition of such disparities between model predictions and experimental data has given rise to the concept of hybrid modeling, which seeks to refine and enhance physics-based models through the integration of measured data. By incorporating real-world observations, it becomes possible to correct and enrich these models, bridging the gap between theoretical formulations and the actual behavior of the system under investigation.

Yet, this idea of using measurements to modify existing physics-based models presents several important questions. Firstly, can the collected data alone be used to construct a model, whether an entirely new model or the enrichment of an existing one, or is it necessary to acquire additional or improved data ? Secondly, how can an existing, well-posed physics-based formulation be modified to incorporate the assimilation of measurement data, without compromising its fundamental properties and integrity?

They questions have been addressed extensively in the literature, both the difficulty of working with partial data [74, 75] and using data to enrich models [186–188]. They are revisited in this chapter, which delves into the realm of hybrid modeling, exploring different strategies to build, enrich, and correct physics-based models using measured data. The chapter is structured into three main sections, each addressing a distinct aspect of hybrid modeling.

The first section addresses the challenge of modeling systems under partial observability. It investigates how incomplete or limited data can be leveraged to construct a reliable model that captures the essential dynamics of the system.

The second section of this chapter explores the process of enriching a physics-based model from data. It explores the learning of source terms or loading terms that can be incorporated into the existing equations, effectively augmenting the model to better match experimental measurements.

Finally, the third section focuses on the correction of a physics-based model from data by identifying and refining the physical properties of the system. By comparing model predictions with measured data, it becomes possible to identify areas of discrepancy and adjust the model parameters to align more closely with reality. This section examines parameter calibration by promoting sparsity, showcasing its potential to enhance the accuracy and predictive power of physics-based models.

Through this exploration of hybrid modeling techniques, this chapter aims to shed light on the challenges, approaches, and opportunities associated with integrating measured data into physics-based models. By harnessing the power of data, we can advance our understanding of complex systems, improve the reliability of predictions, and unlock new possibilities for scientific and industrial applications.

3.1 Modeling systems from partial observations

When proceeding with data for modeling purposes, whether in the context of fully data-driven or bybrid modeling, a recurrent issue concerns data accessibility. Sometimes, the considered system is not globally accessible, only a small part of it being reachable to perform measurements.

The present section addresses a conceptual issue, that will be discussed on an example simple enough to be fully understood, and at the same time complex enough to encompass all the modeling issues discussed here.

If there is a part of a system inaccessible for observation in which a loading that is applied cannot be either observed or measured, and that influences the measures performed in the observable part of the system, different questions arise:

- Is there a model connecting the observable input(s) to the corresponding output(s), knowing that they are impacted by the hidden dynamics of the system? Is it unique?
- Under which conditions such a model could exist? How to find it?
- How to formulate it correctly? Is it well-posed and consistent?
- How to learn it?
- Which is the impact of these hidden dynamics on the learning process?

This section aims at revisiting the construction of models in the domains exhibiting partial observability, in both the steady and transient cases, while following a double approach: the usual algebraic formulation and the one concerned by Machine Learning approaches.

3.1.1 On the existence of models relating observable features

Let us first assume a large system, whose state is described by a number of state variables. The variables involved in the state description are supposed well defined. However, the model governing the state or its time evolution is assumed unknown and the data describing the state is only observable and measurable on part of the system, remaining unattainable in the rest of the system. Previous analysis on the field can be found in [74–77].

For instance, in the case of the two-masses oscillator depicted in Fig. 3.1, it is assumed that the state is perfectly defined by the position and momentum of each mass, however only the state of the second mass is accessible (and thus, measurable). A natural question concerns the possibility to learn the model that governs the observable state (q_2, p_2) while ignoring the state of the first mass (q_1, p_1).

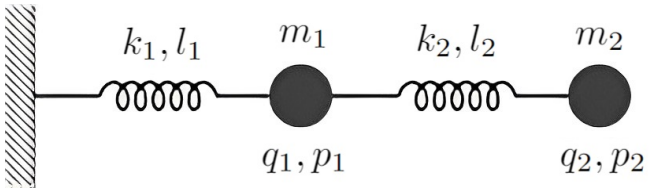


FIGURE 3.1: Oscillator composed of two masses, two linear springs of stiffness k_1 and k_2, reference lengths l_1 and l_2, and whose state is defined by the position and momentum of each mass (q_1, p_1, q_2, p_2).

In what follows, this question is addressed using a quite generic algebraic rationale in two situations: a model that does not depend on time and a transient problem. First, generic settings are considered, then the reasoning will be applied to multiple-mass oscillators.

Time-independent problem

A generic linear time-independent model can be expressed from

$$\mathbf{K}\mathbf{U} = \mathbf{F}, \tag{3.1.1}$$

which, considering the observable variables $\mathbf{U_o}$ and the internal ones $\mathbf{U_i}$, can be rewritten as

$$\begin{pmatrix} \mathbf{K_{oo}} & \mathbf{K_{oi}} \\ \mathbf{K_{io}} & \mathbf{K_{ii}} \end{pmatrix} \begin{pmatrix} \mathbf{U_o} \\ \mathbf{U_i} \end{pmatrix} = \begin{pmatrix} \mathbf{F_o} \\ \mathbf{F_i} \end{pmatrix}. \tag{3.1.2}$$

Developing the last equation yields

$$\mathbf{K_{io}}\mathbf{U_o} + \mathbf{K_{ii}}\mathbf{U_i} = \mathbf{F_i} \rightarrow \mathbf{U_i} = \mathbf{K_{ii}^{-1}}\mathbf{F_i} - \mathbf{K_{ii}^{-1}}\mathbf{K_{io}}\mathbf{U_o}, \tag{3.1.3}$$

and introducing the resulting expression of $\mathbf{U}_i$ into the development of the first, it results (this is known as static condensation [78] or Guyan reduction [79])

$$(\mathbf{K_{oo}} - \mathbf{K_{oi}}\mathbf{K_{ii}^{-1}}\mathbf{K_{io}})\mathbf{U_o} = \mathbf{F_o} - \mathbf{K_{oi}}\mathbf{K_{ii}^{-1}}\mathbf{F_i}, \tag{3.1.4}$$

which can be rewritten as

$$\tilde{\mathbf{K}}_{\mathbf{oo}}\mathbf{U_o} = \mathbf{F_o} - \tilde{\mathbf{F}}_{\mathbf{i}}, \tag{3.1.5}$$

with

$$\begin{cases} \tilde{\mathbf{K}}_{\mathbf{oo}} = (\mathbf{K}_{\mathbf{oo}} - \mathbf{K}_{\mathbf{oi}}\mathbf{K}_{\mathbf{ii}}^{-1}\mathbf{K}_{\mathbf{io}}) \\ \tilde{\mathbf{F}}_{\mathbf{i}} = \mathbf{K}_{\mathbf{oi}}\mathbf{K}_{\mathbf{ii}}^{-1}\mathbf{F}_{\mathbf{i}} \end{cases} . \tag{3.1.6}$$

Remark 1:

- If $\mathbf{F_i} = \mathbf{0}$, a direct relation exists between $\mathbf{U_o}$ and $\mathbf{F_o}$.
- In the case of a 1D system in which only the borders of the interval are accessible (observable), $\mathbf{U_o}$ and $\mathbf{F_o}$ contains two components. By applying ${\mathbf{U_o}}^T = (1, 0)$, the resulting $\mathbf{F_o}$ represents the first column of $\tilde{\mathbf{K}}_{\mathbf{oo}}$, and the solution $\mathbf{F_o}$ related to ${\mathbf{U_o}}^T = (0, 1)$ will represent the second column of $\tilde{\mathbf{K}}_{\mathbf{oo}}$.
- In the same one-dimensional system, when $\mathbf{F_i} \neq \mathbf{0}$, there are two effective internal variables, the components of $\tilde{\mathbf{F}}_{\mathbf{i}}$. Thus, all the richness of $\mathbf{F_i}$ boils up to these two components generating some sort of irreversibility: $\tilde{\mathbf{F}}_{\mathbf{i}}$ can be obtained from $\mathbf{F_i}$, but the last one cannot be obtained from the former. The condensation of the internal degrees of freedom into the observable one produces an entropy increase in the theory of information sense: there are many micro-states $\mathbf{F_i}$ associated to the macro-state $\tilde{\mathbf{F}}_{\mathbf{i}}$.
- Computing these two effective internal variables just described requires an extra-calculation. For example, if $\mathbf{U_o} = \mathbf{0}$, then $\mathbf{F_o} = \tilde{\mathbf{F}}_{\mathbf{i}}$.

Time-dependent problem

A general linear second-order dynamical system can be expressed from

$$\mathbf{M}\ddot{\mathbf{U}} + \mathbf{C}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{F}, \tag{3.1.7}$$

which, applying Fourier transform, becomes

$$-\omega^2\mathbf{M}\mathbb{U} + j\omega\mathbf{C}\mathbb{U} + \mathbf{K}\mathbb{U} = \mathbb{F}, \tag{3.1.8}$$

with j the imaginary ($j^2 = -1$) and $\mathbb{U}$ and $\mathbb{F}$ the Fourier transforms of $\mathbf{U}$ and $\mathbf{F}$ respectively. The previous equation can be rewritten as

$$\mathbf{K}^*\mathbb{U} = \mathbb{F}, \tag{3.1.9}$$

with $\mathbf{K}^* = -\omega^2\mathbf{M} + j\omega\mathbf{C} + \mathbf{K}$, that can be separated in the same way considered in the time-independent case, but now, for each possible frequency (ω) involved in the loading and operating in the complex domain, leading to

$$\tilde{\mathbf{K}}^*_{\mathbf{oo}}\mathbb{U}_{\mathbf{o}} = \mathbb{F}_{\mathbf{o}} - \tilde{\mathbb{F}}_{\mathbf{i}}, \tag{3.1.10}$$

which proves that all the discussion previously addressed in the time-independent case remains valid when the Fourier transform applies.

Thus, one could expect that a model relating observable variables might exist as well (and could be learned from collected data) in the time domain, under certain constraints, as the one referred in *Remark 2* below, due to the dependence of $\tilde{\mathbb{F}}_i$ on the internal loading $\mathbb{F}_i$. This would imply the consideration of the history of the variables, which is naturally implicit in the Fourier transform.

Remark 2:

The just described rationale applies in the forced regime, i.e. far from the transient effects induced by the initial condition. In order to address transient regimes, the Laplace transform could be employed instead of the Fourier one. However, it is well known that the Laplace inverse transform is more challenging from the numerical point of view than Fourier's. It is also important to note that the Fourier transform of the internal loading considered in the training stage should remain invariant to ensure the validity of the learned model.

Neural Network-based modeling

In many cases Artificial Intelligence, and more concretely Machine Learning, aims at extracting the model that relates measured inputs to the corresponding outputs [153, 189]. In general, the measured output depends on the whole internal state. For instance, in a structural dynamics problem where the loading (evolving in time) constitutes the problem's input, the corresponding response is the displacement at each point and time; whereas the corresponding output-data is the measured displacement in a certain observable point of the structure.

In physics-based structural mechanics, the internal response (displacement at any location and time instant) is obtained by discretization of the continuum mechanics model, consisting of the momentum balance and the constitutive equations and; from this internal state, the output of interest is directly extracted at each time instant. Alternatively, Machine Learning looks for the direct relation between observables, the input action and the measured response that, as just mentioned, can depend on the present and past values of a series of non observed internal variables [190].

Recurrent Neural Networks (RNN) [42] and their Long-Short Time Memory counterparts (LSTM) [43] address such situations by trying to model the time evolution of the internal state at the same time it constructs the model relating the observable input and output (action and response).

Addressing time-dependent problems in the time domain

Finally, it is possible to address time-dependent problems modeling but directly operating in the time domain, instead of operating in the Fourier domain as was considered before. For the sake of simplicity, this is illustrated with the first order dynamical system

$$\mathbf{C}\dot{\mathbf{U}} + \mathbf{K}\mathbf{U} = \mathbf{F}, \tag{3.1.11}$$

whose implicit time discretisation reads

$$\mathbf{C}\mathbf{U}^n + \Delta t \mathbf{K}\mathbf{U}^n = \Delta t \mathbf{F}^n + \mathbf{C}\mathbf{U}^{n-1}, \tag{3.1.12}$$

with Δt being the considered time step. This equation can be rewritten in the more compact form

$$\mathbf{K}^*\mathbf{U}^n = \mathbf{F}^{*,n} + \mathbf{C}\mathbf{U}^{n-1}, \tag{3.1.13}$$

with $\mathbf{K}^* = \mathbf{C} + \Delta t \mathbf{K}$ and $\mathbf{F}^{*,n} = \Delta t \mathbf{F}^n$.

The sequencing of these equations can be written, inspired by the Dynamic Mode Decomposition [46], in the matrix form

$$\mathbf{K}^*\left[\mathbf{U}^n, ..., \mathbf{U}^1\right] = \left[\mathbf{F}^{*,n}, ..., \mathbf{F}^{*,1}\right] + \mathbf{C}\left[\mathbf{U}^{n-1}, ..., \mathbf{U}^0\right], \tag{3.1.14}$$

and by defining the extended vectors $\mathcal{U}$ and $\mathcal{F}$,

$$\begin{cases} \mathcal{U}^T = [\mathbf{U}^{n^T}, \mathbf{U}^{n-1^T}, ..., \mathbf{U}^{0^T}] \\ \mathcal{F}^T = [\mathbf{F}^{n^T}, \mathbf{F}^{n-1^T}, ..., \mathbf{F}^{1^T}] \end{cases}, \tag{3.1.15}$$

and the extended matrix $\mathcal{K}$

$$\mathcal{K} = \begin{pmatrix} \mathbf{K}^* & -\mathbf{C} & \mathbf{0} & ... & ... \\ \mathbf{0} & \mathbf{K}^* & -\mathbf{C} & \mathbf{0} & ... \\ ... & ... & ... & ... & ... \end{pmatrix}, \tag{3.1.16}$$

The previous system reads

$$\mathcal{K}\mathcal{U} = \mathcal{F}, \tag{3.1.17}$$

where the solution $\mathcal{U}$ is, in general, computed from the $\mathcal{K}$ matrix pseudo-inverse.

This algebraic system can be addressed by using the same rationale as the one applied before, but this time, the model explicitly involves the time evolution of the input(s) and output(s), reinforcing the result already obtained when using the Fourier transform.

Another alternative formulation, more aligned with the use of machine learning techniques that will be presented afterwards, consists in writing the explicit integration

$$\mathbf{C}\mathbf{U}^n = \Delta t\mathbf{F}^n - \Delta t\mathbf{K}\mathbf{U}^{n-1} + \mathbf{C}\mathbf{U}^{n-1}, \tag{3.1.18}$$

that can be reformulated as

$$\mathbf{U}^n = \mathbf{A}\mathbf{F}^n + \mathbf{B}\mathbf{U}^{n-1}, \tag{3.1.19}$$

perfectly expressible within the RNN architecture. When the model concerns only part of the state (the observable part), RNN and/or LSTM seem especially appealing to carry out the task.

3.1.2 Results for the analytical case in the linear setting

As mentioned, some numerical examples, simple enough to be perfectly understood, but complex enough to underline all the issues and methodological aspects just discussed.

The simplicity of the problem is only apparent. There are forces being applied to the internal masses, unknown and unobserved by the modeler, who, furthermore, totally ignores how many hidden masses are involved in the system. Three masses are considered in the present example, but they could come in any number, from one to the thousands.

When introducing all the system's degrees of freedom - in this case, the state of the three masses - in a model, the last one becomes larger but finally simpler to interpret and to learn, since all the data needed to properly describe the system is there, fully available. On the contrary, when considering only the data associated to one mass, while ignoring all the data related to all the other masses, the model seems simpler from its size, but very intricate nonetheless.

For this reason, and this is the motivation, the simplicity is only apparent, and allows for a more fruitful discussion on the issues and the conceptual questions previously addressed.

Learning in the Fourier space

The N-mass dynamical system considered here is illustrated in Fig. 3.2. The modeling includes inertia, elastic and damping behaviors.

The state of each mass is represented by $\mathbf{z}_i = (q_i, p_i)$, q_i and p_i being the i-th mass position and momentum respectively. The system state is described by the extended vector $\mathbf{Z}^T = (\mathbf{z}_1^T, ..., \mathbf{z}_N^T)$.

The usual model, coming from Newton's equation, can be expressed by

$$\dot{\mathbf{Z}} = \mathbf{T}\mathbf{Z} + \mathbf{J} + \mathbf{F}, \tag{3.1.20}$$

where matrix $\mathbf{T}$ includes the system properties, masses, spring stiffness and viscosity of the dampers. On the other hand, $\mathbf{J}$ is a constant vector (in the linear case addressed below) and $\mathbf{F}$ contains the external forces applied on the different masses, appearing at the odd positions in vector $\mathbf{F}$ (an explicit form of that matrix and those vectors will be given later).

