Deep learning for physical systems: estimation, adaptation and exploration

Matthieu Blanke

2024

Remerciements

Après trois années, ma thèse touche à sa fin et je souhaite remercier les personnes qui m'ont accompagné durant cette aventure, et sans qui rien de tout cela n'aurait été possible.

Je tiens tout d'abord à exprimer ma profonde gratitude envers mon directeur de thèse Marc Lelarge. Ces trois années de thèse sur un sujet nouveau ont été un chemin scientifique stimulant, le long duquel tu as toujours su me donner des conseils précieux. Sur les sujets scientifiques, comme sur les questions extra-scientifiques, ton expertise et ta disponibilité ont été clé pour l'avancée de notre travail. Surtout, la bonnne humeur et les blagues quotidienne qui ont accompagné nos discussions me laisseront pour toujours un excellent souvenir de cette thèse.

Je souhaite également remercier les membres du jury Pauline Bernard, Patrick Gallinari, Pierre Gentine et Gilles Louppe, pour avoir accepté de remplir ce rôle, et pour les précieux retours scientifiques qu'ils m'ont fourni.

Un grand merci à Ronan pour m'avoir accueilli à l'IMT à Brest, pour m'avoir fait découvrir la recherche passionante sur les réseaux de neurones pour l'assimilation de données et l'océanographie, pour sa disponibilité et sa bienveillance. J'ai été très heureux de pouvoir travailler avec toi et j'espère que nous continuerons à collaborer dans le futur.

J'ai une pensée pour les professeurs qui ont nourri chez moi une curiosité et une passion pour les mathématiques et la physique. Je souhaite remercier mes professeurs du collège-lycée de l'Harteloire de Brest, notamment M^{me} Leteuré, M^{me} Corda, M^{me} Quilivic, M^{me} Croguennec, M^{me} Regbaoui, M. Le Bloas, M. Beriven et M. Petibon. Merci à mes professeurs de classes préparatoire M. Joseph, M. Becker, M. Kious, M. Rochet et M^{me} Froloff, et à mes professeurs de l'École polytechnique et de master, en particulier à M^{me}. Allassonnière, M. Mallick, M. Goerbig, M. Oudot, M^{me} Cugliandolo, et M. Fournier. Je remercie également Michael Benzaquen, Jean-Philippe Bouchaud et José Moran qui m'ont encadré dans ma première expérience de recherche à Econophysix, dont je garde un excellent souvenir et qui m'a donné envie de poursuivre dans cette voie.

Pendant ces trois années, j'ai eu la chance de travailler dans un environnement de travail d'exception. À l'Inria, j'ai toujours eu cette impression de me sentir à la maison, et je souhaite remercier tous ceux qui ont participé à cette atmosphère si bienveillante.

Je remercie tout d'abord les membres de mon bureau. Merci à Luca pour avoir été un grand frère de thèse idéal avec toute la bienveillance du monde, et un ami cher. Merci à Bastien, Éric et Mathieu pour avoir été les trois autres frères du Premier Bureau des Boloss, dont je suis très fier de faire partie à vos côtés. Nos fous rires au bureau, mais aussi au CIRM, et à East Mamma resteront gravés comme les premiers souvenirs de thèse. Merci à Ludovic pour nous avoir montré la voie, et merci à Jakob pour avoir pris la suite, et qui a fait perdurer la bonne ambiance au quotidien. Enfin, merci à la nouvelle génération, Killian et Martin. Je suis très heureux de voir en partant du bureau qu'il reste entre de très bonnes mains. Dès votre arrivée, j'ai compris que la dernière année de thèse allait être joyeuse : vous êtes tout de suite devenus des amis, avec qui j'ai eu la chance de partager le même bureau. De la "early cantine" au "late baby", les journées à vos côtés passaient vite même quand le travail était dur. Je vous souhaite une excellente continuation, et je vous remercie pour tout. Merci aussi à Maxime pour son passage au bureau et sa motivation pour toutes les activités sportives.

Merci à Ilia, pour avoir été un frère de thèse. Certains moments pendant la thèse sont plus difficiles que d'autres. Pendant ces moments, j'ai eu la chance immense d'avoir un ami très cher qui me rappelait que «it's not so bad, it's not so bad». Merci à Lucas, Vianney, Thomas, David et Romain pour avoir été des amis toujours bienveillants et de bons conseils.

Un énorme merci à Marina, pour ses sourires qui ont illuminé le quotidien de l'équipe. Ton arrivée a tout de suite rendu les journées à l'Inria plus belles et plus drôles, et ces heures à discuter au bureau

vont me manquer. Merci aussi à Hélène Bessin-Rousseau et à Hélène Milome pour leur bonne humeur et leur aide si précieuse.

Merci à tous les membres de l'équipe Argo, où la bonne l'ambiance a toujours régné, que ce soit pendant les séminaires, autour d'une table à la cantine ou à la pause gâteau. Merci en particulier aux autres doctorants, avec qui nous avons partagé cette belle aventure. Merci à Constantin pour m'avoir beaucoup aidé dans la rédaction de mon mémoire de thèse et à Batiste pour tous ses conseils et ces discussions sur le surf.

Merci à Justin Carpentier et à Stéphane Caron, pour avoir toujours été disponibles pour répondre à mes questions sur le sujet si fascinant qu'est la robotique, et pour les collaborations à venir. Merci à Adrien Taylor pour les discussions sur la recherche. Merci à Thomas Eboli, Thomas Chabal, Bertille, Antonin, Roman, Armand, Alessia, Élise, Maxime, Guillaume, Julia et Nathan qui ont été des amis pendant cette thèse à l'Inria, et que j'espère revoir très vite ici ou ailleurs.

Merci à nos collègues des ressources humaines Alexandre, Marion, Julien, Johanna, Ortenca, Lucile et Sandra, sur qui on a toujours pu compter, pour leur sympathie et leur humour. Merci à Éric Fleury pour avoir été non seulement à la tête, mais aussi au cœur de notre centre Inria. Merci aussi à Éric Sinaman pour son aide précieuse à la fin de la thèse.

En dehors de l'Inria, je remercie Giuliano Orso, Christophe Mora, Thomas Romary et Pauline Bernard pour m'avoir accompagné avec beaucoup de bienveillance dans les missions d'enseignement. Merci à Rudy Morel, Julien Le Sommer, Jean-Noël Thépaut, Nils Thuerey et Éric Passy pour tous leurs conseils. Merci à Nicolas Desassis, Sylvère Bonnabel, Mike Pereira et Mona Buisson-Fenet pour leur accueil et les discussions aux Mines.

Merci à Hugo Yèche et aux autres doctorants rencontrés à ICLR 2024 pour cette belle expérience, j'espère vous revoir bientôt.

Au-delà du monde professionnel, j'aimerais remercier tous mes amis qui m'ont soutenu pendant la thèse, et qui vont beaucoup, beaucoup me manquer.

Merci à mon ami d'enfance Nathan. Merci à Nicolas pour être venu depuis Brest, et pour avoir été un très cher compagnon de voyage. Merci à Franck, Erika, Galy, Ana-Yulia et Anastasia pour toutes ces moments et ces soirées. Merci au groupe des Brestois : Alix, Clara, David, Francis, James, Marion, Martin, Rova et Samuel avec qui on retrouvait toujours un peu de notre Brest natale à Paris ou ailleurs.

Merci à mes amis de classes préparatoires Martin, Eliott, Michaël, Hadrien, Hugo, Quentin et Simon, qui m'ont appris à travailler dur même dans les conditions extrêmes, mais surtout à le faire dans la bonne humeur. Merci à Song, Vincent et Toby, pour avoir illuminé la fin de la prépa et pour avoir été de si bons amis par la suite. Merci aussi à Eulalie, Vénus et Clémentine.

Merci aux amis rencontrés à Polytechnique. Un énorme merci à mes amis de la X5 : Lucas, Timothée Jules, Benoît, et bien sûr aussi Alex. Avec ces foots, ces rassos viaducs, ces parties de Mario Kart, le mansion à Bastille, les repas du midi sur les quais, les vacances à Brest, la conversation au quotidien, toutes ces soirées ... vous m'avez régalé et je n'aurais pas pu imaginer ces trois ans de thèse sans vous. Un grand merci à mes amis marocains Souhail, Adam et Ayman, avec lesquels j'espère pouvoir vite refaire des vacances au ski. Merci aussi à Éloise, Gabriel et Samuel, que j'espère revoir bientôt.

Merci à mes amis de l'ESSCA pour ces vacances et ces soirées déjantées : merci à Elisabeth, Adrien, Nicolas, et Alice, ainsi qu'à Juliette, Marilou et Hugo.

Merci à Joséphine, pour tout.

Enfin, un grand merci à ma famille, sur qui j'ai pu toujours compter, et que j'aime fort.

Résumé

Ces dernières années ont été marquées par une révolution de l'intelligence artificielle, avec des percées remarquables dans des domaines variés tels que la vision par ordinateur, la génération automatique d'images et le traitement du langage naturel. Dans les applications scientifiques, l'apprentissage profond présente un potentiel immense grâce à sa capacité à apprendre à partir des données et à fournir des solutions approximatives à des problèmes physiques complexes, sans nécessiter une modélisation explicite. Cependant, l'application de ces approches dans les sciences rencontre encore des défis techniques importants, tels que le manque de données, le coût des calculs et l'interprétabilité des modèles.