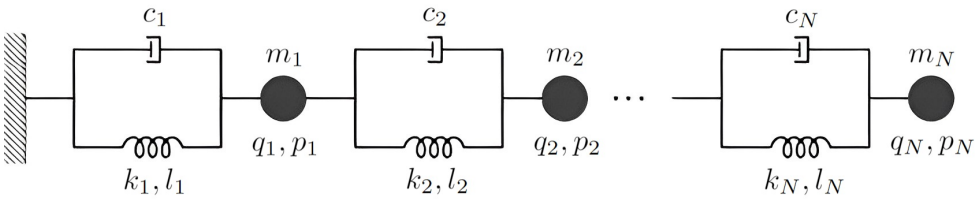


Figure 3.2: N-mass dynamical system

In the forced regime, the Fourier transform becomes a valuable route. The dynamical model in the Fourier domain reads

$$(-\mathbf{T} + j\omega\mathbf{I})\mathbb{Z} = \mathbb{J} + \mathbb{F}. \tag{3.1.21}$$

By defining the effective loading $\mathbb{S} = \mathbb{J} + \mathbb{F}$, and $\tilde{\mathbf{T}} = -\mathbf{T} + j\omega\mathbf{I}$, the matrix form that separates the degrees of freedom related to the measurable position (noted by q) and the derived momentum (p) is now written:

$$\begin{pmatrix} \tilde{\mathbf{T}}_{qq}(\omega) & \mathbf{T}_{qp} \\ \mathbf{T}_{pq} & \tilde{\mathbf{T}}_{pp}(\omega) \end{pmatrix} \begin{pmatrix} \mathbb{Z}_q(\omega) \\ \mathbb{Z}_p(\omega) \end{pmatrix} = \begin{pmatrix} \mathbb{S}_q(\omega) \\ \mathbb{S}_p(\omega) \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbb{S}_p(\omega) \end{pmatrix}. \tag{3.1.22}$$

Since $\mathbb{S}_q(\omega) = \mathbf{0}$, $\mathbb{Z}_p(\omega)$ can be expressed in terms of $\mathbb{Z}_q(\omega)$:

$$\tilde{\mathbf{T}}_{qq}(\omega)\mathbb{Z}_q(\omega) + \mathbf{T}_{qp}\mathbb{Z}_p(\omega) = \mathbf{0} \rightarrow \mathbb{Z}_p(\omega) = -\mathbf{T}_{qp}^{-1}\tilde{\mathbf{T}}_{qq}(\omega)\mathbb{Z}_q(\omega), \tag{3.1.23}$$

that, introduced into the second equation, leads to:

$$\left(\mathbf{T}_{pq} - \tilde{\mathbf{T}}_{pp}(\omega)\mathbf{T}_{qp}^{-1}\tilde{\mathbf{T}}_{qq}(\omega)\right)\mathbb{Z}_q(\omega) = \mathbb{S}_p(\omega), \tag{3.1.24}$$

which can be reshaped into the more compact form:

$$\mathbf{A}(\omega)\mathbb{Q}(\omega) = \mathbb{R}(\omega), \tag{3.1.25}$$

with $\mathbf{A}(\omega) = \mathbf{T}_{pq} - \tilde{\mathbf{T}}_{pp}(\omega)\mathbf{T}_{qp}^{-1}\tilde{\mathbf{T}}_{qq}(\omega)$, $\mathbb{Q}(\omega) \equiv \mathbb{Z}_q(\omega)$ and $\mathbb{R}(\omega) \equiv \mathbb{S}_p(\omega)$.

This way, the momentum no longer appears in the state variables, since it derives directly from the measurable position.

Now, the partition between the internal and the observable degrees of freedom can be enforced:

$$\begin{pmatrix} \mathbf{A_{oo}}(\omega) & \mathbf{A_{oi}}(\omega) \\ \mathbf{A_{io}}(\omega) & \mathbf{A_{ii}}(\omega) \end{pmatrix} \begin{pmatrix} \mathbb{Q}_\mathbf{o}(\omega) \\ \mathbb{Q}_\mathbf{i}(\omega) \end{pmatrix} = \begin{pmatrix} \mathbb{R}_\mathbf{o}(\omega) \\ \mathbb{R}_\mathbf{i}(\omega) \end{pmatrix} \tag{3.1.26}$$

that, following the aforementioned rationale, leads to

$$\tilde{\mathbf{A}}_\mathbf{oo}(\omega)\mathbb{Q}_\mathbf{o}(\omega) = \mathbb{R}_\mathbf{o}(\omega) - \tilde{\mathbb{R}}_\mathbf{i}(\omega), \tag{3.1.27}$$

with $\tilde{\mathbb{R}}_\mathbf{i}(\omega) = \mathbf{A_{oi}}(\omega)\mathbf{A_{ii}}^{-1}(\omega)\mathbb{R}_\mathbf{i}(\omega)$ and $\tilde{\mathbf{A}}_\mathbf{oo}(\omega) = \mathbf{A_{oo}}(\omega) - \mathbf{A_{oi}}(\omega)\mathbf{A_{ii}}^{-1}(\omega)\mathbf{A_{io}}(\omega)$, where the same remarks that were previously discussed apply.

This is illustrated with a system composed of 3 identical masses ($m_1 = m_2 = m_3 = m$), springs ($k_1 = k_2 = k_3 = k$) and dampers ($c_1 = c_2 = c_3 = c$), with the springs having a reference length also identical ($l_1 = l_2 = l_3 = l$). Forces can be applied on both the internal masses (the first two) as well as on the observable one, the third. The following values are considered: $m = 0.5$ kg, $c = 0.8$ Ns/m, $k = 1$ N/m, $l = 1$ m.

The dynamical model reads:

$$\begin{pmatrix} \dot{q}_1 \\ \dot{p}_1 \\ \dot{q}_2 \\ \dot{p}_2 \\ \dot{q}_3 \\ \dot{p}_3 \end{pmatrix} = \begin{pmatrix} 0 & 1/m & 0 & 0 & 0 & 0 \\ -2k & -2c/m & k & c/m & 0 & 0 \\ 0 & 0 & 0 & 1/m & 0 & 0 \\ k & c/m & -2k & -2c/m & k & c/m \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{m} \\ 0 & 0 & k & c/m & -k & -c/m \end{pmatrix} \begin{pmatrix} q_1 \\ p_1 \\ q_2 \\ p_2 \\ q_3 \\ p_3 \end{pmatrix} + \begin{pmatrix} 0 \\ k_1 l_1 - k_2 l_2 \\ 0 \\ k_2 l_2 - k_3 l_3 \\ 0 \\ k_3 l_3 \end{pmatrix} + \begin{pmatrix} 0 \\ F_1(t) \\ 0 \\ F_2(t) \\ 0 \\ F_3(t) \end{pmatrix}, \tag{3.1.28}$$

which, in the linear case and taking into account that $k_1 = k_2 = k_3 = k$ and $l_1 = l_2 = l_3 = l$, after applying Fourier transform leads to:

$$\begin{pmatrix} j\omega & -1/m & 0 & 0 & 0 & 0 \\ 2k & j\omega + 2c/m & -k & -c/m & 0 & 0 \\ 0 & 0 & j\omega & -1/m & 0 & 0 \\ -k & -c/m & 2k & j\omega + 2c/m & -k & -c/m \\ 0 & 0 & 0 & 0 & j\omega & -1/m \\ 0 & 0 & -k & -c/m & k & j\omega + c/m \end{pmatrix} \begin{pmatrix} \hat{q}_1 \\ \hat{p}_1 \\ \hat{q}_2 \\ \hat{p}_2 \\ \hat{q}_3 \\ \hat{p}_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ kl \end{pmatrix} \delta(\omega) + \begin{pmatrix} 0 \\ \hat{F}_1 \\ 0 \\ \hat{F}_2 \\ 0 \\ \hat{F}_3 \end{pmatrix}, \tag{3.1.29}$$

where the *hat* operator, $\hat{\bullet}$, refers to the Fourier transform of the masses positions, momentum and applied forces.

It is important to note that, in the nonlinear case described later on, since the spring stiffnesses depend on the spring elongation and the latter will obviously be different for each node - unlike here in the linear case - , the first vector of the right hand member will contain three non-vanishing spring contributions: $k_1 l_1 - k_2 l_2$, $k_2 l_2 - k_3 l_3$ and $k_3 l_3$.

By reordering the previous system, the position and momentum degrees of freedom can be grouped:

$$\begin{pmatrix} \tilde{\mathbf{T}}_{qq}(\omega) & \mathbf{T}_{qp} \\ \mathbf{T}_{pq} & \tilde{\mathbf{T}}_{pp}(\omega) \end{pmatrix} \begin{pmatrix} \hat{q}_1 \\ \hat{q}_2 \\ \hat{q}_3 \\ \hline \hat{p}_1 \\ \hat{p}_2 \\ \hat{p}_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \hline 0 \\ 0 \\ kl \end{pmatrix} \delta(\omega) + \begin{pmatrix} 0 \\ 0 \\ 0 \\ \hline \hat{F}_1 \\ \hat{F}_2 \\ \hat{F}_3 \end{pmatrix}, \tag{3.1.30}$$

and then, the momentum degrees of freedom $\hat{p}_i$ condensed into the ones related to the masses positions $\hat{q}_i$, as previously discussed:

$$\Big(\mathbf{T}_{pq} - \tilde{\mathbf{T}}_{pp}(\omega)\mathbf{T}_{qp}^{-1}\tilde{\mathbf{T}}_{qq}(\omega)\Big)\mathbb{Z}_q(\omega) = \mathbb{S}_p(\omega). \tag{3.1.31}$$

After separating the internal and observable degrees of freedom, it reads:

$$\left(\begin{array}{cc|c} A_{11}(\omega) & A_{12}(\omega) & A_{13}(\omega) \\ A_{21}(\omega) & A_{22}(\omega) & A_{23}(\omega) \\ \hline A_{31}(\omega) & A_{32}(\omega) & A_{33}(\omega) \end{array}\right)\left(\begin{array}{c} \hat{q}_1 \\ \hat{q}_2 \\ \hline \hat{q}_3 \end{array}\right) = \left(\begin{array}{c} 0 \\ 0 \\ \hline kl \end{array}\right)\delta(\omega) + \left(\begin{array}{c} \hat{F}_1 \\ \hat{F}_2 \\ \hline \hat{F}_3 \end{array}\right) \tag{3.1.32}$$

which allows making the model involving the observable degree of freedom, $\hat{q}_3$, explicit:

$$\left\{A_{33}(\omega) - \begin{pmatrix} A_{31}(\omega) & A_{32}(\omega) \end{pmatrix}\begin{pmatrix} A_{11}(\omega) & A_{12}(\omega) \\ A_{21}(\omega) & A_{22}(\omega) \end{pmatrix}^{-1}\begin{pmatrix} A_{13}(\omega) \\ A_{23}(\omega) \end{pmatrix}\right\}\hat{q}_3(\omega) =$$

$$kl\delta(\omega) + \hat{F}_3 + \begin{pmatrix} A_{31}(\omega) & A_{32}(\omega) \end{pmatrix}\begin{pmatrix} A_{11}(\omega) & A_{12}(\omega) \\ A_{21}(\omega) & A_{22}(\omega) \end{pmatrix}^{-1}\begin{pmatrix} \hat{F}_1(\omega) \\ \hat{F}_2(\omega) \end{pmatrix}, \tag{3.1.33}$$

Rearranging this in a more compact manner yields:

$$\tilde{A}_{33}(\omega)\hat{q}_3(\omega) = kl\delta(\omega) + \hat{F}_3(\omega) + \hat{F}_{i3}(\omega), \tag{3.1.34}$$

which represents the system transfer function.

Now, the final point concerns the data-driven model identification, that is, how to extract from the given data the different model components: $\tilde{A}_{33}$ and $\hat{F}_{\mathbf{i}3}$, for each involved frequency ω. In the last equation, the index $\mathbf{i}$ associated with $\hat{F}_{\mathbf{i}3}$ reflects all the effects coming from the unresolved degrees of freedom (internal unobserved masses).

Conceptually, the system identification could proceed as follows:

1. The free response associated to $F_3 = 0$ (only the loads on the internal masses apply), $q_3^f(t)$, is obtained (measured), the superscript $\bullet^f$ refers to the fact that the observable mass remains free of loading.

2. Then, for a non-null (and measurable) applied loading on the observable mass, $F_3 \neq 0$, the system response $q_3(t)$ is recorded, which is now a consequence of all the loading terms involved in the right hand member of the previous equation.

3. The difference between the forced and free displacement can be obtained from $\Delta q_3(t) = q_3(t) - q_3^f(t)$, allowing for the computation of its Fourier transform $\widehat{\Delta q_3}(\omega)$.

4. Finally, by means of the just calculated $\widehat{\Delta q_3}$ and the Fourier transform of the measurable force $\hat{F}_3(\omega)$, the model coefficient $\tilde{A}_{33}(\omega)$ is learned from

$$\tilde{A}_{33}(\omega) = \frac{\hat{F}_3(\omega)}{\widehat{\Delta q_3}(\omega)}. \tag{3.1.35}$$

In this example, the loading applied to the system reads:

$$\left\{\begin{array}{l} F_1(t) = 2\cos(2\pi t) \\ F_2(t) = 2\cos\left(\dfrac{\pi}{4}t\right) \\ F_3(t) = 2\cos\left(\dfrac{\pi}{2}t\right) \end{array}\right., \tag{3.1.36}$$

The free and forced responses and their Fourier transforms are depicted respectively in Figs. 3.3 and 3.4. This loading is used to generate the synthetic data that will serve to identify the model's output $q_3(t)$ later on as a function of the observed load $F_3(t)$. During the training process of that model, $F_1(t)$ and $F_2(t)$ are fully ignored.

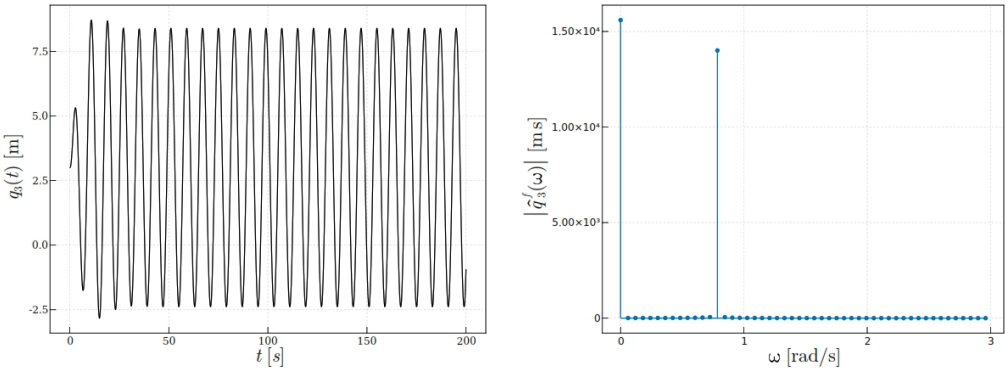


FIGURE 3.3: Free response ($F_3(t) = 0$): (left) $q_3^f(t)$; and (right) $\hat{q}_3^f(\omega)$.

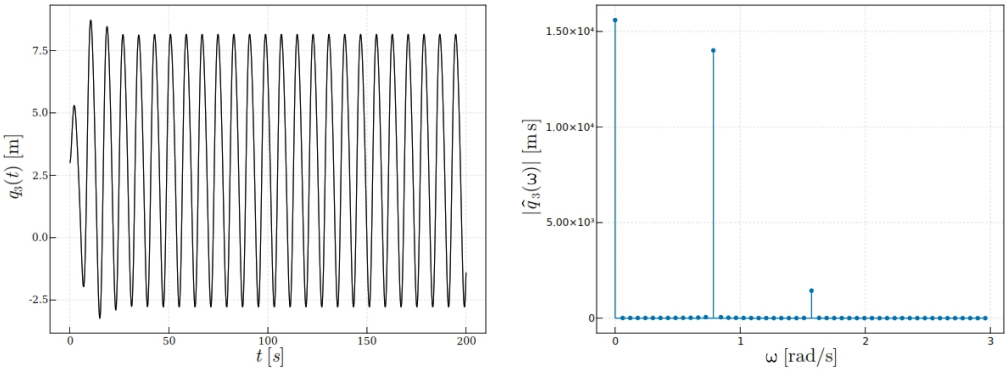


FIGURE 3.4: Response: (left) $q_3(t)$; and (right) $\hat{q}_3(\omega)$.

Figure 3.5 shows the response difference $\Delta q_3(t) = q_3(t) - q_3^f(t)$ and its Fourier transform $\widehat{\Delta q_3}(\omega)$ on the domain in which the difference $\Delta q_3(t)$ becomes almost stabilized, meaning the transient component almost vanishes.

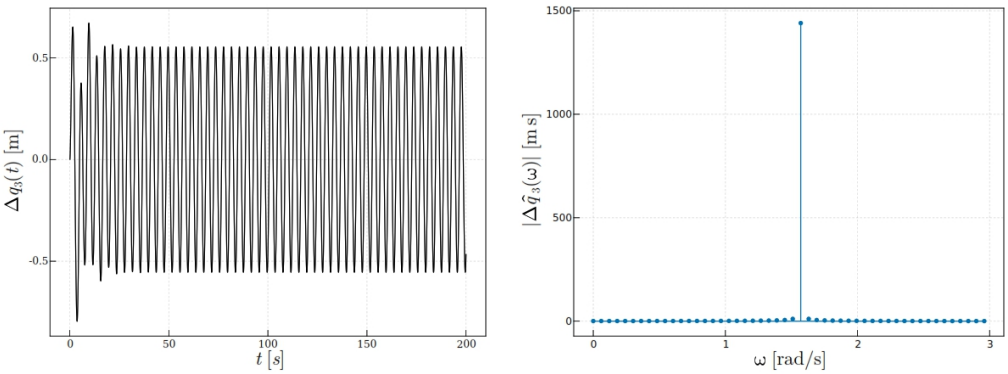


FIGURE 3.5: Response difference: (left) $\Delta q_3(t)$; and (right) $\widehat{\Delta q_3}(\omega)$.

Now, when comparing the reference solution, obtained by the reference analytical

model $\tilde{A}_{33} = -1.4848 + 1.0221j$, to the one obtained from the learned model $\tilde{A}_{33} = -1.5242 + 0.9823j$ (at principal frequency), an excellent accuracy can be noticed.

3.1.3 Results of RNN and LSTM time simulations in both the linear and the nonlinear settings

The 3-masses dynamical system is again considered for the time response scenario. The dynamical problem is integrated numerically to obtain the ground truth, that is, the reference solution. The computed data will be used to train the different neural networks, the RNN and the LSTM.

In both cases, the input data consists of the force F_3 and position q_3 in the previous time steps, that results in the surrogate $\mathcal{H}$:

$$\tilde{q}_3^i = \mathcal{H}\left(\begin{pmatrix} F_3^i \\ F_3^{i-1} \\ \vdots \\ F_3^{i-n+1} \end{pmatrix}, \begin{pmatrix} q_3^{i-1} \\ q_3^{i-2} \\ \vdots \\ q_3^{i-n} \end{pmatrix}\right), \tag{3.1.37}$$

where $\tilde{q}_3^i$ is the prediction of q_3 at time step i.

As Eq. (3.1.37) reflects, different memory lengths from the use of positive integer n, $(n \geq 0)$, are taken into account. For $n \neq 0$, an initialization issue occurs.

In the case considered here, the larger memory is the price paid ignore the internal forces, whose consequences on the observed variables are learned from the time evolution of the last.

The initialization can be carried out following two routes:

- If one is interested in the forced regime, the long-time solution does not depend on the initialization.
- To obtain the transient solution, one could consider a coarser model that updates the state from the just previous state until completing the first n values. Then the LSTM can take over.

In the present case, as previously indicated, the focus is on proving under which conditions a model relating observable inputs and outputs exists, despite the existence of hidden dynamics, resulting in a noticeable larger memory. For that reason, in the simulations considered here, the first n values are assumed known.

Using a simple recurrent neural network

Firstly, a RNN surrogate model is used, with the memory defined in Eq. (3.1.37) $n = 2$. The considered data for training comes from the integration of the dynamical system, in both the linear and nonlinear cases.

The data consists of 10000 states of the observable variables (coming as indicated from the standard integration of the dynamical system). This data are divided into two sets, the training and the testing ones, the former containing 80% of the points and the latter the remaining 20%.

The RNN consists of a single layer with one output $\tilde{q}_3^i$, in reference to Eq. (3.1.37). The network parameters and the initialization choices are the ones reported in [191]. The algorithm is trained during 1500 epochs, although the use of fewer epochs also leads to similar results.

The linear problem considers, once more: $m_1 = m_2 = m_3 = 0.5kg$, $c = 0.8Ns/m$, $k_1 = k_2 = k_3 = 1N/m$, and $l_1 = l_2 = l_3 = 1m$, expressing the applied loading the following way:

$$\begin{cases} F_1(t) = 2\cos(2\pi t) \\ F_2(t) = 2\cos\left(\frac{\pi}{4}t\right) \\ F_3(t) = \frac{t}{t_{max}} + \cos\left(\frac{\pi}{2}t\right) \end{cases}, \tag{3.1.38}$$

with $t_{max} = 500s$. This loading is used to generate the synthetic data that will serve afterwards to identify the model $q_3(t)$ as a function of the observed load $F_3(t)$. During the training process of that model, $F_2(t)$ and $F_3(t)$ are again completely neglected.

The computed results from the trained network are given in Fig. 3.6, the mean absolute percentage errors (MAPE) are 1.38% on the training set and 2.18% on the testing set.

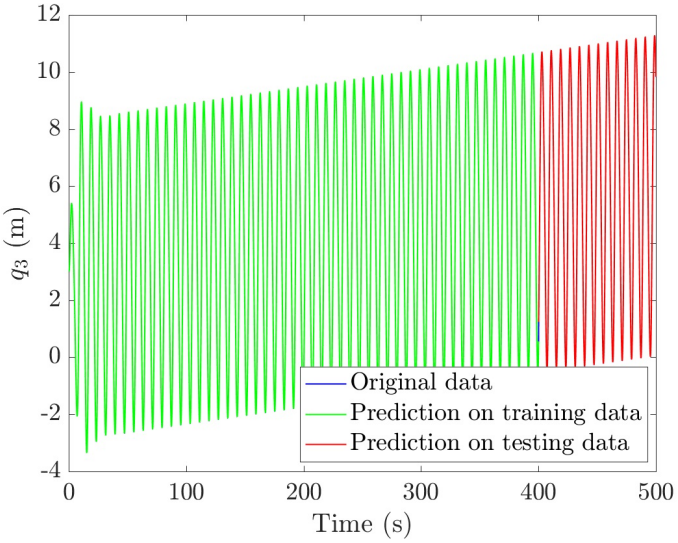


FIGURE 3.6: Prediction of the observable position $\tilde{q}_3(t)$ computed from a trained RNN with $n = 2$ (colors green and red mark the training and testing sets, respectively). It can be noted that the blue curve is not visible because it is almost exactly under the green and red curves.

The same RNN (now with $n = 3$ in reference to Eq. (3.1.37)) is now employed to tackle a nonlinear dynamical system, with similar parameters to the ones considered in the linear case, except in what concerns the springs stiffnesses, now given by:

$$\begin{cases} k_1 = k_{01}(1 + \alpha\Delta l_1) \\ k_2 = k_{02}(1 + \alpha\Delta l_2) \\ k_3 = k_{03}(1 + \alpha\Delta l_3) \end{cases}, \tag{3.1.39}$$

with $k_{01} = k_{02} = k_{03} = 10\ N/kg$, $\alpha = 10^{-4}\ m^{-1}$ (arbitrary, although carefully tuned to maintain the stability of the simulation) and where $\Delta l_\bullet$ is the elongation of the corresponding spring, i.e. $\Delta l_2 = q_2 - q_1 - f_2$, $\Delta l_3 = q_3 - q_2 - f_3$ and $\Delta l_1 = q_1 - l_1$.

The considered loading reads:

$$\begin{cases} F_1(t) = 2\cos(2\pi t) \\ F_2(t) = 2\cos\left(\frac{\pi}{4}t\right) \\ F_3(t) = \cos\left(\frac{\pi}{2}t\right) \end{cases} . \tag{3.1.40}$$

The results concerning the nonlinear dynamical system are reported in Fig. 3.7, and, for the sake of clarity, the associated absolute error in reported in Fig. 3.8, with a mean absolute percentage error (MAPE) of 1.34% in the training set and 1.29% in the testing set. The error is slightly larger in the training set, probably due to the larger transient phase presenting higher peaks.

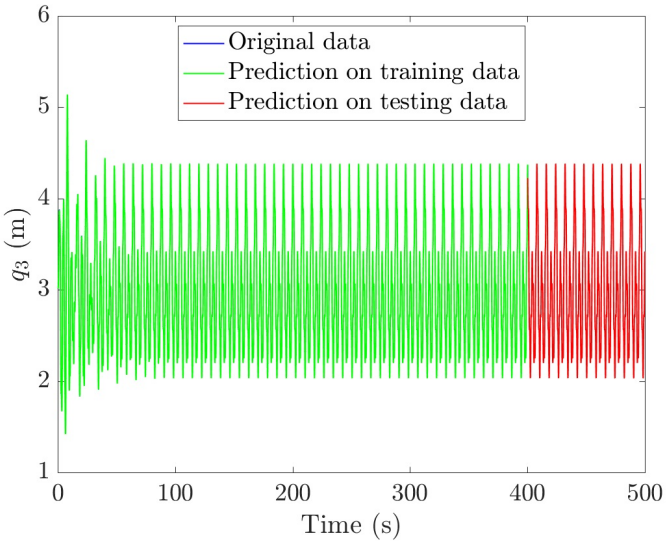


FIGURE 3.7: Prediction of the observable position $\tilde{q}_3(t)$ in the nonlinear case, computed by a trained RNN with $n = 3$ (again, colors refer to the training and testing sets). It can be noted that the blue curve is not visible because it is almost exactly under the green and red curves.

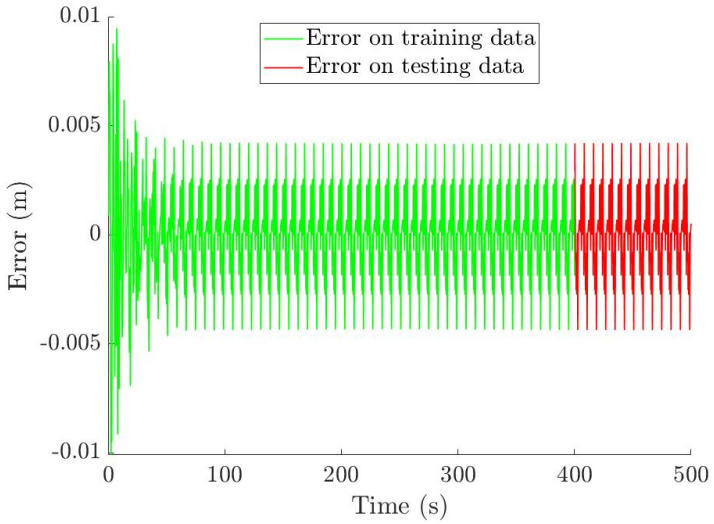


FIGURE 3.8: Error in the prediction of the observable position $\tilde{q}_3(t)$ in the nonlinear case, computed by a trained RNN with $n = 2$ (the same color code applies).

Using a LSTM recurrent neural network

The same linear and nonlinear dynamical systems are now processed by LSTM cells, with the same network parameters and initializations used for the RNN.

When addressing the linear case, the computed results are given in Fig. 3.9, with a MAPE of 0.84% in the training set and 1.33% in the testing set. The results in the nonlinear case are reported in Fig. 3.10, and again, for the sake of clarity, the associated absolute error is presented in Fig. 3.11, presenting a MAPE of 0.15% in the training set and 0.14% in the testing set. The error is again slightly larger in the training set for the same reasons given before.

As expected, LSTM outperforms RNN for a large number of epochs. It was noticed that, by reducing the number of epochs, RNN outperforms LSTMs because convergence is more easily achieved using a lower number of parameters. The error in the linear case is larger, possibly due to the fact that it involves close to zero values which negatively impact the error calculation.

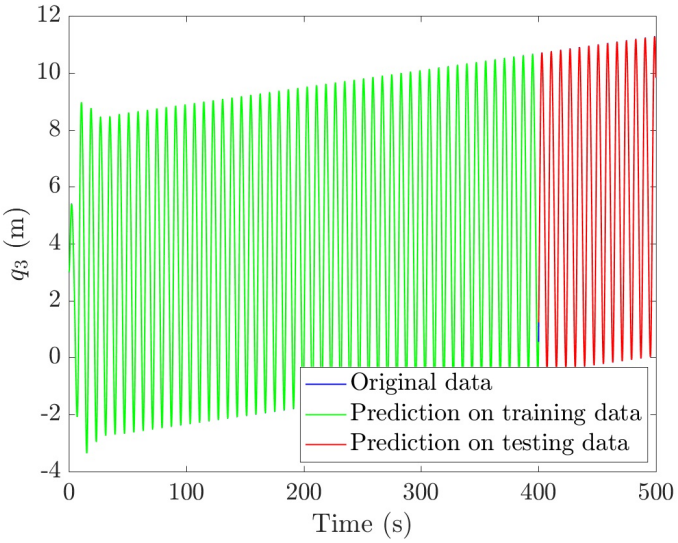


Figure 3.9: Prediction of the observable position $\tilde{q}_3(t)$ computed by a trained LSTM with $n = 2$ (the same color code is employed). It can be noted that the blue curve is not visible because it is almost exactly under the green and red curves.

It must be noted that several experiments with various number of elements, different damping coefficients, stiffnesses, lengths and masses have been carried out with similarly satisfactory results (MSE error always below 0.07 for both training and testing).

3.2 Hybrid modeling by learning a source term

After addressing the issue of collecting the right data to create models, arises the question of how to combine the data at hand with a physics-based model to create a hybrid model.

As mentioned in section 1.2.2, the PINN is a tempting route to answer this question in a simple manner: the equations coming from the physics can be embedded in the loss function to define the solution of the well-posed physical problem, and then by

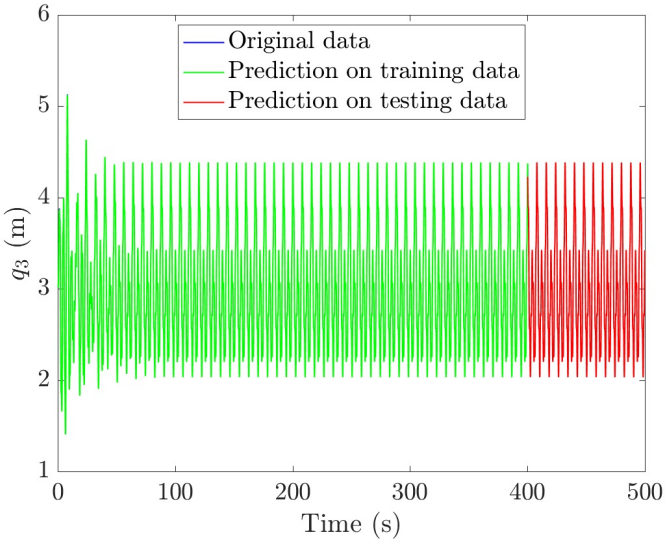


FIGURE 3.10: Prediction of the observable position $\tilde{q}_3(t)$ in the nonlinear case, computed by a trained LSTM with $n = 3$ (same color code). It can be noted that the blue curve is not visible because it is almost exactly under the green and red curves.

adding constraints coming from data, the solution becomes a compromise between all the constraints, trying to model the data while retaining physical sense.

Although it is sufficient for many applications, the downsides of such an approach is that it produces a hybrid solution but not a hybrid model. This may seem like a minor detail, but creating a hybrid model has two advantages compared to computing a hybrid solution:

- The model can be used to obtain solutions, not only in the setting in which it was built, but also other settings (being cautious about extrapolation).
- The nature of the correction applied to the model can in certain conditions be interpreted in a physical sense (this point is emphasized in section 3.3).

The present section proposes a hybridization strategy applied to classical discretization methods then improved within the PINN rationale, which corrects a physical model thanks to data measured on a real system. It is first described, and then applied to an industrial case in the context of Hybrid Twins for power transformers.

3.2.1 Proposed methodology

Consider a system described by a state $u(x,t)$ governed by a partial differential equation in the form:

$$\begin{cases} \mathcal{L}(u)(x,t) = f(\boldsymbol{p}(x,t)) & \forall (x,t) \in \Omega \times [0,T] \\ u(x,t) = f_{\mathbf{d}}(x,t) & \forall (x,t) \in \Gamma_{\mathbf{d}} \times [0,T] \\ \partial_{\mathbf{n}} u(x,t) = f_{\mathbf{n}}(x,t) & \forall (x,t) \in \Gamma_{\mathbf{n}} \times [0,T] \\ u(x,0) = u_{\mathbf{0}}(x) & \forall x \in \Omega \end{cases} \tag{3.2.1}$$

This problem is assumed to have a unique solution u which perfectly fits within all the constraints, which could be computed methods such as finite elements, finite differences or PINN.

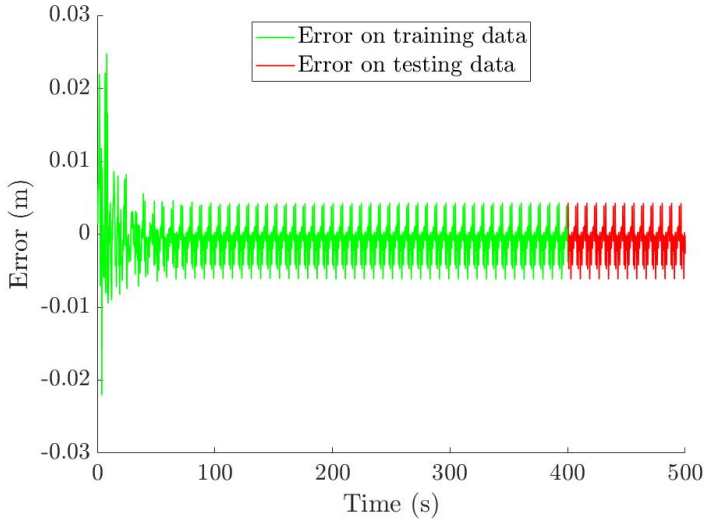


FIGURE 3.11: Error in the prediction of the observable position $\tilde{q}_3(t)$ in the nonlinear case, computed by a trained LSTM with $n = 3$ (same color code).

Once measurement data becomes available and a discrepancy between the simulation and reality is observed, the hybrid model comes into play.

Since the model defined by Eq. 3.2.1 is not able to represent reality, it must be wrong, or at least imprecise, and should therefore be modified to fit the measurements obtained on the real system.

Therefore a different strategy than the one described in section 1.2.2 is proposed, where the enrichment is applied directly to the model rather than to the solver. Eq. 3.2.1 is modified in the following fashion:

$$\begin{cases} \mathcal{L}(u)(x,t) = f(\boldsymbol{p}(x,t)) + g(u(x,t), \boldsymbol{p}(x,t)) & \forall (x,t) \in \Omega \times [0,T] \\ u(x,t) = f_{\mathbf{d}}(x,t) & \forall (x,t) \in \Gamma_{\mathbf{d}} \times [0,T] \\ \partial_{\mathbf{n}} u(x,t) = f_{\mathbf{n}}(x,t) & \forall (x,t) \in \Gamma_{\mathbf{n}} \times [0,T] \\ u(x,0) = u_{\mathbf{0}}(x) & \forall x \in \Omega \end{cases} \tag{3.2.2}$$

where g is the enrichment term which will be used to learn the data-based correction.

Hybrid model using a classical solver

If enough data is available and the problem is low-dimensional, g could be learned from the solution generated by a classical solver. From measured data sampled in various locations and at different times according to a mesh that is fine enough: $u_{\mathbf{m}}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}), i = 1, \dots, N_{\mathbf{m}}$, an estimation of the residual of Eq. 3.2.1 is computed by approximating $\mathcal{L}$ with a discretization method on the mesh related to the sampling:

$$r^i = \mathcal{L}(u)(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}) - f(\boldsymbol{p}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}})) \tag{3.2.3}$$

Now, a regression model $\hat{g}$ is created to link u and $\boldsymbol{p}$ to the residual:

$$\forall i = 1, \dots, N_{\mathbf{m}}, \quad \hat{g}(u(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}), \boldsymbol{p}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}})) = r^i \tag{3.2.4}$$

Since a lot of data is available, the regression method of choice is usually the neural network, which can be trained extensively without real risk of overfitting, unless the dimension of $\boldsymbol{p}$ is very high.

Once the model $\hat{g}$ is built, it can be introduced in the solver according to Eq. 3.2.2 to produce the enriched solution.

Although appealing, this route is rarely applicable in practice, except to tackle ODE problems, which are one-dimensional and do not require full-fied measurements in space.

Hybrid model strategy based on the PINN

In most cases, the data availability is quite reduced. In this case, it is impossible to approximate $\mathcal{L}(u_{\mathbf{m}})$. However, the PINN formulation allows to circumvent this issue.