Cette thèse aborde divers aspects de l'application de l'apprentissage profond aux systèmes physiques, où ces défis techniques sont particulièrement prononcés. Nous étudions les problèmes de l'identification de système, du plan d'expérience optimal et de l'estimation d'état, trois sujets interdépendants qui, ensemble, décrivent le processus de modélisation statistique d'un système physique. Notre objectif est de démontrer comment l'apprentissage profond peut accélérer les méthodes existantes pour ces problèmes, tout en respectant les contraintes techniques imposées par les systèmes physiques. Pour ce faire, nous proposons des méthodes interprétables qui puissent opérer dans un régime de faible nombre de données et de puissance de calcul limitée, contrairement à l'approche «boîte noire» traditionnelle des réseaux de neurones.

Notre première contribution traite du problème d'identification de système dans un contexte où les données sont limitées et non identiquement distribuées. Pour répondre à ces contraintes, nous proposons un modèle de méta-apprentissage qui incorpore la structure physique du système, permettant ainsi une modélisation statistique interprétable et à moindre coût.

Ensuite, nous abordons le problème du plan d'expérience pour les systèmes physiques, en nous concentrant sur les applications aux systèmes dynamiques embarqués tels que les robots. Ces systèmes doivent fonctionner en temps réel et imposent donc des contraintes strictes en termes de quantités de données disponibles et de capacités de calcul. En utilisant la théorie de l'information, nous concevons un algorithme d'exploration compatible avec ces contraintes, que nous appliquons d'abord à des modèles linéaires de la dynamique, puis à des modèles plus complexes tels que les réseaux de neurones.

Enfin, notre dernière contribution s'attaque au problème de l'assimilation de données, qui est particulièrement crucial en météorologie. Nous proposons un algorithme neuronal pour résoudre ce problème inverse, et explorons comment cette approche peut améliorer la précision des méthodes traditionnelles tout en réduisant leur coût de calcul.

Mots-clés: apprentissage profond, systèmes physiques, identification de système, apprentissage actif, plan d'expériences, assimilation de données

Abstract

In recent years, we have witnessed an artificial intelligence revolution, with remarkable breakthroughs in various fields such as computer vision, automatic image generation, and natural language processing. In scientific applications, deep learning has immense potential due to its ability to learn from data and provide approximate solutions to complex physical problems without the need for explicit modeling. However, the application of these approaches in the sciences still faces significant technical challenges, such as data scarcity, computational complexity, and model interpretability.

This thesis addresses various aspects of deep learning applications to physical systems, where these technical challenges are particularly prominent. We explore the problems of system identification, optimal experimental design, and state estimation, three interrelated issues that together describe the statistical modeling process of a physical system. Our goal is to demonstrate how deep learning can accelerate existing methods for these problems while respecting the applicability constraints imposed by physical systems. To achieve this, we propose interpretable methods that can operate in a regime of limited data and computational power, unlike the traditional "black-box" approach of deep learning.

Our first contribution addresses the problem of system identification in a context where observations are limited in number and not identically distributed. To meet these constraints, we propose a meta-learning model that incorporates the physical structure of the system, enabling interpretable learning at a lower cost.

Next, we tackle the problem of experimental design for physical systems, focusing on applications to embedded dynamical systems such as robots. These systems must operate in real-time, imposing strict constraints on the amount of available data and computational capacity. Using information theory, we design an exploration algorithm compatible with these constraints, which we first apply to linear dynamics and then to more complex models such as neural networks.

Finally, our last contribution addresses the problem of data assimilation, which is particularly crucial in meteorology. We propose a neural network-based approach to solve this inverse problem and explore how it can improve the accuracy of traditional methods while reducing their computational cost.

Keywords: deep learning, physical systems, system identification, active learning, experimental design, data assimilation

Contents

Thesis outline

Chapter-by-chapter summary

The following is a chapter-by-chapter summary of this thesis.

Chapter 1 The opening chapter of this thesis provides an overview of statistical learning and deep learning for physical systems. Beginning with a historical perspective on statistical estimation in physical experiments, we discuss the role of machine learning in the sciences in the era of artificial intelligence. While artificial intelligence has the potential to accelerate science with a wealth of applications, the complexity of physical systems also poses its own challenges. With these challenges in mind, we outline the contributions of this thesis in the field of deep learning for physical systems.

Chapter 2 In this chapter, we lay the mathematical framework on which we build in this thesis. We introduce the state-space representation of physical systems, and the associated statistical model. Based on this formalism, and drawing on a number of examples, we present various statistical problems around physical systems and machine learning, which are addressed in the remainder of this thesis. Although these are separate problems, we show how they are interconnected and together form a unified statistical modeling process for physical experiments.

Chapter 3 This chapter deals with the multi-environment learning problem, where a physical system is learned with data collected from inhomogeneous experimental conditions. The aim is to design a learning architecture for system identification that can adapt and generalize to the variability of physical systems. Recently, meta-learning approaches have made significant progress in multi-task learning, but they rely on black-box neural networks, resulting in high computational costs and limited interpretability. Leveraging the physical structure of the learning problem, we argue that multi-environment generalization can be achieved using a simpler learning model, with an affine structure with respect to the learning task. Crucially, we prove that this architecture can identify the physical parameters of the system, enabling interpretable learning. We demonstrate the competitive generalization performance and the low computational cost of our method by comparing it to state-of-the-art algorithms on physical systems, ranging from toy models to complex non-analytical systems. The interpretability of our method is illustrated with original applications to parameter estimation and to adaptive control.

Chapter 4 This chapter tackles the problem of active learning and experimental design for system identification, where the experimenter seeks to choose the most informative experiments for learning the system. We focus on dynamical systems, which adds an online decision-making constraint. For multi-input multi-output linear time-invariant dynamical systems, we use an information-theoretic framework and introduce an online greedy exploration policy where the control maximizes the information of the next step. In a setting with a limited number of observations, our algorithm has low complexity and shows experimentally competitive performances compared to more elaborate gradient-based methods.

Chapter 5 This chapter aims to extend the online exploration policy of the previous chapter to nonlinear learning models, including neural networks. The complexity of nonlinear dynamics and the computational limites of real systems make it challenging to formulate and optimize an exploration objective. In this chapter, we introduce FLEX, an exploration algorithm for nonlinear dynamics based on optimal experimental design. Our policy maximizes the information of the next step and results in an adaptive exploration algorithm, compatible with generic parametric learning models and requiring minimal resources. We test our method on a number of nonlinear environments covering different settings, including time-varying dynamics. Keeping in mind that exploration is intended to serve an exploitation objective, we also test our algorithm on downstream model-based classical control tasks and compare it to other state-of-the-art model-based and model-free approaches. The performance achieved by FLEX is competitive, and its computational cost is low.

Chapter 6 This chapter studies the data assimilation problem and introduces preliminary work on a neural network approach. Data assimilation is a central problem in many geophysical applications, such as weather forecasting. It aims to estimate the state of a potentially large system, such as the atmosphere, from sparse observations, supplemented by prior physical knowledge. The size of the systems involved and the complexity of the underlying physical equations make it a challenging task from a computational point of view. Neural networks represent a promising method of emulating the physics at low cost, and therefore have the potential to considerably improve and accelerate data assimilation. We introduce a deep learning approach where the physical system is modeled as a sequence of coarse-to-fine Gaussian prior distributions parametrized by a neural network. This allows us to define an assimilation operator, which is trained in an end-to-end fashion to minimize the reconstruction error on a dataset with different observation processes. We illustrate our approach on chaotic dynamical physical systems with sparse observations and compare it to traditional variational data assimilation methods.

Chapter 7 The final chapter of this thesis summarizes our contributions and discusses several limites and possible extensions.

Related publications and preprints

Publications and preprints related to this thesis are listed below.

Chapter 3 is based on the article *Interpretable Meta-Learning of Physical Systems* (Blanke and Lelarge, 2024), published in The Twelfth International Conference on Learning Representations (ICLR 2024).

Chapter 4 is based on the article *Online greedy identification of linear dynamical systems* (Blanke and Lelarge, 2022), published in the Proceedings of the IEEE 61st Conference on Decision and Control (CDC 2022).

Chapter 5 is based on the article *FLEX: an Adaptive Exploration Algorithm for Nonlinear Systems* (Blanke and Lelarge, 2023), published in the Proceedings of the 40th International Conference on Machine Learning (ICML 2023).

Chapter 6 is based on the article *Incremental Neural Data Assimilation* (Blanke et al., 2024), accepted at the ICML 2024 AI for Science workshop (ICML 2024 AI4Science).

Software

In the course of this thesis, we have paid particular attention to the development of open-source numerical tools and reproducible experiments. Our software contributions are summarized below.

CAMEL A deep meta-learning architecture for multi-environment learning of physical systems (Chapter 3), based on PyTorch (Paszke et al., 2019). `https://github.com/MB-29/CAMEL`

Greedy identification Online active identification algorithm for linear dynamical systems (Chapter 4), based on PyTorch. `https://github.com/MB-29/greedy-identification`

FLEX A fast, adaptive and flexible model-based reinforcement learning exploration algorithm (Chapter 5), based on PyTorch. `https://github.com/MB-29/FLEX`

Neural data assimilation A neural method for the data assimilation problem (Chapter 6), based on JAX (Bradbury et al., 2018). `https://github.com/MB-29/assimilation`

Résumé substantiel

Chapitre 1

Ce chapitre explore le rôle croissant de l'intelligence artificielle (IA) dans les sciences physiques, en mettant particulièrement l'accent sur l'apprentissage profond. L'IA, notamment avec les réseaux neuronaux profonds, a transformé de nombreux domaines tels que la vision par ordinateur, la génération d'images et le traitement du langage naturel. Ces modèles, inspirés par la structure du cerveau humain, permettent de traiter des données complexes et de trouver des solutions approximatives à des problèmes auparavant inaccessibles avec des méthodes traditionnelles.

Dans les sciences, des applications marquantes de l'IA ont émergé, telles que la prédiction la structure des protéines, ou des odèles avancés de prévision météorologique. Cependant, l'intégration de l'apprentissage profond aux systèmes physiques pose plusieurs défis. Ces systèmes sont complexes, non linéaires et souvent limités par la rareté des données expérimentales. Les réseaux neuronaux, bien qu'efficaces, restent des « boîtes noires » difficiles à interpréter, ce qui complique leur adoption dans des contextes critiques où les erreurs peuvent avoir des conséquences majeures.

Le chapitre aborde également l'évolution de la modélisation statistique, des méthodes historiques comme les moindres carrés aux techniques modernes d'apprentissage profond. Ces dernières, grâce à leur flexibilité, complètent ou remplacent parfois les modèles physiques traditionnels, en apprenant directement à partir des données. Elles permettent ainsi de modéliser des phénomènes complexes comme la dynamique des fluides ou les interactions robotiques, tout en réduisant les coûts computationnels.

Enfin, ce chapitre met en lumière les défis liés à la quantification des incertitudes et à l'exigence de modèles robustes face aux variations des environnements expérimentaux, tout en posant les bases des contributions de la thèse.

Chapitre 2

Ce chapitre établit le cadre statistique fondamental sur lequel repose cette thèse. En utilisant la représentation d'espace d'état, un formalisme issu de la théorie du contrôle, il propose une approche unifiée pour modéliser les systèmes physiques. Ce cadre permet de modéliser les relations complexes entre les variables physiques observées et les variables non observées, tout en tenant compte des incertitudes inhérentes aux systèmes et aux mesures.

Représentation d'espace d'état Un système physique est décrit par un vecteur d'état regroupant toutes les variables physiques qui le caractérisent. Ces variables évoluent en fonction des contrôles expérimentaux appliqués par l'utilisateur et des lois physiques sous-jacentes, souvent modélisées par des équations paramétriques. Cependant, ces modèles sont rarement parfaits : des effets non modélisés et des bruits expérimentaux ajoutent des incertitudes. Par ailleurs, les observations sont généralement partielles et bruitées, nécessitant une modélisation probabiliste pour rendre compte de cette complexité.

Nous proposons des exemples illustrant cette représentation, comme celui du champ électrostatique généré par des charges ponctuelles. Ce système simple permet d'introduire les concepts fondamentaux de l'estimation statistique. D'autres exemples, comme les systèmes dynamiques non linéaires (par exemple, le pendule) ou des modèles atmosphériques complexes, montrent comment cette approche s'étend à des systèmes réels et plus difficiles à modéliser.

Problèmes statistiques clés Le chapitre met en avant trois problèmes statistiques fondamentaux liés à la modélisation des systèmes physiques. Tout d'abord, le problèem d'identification de système consiste à estimer les paramètres d'un modèle à partir de données expérimentales. L'objectif est de construire un modèle paramétrique qui représente fidèlement le système physique. Cela inclut des défis tels que la gestion de données hétérogènes provenant de conditions expérimentales variées et l'intégration de modèles physiques et de modèles appris par des réseaux neuronaux. Ensuite le problème de plan d'expérience optimal consiste à déterminer une stratégie pour choisir les expériences qui maximisent l'information recueillie sur le système. Ce problème est crucial lorsque les ressources expérimentales sont limitées ou coûteuses. Le chapitre introduit des approches théoriques basées sur des critères d'optimalité, tels que l'information de Fisher, pour guider ces choix. Enfin, nous présentons le problème d'assimilation de données, qui vise à reconstruire les états d'un système physique à partir d'observations incomplètes et bruitées. Ce cadre est essentiel dans des domaines tels que la météorologie, où les observations sont clairsemées et les systèmes très complexes. Des méthodes probabilistes, comme le filtre de Kalman ou ses extensions non linéaires, sont utilisées pour intégrer les observations aux connaissances a priori.

Défis Ce chapitre identifie les principaux défis qui feront l'objet des contributions de cette thèse : la gestion de données limitées, l'interprétabilité des modèles d'apprentissage, et la quantification des incertitudes. Les modèles traditionnels, souvent linéaires, sont robustes mais limités dans leur capacité à représenter des phénomènes non linéaires complexes. Les réseaux neuronaux profonds, bien qu'expressifs, posent des problèmes d'interprétabilité et nécessitent de grandes quantités de données pour éviter le surapprentissage. Ce chapitre présente l'enjeu de ces défis dans les problèmes statistiques qui nous occupent, en se fondant sur le cadre mathématique de la représentation d'espace d'états.

Chapitre 3

Ce chapitre explore le problème de l'apprentissage de systèmes physiques dans des environnements multiples, un défi courant dans les sciences physiques où les données proviennent de conditions expérimentales variées. Ces variations rendent nécessaire le développement d'algorithmes d'apprentissage capables de généraliser efficacement et de s'adapter rapidement à de nouveaux environnements. Ce chapitre introduit une méthode appelée CAMEL (Context-Affine Multi-Environment Learning), conçue pour relever ces défis tout en garantissant l'interprétabilité du modèle d'apprentissage.

Problème Les systèmes physiques sont souvent influencés par des paramètres qui varient selon les environnements expérimentaux. Ces variations, parfois subtiles, compliquent l'identification des systèmes et nécessitent des modèles d'apprentissage capables de gérer des données non homogènes. Les approches existantes basées sur l'apprentissage profond, comme les algorithmes de méta-apprentissage, sont performantes mais souvent coûteuses en calcul et manquent d'interprétabilité. Ces limites rendent leur application difficile dans des contextes où les données sont rares et les contraintes computationnelles importantes, comme en robotique ou dans la modélisation des systèmes dynamiques.

Contributions Le chapitre propose une approche simplifiée et efficace basée sur une architecture d'apprentissage affine en un paramètre de contexte. L'architecture CAMEL exploite la structure physique sous-jacente des systèmes pour améliorer la généralisation tout en réduisant les coûts

computationnels. Elle repose sur l'idée que de nombreux systèmes physiques peuvent être modélisés par des relations affines en fonction des paramètres spécifiques aux tâches. Cela permet d'aligner le modèle statistique avec les structures physiques connues, facilitant ainsi l'identification des paramètres. En outre, CAMEL se distingue par sa simplicité computationnelle : il évite les optimisations complexes des approches traditionnelles et s'adapte efficacement aux nouvelles tâches à l'aide de solutions analytiques directes.

Validation expérimentale La méthode est évaluée sur plusieurs systèmes physiques, allant de modèles analytiques simples à des systèmes non analytiques complexes. Les expériences montrent que CAMEL offre des performances comparables, voire supérieures, aux méthodes existantes, avec une capacité remarquable à identifier les paramètres physiques. Par exemple, dans des applications robotiques, CAMEL permet un contrôle adaptatif tout en identifiant simultanément les paramètres dynamiques des robots, même dans des conditions expérimentales difficiles.

Perspectives L'architecture que nous présentons ouvre des perspectives prometteuses pour l'apprentissage dans les sciences physiques, en particulier dans les domaines nécessitant des solutions interprétables et économes en calcul. Des extensions potentielles pourraient être étudiées, comme l'intégration de contraintes physiques plus complexes ou l'utilisation d'approches d'apprentissage actif pour maximiser l'efficacité des données disponibles.

Chapitre 4

Ce chapitre traite de l'identification active des systèmes dynamiques linéaires, un problème crucial dans des domaines comme l'aéronautique et la robotique. L'objectif est de concevoir une méthode d'exploration efficace pour estimer les paramètres d'un système inconnu tout en minimisant le nombre d'observations nécessaires. Ce travail se concentre sur les systèmes linéaires multivariés invariants dans le temps, qui, malgré leur simplicité, peuvent représenter de nombreux systèmes réels, y compris des systèmes non linéaires localement linéarisés.

Problème L'identification des systèmes physiques consiste à estimer leurs paramètres en recueillant des données expérimentales. Cependant, ces expériences sont souvent coûteuses en termes de temps, de ressources et de risques. Ce chapitre s'intéresse à une approche d'identification active, où l'expérimentateur choisit des actions pour maximiser les informations collectées à chaque étape, permettant ainsi d'améliorer la précision de l'estimation tout en réduisant les coûts.

Contributions Le chapitre présente une nouvelle politique d'identification active basée sur la théorie de l'informaiton, en adoptant une approche gloutonne. Contrairement aux méthodes existantes qui optimisent sur des horizons longs, cette approche simplifie le problème en se concentrant sur l'optimisation des informations recueillies à la prochaine étape. Cette stratégie offre plusieurs avantages. Premièrement, elle fonctionne en temps réel, avec des contraintes computationnelles minimales. Deuxièmement, elle est robuste dans des contextes où les ressources d'observation et de calcul sont limitées. Enfin, elle surpasse, dans certains cas, des méthodes plus complexes basées sur des méthodes d'optimisation de descente de gradient, en termes de précision et de coût.