Following the PINN rationale, the solution is approximated by a neural network $\hat{u}(x,t)$. An additional neural network $\hat{g} : \mathbb{R} \times \mathbb{R}^d \mapsto \mathbb{R}$ is defined to approximate function g. Both neural networks are trained simultaneously by minimizing the loss function $\mathtt{L}_{\mathcal{H}}$ defined in Eq. 3.3.18. However the first term $\mathtt{L}_{\mathbf{\Omega}}$ is adapted from Eq. 1.2.3 to include the change in the model and allow to train $\hat{g}$:

$$\mathtt{L}_{\mathbf{\Omega}} = \frac{1}{N_{\mathbf{\Omega}}} \sum_{i=1}^{N_{\mathbf{\Omega}}} \left| \mathcal{L}(\hat{u})(x^i_{\mathbf{\Omega}}, t^i_{\mathbf{\Omega}}) - f(\boldsymbol{p}(x^i_{\mathbf{\Omega}}, t^i_{\mathbf{\Omega}})) - \hat{g}(\hat{u}(x^i_{\mathbf{\Omega}}, t^i_{\mathbf{\Omega}}), \boldsymbol{p}(x^i_{\mathbf{\Omega}}, t^i_{\mathbf{\Omega}})) \right|^2 \tag{3.2.5}$$

When the training is complete, the hybrid solution u is obtained, but also function g and therefore a hybrid model. In certain cases, g could be analyzed by the modeler to understand the nature of the enrichment which was necessary for the model to match the data. It could also be used to compute the solution for a different loading or source by changing $\boldsymbol{p}$, or for different initial and boundary conditions.

3.2.2 Application to an industrial problem

Power transformers play a vital role in transmitting electrical energy and demand special focus. Monitoring and predicting the oil temperature of power transformers is crucial because it significantly impacts the expected lifespan of the insulation and therefore of the transformer. In particular, excessively high or low oil temperatures can cause irreversible harm to the transformer.

Since the usual lifespan of power transformers is between 20 and 60 years, even if a transformer initially has a well calibrated model, it is not expected to match reality forever. The hybrid modeling route is therefore very appealing, to enrich and correct the model whenever the measurements performed on the real system diverge from the model's predictions. This could have two major advantages:

- Quantifying the correction necessary to match the measured data is a tool for diagnosis, because the discrepancy between the model and reality is often due to aging or degradation.
- Having a model which matches reality enables prognosis and testing the impact of various scenarios of operation on oil temperature and therefore aging of the system.

Physics-based model

The IEC and IEEE dynamic thermal models [192, 193] pertain to the standardization of the physical modeling of oil temperature in transformers. Despite being aware of their limitations, such as the lack of information regarding cooling conditions and changes in material properties, an IEC/IEEE thermal model will serve as a benchmark physics-based model for the construction of the hybrid model and to compare the results. Interested readers can refer to [194] for a comprehensive overview of the physics-based thermal modeling process in a power transformer. The physics-based model utilized for this application is derived from the IEC and IEEE standards.

Based on the heat transfer equations, the proposed physics-based thermal model is composed of the following first-order system:

$$P(K^{load}, \Theta^W) = C_{th}(\Theta^{TO})\frac{d\Delta\Theta^{TO}}{dt} + \frac{\Delta\Theta^{TO}}{R_{th}(\Theta^{TO})} \tag{3.2.6}$$

$$\Theta^{TO} = T^{amb} + \Delta\Theta^{TO} \tag{3.2.7}$$

$$\Theta^W = \Theta^{TO} + \Delta\Theta^{OW} \tag{3.2.8}$$

where $\Delta\Theta^{TO}$ represents the predicted temperature rise of the top oil, Θ^W corresponds to the estimated average temperature of the windings, P denotes the total estimated losses generated by the transformer, T^{amb} represents the ambient temperature, $\Delta\Theta^{OW}$ represents the constant oil-to-winding temperature rise (provided by commissioning data), and C_{th} and R_{th} refer to the overall estimated thermal capacitance and resistance, respectively.

In order to enhance the accuracy of the model, it is important to consider that each parameter is temperature-dependent. The following observations can be made:

- Parameter P depends on the transformer load and winding temperature due to the variation in material resistivity as a function of temperature.
- The thermal capacitance C_{th} is influenced by the temperature of the top oil, as the heat capacity of the transformer's mineral oil changes with temperature.
- The thermal resistance R_{th} is updated because the viscosity of the oil changes with temperature.

The primary advantage of this physics-based model is its ability to obtain parameters solely based on commissioning data. The model only requires two inputs: the load factor K^{load} and the ambient temperature T^{amb}.

Figure 3.12 provides a representation of the given inputs and compares the measured and estimated top-oil temperature for one of RTE's transformers. For the selected 2000 data points, the mean absolute percentage error (MAPE) is 10.4%. As anticipated, given the limited knowledge of the transformer's parameters and the model's simplicity, there is a significant difference between the measured and estimated top-oil temperature, rendering it an inadequate estimation. Notably, when the top-oil temperature reaches 45°C (around the 1600th data point), a second group of fans is automatically activated, which is not accounted for in the proposed model. Although it is possible, as demonstrated in the example presented by [195], to enhance the model's accuracy by considering variable cooling conditions, it often necessitates heuristic or regression-based parameter tuning, which cannot be universally applied to every transformer.

In conclusion, despite its inaccuracies, the proposed physics-based model serves as the reference for this application of the hybrid modeling concept, as it captures a significant portion of the system's behavior and retains its physical significance.

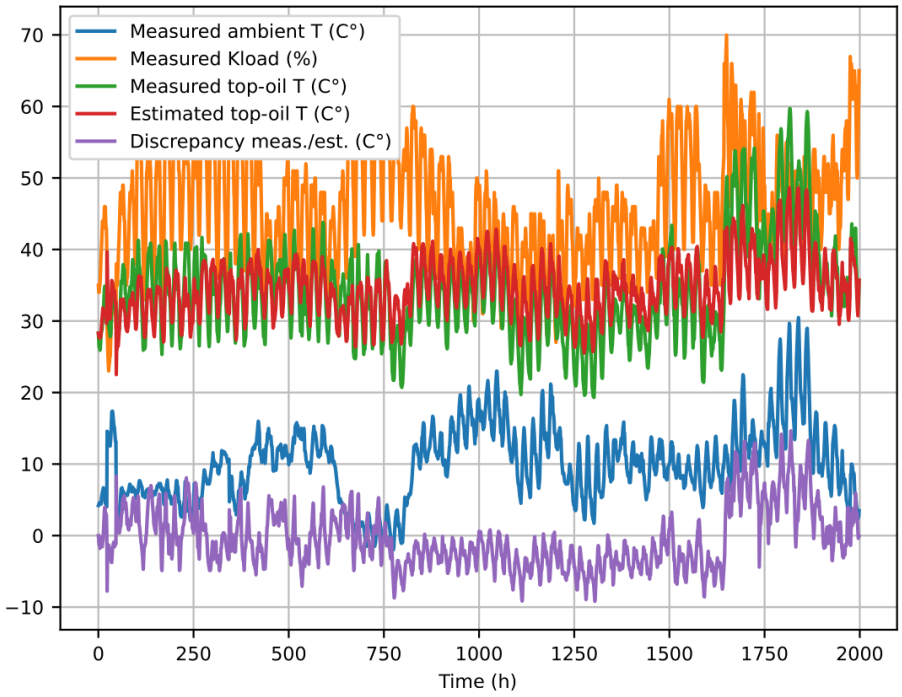


FIGURE 3.12: Physics-based top-oil temperature prediction

HYBRID MODEL

The strategy proposed in 3.2.1 is applied to the prediction of a power transformer's oil temperature using the model described in 3.2.2 and data collected on a transformer in operation.

The quantities measured are ambient temperature, load factor and top-oil temperature, at an hourly rate. The hourly load factor being a rapidly fluctuating variable, it is calculated as the average of the values sampled every 5 minutes. In this work, the presence of a time delay in measurements is not considered, as it is negligible compared to the one-hour sampling period used.

As suggested by the method, the source term in Eq. 3.2.6 is enriched with an unknown function g:

$$P^E = P(K^{load}, \Theta^W) + g\left(K^{load}, T^{amb}, \Theta^{TO}, \frac{d\Delta\Theta^{TO}}{dt}\right). \tag{3.2.9}$$

Eq. 3.2.6 therefore becomes :

$$P^E = C_{th}(\Theta^{TO})\frac{d\Delta\Theta^{TO}}{dt} + \frac{\Delta\Theta^{TO}}{R_{th}(\Theta^{TO})}. \tag{3.2.10}$$

A fully-connected neural network is used to model g and is trained by discretizing the equations with finite differences and then minimizing the difference between g and the residual of Eq. 3.2.6.

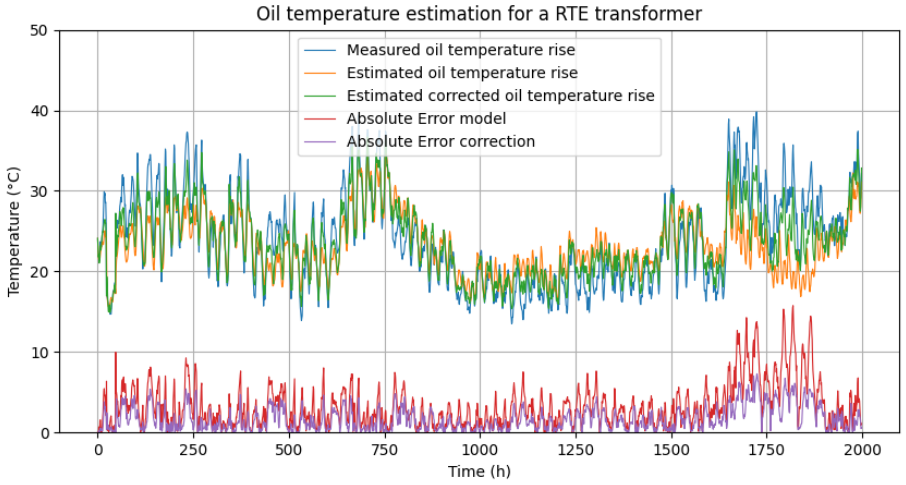


FIGURE 3.13: Hybrid-based top-oil estimation using the source term correction approach

In the case at hand, this approach allows to partially correct the model. Although the oil temperature predicted by the corrected model does not perfectly match reality, it performs better than the physical model. The mean absolute error is $3.2°C$ for the physics-based model and $1.7°C$ for the hybrid model. The remaining error suggests that the discrepancy between the model and reality can not be entirely described by the extra source term g included in the model. Another approach may have produced better results, but it is also possible that some of the real behavior is due to unknown outside factors such as wheather conditions.

3.3 Hybrid modeling by identifying properties

In the pursuit of refining physics-based models through hybrid modeling, an alternative approach emerges: one that assumes the model itself is accurate but acknowledges the uncertainty surrounding the physical properties associated with the system under study. This uncertainty may arise from modeling errors or approximations, the use of coarse representations, or the presence of damage or aging within the system. This section delves into the latter scenario, focusing on the specific case of localized damage. Works aiming at correcting models which suffered damage have been reviewed in [196].

When a system exhibits localized damage, accurately predicting its behavior becomes a complex task. However, by leveraging the framework of hybrid modeling, it becomes possible to correct the physical parameters associated with the system, enabling the development of a model capable of accurately capturing the behavior of the damaged system. Additionally, this approach offers the opportunity to not only correct the model but also to identify the nature and location of the damage itself.

The model correction methodology discussed in this section holds particular relevance in the context of Structural Health Monitoring (SHM) [80]. Extensive research efforts

have been dedicated to SHM, with a focus on investigating and addressing similar objectives. These studies often involve the analysis of the system in either the time domain [197–202] or the frequency domain [203–207] to detect and assess the presence of damage. Furthermore, the incorporation of machine learning techniques in SHM has gained considerable popularity, and notable applications of such techniques can be found in the literature [208].

This section introduces a novel methodology centered around the idea of parameter correction. By updating the physical parameters within the existing model, it becomes possible to refine its predictive capabilities, aligning it more closely with the observed behavior of the damaged system. The aim is to obtain a model that not only accounts for the presence of damage but also accurately predicts its impact on the system's overall behavior.

Moreover, this parameter correction methodology offers the means to conduct an in-depth analysis of the updated model. By comparing the updated model predictions with measured data, it becomes possible to identify the nature and location of the damage. This information is invaluable for understanding the underlying causes of the observed discrepancies and provides crucial insights for maintenance and repair strategies.

Key to the success of this approach is the utilization of sparse regularization techniques, which makes this methodology original. Sparse regularization allows for the selection of the most probable damage scenario from a range of possibilities that could explain the measurement data. By employing regularization techniques that favor sparse solutions, the methodology can effectively pinpoint the most likely damage locations and characterize the severity and extent of the localized damage within the system.

This approach is developed for two different types of numerical methods: the Finite Element Method in the context of linear equations and Physics-Informed Neural Networks.

3.3.1 Finite elements approach

The methodology is described on a linear elasticity problem. Let K be the original stiffness matrix representing the structural behavior. $\mathbf{K}$ is an $N \times N$ matrix, where N is the number of degrees of freedom in the structure. Let $\mathbf{U}$ be the nominal displacement vector obtained from the original model. $\mathbf{U}$ is an $N \times 1$ vector. Let $\mathbf{F}$ be the applied loading vector. $\mathbf{F}$ is an $N \times 1$ vector. The linear model describing the sytem reads

$$\mathbf{KU} = \mathbf{F}. \tag{3.3.1}$$

Eq. 3.3.1 is assumed to perfectly model the nominal system at the beginning of its lifecycle. However, after some time, a discrepancy is observed between the model's predictions and the strain or displacement measured on the system. This discrepancy is assumed to be due to a localized impact on the structure which deteriorates its mechanical properties. The objective is to find a corrected stiffness matrix $\mathbf{\Delta K}$ and a corrected displacement vector $\mathbf{\Delta U}$ that minimize the difference between the predicted and measured displacements. This can be achieved by solving the following system of equations:

$$(\mathbf{K} + \mathbf{\Delta K})(\mathbf{U} + \mathbf{\Delta U}) = \mathbf{F}. \tag{3.3.2}$$

Expanding the product and introducing Eq. 3.3.1 yields:

$$\mathbf{K}\Delta\mathbf{U} + \Delta\mathbf{K}\mathbf{U} + \Delta\mathbf{K}\Delta\mathbf{U} = \mathbf{0}. \tag{3.3.3}$$

To linearize the problem, the following approximation is considered:

$$\mathbf{K}\Delta\mathbf{U} + \Delta\mathbf{K}\mathbf{U} \approx \mathbf{0} \tag{3.3.4}$$

In order to maintain the physical mathematical properties and physical meaning of the stiffness matrix, its correction term ΔK is approximated as:

$$\Delta\mathbf{K} = \sum_{e=1}^{E} \alpha_e \mathbf{K}_e \tag{3.3.5}$$

where E is the number of elements involved in the mesh that covers the domain Ω, α_e is the correction coefficient for element e, and $\mathbf{K}_e$ is the nominal stiffness matrix related to element Ω_e expressed in the global nodal numbering. This can be interpreted as a element-wise constant correction of the system's Young Modulus.

Alongside Eq. 3.3.4, the correction should also satisfy that the corrected displacement matches the measured displacement in the sensor locations:

$$\widetilde{\mathbf{U}} + \widetilde{\Delta\mathbf{U}} = \mathbf{U}^m \tag{3.3.6}$$

where the tilde $\widetilde{\bullet}$ refers to the extraction of the part of the vector that correspond to the sensor locations and $\mathbf{U}^m$ contains the displacement measurements in the sensor locations.

Remark: when strain measurements are performed instead of displacement measurements, Eq. 3.3.6 is adapted using the discrete (linear) strain operator $\mathcal{E}$:

$$\widetilde{\mathcal{E}\mathbf{U}} + \widetilde{\mathcal{E}\Delta\mathbf{U}} = \boldsymbol{\varepsilon}^m \tag{3.3.7}$$

△

The system of equations thus obtained enforces both the physics and the measurements while the constraints on the value of the Young's modulus are removed. However, in most cases, this creates an under-determined system which therefore has an inifinite number of solutions. To circumvent this issue, the constraints of the Young's modulus can be introduced back in a relaxed form. Moreover, the assumption that the degradation of properties is localized motivates the use of the ℓ^1-norm which promotes sparsity in the solution.

The optimization problem can thus be formulated as follows:

$$\begin{aligned} \min_{\boldsymbol{\alpha},\Delta\mathbf{U}} \quad & \|\boldsymbol{\alpha}\|_1 \\ \text{s.t.} \quad & \mathbf{K}\Delta\mathbf{U} + \sum_{e=1}^{E} \alpha_e \mathbf{K}_e \mathbf{U} = \mathbf{0} \\ & \widetilde{\mathbf{U}} + \widetilde{\Delta\mathbf{U}} = \mathbf{U}^m \end{aligned} \tag{3.3.8}$$

This problem can be transformed in the following linear program:

$$\begin{aligned} \min_{\mathbf{z},\boldsymbol{\alpha},\Delta\mathbf{U}} \quad & \mathbf{1}^T\mathbf{z} \\ \text{s.t.} \quad & \boldsymbol{\alpha} \leq \mathbf{z} \\ & -\boldsymbol{\alpha} \leq \mathbf{z} \\ & \mathbf{K}\Delta\mathbf{U} + \sum_{e=1}^{E} \alpha_e \mathbf{K}_e \mathbf{U} = \mathbf{0} \\ & \widetilde{\mathbf{U}} + \widetilde{\Delta\mathbf{U}} = \mathbf{U}^m \end{aligned} \tag{3.3.9}$$

or its equivalent form:

$$\begin{aligned} \min_{\boldsymbol{\alpha}^+,\boldsymbol{\alpha}^-,\Delta\mathbf{U}} \quad & \mathbf{1}^T(\boldsymbol{\alpha}^+ + \boldsymbol{\alpha}^-) \\ \text{s.t.} \quad & \boldsymbol{\alpha}^+ \geq \mathbf{0} \\ & \boldsymbol{\alpha}^- \geq \mathbf{0} \\ & \mathbf{K}\Delta\mathbf{U} + \sum_{e=1}^{E} (\alpha_e^+ - \alpha_e^-) \mathbf{K}_e \mathbf{U} = \mathbf{0} \\ & \widetilde{\mathbf{U}} + \widetilde{\Delta\mathbf{U}} = \mathbf{U}^m \end{aligned} \tag{3.3.10}$$

where $\boldsymbol{\alpha} = \boldsymbol{\alpha}^+ - \boldsymbol{\alpha}^-$ similarly as in [209].