Validation expérimentale Les performances de l'algorithme glouton sont évaluées sur des systèmes linéaires simulés ainsi que sur des applications réalistes, comme l'identification des paramètres d'un système aéronautique. Les résultats montrent que cette méthode peut rivaliser avec des algorithmes sophistiqués tout en étant beaucoup plus économe en ressources. Par exemple, dans des expériences simulant des mouvements latéraux d'un avion, l'approche gloutonne obtient des performances similaires à celles d'un oracle tout en réduisant considérablement le temps de calcul.

Perspectives Bien que cette approche repose sur des hypothèses simplifiées (comme l'accès complet aux états du système), elle offre une base solide pour une application à des systèmes plus complexes. Dans des contextes réels, des contraintes supplémentaires, comme des échantillons limités ou des restrictions sur les entrées, pourraient être prises en compte. Enfin, le chapitre ouvre des perspectives sur la généralisation de cette méthode à des systèmes non linéaires, une problématique abordée dans le chapitre suivant.

Chapitre 5

Ce chapitre traite de l'exploration active de systèmes dynamiques non linéaires, un défi central pour de nombreuses applications en robotique, aéronautique et systèmes énergétiques. Contrairement aux systèmes linéaires, les systèmes non linéaires présentent des dynamiques complexes qui rendent leur apprentissage coûteux en termes de données et de calcul. Le chapitre propose un algorithme appelé FLEX, conçu pour explorer efficacement ces systèmes tout en respectant les contraintes pratiques des systèmes réels, comme la mémoire et la vitesse de calcul.

Problème L'exploration active vise à collecter des données informatives pour modéliser un système inconnu avec un minimum d'échantillons. Cela est essentiel dans des contextes où les expériences sont coûteuses, par exemple lors des tests de systèmes embarqués ou des simulations aéronautiques. Les approches actuelles, bien que puissantes, souffrent souvent de limites : elles nécessitent de longs horizons de planification, sont lentes ou ne s'adaptent pas aux changements dynamiques en temps réel.

Contributions La politique d'exploration présentée se distingue par son adaptabilité et sa simplicité computationnelle. FLEX utilise des principes issus de la théorie de l'information pour maximiser l'efficacité des données recueillies. Plutôt que de planifier sur de longs horizons, FLEX adopte une approche gloutonne, optimisant les décisions à chaque étape pour s'adapter rapidement aux nouvelles observations. Il est compatible avec des modèles paramétriques variés, y compris les réseaux de neurones, et peut être utilisé dans des environnements où les dynamiques changent au fil du temps.

Validation expérimentale Les performances de FLEX sont évaluées dans plusieurs environnements non linéaires, comme le pendule amorti ou des systèmes à dynamique variable dans le temps. Les résultats montrent que FLEX surpasse les approches traditionnelles, tant en termes d'efficacité des échantillons que de coût computationnel. Par exemple, dans des environnements à dynamique varibale, l'algorithme dirige les actions vers des zones d'intérêt où les données sont les plus informatives, tout en maintenant une faible charge de calcul.

Perspectives Ce chapitre ouvre des perspectives prometteuses pour l'exploration de systèmes complexes. FLEX pourrait être étendu à des contextes plus réalistes, comme les systèmes où les états ne sont que partiellement observables. De plus, l'intégration de nouvelles avancées en différentiation automatique pourrait encore réduire les coûts de calcul. Enfin, l'algorithme pourrait également être adapté pour équilibrer exploration et exploitation dans des tâches où l'objectif est de maximiser les performances de contrôle basées sur les modèles appris.

Chapitre 6

Ce chapitre explore l'utilisation des réseaux neuronaux pour résoudre le problème d'assimilation de données, un défi clé dans les applications géophysiques comme la prévision météorologique. L'assimilation de données consiste à reconstruire l'état d'un système physique complexe à partir d'observations partielles et bruitées, en intégrant des connaissances a priori sur la physique sous-jacente. Bien que

les méthodes traditionnelles, comme l'algorithme 4D-Var, soient efficaces, elles souffrent d'un coût computationnel élevé. Ce chapitre propose une approche neuronale pour réduire ces coûts tout en maintenant une précision élevée.

Problème L'assimilation de données est essentielle pour des systèmes comme l'atmosphère terrestre et l'océan, où les observations sont rares et souvent imprécises. Les algorithmes traditionnels, bien qu'exacts, nécessitent des calculs intensifs, en particulier pour simuler et différencier des modèles physiques complexes. En réponse à ces limites, les réseaux neuronaux offrent une méthode prometteuse pour représenter les dynamiques physiques à un coût réduit.

Contributions Ce chapitre introduit une méthode neuronale qui repose sur un opérateur d'assimilation paramétré par un réseau neuronal. L'approche adopte un apprentissage de bout en bout pour minimiser l'erreur de reconstruction des états physiques à partir des observations. Contrairement aux approches classiques, le modèle utilise des distributions gaussiennes locales comme a priori, garantissant ainsi un calcul de solution efficace. De plus, cette méthode peut être itérative, améliorant progressivement la qualité de la reconstruction en affinant les estimations.

Validation expérimentale Les performances de cette méthode sont évaluées sur des systèmes physiques simulés, notamment le pendule non linéaire et le système de Lorenz. Ces expériences montrent que l'approche neuronale offre des gains significatifs en termes de coût computationnel tout en maintenant une précision comparable à celle des méthodes traditionnelles. De plus, l'intégration de l'estimation neuronale dans des algorithmes comme le 4D-Var réduit le nombre d'itérations nécessaires, combinant ainsi le meilleur des deux mondes : l'efficacité neuronale et la robustesse des équations de la physique.

Perspectives Bien que prometteuse, cette méthode présente des limites dans les systèmes de grande dimension, où les réseaux neuronaux peuvent manquer de généralisation en dehors des données d'entraînement. Pour des systèmes plus complexes, une approche hybride combinant réseaux neuronaux et modèles physiques semble être une solution idéale. Enfin, l'importance de la quantification des incertitudes est soulignée, ouvrant la voie à des recherches futures pour améliorer la fiabilité des prédictions.

Chapitre 7

Le dernier chapitre de cette thèse récapitule nos contributions et discute de leurs limites, ainsi que des pistes d'extension possibles. Nous résumons le cœur de notre approche, en expliquant comment les réseaux neuronaux peuvent compléter les approches statistiques classiques tout en conservant une certaine simplicité, et en offrant une meilleure interprétabilité, essentielle pour des applications scientifiques critiques.

Bien que les approches proposées aient montré des performances prometteuses sur des simulations, nous insistons sur la nécessité d'évaluations sur des systèmes réels de plus grande échelle, comme des robots ou des modèles géophysiques complexes, pour valider leur robustesse.

Parmi les pistes de recherche, plusieurs défis demeurent. Rendre les modèles plus transparents est crucial pour renforcer leur adoption dans des domaines tels que la modélisation climatique. Par ailleurs, l'intégration des connaissances physiques dans les modèles pourrait réduire la dépendance à de grands ensembles de données, notamment pour des tâches nécessitant peu d'échantillons. L'utilisation de techniques comme la compression des signaux physiques pourrait permettre de réduire les ressources nécessaires pour traiter des systèmes complexes. Enfin, les modèles génératifs, comme les modèles de diffusion, pourraient fournir une méthode robuste pour mesurer les incertitudes dans les prédictions scientifiques.

En conclusion, cette thèse ouvre de nouvelles perspectives pour l'application de l'apprentissage profond aux sciences physiques, tout en soulignant la nécessité de solutions robustes, efficaces et interprétables pour les systèmes complexes.

List of Symbols

d	state dimension
n	parameter dimension
m	observation dimension
k	input dimension
$x \in \mathbb{R}^d$	state
$u \in \mathbb{R}^k$	input
$y \in \mathbb{R}^m$	observation
$\theta \in \mathbb{R}^n$	parameter vector
$M : (u, \theta) \in \mathbb{R}^k \times \mathbb{R}^n \mapsto M(u; \theta) \in \mathbb{R}^d$	state model
$h : x, u \in \mathbb{R}^d \times \mathbb{R}^k \mapsto h(x, u) \in \mathbb{R}^m$	observation function
$\eta \in \mathbb{R}^d$	model noise
$\xi \in \mathbb{R}^m$	observational noise
z	regression variable
$f(z; \theta)$	observation model
$A \times B$	matrix multiplication of matrices A and B
$A^\top$	matrix transposition
$\mathrm{Tr}(A)$	trace of matrix A
∇	gradient
p	probability density
$\mathbb{E}$	expectation
$\mathcal{N}(\mu, P)$	normal distribution with mean μ and covariance P
$\ell : \mathbb{R}^n \to \mathbb{R}_+$	loss function
Π	policy
γ	learning rate

Chapter 1

Introduction

Recent years have seen an artificial intelligence revolution with outstanding breakthroughs across various domains, such as computer vision, automatic image generation, and natural language processing. In scientific applications, despite its great potential, deep learning still faces significant technical challenges. This chapter discusses the applications and challenges of deep learning for physical systems and outlines the contributions of this thesis.

Chapter organization This chapter is organized as follows. Section 1.1 discusses the role of scientific applications in the context of the artificial intelligence revolution. Section 1.2 explains the importance of statistical modeling in the sciences, from historical methods to deep learning approaches. Section 1.3 highlights several challenges in applying deep learning to physical systems. Section 1.4 outlines our contributions to addressing these challenges.

1.1. The sciences in the age of artificial intelligence

Artificial Intelligence (AI) is booming and full of promise, revolutionizing many technological sectors with significant societal impact. In certain fields, AI is widely recognized as state of the art, particularly in areas like computer vision (Szeliski, 2022), automatic image generation (Ho et al., 2020), and natural language processing (Vaswani et al., 2017), where AI has made substantial strides toward mainstream use.

While AI is a broad and encompassing term, many of the recent breakthroughs are driven by a specific subset of AI known as deep learning. Deep learning involves training very large statistical models known as deep neural networks on extensive datasets. Deep learning leverages multiple layers of interconnected neurons, inspired by the human brain's architecture, to learn complex patterns and representations from data (LeCun et al., 2015). Once trained, these neural networks can make predictions or generate outputs at a relatively low computational cost. The underlying general idea is that a neural network properly trained on a sufficiently large dataset offers the possibility of finding approximate solutions to problems too complex for traditional computational methods. The successes of AI highlight the versatility and power of deep learning, setting the stage for its potential impact in more complex and traditionally challenging areas such as physical systems.

The potential for deep learning in the sciences is considerable. There has been growing interest and substantial research investment in applying AI to disciplines such as biology, robotics, and climate science. Notable examples include AlphaFold (Jumper et al., 2021), an AI that predicts protein structure, and the various weather forecast models developed by major AI companies, such as GraphCast (Lam et al., 2022) or FourCastNet (Kurth et al., 2023). In these fields, modeling the underlying physical phenomena is particularly challenging due to their inherent complexity. Consequently, neural networks are appealing for their ability to provide approximate solutions where traditional physics-based models struggle.

A vast field of research is therefore emerging at the interface between different fields of science and artificial intelligence, statistics, and mathematics. However, the complexity of physical systems often poses significant barriers to the immediate application of neural networks. It is thus of great interest to overcome these challenges and combine domain expertise with deep learning to achieve better results than each of the disciplines separately (Karniadakis et al., 2021).

1.2. Learning physical systems: from least squares to deep learning

Statistical methods have always been part of physical experimentation, long before the rise of machine learning. The celebrated method of least squares, for example, finds its roots in astronomy, as scientists sought to aid navigators during the Age of Discovery. Accurate descriptions of celestial bodies were essential for safe navigation across uncharted seas. The method, pioneered by Legendre (1806) and Gauss (1809), was among the first mathematically rigorous methods for statistical estimation, with connections to the normal distribution.

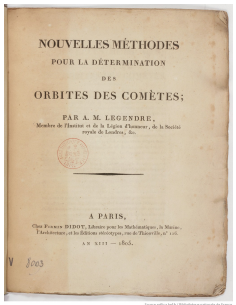
NOUVELLES MÉTHODES
POUR LA DÉTERMINATION
DES
ORBITES DES COMÈTES;
PAR A. M. LEGENDRE,
A PARIS,

Figure 1.1. First page of Legendre's memoir on the estimation of comet orbits.

Consequently, the first mathematically rigorous applications of statistical learning emerged from physics, where these methods are used to estimate physical parameters. Physical models generally rely on such parameters, which may not always be known beforehand. During experimentation, these parameters are inferred from measurements, which are inevitably subject to errors, due to model imprecision or to the observation process. A robust and efficient statistical treatment of these errors is essential to validate experimental results (Pugh and Winslow, 1966).

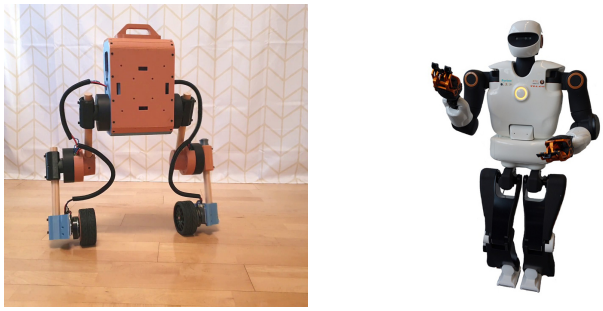

Figure 1.2. **Left** Upkie, credit: Stéphane Caron. **Right** Pyrene humanoid, credit: CNRS.

Least-squares parameter estimation remains the preferred method for many applications due to its simplicity of solution and computational efficiency. In robotics, for example, least-squares estimation is used to estimate the system's dynamic parameters: as a robot moves, its inertia and mass may change over time (consider a robot picking up, carrying and dropping a load). Accurate estimation of these parameters is essential for real-time control of the robot (Tedrake, 2022).

Today, statistical learning methods have undergone remarkable advancement, serving as the backbone of the AI revolution. The development of learning architectures based on neural networks, coupled with the availability of large datasets and powerful computing resources, as well as the emergence of new optimization algorithms, has extended the scope of least-squares regression to highly sophisticated deep learning models (Kingma and Ba, 2017). For physical systems, such powerful statistical models are appealing for several types of applications, which we briefly describe next.

Beyond least-squares estimation of physical parameters, more advanced statistical models such as neural networks can be introduced to complement a theoretical model. While theoretical models are often based on explicit physical principles and equations, they may fail to capture the full complexity of real-world phenomena. In contrast, deep learning models can autonomously learn patterns and relationships directly from data, without the need for explicit equations or assumptions about underlying processes. This flexibility allows deep learning models to adapt to complex and nonlinear relationships that may be challenging for traditional theoretical models to capture accurately (Brunton and Kutz, 2022).

Consider the example of robotics. A fundamental aspect of robot control algorithms involves planning future movements based on desired motions, requiring a faithful model of robot dynamics. Figure 1.2 shows two biped robots, for which the mere task of balancing requires an accurate model. While physics provides a solid foundation for describing robot motion equations (Spong et al., 2020), inherent complexities such as friction introduce significant errors, which may considerably impair control performance. It is therefore crucial to account for these effects as accurately as possible. When these phenomena are too complex for direct physics-based modeling, statistical methods offer a viable alternative by fitting a model to data. Deep learning emerges as a promising option for learning these nonlinear effects, allowing the use of hybrid models where neural networks learn only the unknown component of the dynamics (Zeng et al., 2020; Gao et al., 2024).

Moreover, deep learning models have demonstrated the capability to not only complement but even replace traditional physical models in certain scenarios. Indeed, the computational efficiency of neural networks enables them to outperform complex physical simulators in terms of speed, as they are cheaper to evaluate once trained. This advantage has led to the development of surrogate models (Willard et al., 2020; Haghighat et al., 2021), which use neural networks to approximate the behavior of complex systems at a fraction of the computational cost. For example, fluid dynamics (Lemarié-Rieusset, 2018), which represent particularly complex physical systems (Carlson et al., 2006), have been effectively modeled using neural networks (Thuerey et al., 2020). In atmospheric physics and in oceanography, deep learning has been applied for weather and climate forecasting (Lam et al., 2022; Hoyer et al., 2023;

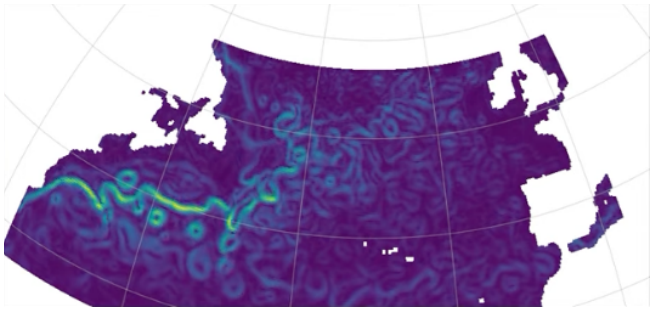

Figure 1.3. Ocean modeling in the Gulf Stream region, using deep neural networks for data assimilation (Beauchamp et al., 2023).

Ben Bouallègue et al., 2024), as well as ocean modeling (Fablet et al., 2021a), offering a promising approach for improving the accuracy and efficiency of traditional methods that rely on the numerical integration of the Navier-Stokes equations. Figure 1.3 shows the reconstruction of the sea surface height in the Gulf Stream region by a neural network, illustrating the potential to learn complex physical processes from data. The computational gain of deep learning can be crucial in many scenarios. For instance, fast predictions are valuable in real-time applications, such as extreme event prediction (Ghil et al., 2011), or clinical diagnosis, which uses magnetic resonance imaging-based computational fluid dynamics (Peper et al., 2022). The computational savings of deep learning also allow for many-query applications, where the model needs to be evaluated a large number of times. Such applications include optimal experimental design (Rainforth et al., 2024), data assimilation (Fablet et al., 2021b), and uncertainty quantification (Abdar et al., 2021). It should be noted, however, that there is a considerable upstream cost in the training of the neural network on immense quantities of high-dimensional data (Strubell et al., 2020).

1.3. Challenges of learning physical systems

While the examples above demonstrate the great potential of deep learning in the sciences, this field also presents significant technical challenges. The range of applications for artificial intelligence is vast, and each domain comes with its own technical constraints. We highlight several challenges common to many physical systems through various examples.