The solution of this problem is not always completely suitable because there are sometimes still too many non-zero elements in $\boldsymbol{\alpha}$. When this happens, a relevant solution is to partition the domain into a certain number of parts, and to remove the degrees of freedom in $\boldsymbol{\alpha}$ related to the parts in which $\boldsymbol{\alpha}$ is smaller than a certain threshold, before solving the problem again. This can be iterated a few times until convergence.

Numerical experiment

To test the method, a numerical experiment is carried out by considering a homogeneous plate depicted in Fig. 3.14 and its damaged counterpart described in Fig. 3.15, from which simulations are performed to generate the "measured" data.

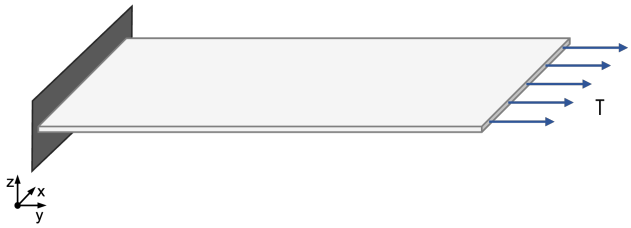


Figure 3.14: Structural model considered for illustrating the local model correction methodology.

The plate is discretized with a tetrahedral mesh, depicted in Figure 3.15, upon which the different mechanical fields (displacement, strain and stress) are computed using finite elements. The damaged elements are represented in red in Fig. 3.15; in these elements, the Young's modulus is reduced to 10% of its healthy state. For the sake of simplicity, the sensors are placed on nodes of the mesh, representd in blue in Fig. 3.15.

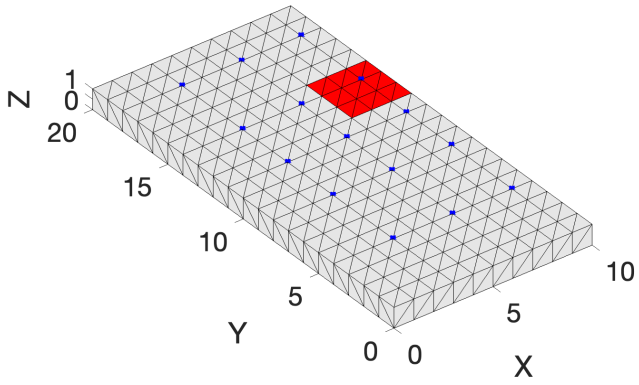
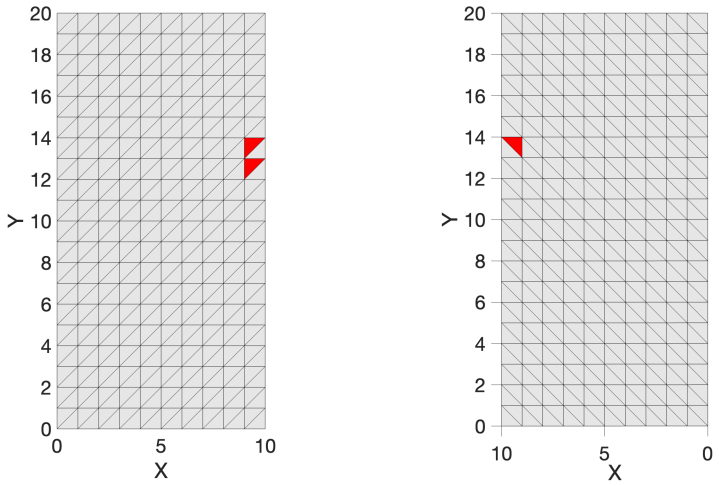


FIGURE 3.15: Damaged elements (red) and location of the 30 sensors (blue).

After solving the minimization problem (3.3.9), the elements in which $|\alpha_e| > 0.7 \max_e |\alpha_e|$ are identified and highlighted in red in Figure 3.16. It can be observed that the damage location is identified although its distribuion is not exactly recovered.



(A) (B)

FIGURE 3.16: Identified elements most affected by the damage (in red). (A) Top view; (B) bottom view.

The solution of the elasticity problem in then obtained with the corrected model and compared with the reference solution, as depicted in Figs. 3.17 and 3.18. The proposed procedure shows excellent accuracy to correct the model.

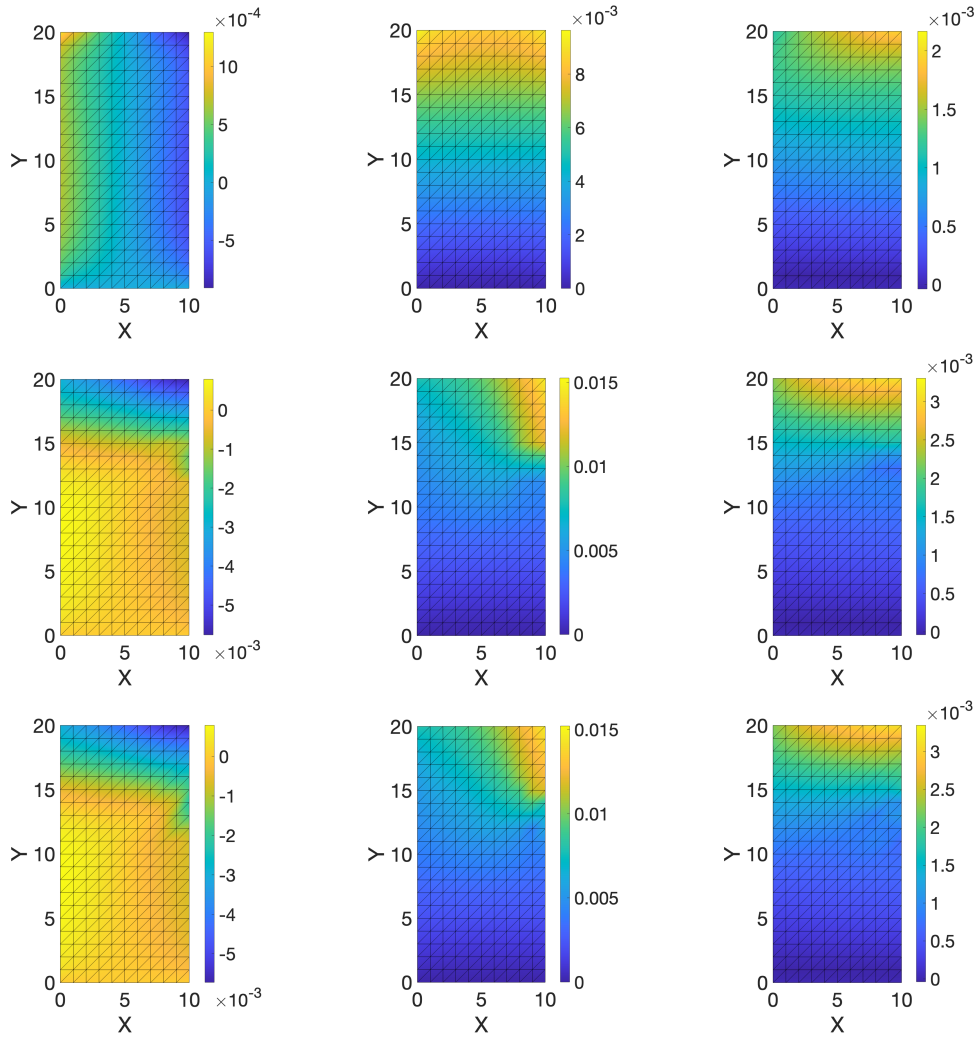


FIGURE 3.17: Top view of x (**left**), y (**center**) and z (**right**) components of the displacement field obtained with the nominal model (**top**); reference model that takes into account the real damaged region (**middle**) and corrected model (**bottom**).

The errors between the corrected and the reference (the nominal and the reference) displacement fields, for x, y and z, components are as follows:

- $\left\|U_x^{\text{Corr}} - U_x^{\text{Ref}}\right\|_2 = 2.2{\cdot}10^{-4}(3.3{\cdot}10^{-2})$, $err_{\%}^{x} = \frac{\left\|U_x^{\text{Corr}} - U_x^{\text{Ref}}\right\|_2}{\left\|U_x^{\text{Ref}}\right\|_2}{\cdot}100 = 6.4\%\ (98.2\%)$
- $\left\|U_y^{\text{Corr}} - U_y^{\text{Ref}}\right\|_2 = 3.8{\cdot}10^{-4}(3.3{\cdot}10^{-2})$, $err_{\%}^{y} = \frac{\left\|U_y^{\text{Corr}} - U_y^{\text{Ref}}\right\|_2}{\left\|U_y^{\text{Ref}}\right\|_2}{\cdot}100 = 2.9\%\ (26.1\%)$
- $\left\|U_z^{\text{Corr}} - U_z^{\text{Ref}}\right\|_2 = 9.3{\cdot}10^{-5}\ (9.6{\cdot}10^{-3})$, $err_{\%}^{z} = \frac{\left\|U_z^{\text{Corr}} - U_z^{\text{Ref}}\right\|_2}{\left\|U_z^{\text{Ref}}\right\|_2}{\cdot}100 = 3.1\%\ (32.6\%)$

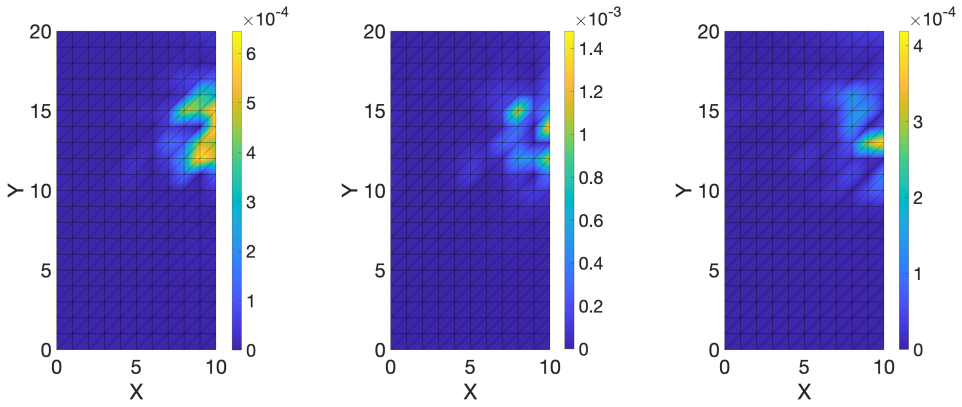


FIGURE 3.18: Top view of the difference in absolute value between the corrected and the reference displacement field, for x (**left**), y (**middle**) and z (**right**) components.

Since the quality of the technique is strongly correlated with the number and the positioning of the sensors, a comparative study shows the relative errors between the reference displacement and the one obtained with the corrected model for different sensor configurations, shown in Fig. 3.19. As expected, the relative error $\left\|U^{\text{Corr}}_{x,y,z} - U^{\text{Ref}}_{x,y,z}\right\|_2 / \left\|U^{\text{Ref}}_{x,y,z}\right\|_2$ decreases with an increase of the number of sensors, as Fig. 3.20 shows.

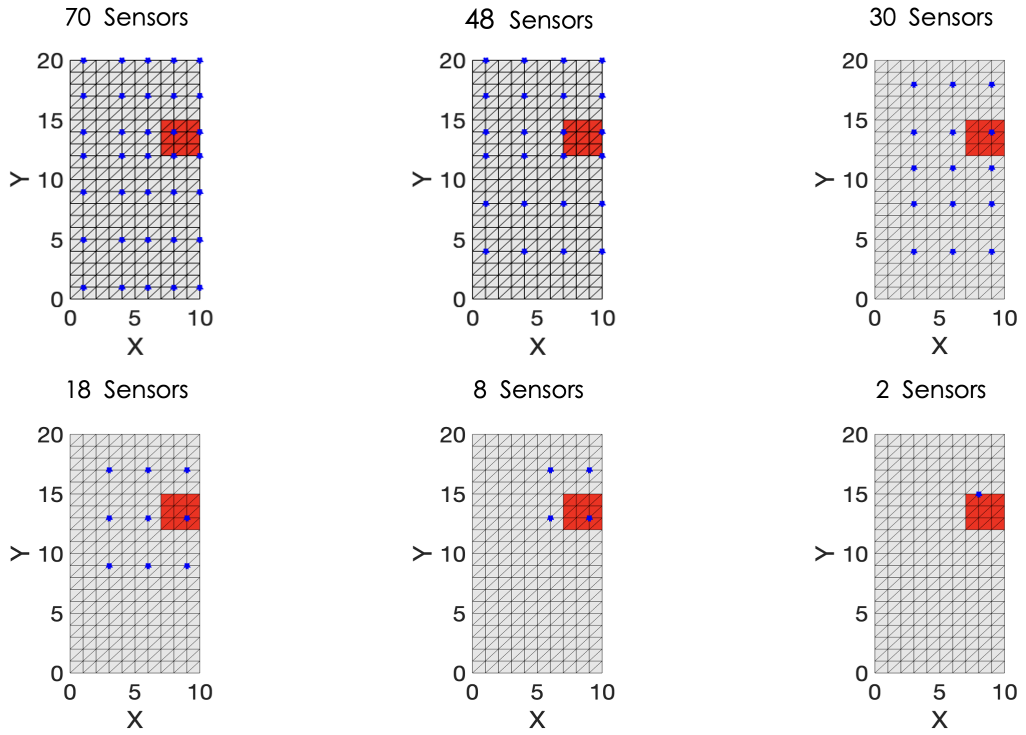


FIGURE 3.19: Top view of sensor locations for different configurations.

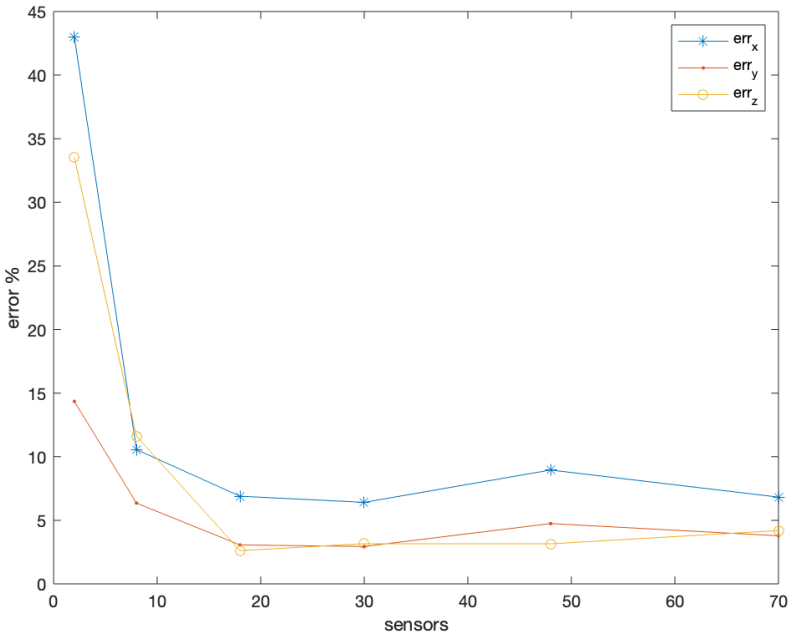


FIGURE 3.20: Relative error $\left\|U^{\text{Corr}}_{x,y,z} - U^{\text{Ref}}_{x,y,z}\right\|_2 / \left\|U^{\text{Ref}}_{x,y,z}\right\|_2$ with respect to number of sensors in the three directions (x in blue, y in red and z in yellow). The 6 configurations are the ones shown in Fig. 3.19.

3.3.2 PINN-BASED APPROACH

A similar approach can be formulated in the PINN setting. In this case, the problem is not discretized and directly uses the continuous equations of solid mechanics. The following problem is considered:

$$\begin{cases} \nabla \cdot \boldsymbol{\sigma} + \boldsymbol{b} = 0 & \boldsymbol{x} \in \Omega \\ \boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x}) & \boldsymbol{x} \in \Gamma_d \\ \boldsymbol{\sigma} \cdot \boldsymbol{n} = \boldsymbol{\psi}(\boldsymbol{x}) & \boldsymbol{x} \in \Gamma_n \\ \boldsymbol{\varepsilon} = \frac{1}{2}(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T) & \\ \boldsymbol{\sigma} = \mathsf{C} : \boldsymbol{\varepsilon} & \end{cases} \tag{3.3.11}$$

where $\mathbf{x}$ is the spatial coordinates, $\Omega \subset \mathbb{R}^2$ the computational domain and $\Gamma = \Gamma_d \cup \Gamma_n$ its boundary. $\boldsymbol{\sigma}$ is the Cauchy stress tensor, $\boldsymbol{\varepsilon}$ the infinitesimal strain tensor and C the fourth-order elasticity tensor. $\boldsymbol{b}$ is the force density, and $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ are respectively the Dirichlet and Neumann boudary conditions applied to the system. C is defined from the Young's modulus E and Poisson's ratio ν such that:

$$\begin{bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{bmatrix} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 1-\nu \end{bmatrix} \begin{bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \varepsilon_{12} \end{bmatrix} \tag{3.3.12}$$

The solution is approximated using two fully-connected neural networks to represent $\boldsymbol{u}$ and $\boldsymbol{\sigma}$, respectively denoted as $\hat{\boldsymbol{u}}$ and $\hat{\boldsymbol{\sigma}}$.

To solve Eq. 3.3.11 with the PINN, the loss function L_Φ is defined as:

$$\mathrm{L}_\Phi = \lambda_{\boldsymbol{\Omega}} \mathrm{L}_{\boldsymbol{\Omega}} + \lambda_{\mathbf{d}} \mathrm{L}_{\mathbf{d}} + \lambda_{\mathbf{n}} \mathrm{L}_{\mathbf{n}} + \lambda_{\mathbf{f}} \mathrm{L}_{\mathbf{f}} \tag{3.3.13}$$

where:

$$\mathtt{L}_{\boldsymbol{\Omega}} = \frac{1}{N_{\boldsymbol{\Omega}}} \sum_{i=1}^{N_{\boldsymbol{\Omega}}} \left\| \nabla \cdot \hat{\boldsymbol{\sigma}}(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) - \boldsymbol{b}(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) \right\|^2 \tag{3.3.14}$$

$$\mathtt{L}_{\mathbf{d}} = \frac{1}{N_{\mathbf{d}}} \sum_{i=1}^{N_{\mathbf{d}}} \left\| \hat{\boldsymbol{u}}(\boldsymbol{x}^i_{\mathbf{d}}) - \boldsymbol{\phi}(\boldsymbol{x}^i_{\mathbf{d}}) \right\|^2 \tag{3.3.15}$$

$$\mathtt{L}_{\mathbf{n}} = \frac{1}{N_{\mathbf{n}}} \sum_{i=1}^{N_{\mathbf{n}}} \left\| \hat{\boldsymbol{\sigma}}(\boldsymbol{x}^i_{\mathbf{n}}) \cdot \boldsymbol{n} - \boldsymbol{\psi}(\boldsymbol{x}^i_{\mathbf{n}}) \right\|^2 \tag{3.3.16}$$

$$\mathtt{L}_{\mathbf{f}} = \frac{1}{N_{\boldsymbol{\Omega}}} \sum_{i=1}^{N_{\boldsymbol{\Omega}}} \left\| \hat{\boldsymbol{\sigma}}(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) - \mathsf{C} : \hat{\boldsymbol{\varepsilon}}(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) \right\|^2 \tag{3.3.17}$$

and $\lambda_{\boldsymbol{\Omega}}, \lambda_{\mathbf{d}}, \lambda_{\mathbf{n}}, \lambda_{\mathbf{f}} \in \mathbb{R}^*_+$ are the weights associated to each term. $\hat{\boldsymbol{\varepsilon}}$ is obtained by applying the adequate derivatives with respect to the inputs of neural network $\hat{\boldsymbol{u}}$.

Displacement measurements $\boldsymbol{u}_{\mathbf{m}}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}), i = 1, \dots, N_{\mathbf{m}}$ (or strain measurements $\boldsymbol{\varepsilon}_{\mathbf{m}}(x^i_{\mathbf{m}}, t^i_{\mathbf{m}}), i = 1, \dots, N_{\mathbf{m}}$) are collected on the real structure. To make the model coincide with this data, the Young's modulus E is once again assumed to have been degraded in a localized area and the displacement (or strain) is modified to match the value of the measurements in the sensor locations.

This is performed by adding a third neural network $\hat{E}$ to represent the Young's modulus and replacing the loss function with the following hybrid one, in which two terms are added and one is modified:

$$\mathtt{L}_{\mathcal{H}} = \lambda_{\boldsymbol{\Omega}} \mathtt{L}_{\boldsymbol{\Omega}} + \lambda_{\mathbf{d}} \mathtt{L}_{\mathbf{d}} + \lambda_{\mathbf{n}} \mathtt{L}_{\mathbf{n}} + \lambda_{\mathbf{f}} \mathtt{L}_{\mathbf{f}} + \lambda_{\mathbf{m}} \mathtt{L}_{\mathbf{m}} + \lambda_{\mathbf{r}} \mathtt{L}_{\mathbf{r}} \tag{3.3.18}$$

where:

$$\mathtt{L}_{\mathbf{f}} = \frac{1}{N_{\boldsymbol{\Omega}}} \sum_{i=1}^{N_{\boldsymbol{\Omega}}} \left\| \hat{\boldsymbol{\sigma}}(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) - \hat{\mathsf{C}} : \hat{\boldsymbol{\varepsilon}}(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) \right\|^2 \tag{3.3.19}$$

$$\mathtt{L}_{\mathbf{m}} = \frac{1}{N_{\mathbf{m}}} \sum_{i=1}^{N_{\mathbf{m}}} \left\| \hat{\boldsymbol{u}}(\boldsymbol{x}^i_{\mathbf{m}}) - \boldsymbol{u}_{\mathbf{m}}(\boldsymbol{x}^i_{\mathbf{m}}) \right\|^2 \tag{3.3.20}$$

$$\mathtt{L}_{\mathbf{r}} = \frac{1}{N_{\boldsymbol{\Omega}}} \sum_{i=1}^{N_{\boldsymbol{\Omega}}} \left| \hat{E}(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) - E(\boldsymbol{x}^i_{\boldsymbol{\Omega}}) \right| \tag{3.3.21}$$

and $\lambda_{\mathbf{m}}, \lambda_{\mathbf{r}} \in \mathbb{R}^*_+$ are the weights associated to each term. $\hat{\mathsf{C}}$ is obtained from Eq. 3.3.12 applied to $\hat{E}$. The term $\mathtt{L}_{\mathbf{r}}$ corresponds to the regularization discussed in the section 3.3.1. In this case, it aims at applying a L^1-norm regularization to $\hat{E} - E$ in order to reduce its support, and therefore to have the nominal value of the Young's modulus in most of the domain.

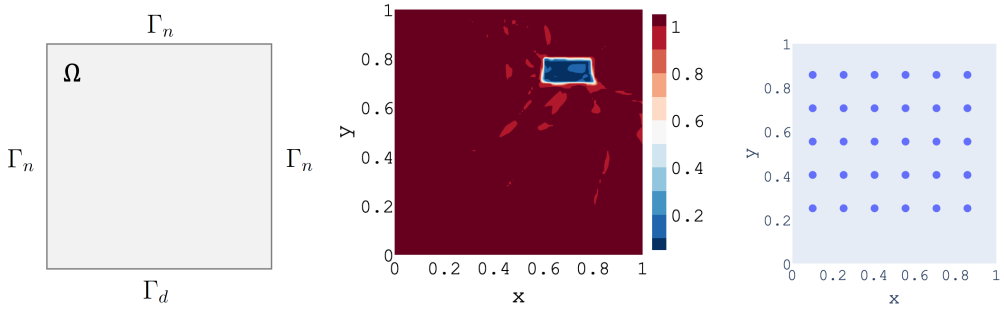
Remark: when the strain is measured and not the displacement, $\mathtt{L}_{\mathbf{m}}$ becomes:

$$\mathtt{L}_{\mathbf{m}} = \frac{1}{N_{\mathbf{m}}} \sum_{i=1}^{N_{\mathbf{m}}} \left\| \hat{\boldsymbol{\varepsilon}}(\boldsymbol{x}^i_{\mathbf{m}}) - \boldsymbol{\varepsilon}_{\mathbf{m}}(\boldsymbol{x}^i_{\mathbf{m}}) \right\|^2. \tag{3.3.22}$$

△

Numerical experiment

To test the method, a numerical experiment is carried out by considering a homogeneous plate with a Young's modulus of $1MPa$ depicted in Fig. 3.21a and its damaged counterpart described in Fig. 3.21b, from which direct simulations are performed to generate the "measured" data.



(A) Domain (B) Damaged Young's modulus E (C) Sensors location

Figure 3.21: Problem configuration

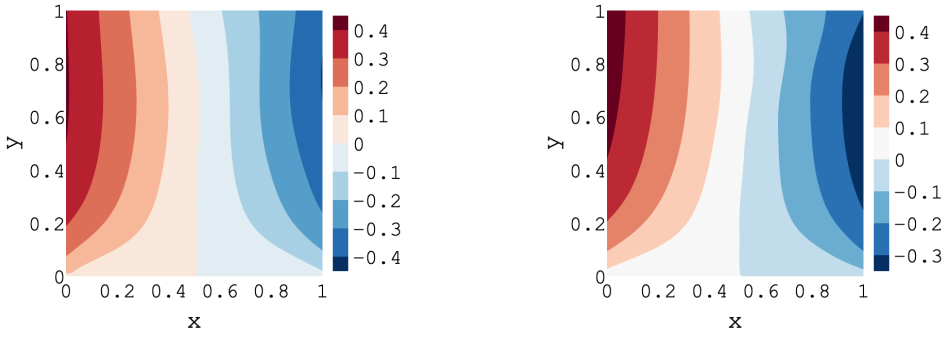


Figure 3.22: Comparison between the real (left) and computed (right) x-displacement

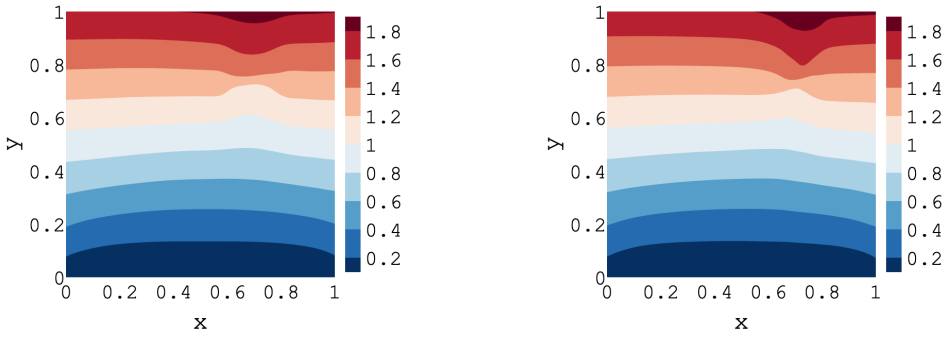


Figure 3.23: Comparison between the real (left) and computed (right) y-displacement

The effect of the number of measurement points in the inversion scheme is now investigated. Fig.3.25 shows the domain equipped with different measurement points, and the Young's modulus obtained with the neural network.

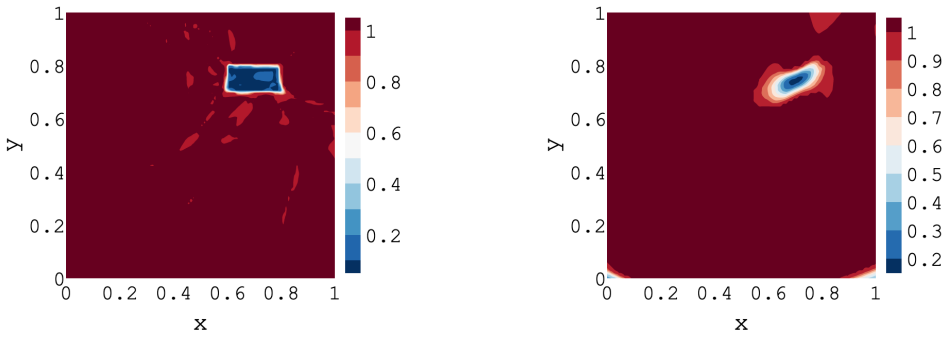


FIGURE 3.24: Comparison between the real (left) and computed (right) Young's modulus

This approach performs quite well, it reconstructs the full displacement field from the sparse sensor measurements, and it is able to locate the damage even when the number of sensors is extremely small.

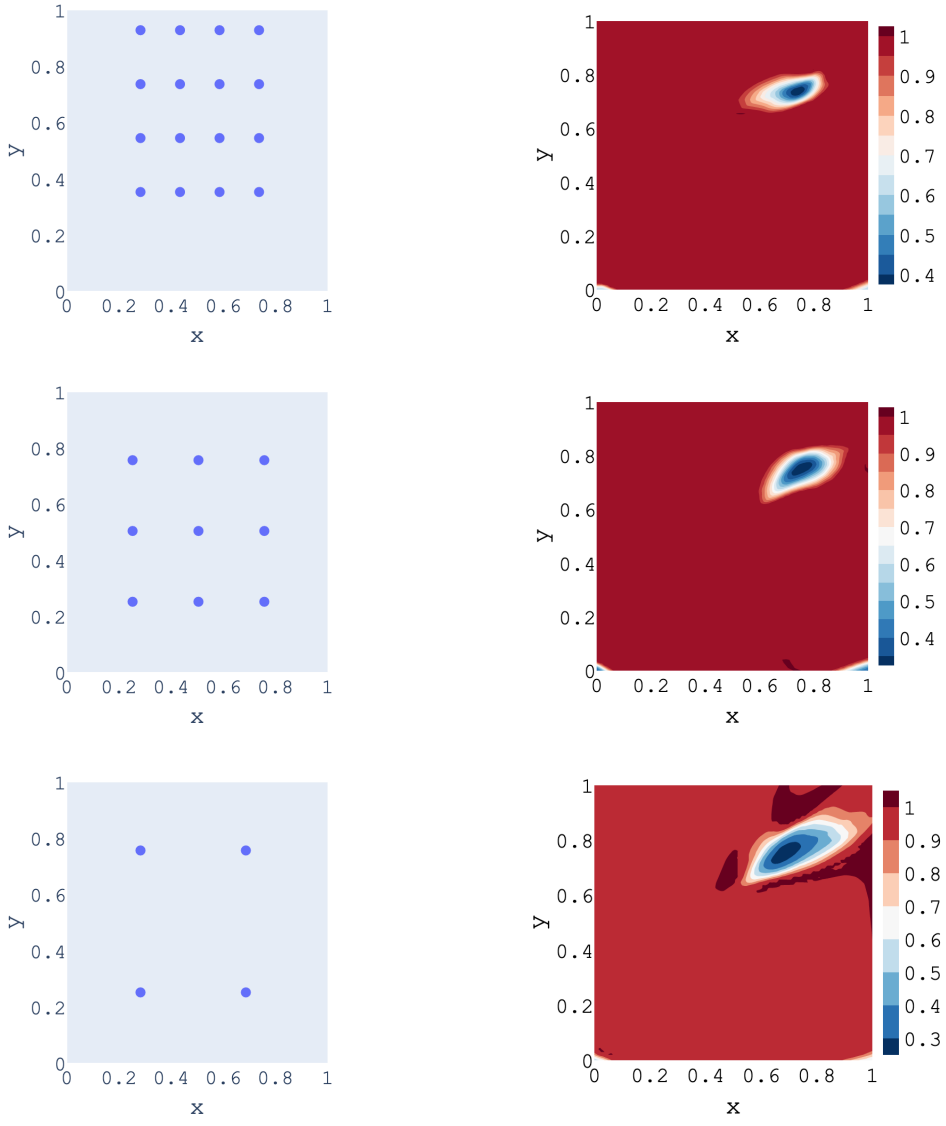


FIGURE 3.25: Sensors location (left) for different configurations, and computed Young's modulus (right)

Conclusion

This thesis has made contributions to the field of digital twins by introducing, developing, and improving methodologies that enable the construction of digital and hybrid twins. The work can be divided into several parts, each addressing specific aspects and challenges related to hybrid twinning.

- Non-intrusive model order reduction: This part of the work focused on improving the accuracy and efficiency of non-intrusive model order reduction (MOR) techniques for hybrid twins. The existing methods based on separated reperesentations were improved by using regularization techniques to increase the power of representation while limiting overfitting. ANOVA was used to study the effect of the different parameters separately in order to construct more robust approximation bases and thus improve the quality of the surrogate models. Future research on this topic could explore the construction of even better bases, that could be specifically design to be combined with the regularization technique proposed in s^2-PGD and rs-PGD. Such applications could make use of tools such as non-negative matrix factorization or techniques using sparsity promoting norms.

- Data representation: This part of the work focused on developing new methods for representing data in order to build better reduced models. The technologies based on curve alignment and optimal transport proved to be effective in dealing with a priori irreducible data. While the methodologies dealing with curves are simple and easy to apply to any kind of problem, they are not automatic, therefore some work could be performed to decrease the number of tasks required to be performed by the modeler. Meanwhile, Optimal Transport allows an automatic processing of the data but it is very complex and the techniques proposed to allow to extend its domain of applicability are even more intricate. Studies should be performed to understand how to apply Optimal Transport based methodologies to general data.

- Partial observability: This part of the thesis involved a theoretical study on creating models from partial observability. This investigation explored the creation of machine learning models for dynamical systems in scenarios where the system cannot be completely observed. While a simple case study was presented, future research could expand upon this work to explore more complex scenarios and test the limitations of modeling under partial observability.

- Hybrid models: This part of the thesis focused on constructing hybrid models from physics-based models and measurement data. The developed strategies aimed at creating hybrid models rather than hybrid solutions, emphasizing explicability and generalization of the model. Two methodologies were proposed

to enrich models based on partial differential equations. One approach introduced an unknown as a source term in the problem, while the other aimed to identify properties of the physical system. Both methods successfully improved the model's fit to the data. Furthermore, the second methodology demonstrated the utility of model explicability in Structural Health Monitoring and damage detection. These techniques were formulated using classical discretization methods as well as Physics-Informed Neural Networks. Leveraging the flexibility of PINNs could yield original and powerful extensions to the techniques presented in the thesis.

In conclusion, this thesis has laid the foundation for advancing the field of hybrid twinning by introducing novel methodologies and exploring various aspects of model construction, representation, data-driven considerations, and hybridization. The contributions made in this work provide a valuable framework for further research and applications in the development of hybrid twins for addressing industrial challenges.

Bibliography

[1] W. H. Schilders, H. A. Van der Vorst, and J. Rommes, *Model order reduction: theory, research aspects and applications*, vol. 13, Springer, 2008.

[2] F. Chinesta, A. Huerta, G. Rozza, and K. Willcox, *Model order reduction*, Encyclopedia of computational mechanics, (2016).

[3] T. Lassila, A. Manzoni, A. Quarteroni, and G. Rozza, *Model order reduction in fluid dynamics: challenges and perspectives*, Reduced Order Methods for modeling and computational reduction, (2014), pp. 235–273.

[4] R. A. Fisher, *Design of experiments*, British Medical Journal, 1 (1936), p. 554.

[5] M. Cavazzuti, *Design of experiments*, Optimization methods: from theory to design scientific and technological aspects in mechanics, (2013), pp. 13–42.

[6] A. Vabalas, E. Gowen, E. Poliakoff, and A. J. Casson, *Machine learning algorithm validation with a limited sample size*, PloS one, 14 (2019), p. e0224365.

[7] M. Evchenko, J. Vanschoren, H. H. Hoos, M. Schoenauer, and M. Sebag, *Frugal machine learning*, arXiv preprint arXiv:2111.03731, (2021).

[8] D. M. Hawkins, *The problem of overfitting*, Journal of chemical information and computer sciences, 44 (2004), pp. 1–12.