The use of deep learning methods relies on the availability of sufficiently large datasets to train complex models. However, acquiring data for physical systems often comes at a high cost due to the resource-intensive nature of experiments. Consider geophysical systems like the ocean, where obtaining observations requires in situ measurements or costly expeditions aboard oceanographic vessels, each costing millions of dollars (Li et al., 2023). In fields like material discovery, which aims to identify and develop new materials with desired properties, data generation requires specialized equipment and skilled labor for experimental synthesis (Nandy et al., 2022). Even on a smaller scale, conducting robotic experiments entails considerable investments of time, energy, and potential risk of accidents. Moreover, when data is obtained from physical simulators, computational expenses and slow processing times can pose significant barriers, particularly for complex systems like numerical weather prediction (Palmer, 2012) or molecular dynamics simulations (Durrant and McCammon, 2011). Consequently, learning methods must operate in a low-data regime and compensate for the scarcity of observations by incorporating physical knowledge into the model (Karniadakis et al., 2021).

Another significant limitation is the computational cost of modeling physical systems. Physical systems may be complex and large, resulting in high-dimensional estimation problems. For instance, in numerical weather prediction problems, the spatial grid used for forecasting may store a number

of variables on the order of one million (Bocquet et al., 2014). This places severe limitations on the computational budget required by any new learning method. In other applications, it is not merely the size, but rather the computation time that poses a challenge. In robotics, for example, the calculations executed by robots or other embedded systems must run in real time, at a typical operating frequency of 100 Hz, with limited computational power (Wang, 2009). As a consequence, running large-scale models becomes impractical, making it crucial to develop smaller, more computationally efficient architectures.

Traditional methods of parameter estimation, as proposed by Gauss or Legendre in the method of least squares, rely on a physical model and aim to minimize the error on the model's physical parameters. In contrast, neural network parameters have a statistical meaning but no physical interpretation: these models are referred to as "black-box" models. Consequently, it is not possible to understand the learned parameters in terms of physical quantities. While the lack of interpretability is a recognized challenge across many deep learning applications (Lipton, 2018), it is particularly critical for physical systems due to the immediate material risks (such as equipment damage or safety hazards) associated with experimental outcomes, where incorrect predictions can lead to tangible and potentially dangerous consequences. Interpreting the learned model in terms of the system's physical quantities is essential for enhancing model explainability (Linardatos et al., 2021; Grojean et al., 2022), enabling more understandable scientific discovery and making downstream model-based applications safer (Chakraborty et al., 2017; Carter et al., 2023).

Another important challenge for learning physical systems is uncertainty quantification, which is closely related to model interpretability. Given the inherent lack of interpretability of neural networks, extracting a measure of their error is complex. Yet, estimating the error of a statistical model is crucial, as any decision-making using the model requires not only the model output but also confidence intervals to estimate the prediction variability and to anticipate possible risks (Abdar et al., 2021). Moreover, estimating model uncertainty helps to overcome the challenge of data scarcity by selecting training data in the regions where model uncertainty is greatest (MacKay, 1992).

1.4. Overview of contributions

In this thesis, we explore various challenges related to the application of deep learning in physical systems. In Chapter 2, we introduce a state-space statistical model and use it to formulate some key statistical problems, which are discussed in detail in the subsequent chapters. Although these are separate problems, we will see in the next chapter that they are interconnected and together form a unified process of statistical modeling of a physical experiment.

The rise of neural networks has led to the development of neural emulators for physical systems, where neural networks either augment or entirely replace traditional physical models. However, due to the high cost of conducting physical experiments, the available data is often limited. Additionally, experimental conditions are likely to vary from one experiment to another, resulting in variability in the data distribution. To overcome the challenge of data scarcity and make the most of available data resources, it is necessary to make deep learning models robust to these environmental changes. In Chapter 3, we address this adaptation problem by incorporating physical prior knowledge into a deep learning architecture. Concerned with the interpretability issue of neural networks, we propose an interpretable method where model parameters can be connected to physical parameters. Our method is applied to learning partial differential equations and to robotics.

Another crucial aspect of learning in a low-data regime is active learning, also known as optimal experimental design. This approach involves selecting the most informative experiments during data collection to maximize learning efficiency. In Chapters 4 and 5, we study the experimental design problem in the context of dynamical systems, where it is referred to as the exploration problem. For applications to embedded systems, such as in robotics, computational resources are limited, as the

exploration algorithm must run in real-time environments. Chapter 4 addresses the experimental design problem in the simplest scenario of linear dynamic models, for which we develop an efficient, online exploration algorithm with low computational complexity. Chapter 5 extends this exploration algorithm to nonlinear models, such as neural networks. We demonstrate the application of our methods to simulated aeronautical and robotic systems, highlighting their practical implications and effectiveness.

Finally, Chapter 6 studies the data assimilation problem, which is of paramount importance in geophysics. Data assimilation involves estimating the state of a geophysical system from sparse observations, collected by various instruments such as satellites, buoys, and ground-based sensors. This estimation is achieved by finding the most likely state given the available observations and the governing physical laws. Critically, the large size of geophysical systems involves substantial computational costs. Additionally, the underlying physics, predominantly fluid dynamics, is inherently complex and computationally intensive to simulate. Consequently, data assimilation presents a great computational challenge. Neural networks, with their capability to approximate complex signals at a relatively low cost, offer promising solutions to enhance the accuracy of data assimilation while significantly reducing its computational burden. We study a neural data assimilation algorithm and evaluate its performance on simulated chaotic physical systems.

Chapter 2

Statistical estimation of physical systems

In this chapter, we establish the mathematical framework that underpins this thesis. Building on this formalism and drawing on a number of examples, we provide an overview of the different statistical learning problems addressed in this thesis. We then outline our specific contributions.

Chapter organization This chapter is organized as follows. Section 2.1 introduces the statistical framework that we will use to model physical systems. Building on this framework, Sections 2.2, 2.3, and 2.4 briefly introduce three statistical problems related to physical systems, which we address in the following chapters of this thesis, along with the associated technical challenges. Section 2.5 summarizes our contributions to these problems.

2.1. Statistical modeling framework

We propose to model physical systems using the state-space representation (Friedland, 2012), which provides a versatile framework for formulating various problems related to physical systems. This approach enables us to establish a unified mathematical foundation for addressing the diverse challenges at the intersection of physics, control theory, statistics, and machine learning.

2.1.1 State-space formalism

A physical system is represented by its state vector $x \in \mathbb{R}^d$, encompassing all the physical variables that characterize it. The state of this system depends on some experimental control variables, or input, $u \in \mathbb{R}^k$, that are chosen by the experimenter. Prior physical knowledge of the system allows us to predict the state as a function of the control with a model, up to some parameters to be determined and to some randomness. The model is formalized as a parametric function $M(u, \theta)$, with $\theta \in \mathbb{R}^n$ being the vector of parameters that can be estimated. Additionally, unmodeled effects are accounted for with a random variable $\eta \in \mathbb{R}^d$. In principle, the model may incorporate physical laws, and may be as complex as a physical simulator. Depending on the model, the randomness η may take many forms. For simplicity, we will restrict ourselves to additive model noise, and we model the state as $x = M(u; \theta) + \eta$. The assumption of additive noise may not always hold true, and we will discuss it in Section 2.1.3.

Most of the time, physical systems are only partially and indirectly observed: only part of the variables of x are measured by sensors. The observation process is designed by the experimenter, so the observations also depend on the system input. The observation process takes the form of a function $h(x, u)$ of the state and the control, with observational noise ξ. The vector of observation is then modeled as $y = h(x, u) + \xi \in \mathbb{R}^m$. Depending on the applications, the observations may be more or less sparse and noisy. We assume for simplicity that the observation function h is known, although it may be only partially known in some cases, such as remote sensing (Liang, 2005) or medical imaging (Rangayyan and Krishnan, 2024).

Our state-space statistical model then takes the following form:

$$\begin{aligned} x &= M(u; \theta) + \eta \\ y &= h(x, u) + \xi. \end{aligned} \tag{2.1.1}$$

We will illustrate our mathematical framework on several examples, from toy physical systems, where the physics is known, and whose simplicity allows us to clearly highlight our points (Reutlinger et al., 2018), to more complex real-world use cases.

Example 2.1.1 [Electric point charges and Coulomb's law] Let us model the electrostatic field created by n point charges of unknown values fixed at some known locations $\{\zeta_1, \dots \zeta_n\}$. The values of the electric charges are represented by a parameter vector $\theta \in \mathbb{R}^n$ to be estimated from measurements. The state consists of the field values everywhere in the domain Ω, which we assume to be modeled by a grid of d points: $\Omega = \mathbb{R}^d$ and $x = (x(z), z \in \Omega) \in \mathbb{R}^d$. We assume that the field is fixed throughout all the experiments with no external randomness: $\eta = 0$, so the model does not depend on any control variables nor on noise. This system is illustrated in Figure 2.1 for a two-dimensional domain: $\Omega \subset \mathbb{R}^2$ and $z = (z_1, z_2)$. According to Coulomb's law, the state is modeled as a deterministic parametric function: $x = M(\theta)$ with

$$M(\theta) = \Big(\sum_{j=1}^{n} \frac{\theta_j}{|z - \zeta_j|} \Big)_{z \in \Omega} \quad \in \mathbb{R}^d. \tag{2.1.2}$$

Our experiment consists in measuring the field at some chosen location $u \in \Omega$, which is the only control variable of our experiment. Hence, the measurement function is

$$h(x, u) = x(u) \in \mathbb{R}. \tag{2.1.3}$$

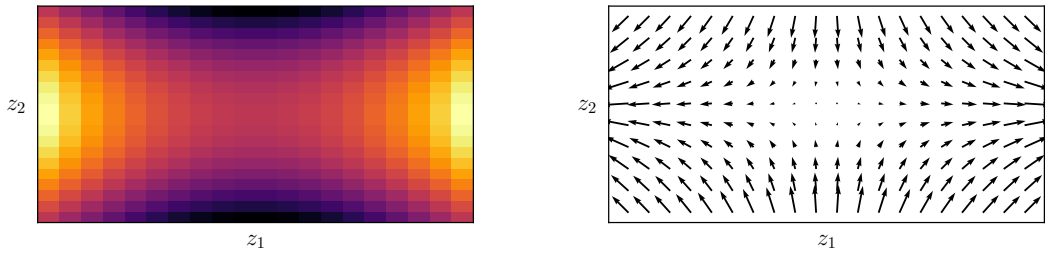

Figure 2.1. **Left** Electrostatic potential $x(z)$ generated by a set of 4 point charges, located at $\{\zeta_j = (\pm 2, \pm 1)\}$. **Right** The electrostatic vector field $-\nabla x(z)$.

The sensors are assumed to be accurate up to some precision ρ, which is modeled with Gaussian noise: $\xi \sim \mathcal{N}(0, \rho^2)$. Our experiment can then be summarized in the state-space form (2.1.1) as follows

$$\begin{aligned} x &= \Big(\sum_{j=1}^{n} \frac{\theta_j}{|z - \zeta_j|}\Big)_{z \in \Omega} && \in \mathbb{R}^{\Omega}, \\ y &\sim x(u) + \mathcal{N}(0, \rho^2) && \in \mathbb{R}. \end{aligned} \tag{2.1.4}$$

We will see in Section 2.2 how this modeling framework allows us to formulate the point charge estimation as a maximum likelihood problem, which takes the form of a linear regression as (2.1.2) is a linearly parametrized model.

Black-box models In Example 2.1.1, prior physical knowledge enables the field to be modeled by a known function, down to a few physical parameters. In deep learning, on the other hand, the idea is to model problems where little to no prior knowledge is available, and that are too complex to be described by a simple model. The system is then modeled with a "black box" model, such as a neural network, described by a large parameter vector $\theta \in \mathbb{R}^n$ (Ljung, 2010). These parameters have no physical meaning: they can be seen as cursors, which are adjusted to fit the dataset. The flexibility of such a statistical model makes it possible to model arbitrarily complex functions from data alone (Cybenko, 1989), with little prior information about the problem structure, provided that the training dataset is sufficiently large. With these powerful learning models on the table, an emerging ambition is to directly approximate complex physical systems from data, with little to no physical modeling. The resulting approximators emulating physical systems are called surrogate models, and may then be used for prediction or control tasks. We will provide a more detailed description of such models in the following section.

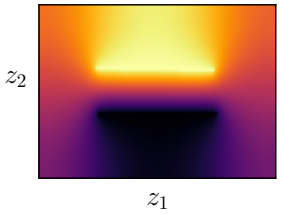

Figure 2.2. Electrostatic potential $x(z)$ created by a capacitor.

Example 2.1.2 [Laplace equation] Taking the example of electrostatics a step further, we now consider the electrostatic potential created by an arbitrary distribution of electric charges on Ω, with boundaries $\partial\Omega$ and boundary condition $b(z)$ for $z \in \partial\Omega$. According to Maxwell's equations, the electrostatic field solves the Laplace equation, yielding the boundary value problem

$$\Delta x = 0 \quad \text{on } \Omega, \qquad x(z) = b(z) \quad \text{on } \partial\Omega. \tag{2.1.5}$$

Traditionally, the field is calculated by integrating (2.1.5) numerically, using the physical knowledge of the partial differential equation. Figure 2.2 illustrates the electrostatic field created by a capacitor, and computed by the numerical integration of (2.1.5). On the other hand, a surrogate model aims to

replace the numerical integration of (2.1.5) with a complex parametric model $x(z) = f(z;\theta)$ trained on data (Brunton and Kutz, 2022). In our state-space representation, this may be described by the model

$$M(\theta) = \big(f(z;\theta)\big)_{z\in\Omega} \tag{2.1.6}$$

Unlike the physical model of (2.1.2), where $f(z;\theta)$ is a simple, well-identified function of physical parameters θ, the neural network is a complex function and its parameters θ have a purely statistical role. The lack of physical structure is compensated for by the model's high learning expressivity.

The field of scientific deep learning for partial differential equations is vast and a variety of models exist. Recent advances include neural operators and implicit neural representations, in which the geometry of the boundary $\partial\Omega$ is also taken as an input of the model, allowing for a generalization of the learning model across different geometries and different boundary conditions (Li et al., 2020; Yin et al., 2023).

2.1.2 Dynamical systems

A large class of physical systems of interest are dynamical systems, where the state changes with time. The state-space formalism is naturally extended to dynamical systems (Goodwin and Payne, 1977) as follows. The state is time-dependent and therefore the state-space model evolves with time and may be defined for each time step. We proceed in discrete time by defining the instantaneous state at time t as x_t, and the global state of the system at time t as the past trajectory $x_{0:t} = (x_0, \dots, x_t) \in \mathbb{R}^{d\times(t+1)}$, consisting of the collection of states at all past times. The time-dependent model $M_t(u_t;\theta)$ approximates the transition from one state to the next one. The state-space representation takes the form

$$\begin{aligned} x_{t+1} &= M_t(u_t;\theta) + \eta_t \\ y_t &= h_t(x_t, u_t) + \xi_t. \end{aligned} \tag{2.1.7}$$

Example 2.1.3 [Linear-Gaussian dynamical model] The simplest model for dynamical systems is arguably a time-invariant linear-Gaussian model, that is a model where the transitions from one state to the next are given by a fixed linear map, with a linear observation function and Gaussian noise:

$$\begin{aligned} x_{t+1} &= Ax_t + Bu_t + \eta_t, \quad & \eta_t &\sim \mathcal{N}(0,\Sigma) \\ y_t &= Hx_t + \xi_t, & \xi_t &\sim \mathcal{N}(0,R), \end{aligned} \tag{2.1.8}$$

where $u_t \in \mathbb{R}^k$ is the control input, $A \in \mathbb{R}^{d\times d}$ and $B \in \mathbb{R}^{d\times k}$ are transition matrices, $H \in \mathbb{R}^{m\times d}$ is the observation matrix, η_t is a Gaussian noise modeling the dynamical effects that cannot be captured by a linear map and ξ_t is a Gaussian noise modeling the sensor uncertainty. A linear dynamical model may describe nonlinear systems in the vicinity of a stable equilibrium, with the model noise accounting for unmodeled nonlinear effects. Defining $\theta := (A, B) \in \mathbb{R}^{d\times(d+k)}$, the dynamics of (2.1.8) defines a time-dependent parametric model $x_{t+1} = M_t(u_t;\theta) + \eta_t$ with $M_t(u;\theta) = Ax_t + Bu$.

Example 2.1.4 [The pendulum] The pendulum is one of physics' most famous toy systems (Figure 2.3). Although very simple, it is representative of many characteristics of the dynamical systems that we are aiming to learn. It can be seen as the basic element of manipulator robots, whose general equations are more complex but have the same form. Denoting by φ_1 the pendulum's inertia, by φ_2 its weight, and by τ the applied torque, the pendulum's angle q obeys an equation of the form

$$\varphi_1\ddot{q} + \varphi_2 \sin q + \text{friction}(q,\dot{q}) = \tau, \tag{2.1.9}$$

where $\text{friction}(q,\dot{q})$ represents the torque applied by friction forces, which are notoriously difficult to model (Marques et al., 2016). Letting $x = (q,\dot{q})$, a linear model for the pendulum dynamics can be obtained by linearizing $\sin q \simeq q$ for $|q| \ll 1$, and by modeling the unknown friction (and potentially other unknown forces) by Gaussian noise $\eta \sim \mathcal{N}(0,\sigma^2)$. Assuming that only the angle of the pendulum is measured, with some precision ρ, the resulting model is then linear-Gaussian, with $\Sigma = \sigma^2 I_d$, $H = (1,0)$, and $R = \rho^2 I_m$, and transition matrices A and B, for which we don't give explicit expressions here for simplicity.

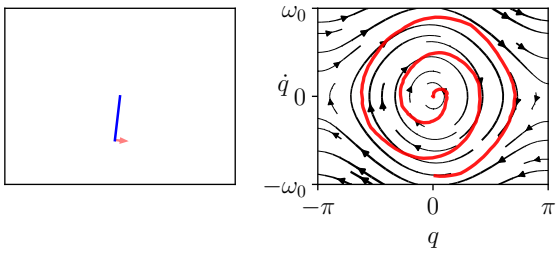

Figure 2.3. The actuated pendulum (left) and a trajectory in phase space (right).

The class of linear-Gaussian models has the advantage of being easy to estimate: as we shall see in Section 2.2 and Section 2.4, the estimation of either the parameters θ or the state x reduces to the minimization of a quadratic form. Although it can be argued that linear models are too simple to account for complex physical systems, it should be noted that this simplicity allows efficient implementation at low cost. Thus, the model's lack of complexity and its prediction errors can be compensated for by high-frequency adjustments in a feedback loop. This is how linear models find operational applications in many complex real-world systems. Examples include the celebrated Kalman filter (Kalman and Bucy, 1961), and linear model predictive control in robotics (Muske and Rawlings, 1993). Note that linear models may also be allowed to depend on time, with the so-called linear time-varying systems (Liu, 1997).