[9] P. Barbiero, G. Squillero, and A. Tonda, *Modeling generalization in machine learning: A methodological and computational study*, arXiv preprint arXiv:2006.15680, (2020).

[10] C. Rudin, C. Chen, Z. Chen, H. Huang, L. Semenova, and C. Zhong, *Interpretable machine learning: Fundamental principles and 10 grand challenges*, Statistic Surveys, 16 (2022), pp. 1–85.

[11] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, *Explainable ai: A review of machine learning interpretability methods*, Entropy, 23 (2020), p. 18.

[12] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, *Active learning with statistical models*, Journal of artificial intelligence research, 4 (1996), pp. 129–145.

[13] M. Raissi, P. Perdikaris, and G. E. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations*, Journal of Computational physics, 378 (2019), pp. 686–707.

[14] J. W. Tukey, *Comparing individual means in the analysis of variance*, Biometrics, (1949), pp. 99–114.

[15] H. Scheffe, *The analysis of variance*, vol. 72, John Wiley & Sons, 1999.

[16] C. Villani et al., *Optimal transport: old and new*, vol. 338, Springer, 2009.

[17] F. Santambrogio, *Optimal transport for applied mathematicians*, Birkäuser, NY, 55 (2015), p. 94.

[18] D. Goldsman, R. E. Nance, and J. R. Wilson, *A brief history of simulation revisited*, in Proceedings of the 2010 Winter Simulation Conference, IEEE, 2010, pp. 567–574.

[19] M. A. Bossak, *Simulation based design*, Journal of Materials Processing Technology, 76 (1998), pp. 8–11.

[20] J. Peiró and S. Sherwin, *Finite difference, finite element and finite volume methods for partial differential equations*, Handbook of Materials Modeling: Methods, (2005), pp. 2415–2446.

[21] C. Mattiussi, *An analysis of finite volume, finite element, and finite difference methods using some concepts from algebraic topology*, Journal of Computational Physics, 133 (1997), pp. 289–309.

[22] O. C. Zienkiewicz and P. Morice, *The finite element method in engineering science*, vol. 1977, McGraw-hill London, 1971.

[23] K.-J. Bathe, *Finite element method*, Wiley encyclopedia of computer science and engineering, (2007), pp. 1–12.

[24] K. Jordán, *Calculus of finite differences*, vol. 33, American Mathematical Soc., 1965.

[25] N. EH, *The calculus of finite differences*, Nature, 134 (1934), pp. 231–233.

[26] H. K. Versteeg and W. Malalasekera, *An introduction to computational fluid dynamics: the finite volume method*, Pearson education, 2007.

[27] R. Eymard, T. Gallouët, and R. Herbin, *Finite volume methods*, Handbook of numerical analysis, 7 (2000), pp. 713–1018.

[28] G. Constable and B. Somerville, *A century of innovation: Twenty engineering achievements that transformed our lives*, Joseph Henry Press, 2003.

[29] A. M. Madni, C. C. Madni, and S. D. Lucero, *Leveraging digital twin technology in model-based systems engineering*, Systems, 7 (2019), p. 7.

[30] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, *Physics-informed machine learning*, Nature Reviews Physics, 3 (2021), pp. 422–440.

[31] Z. Liu, Y. Chen, Y. Du, and M. Tegmark, *Physics-augmented learning: A new paradigm beyond physics-informed learning*, arXiv preprint arXiv:2109.13901, (2021).

[32] X. Wan, *The effect of regularization coefficient on polynomial regression*, in Journal of Physics: Conference Series, vol. 1213, IOP Publishing, 2019, p. 042054.

[33] A. N. Gorban and D. C. Wunsch, *The general approximation theorem*, in 1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98CH36227), vol. 2, IEEE, 1998, pp. 1271–1274.

[34] M. Tschannen, O. Bachem, and M. Lucic, *Recent advances in autoencoder-based representation learning*, arXiv preprint arXiv:1812.05069, (2018).

[35] M. Ringnér, *What is principal component analysis?*, Nature biotechnology, 26 (2008), pp. 303–304.

[36] B. Schölkopf, A. Smola, and K.-R. Müller, *Kernel principal component analysis*, in Artificial Neural Networks—ICANN'97: 7th International Conference Lausanne, Switzerland, October 8–10, 1997 Proceeedings, Springer, 2005, pp. 583–588.

[37] S. T. Roweis and L. K. Saul, *Nonlinear dimensionality reduction by locally linear embedding*, science, 290 (2000), pp. 2323–2326.

[38] G. E. Hinton and S. Roweis, *Stochastic neighbor embedding*, Advances in neural information processing systems, 15 (2002).

[39] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, *Generative adversarial networks: An overview*, IEEE signal processing magazine, 35 (2018), pp. 53–65.

[40] N. Aloysius and M. Geetha, *A review on deep convolutional neural networks*, in 2017 international conference on communication and signal processing (ICCSP), IEEE, 2017, pp. 0588–0592.

[41] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, *Graph neural networks: A review of methods and applications*, AI open, 1 (2020), pp. 57–81.

[42] L. R. Medsker and L. Jain, *Recurrent neural networks*, Design and Applications, 5 (2001), pp. 64–67.

[43] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural computation, 9 (1997), pp. 1735–1780.

[44] A. Elhassouny and F. Smarandache, *Trends in deep convolutional neural networks architectures: A review*, in 2019 International conference of computer science and renewable energies (ICCSRE), IEEE, 2019, pp. 1–8.

[45] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, *Neural ordinary differential equations*, Advances in neural information processing systems, 31 (2018).

[46] J. H. Tu, *Dynamic mode decomposition: Theory and applications*, PhD thesis, Princeton University, 2013.

[47] A. Mauroy, Y. Susuki, and I. Mezić, *Koopman operator in systems and control*, Springer, 2020.

[48] K. Nakajima and I. Fischer, *Reservoir computing*, Springer, 2021.

[49] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, *Learning nonlinear operators via deeponet based on the universal approximation theorem of operators*, Nature machine intelligence, 3 (2021), pp. 218–229.

[50] Q. Hernandez, A. Badias, D. Gonzalez, F. Chinesta, and E. Cueto, *Deep learning of thermodynamics-aware reduced-order models from data*, Computer Methods in Applied Mechanics and Engineering, 379 (2021), p. 113763.

[51] J. Glassey and M. Von Stosch, *Hybrid modeling in process industries*, CRC Press, 2018.

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

[53] G. Berkooz, P. Holmes, and J. L. Lumley, *The proper orthogonal decomposition in the analysis of turbulent flows*, Annual Review of Fluid Mechanics, 25 (1993), pp. 539–575.

[54] F. Chinesta, P. Ladeveze, and E. Cueto, *A short review on model order reduction based on proper generalized decomposition*, Archives of Computational Methods in Engineering, 18 (2011), p. 395.

[55] 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, 199 (2010), pp. 1287–1296. Multiscale Models and Mathematical Aspects in Solid and Fluid Mechanics.

[56] M. Tezzele, N. Demo, and G. Rozza, *Shape optimization through proper orthogonal decomposition with interpolation and dynamic mode decomposition enhanced by active subspaces*, arXiv preprint arXiv:1905.05483, (2019).

[57] R. R. Rama, *Proper orthogonal decomposition with interpolation-based real-time modelling of the heart*, PhD thesis, University of Cape Town, 2017.

[58] R. Ibáñez, E. Abisset-Chavanne, A. Ammar, D. González, E. Cueto, A. Huerta, J. L. Duval, F. Chinesta, et al., *A multidimensional data-driven sparse identification technique: the sparse proper generalized decomposition*, Complexity, 2018 (2018).

[59] M. Köppen, *The curse of dimensionality*, in 5th online world conference on soft computing in industrial applications (WSC5), vol. 1, 2000, pp. 4–8.

[60] M. Verleysen and D. François, *The curse of dimensionality in data mining and time series prediction*, in Computational Intelligence and Bioinspired Systems: 8th International Work-Conference on Artificial Neural Networks, IWANN 2005, Vilanova i la Geltrú, Barcelona, Spain, June 8-10, 2005. Proceedings 8, Springer, 2005, pp. 758–770.

[61] S.-M. Udrescu, A. Tan, J. Feng, O. Neto, T. Wu, and M. Tegmark, *Ai feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity*, Advances in Neural Information Processing Systems, 33 (2020), pp. 4860–4871.

[62] S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Discovering governing equations from data by sparse identification of nonlinear dynamical systems*, Proceedings of the national academy of sciences, 113 (2016), pp. 3932–3937.

[63] R. Ibañez, E. Abisset-Chavanne, E. Cueto, A. Ammar, J. L. Duval, and F. Chinesta, *Some applications of compressed sensing in computational mechanics: model order reduction, manifold learning, data-driven applications and nonlinear dimensionality reduction*, Computational Mechanics, 64 (2019), pp. 1259–1271.

[64] A. Takemura, *Tensor analysis of anova decomposition*, Journal of the American Statistical Association, 78 (1983), pp. 894–900.

[65] M. Stitson, A. Gammerman, V. Vapnik, V. Vovk, C. Watkins, and J. Weston, *Support vector regression with anova decomposition kernels*, Advances in kernel methods—Support vector learning, (1999), pp. 285–292.

[66] X. Yang, M. Choi, G. Lin, and G. E. Karniadakis, *Adaptive anova decomposition of stochastic incompressible and compressible flows*, Journal of Computational Physics, 231 (2012), pp. 1587–1614.

[67] K. Tang, P. M. Congedo, and R. Abgrall, *Sensitivity analysis using anchored anova expansion and high-order moments computation*, International Journal for Numerical Methods in Engineering, 102 (2015), pp. 1554–1584.

[68] Z. Zhang, M. Choi, and G. E. Karniadakis, *Anchor points matter in anova decomposition*, Spectral and High Order Methods for Partial Differential Equations, 76 (2011), pp. 347–355.

[69] Z. Gao and J. S. Hesthaven, *On anova expansions and strategies for choosing the anchor point*, Applied Mathematics and Computation, 217 (2010), pp. 3274–3285.

[70] F. Nielsen and F. Nielsen, *Hierarchical clustering*, Introduction to HPC with MPI for Data Science, (2016), pp. 195–211.

[71] F. Murtagh and P. Contreras, *Algorithms for hierarchical clustering: an overview*, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2 (2012), pp. 86–97.

[72] S. B. Kotsiantis, I. Zaharakis, P. Pintelas, et al., *Supervised machine learning: A review of classification techniques*, Emerging artificial intelligence applications in computer engineering, 160 (2007), pp. 3–24.

[73] S. H. Lee and W. Chen, *A comparative study of uncertainty propagation methods for black-box-type problems*, Structural and multidisciplinary optimization, 37 (2009), pp. 239–253.

[74] I. Ayed, E. de Bézenac, A. Pajot, J. Brajard, and P. Gallinari, *Learning dynamical systems from partial observations*, arXiv preprint arXiv:1902.11136, (2019).

[75] Y. Li, J. Wu, J.-Y. Zhu, J. B. Tenenbaum, A. Torralba, and R. Tedrake, *Propagation networks for model-based control under partial observation*, in 2019 International Conference on Robotics and Automation (ICRA), IEEE, 2019, pp. 1205–1211.

[76] D. González, F. Chinesta, and E. Cueto, *Learning non-markovian physics from data*, Journal of Computational Physics, 428 (2021), p. 109982.

[77] B. Moya, A. Badias, D. Gonzalez, F. Chinesta, and E. Cueto, *Physics-informed reinforcement learning for perception and reasoning about fluids*, arXiv preprint arXiv:2203.05775, (2022).

[78] E. L. Wilson, *The static condensation algorithm*, International Journal for Numerical Methods in Engineering, 8 (1974), pp. 198–203.

[79] S.-H. Chen and H. Pan, *Guyan reduction*, Communications in applied numerical methods, 4 (1988), pp. 549–556.

[80] D. Balageas, C.-P. Fritzen, and A. Güemes, *Structural health monitoring*, vol. 90, John Wiley & Sons, 2010.

[81] W.-L. Loh, *On latin hypercube sampling*, The annals of statistics, 24 (1996), pp. 2058–2080.

[82] A. L. Herman and B. A. Conway, *Direct optimization using collocation based on high-order gauss-lobatto quadrature rules*, Journal of Guidance, Control, and Dynamics, 19 (1996), pp. 592–599.

[83] S. A. Smolyak, *Quadrature and interpolation formulas for tensor products of certain classes of functions*, in Doklady Akademii Nauk, vol. 148, Russian Academy of Sciences, 1963, pp. 1042–1045.

[84] C. Williams and C. Rasmussen, *Gaussian processes for regression*, Advances in neural information processing systems, 8 (1995).

[85] R. A. Fisher, *The statistical utilization of multiple measurements*, Annals of eugenics, 8 (1938), pp. 376–386.

[86] Q. Hernández, A. Badías, D. González, F. Chinesta, and E. Cueto, *Structure-preserving neural networks*, Journal of Computational Physics, 426 (2021), p. 109950.

[87] V. Nasteski, *An overview of the supervised machine learning methods*, Horizons. b, 4 (2017), pp. 51–62.

[88] M. E. Celebi and K. Aydin, *Unsupervised learning algorithms*, vol. 9, Springer, 2016.

[89] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, and J. Tang, *Self-supervised learning: Generative or contrastive*, IEEE Transactions on Knowledge and Data Engineering, 35 (2021), pp. 857–876.

[90] Y. Reddy, P. Viswanath, and B. E. Reddy, *Semi-supervised learning: A brief review*, Int. J. Eng. Technol, 7 (2018), p. 81.

[91] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, *A comprehensive survey on transfer learning*, Proceedings of the IEEE, 109 (2020), pp. 43–76.

[92] Y. Li, *Deep reinforcement learning: An overview*, arXiv preprint arXiv:1701.07274, (2017).

[93] A. Sancarlos, V. Champaney, E. Cueto, and F. Chinesta, *Regularized regressions for parametric models based on separated representations*, Advanced Modeling and Simulation in Engineering Sciences, 10 (2023), p. 4.

[94] C. Ghnatios, V. Champaney, A. Pasquale, and F. Chinesta, *A regularized real-time integrator for data-driven control of heating channels*, Computation, 10 (2022), p. 176.

[95] C. Ghnatios, D. di Lorenzo, V. Champaney, E. Cueto, and F. Chinesta, *Optimal velocity planning based on the solution of the euler-lagrange equations with a neural network based velocity regression*, Discrete and Continuous Dynamical Systems-S, (2023), pp. 0–0.

[96] V. Champaney, A. Pasquale, A. Ammar, and F. Chinesta, *Parametric curves metamodelling based on data clustering, data alignment, pod-based modes extraction and pgd-based nonlinear regressions*, Frontiers in Materials, 9 (2022), p. 363.

[97] S. Torregrosa, V. Champaney, A. Ammar, V. Herbert, and F. Chinesta, *Surrogate parametric metamodel based on optimal transport*, Mathematics and Computers in Simulation, 194 (2022), pp. 36–63.

[98] K. Derouiche, S. Garois, V. Champaney, M. Daoud, K. Traidi, and F. Chinesta, *Data-driven modeling for multiphysics parametrized problems-application to induction hardening process*, Metals, 11 (2021), p. 738.

[99] M. El Fallaki Idrissi, F. Praud, V. Champaney, F. Chinesta, and F. Meraghni, *Multiparametric modeling of composite materials based on non-intrusive pgd informed by multiscale analyses: Application for real-time stiffness prediction of woven composites*, Composite Structures, 302 (2022), p. 116228.

[100] A. Pasquale, V. Champaney, Y. Kim, N. Hascoët, A. Ammar, and F. Chinesta, *A parametric metamodel of the vehicle frontal structure accounting for material properties and strain-rate effect: application to full frontal rigid barrier crash test*, Heliyon, 8 (2022), p. e12397.

[101] S. Vermiglio, V. Champaney, A. Sancarlos, F. Daim, J. C. Kedzia, J. L. Duval, P. Diez, and F. Chinesta, *Parametric electromagnetic analysis of radar-based advanced driver assistant systems*, Sensors, 20 (2020), p. 5686.

[102] M. Jacot, V. Champaney, F. Chinesta, and J. Cortial, *Parametric damage mechanics empowering structural health monitoring of 3d woven composites*, Sensors, 23 (2023), p. 1946.

[103] T. Loreau, V. Champaney, N. Hascoët, P. Mourgue, J.-L. Duval, and F. Chinesta, *Learning the parametric transfer function of unitary operations for real-time evaluation of manufacturing processes involving operations sequencing*, Applied Sciences, 11 (2021), p. 5146.

[104] A. Reille, V. Champaney, F. Daim, Y. Tourbier, N. Hascoet, D. Gonzalez, E. Cueto, J. L. Duval, and F. Chinesta, *Learning data-driven reduced elastic and inelastic models of spot-welded patches*, Mechanics & Industry, 22 (2021), p. 32.

[105] T. Loreau, V. Champaney, N. Hascoet, J. Lambarri, M. Madarieta, I. Garmendia, and F. Chinesta, *Parametric analysis and machine learning-based parametric modeling of wire laser metal deposition induced porosity*, International Journal of Material Forming, 15 (2022), p. 33.

[106] A. Runacher, M.-J. Kazemzadeh-Parsi, D. Di Lorenzo, V. Champaney, N. Hascoet, A. Ammar, and F. Chinesta, *Describing and modeling rough composites surfaces by using topological data analysis and fractional brownian motion*, Polymers, 15 (2023), p. 1449.

[107] V. Champaney, V. J. Amores, S. Garois, L. Irastorza-Valera, C. Ghnatios, F. J. Montáns, E. Cueto, and F. Chinesta, *Modeling systems from partial observations*, Frontiers in Materials, 9 (2022).

[108] X. Kestelyn, G. Denis, V. Champaney, N. Hascoet, C. Ghnatios, and F. Chinesta, *Towards a hybrid twin for infrastructure asset management: Investigation on power transformer asset maintenance management*, in 2022 7th International Advanced Research Workshop on Transformers (ARWtr), IEEE, 2022, pp. 109–114.

[109] S. Torregrosa, V. Champaney, A. Ammar, V. Herbert, and F. Chinesta, *Hybrid twins based on optimal transport*, Computers & Mathematics with Applications, 127 (2022), pp. 12–24.

[110] D. Di Lorenzo, V. Champaney, C. Germoso, E. Cueto, and F. Chinesta, *Data completion, model correction and enrichment based on sparse identification and data assimilation*, Applied Sciences, 12 (2022), p. 7458.

[111] D. Di Lorenzo, V. Champaney, J. Marzin, C. Farhat, and F. Chinesta, *Physics informed and data-based augmented learning in structural health diagnosis*, Computer Methods in Applied Mechanics and Engineering, 414 (2023), p. 116186.

[112] V. Champaney, A. Sancarlos, F. Chinesta, E. Cueto, D. González, I. Alfaro, S. Guevelou, J. L. Duval, A. Chambard, and P. Mourgue, *Hybrid twins-a highway towards a performance-based engineering. part i: Advanced model order reduction enabling real-time physics*, in ESAFORM 2021, Prof. Anne Marie Habraken, 2021.

[113] V. Champaney, F. Chinesta, and E. Cueto, *Engineering empowered by physics-based and data-driven hybrid models: A methodological overview*, International Journal of Material Forming, 15 (2022), p. 31.

[114] D. Hilberg, W. Lazik, and H. Fiedler, *The application of classical pod and snapshot pod in a turbulent shear layer with periodic structures*, Applied scientific research, 53 (1994), pp. 283–290.

[115] S. Chaturantabut and D. C. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM Journal on Scientific Computing, 32 (2010), pp. 2737–2764.

[116] B. Peherstorfer, D. Butnaru, K. Willcox, and H.-J. Bungartz, *Localized discrete empirical interpolation method*, SIAM Journal on Scientific Computing, 36 (2014), pp. A168–A192.

[117] D. Ryckelynck, *Hyper-reduction of mechanical models involving internal variables*, International Journal for Numerical Methods in Engineering, 77 (2009), pp. 75–89.

[118] C. Farhat, T. Chapman, and P. Avery, *Structure-preserving, stability, and accuracy properties of the energy-conserving sampling and weighting method for the hyper reduction of nonlinear finite element dynamic models*, International journal for numerical methods in engineering, 102 (2015), pp. 1077–1110.

[119] H. V. Ly and H. T. Tran, *Modeling and control of physical processes using proper orthogonal decomposition*, Mathematical and computer modelling, 33 (2001), pp. 223–236.

[120] M. Hamdaoui, G. Le Quilliec, P. Breitkopf, and P. Villon, *Pod surrogates for real-time multi-parametric sheet metal forming problems*, International journal of material forming, 7 (2014), pp. 337–358.

[121] M.-N. Nguyen and H.-G. Kim, *An efficient podi method for real-time simulation of indenter contact problems using rbf interpolation and contact domain decomposition*, Computer Methods in Applied Mechanics and Engineering, 388 (2022), p. 114215.

[122] A. Kovárnová, P. Krah, J. Reiss, and M. Isoz, *Shifted proper orthogonal decomposition and artificial neural networks for time-continuous reduced order models of transport-dominated systems*, Topical Problems of Fluid Mechanics, (2022).

[123] D. Borzacchiello, J. V. Aguado, and F. Chinesta, *Non-intrusive sparse subspace learning for parametrized problems*, Archives of Computational Methods in Engineering, 26 (2017).

[124] L. C. Evans, *Partial differential equations*, vol. 19, American Mathematical Society, 2022.

[125] S. Wang, Y. Teng, and P. Perdikaris, *Understanding and mitigating gradient flow pathologies in physics-informed neural networks*, SIAM Journal on Scientific Computing, 43 (2021), pp. A3055–A3081.

[126] A. Jacot, F. Gabriel, and C. Hongler, *Neural tangent kernel: Convergence and generalization in neural networks*, Advances in neural information processing systems, 31 (2018).

[127] S. Wang, X. Yu, and P. Perdikaris, *When and why pinns fail to train: A neural tangent kernel perspective*, Journal of Computational Physics, 449 (2022), p. 110768.

[128] A. Ammar, B. Mokdad, F. Chinesta, and R. Keunings, *A new family of solvers for some classes of multidimensional partial differential equations encountered in kinetic theory modeling of complex fluids*, Journal of non-Newtonian fluid Mechanics, 139 (2006), pp. 153–176.

[129] F. Chinesta, R. Keunings, and A. Leygue, *The proper generalized decomposition for advanced numerical simulations: a primer*, Springer Science & Business Media, 2013.

[130] A. Papritz and A. Stein, *Spatial prediction by linear kriging*, Spatial statistics for remote sensing, (2002), pp. 83–113.

[131] S. E. Ahmed and O. San, *Breaking the kolmogorov barrier in model reduction of fluid flows*, Fluids, 5 (2020), p. 26.

[132] B. Peherstorfer, *Breaking the kolmogorov barrier with nonlinear model reduction*, Notices of the American Mathematical Society, 69 (2022), pp. 725–733.

[133] J. Barnett and C. Farhat, *Quadratic approximation manifold for mitigating the kolmogorov barrier in nonlinear projection-based model order reduction*, Journal of Computational Physics, 464 (2022), p. 111348.

[134] A. Kolmogoroff, *Uber die beste annaherung von funktionen einer gegebenen funktionenklasse*, Annals of Mathematics, (1936), pp. 107–110.

[135] A. Pinkus, *N-widths in Approximation Theory*, vol. 7, Springer Science & Business Media, 2012.

[136] K. Wang and T. Gasser, *Alignment of curves by dynamic time warping*, The annals of Statistics, 25 (1997), pp. 1251–1276.

[137] C. Bork, K. Ng, Y. Liu, A. Yee, and M. Pohlscheidt, *Chromatographic peak alignment using derivative dynamic time warping*, Biotechnology Progress, 29 (2013), pp. 394–402.

[138] T. Taddei, *A registration method for model order reduction: data compression and geometry reduction*, SIAM Journal on Scientific Computing, 42 (2020), pp. A997–A1027.

[139] T. Blickhan, *A registration method for reduced basis problems using linear optimal transport*, arXiv preprint arXiv:2304.14884, (2023).

[140] P. J. Bickel, B. Li, A. B. Tsybakov, S. A. van de Geer, B. Yu, T. Valdés, C. Rivero, J. Fan, and A. van der Vaart, *Regularization in statistics*, Test, 15 (2006), pp. 271–344.

[141] B. Moya, A. Badías, I. Alfaro, F. Chinesta, and E. Cueto, *Digital twins that learn and correct themselves*, International Journal for Numerical Methods in Engineering, 123 (2022), pp. 3034–3044.

[142] A. Sancarlos, M. Cameron, A. Abel, E. Cueto, J.-L. Duval, and F. Chinesta, *From rom of electrochemistry to ai-based battery digital and hybrid twin*, Archives of Computational Methods in Engineering, 28 (2021), pp. 979–1015.

[143] F. Chinesta, E. Cueto, E. Abisset-Chavanne, J. L. Duval, and F. E. Khaldi, *Virtual, digital and hybrid twins: a new paradigm in data-based engineering and engineered data*, Archives of computational methods in engineering, 27 (2020), pp. 105–134.

[144] B. Moya, I. Alfaro, D. Gonzalez, F. Chinesta, and E. Cueto, *Physically sound, self-learning digital twins for sloshing fluids*, PLoS One, 15 (2020), p. e0234569.

[145] B. Moya, D. González, I. Alfaro, F. Chinesta, and E. Cueto, *Learning slosh dynamics by means of data*, Computational Mechanics, 64 (2019), pp. 511–523.

[146] R. B. Laughlin and D. Pines, *The theory of everything*, Proceedings of the national academy of sciences of the United States of America, 97 (2000), p. 28.

[147] G. H. Golub, P. C. Hansen, and D. P. O'Leary, *Tikhonov regularization and total least squares*, SIAM journal on matrix analysis and applications, 21 (1999), pp. 185–194.

[148] H. Zou and T. Hastie, *Regularization and variable selection via the elastic net*, Journal of the royal statistical society: series B (statistical methodology), 67 (2005), pp. 301–320.

[149] C. De Mol, E. De Vito, and L. Rosasco, *Elastic-net regularization in learning theory*, Journal of Complexity, 25 (2009), pp. 201–230.

[150] M. W. Browne, *Cross-validation methods*, Journal of mathematical psychology, 44 (2000), pp. 108–132.

[151] T. Fushiki, *Estimation of prediction error by using k-fold cross-validation*, Statistics and Computing, 21 (2011), pp. 137–146.

[152] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, vol. 2, Springer, 2009.

[153] S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: Machine learning, dynamical systems, and control*, Cambridge University Press, 2022.

[154] V. Kaarnioja, *Smolyak quadrature*, Master's thesis, Helsingfors universitet, 2013.

[155] J. Bernardo, J. Berger, A. Dawid, A. Smith, et al., *Regression and classification using gaussian process priors*, Bayesian statistics, 6 (1998), p. 475.

[156] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, *Support vector regression machines*, Advances in neural information processing systems, 9 (1996).

[157] S.-i. Amari and S. Wu, *Improving support vector machine classifiers by modifying kernel functions*, Neural Networks, 12 (1999), pp. 783–789.

[158] F. Javed, G. S. Chan, A. V. Savkin, P. M. Middleton, P. Malouf, E. Steel, J. Mackie, and N. H. Lovell, *Rbf kernel based support vector regression to estimate the blood volume and heart rate responses during hemodialysis*, in 2009 annual international conference of the IEEE engineering in medicine and biology society, IEEE, 2009, pp. 4352–4355.

[159] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, *Fast and accurate deep network learning by exponential linear units (elus)*, arXiv preprint arXiv:1511.07289, (2015).

[160] A. Kamoulakos, *The ESI-Wilkins-Kamoulakos (EWK) Rupture Model*, John Wiley & Sons, Ltd, 06 2005, pp. 795 – 804.

[161] A. Mammone, M. Turchi, and N. Cristianini, *Support vector machines*, Wiley Interdisciplinary Reviews: Computational Statistics, 1 (2009), pp. 283–289.

[162] G. Biau and E. Scornet, *A random forest guided tour*, Test, 25 (2016), pp. 197–227.

[163] C. Villani, *Topics in optimal transportation*, vol. 58, American Mathematical Soc., 2021.

[164] T. Séjourné, G. Peyré, and F.-X. Vialard, *Unbalanced optimal transport, from theory to numerics*, Handbook of Numerical Analysis, 24 (2023), pp. 407–471.

[165] W. Gangbo, W. Li, S. Osher, and M. Puthawala, *Unnormalized optimal transport*, Journal of Computational Physics, 399 (2019), p. 108940.

[166] S. Ferradans, N. Papadakis, G. Peyré, and J.-F. Aujol, *Regularized discrete optimal transport*, SIAM Journal on Imaging Sciences, 7 (2014), pp. 1853–1882.

[167] G. Peyré, M. Cuturi, et al., *Computational optimal transport*, Center for Research in Economics and Statistics Working Papers, (2017).

[168] G. Peyré, M. Cuturi, et al., *Computational optimal transport: With applications to data science*, Foundations and Trends® in Machine Learning, 11 (2019), pp. 355–607.

[169] Q. Mérigot, *A multiscale approach to optimal transport*, in Computer Graphics Forum, vol. 30, Wiley Online Library, 2011, pp. 1583–1592.

[170] R. Flamary, N. Courty, A. Gramfort, M. Z. Alaya, A. Boisbunon, S. Chambon, L. Chapel, A. Corenflos, K. Fatras, N. Fournier, et al., *Pot: Python optimal transport*, The Journal of Machine Learning Research, 22 (2021), pp. 3571–3578.

[171] V. M. Panaretos and Y. Zemel, *An invitation to statistics in Wasserstein space*, Springer Nature, 2020.

[172] M. Kell, *On interpolation and curvature via wasserstein geodesics*, Advances in Calculus of Variations, 10 (2017), pp. 125–167.

[173] S. Chewi, J. Clancy, T. Le Gouic, P. Rigollet, G. Stepaniants, and A. Stromme, *Fast and smooth interpolation on wasserstein space*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 3061–3069.

[174] Q. Mérigot, A. Delalande, and F. Chazal, *Quantitative stability of optimal transport maps and linearization of the 2-wasserstein space*, in International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 3186–3196.

[175] C. Sarrazin and B. Schmitzer, *Linearized optimal transport on manifolds*, arXiv preprint arXiv:2303.13901, (2023).

[176] P. Petersen, *Riemannian geometry*, vol. 171, Springer, 2006.

[177] L. Rüschendorf, *The wasserstein distance and approximation theorems*, Probability Theory and Related Fields, 70 (1985), pp. 117–129.

[178] M. Nutz, *Introduction to entropic optimal transport*, Lecture notes, Columbia University, (2021).

[179] W. Wang, D. Slepčev, S. Basu, J. A. Ozolek, and G. K. Rohde, *A linear optimal transportation framework for quantifying and visualizing variations in sets of images*, International journal of computer vision, 101 (2013), pp. 254–269.

[180] J. Lott, *Some geometric calculations on wasserstein space*, arXiv preprint math/0612562, (2006).

[181] J. J. Monaghan, *Smoothed particle hydrodynamics*, Reports on progress in physics, 68 (2005), p. 1703.

[182] L. Ambrosio, E. Mainini, and S. Serfaty, *Gradient flow of the chapman–rubinstein–schatzman model for signed vortices*, in Annales de l'IHP Analyse non linéaire, vol. 28, 2011, pp. 217–246.

[183] E. Mainini, *A description of transport cost for signed measures*, Journal of Mathematical Sciences, 181 (2012), pp. 837–855.

[184] T. Fischer, *Existence, uniqueness, and minimality of the jordan measure decomposition*, arXiv preprint arXiv:1206.5449, (2012).

[185] A. Figalli, *The optimal partial transport problem*, Archive for rational mechanics and analysis, 195 (2010), pp. 533–560.

[186] M. Imregun, *A review of model updating techniques*, Shock and Vibration digest, 23 (1991), pp. 9–20.

[187] Y. Yang and Y. Chen, *A new direct method for updating structural models based on measured modal data*, Engineering Structures, 31 (2009), pp. 32–42.

[188] S. Sehgal and H. Kumar, *Structural dynamic model updating techniques: A state of the art review*, Archives of Computational Methods in Engineering, 23 (2016), pp. 515–533.

[189] Z. Liu and M. Tegmark, *Machine learning conservation laws from trajectories*, Physical Review Letters, 126 (2021), p. 180604.

[190] K. Lee and K. T. Carlberg, *Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders*, Journal of Computational Physics, 404 (2020), p. 108973.

[191] X. Glorot and Y. Bengio, *Understanding the difficulty of training deep feedforward neural networks*, in Proceedings of the thirteenth international conference on artificial intelligence and statistics, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[192] I. E. C. Standards, *Power transformer- part 7: Loading guide for mineral-oil-immersed power transformers*, IEC60076-7, (2018).

[193] I. E. E. E. Standards Association, *Ieee guide for loading mineral-oil-immersed transformers and step-voltage regulators*, IEEE Std C57.91-2011 (Revision of IEEE Std C57.91-1995), (2012).

[194] D. L. Alvarez, S. R. Rivera, and E. E. Mombello, *Transformer thermal capacity estimation and prediction using dynamic rating monitoring*, IEEE Transactions on Power Delivery, 34 (2019), pp. 1695–1705.

[195] L. Paulhiac and R. Desquiens, *Dynamic thermal model for oil directed air forced power transformers with cooling stage representation*, IEEE Transactions on Power Delivery, 37 (2022), pp. 4135–4144.

[196] N. F. Alkayem, M. Cao, Y. Zhang, M. Bayat, and Z. Su, *Structural damage detection using finite element model updating with evolutionary algorithms: a survey*, Neural Computing and Applications, 30 (2018), pp. 389–411.

[197] J. Cattarius and D. Inman, *Time domain analysis for damage detection in smart structures*, Mechanical Systems and signal processing, 11 (1997), pp. 409–423.

[198] S. R. Ibrahim and E. Mikulcik, *A method for the direct identification of vibration parameters from the free response*, The Shock and Vibration Inform. Ctr. Shock and Vibration Bull. Part. 4: Sep. 1977, (1977).

[199] J.-N. Juang and R. S. Pappa, *An eigensystem realization algorithm for modal parameter identification and model reduction*, Journal of guidance, control, and dynamics, 8 (1985), pp. 620–627.

[200] S. R. Ibrahim, *Double least squares approach for use in structural modal identification*, AIAA journal, 24 (1986), pp. 499–503.

[201] B. Peeters and G. De Roeck, *Stochastic system identification for operational modal analysis: a review*, J. Dyn. Sys., Meas., Control, 123 (2001), pp. 659–667.

[202] J. Vandiver, A. Dunwoody, R. Campbell, and M. Cook, *A mathematical basis for the random decrement vibration signature analysis technique*, Journal of Mechanical Design, (1982).

[203] N. Maia, J. Silva, and R. Sampaio, *Localization of damage using curvature of the frequency-response-functions*, in Proceedings of the 15th international modal analysis conference, vol. 3089, 1997, p. 942.

[204] X. Liu, N. Lieven, and P. J. Escamilla-Ambrosio, *Frequency response function shape-based methods for structural damage localisation*, Mechanical systems and signal processing, 23 (2009), pp. 1243–1259.

[205] S. Mohan, D. K. Maiti, and D. Maity, *Structural damage assessment using frf employing particle swarm optimization*, Applied Mathematics and Computation, 219 (2013), pp. 10387–10400.

[206] S. Rucevskis, R. Janeliukstis, P. Akishin, and A. Chate, *Mode shape-based damage detection in plate structure without baseline data*, Structural Control and Health Monitoring, 23 (2016), pp. 1180–1193.

[207] G. Sha, M. Radzieński, M. Cao, and W. Ostachowicz, *A novel method for single and multiple damage detection in beams using relative natural frequency changes*, Mechanical Systems and Signal Processing, 132 (2019), pp. 335–352.

[208] M. Azimi, A. D. Eslamlou, and G. Pekcan, *Data-driven structural health monitoring and damage detection through deep learning: State-of-the-art review*, Sensors, 20 (2020), p. 2778.

[209] A. M. Tillmann, *Equivalence of linear programming and basis pursuit*, PAMM, 15 (2015), pp. 735–738.

List of Figures