In contrast, neural networks are too complex for obtaining closed-form estimators or theoretical guarantees. However, their high approximation ability may be used to complement or even replace traditional physical models.

Example 2.1.5 [Learning nonlinear residual physics] As we mentioned in Chapter 1, physical equations accurately describe the dynamics governed by gravitational forces and inertial effects, but do not take into account certain complex phenomena such as frictional forces. To obtain a more accurate controller, we may augment a physical model with a neural network $f(x, u; \theta)$. For example, an accurate model for the pendulum of Example 2.1.4 may be obtained as

$$x_{t+1} = Ax_t + Bu_t + f(x_t, u_t; \theta), \tag{2.1.10}$$

with A and B obtained with the pendulum equation, and f modeling nonlinear phenomena such as friction. Learning nonlinear residual effects can then considerably enhance the model prediction accuracy (Zeng et al., 2020; Gao et al., 2024), and plan actions using non-convex optimization (Qin and Badgwell, 2000).

Example 2.1.6 [Atmospheric model] In meteorology, the atmosphere dynamics can be modeled in the state space form (2.1.7), with x the physical variables on a grid and, y a vector of sparse observations and M a simulator of the Navier Stokes equations. The complexity of the problem stems from the size of the state, with d potentially of the order 10^7, the low observation rate, and the high computational cost of simulating the Navier Stokes equations. Moreover, fine-scale phenomena such as cloud formation cannot be simulated at sub-mesh resolution. The physics is hence approximated by parametric models called parameterizations, which are parameters θ to be estimated. One of the major impacts of artificial intelligence is the substitution of deep learning models for part, or even all, of the physical model, within a fully learned $M(x; \theta)$. The resulting surrogate model can then be used for weather forecasting and data assimilation (Hoyer et al., 2023).

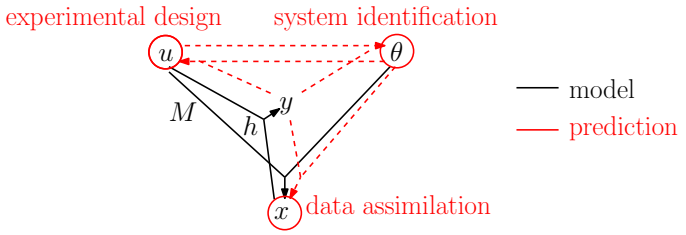

Figure 2.4. Schematic view of the state-space representation, and the interdependent statistical problems of system identification, experimental design and data assimilation.

2.1.3 Statistical model

In the remainder of this thesis, we will assume the noise to be additive, centered and Gaussian, both for the model noise and for the observational noise:

$$\begin{aligned} \eta &\sim \mathcal{N}(0, \Sigma), \\ \xi &\sim \mathcal{N}(0, R). \end{aligned} \tag{2.1.11}$$

Although the noise distribution may deviate significantly from a Gaussian, the choice of adopting the normal distribution is the most widespread and lends itself well to a wide variety of calculations, whether in physics (Stigler, 1981), in signal processing (Mallat, 1999) or in artificial intelligence for generative models (Ho et al., 2020). Arbitrary noise distributions can be modeled more finely (Legin et al., 2023), but we shall restrict ourselves to Gaussian noise for simplicity.

Under these assumptions, our state-space formulation (2.1.1) defines the following statistical model

$$\begin{aligned} \eta &\sim \mathcal{N}(0, \Sigma), \\ \xi &\sim \mathcal{N}(0, R), \\ x|u, \theta &\sim M(u; \theta) + \eta \\ y|x, u &\sim h(x, u) + \xi \end{aligned} \tag{2.1.12}$$

This probabilistic and statistical framework will enable us to mathematically pose various estimation and decision-making problems in the following sections. Specifically, Sections 2.2, 2.3 and 2.4 describe three statistical problems related to physical systems and discuss the challenges of applying deep learning to these problems. Each section follows the same structure. The first subsection introduces a mathematical formulation of the problem. The second subsection provides a fundamental example in the case of a linear-Gaussian model, where the problem can be solved analytically. The third subsection explores the extension of the problem to a nonlinear model. Finally, the fourth subsection discusses the challenges posed by integrating complex nonlinear models into these problems. The connections between these problems and the state-space representation is illustrated in Figure 2.4.

2.2. System identification

The first problem that we will consider is system identification, which aims to answer the following question.

Problem 1 [System identification] "Given a dataset of observations of a physical system $y_1, \dots, y_N$ collected from inputs $u_1, \dots, u_N$, how can we design and learn a parametric model $M(u; \theta)$ for the system that best predicts y as a function of u?"

The name "system identification" originates from the automatic control community (Ljung, 1986), and broadly describes the problem of designing and estimating a model for a physical system from

data. It is therefore a problem as old as the first experiments in estimating physical parameters, and encompasses everything from the comet trajectory estimation problem of Legendre (1806) to deep foundation models for scientific machine learning (Subramanian et al., 2024). In this section, we briefly describe the system identification problem, which we illustrate through the fundamental example of linear regression. We then discuss the challenges of learning physical systems with neural networks.

Motivation Incorporating deep learning into system identification may considerably enhance modeling capabilities for physical systems (Ljung et al., 2020). The rise of machine learning has led to a paradigm shift, where black-box models can be used to predict complex physical systems. While the learning power of neural networks is widely recognized, we have seen in Section 1 that there are still a number of limitations for deep learning models to reach an operational status. Therefore, there is still progress to be made in the field of system identification to bridge the gap between the promise of deep learning and its limitations in physical system applications.

We now turn to the mathematical formulation of the system identification problem.

2.2.1 Mathematical formulation

The goal of system identification is to design a model for the system observations y, and to estimate the model's parameters that best fit the observations. Building on the state-space representation introduced in Section 2.1, we adopt a probabilistic point of view, and we cast system identification as a maximum likelihood estimation problem.

We can define the probability of the observations conditioned on the control variable and the parameters:

$$y|u,\theta \sim p(y|u,\theta). \tag{2.2.1}$$

Following our state-space representation, the observation vector is defined as a function of the unobserved state x, which itself is a random variable. Therefore, the observation distribution can be computed by integrating over all the possible states:

$$p(y|u,\theta) = \int p(y|x,u,\theta)p(x|u,\theta)\mathrm{d}x. \tag{2.2.2}$$

We may omit the dependence on the input u when it is not explicitly involved in the calculations, and denote $p(y|\theta)$ instead of $p(y|\theta,u)$.

As is customary in statistics, the parameters of the statistical model can be inferred from a set of observations $y_{1:N} := (y_1,\dots y_N) \in \mathbb{R}^{m\times N}$, by the so-called maximum likelihood estimation principle (Cox, 2006). Specifically, assuming the statistical model (2.2.1), the maximum likelihood estimator $\hat{\theta}(y_{1:N})$ is the value of θ that makes the observations $y_{1:N}$ the most likely. We denote by $p(y_{1:N}) = p(y_1,\dots,y_N)$ the joint probability density for N experiments at observation values $y_1,\dots,y_N$. We denote by $D = \{(z_1,y_1),\dots,(z_N,y_N)\}$ the dataset of input and observation pairs.

Mathematically, the maximum likelihood estimator solves the following optimization problem

$$\underset{\theta\in\mathbb{R}^n}{\text{maximize}}\ \log p(y_{1:N}|u_{1:N},\theta). \tag{2.2.3}$$

We may formulate maximum likelihood estimation as a minimization problem by defining a loss function as the negative log-likelihood of the observations (up to constants with respect to θ):

$$\ell(\theta;D) := -\log p(y_{1:N}|u_{1:N},\theta). \tag{2.2.4}$$

We may omit the dependence on the dataset D when it is not explicitly involved in the calculations, and simply write $\ell(\theta)$. Adopting a loss minimization point of view, as is common in machine learning, the estimation of θ takes the form of the following training objective:

$$\underset{\theta\in\mathbb{R}^n}{\text{minimize}}\ \ell(\theta; D). \tag{2.2.5}$$

When (2.2.5) admits a unique solution, we shall denote it by

$$\hat{\theta}(y_{1:N}) := \underset{\theta\in\mathbb{R}^n}{\text{argmin}}\ \ell(\theta; D). \tag{2.2.6}$$

Depending on the model, it may be that (2.2.4) does not admit a unique minimizer, and hence that (2.2.6) is not well-defined. In this case we will assume that an approximate minimizer can always be found, and we will denote it by $\hat{\theta}(y_{1:N})$. We will discuss the minimizing algorithms in the next sections.

Statistical regression In many cases, the distribution of the data y given an input u can be described by a deterministic parametric function $f(z;\theta)$ of a known explanatory variable $z = (u, \dots)$ containing u and possibly other variables about the system's state. We denote this dependence as

$$y|z,\theta \sim f(z;\theta) + \nu, \tag{2.2.7}$$

where ν is independent noise. Assuming identically distributed centered noise the likelihood factorizes as

$$\begin{aligned} p(y_{1:N}|u_{1:N},\theta) &= \prod_{i=1}^{N} p(y_i|z_i,\theta) \\ &= \prod_{i=1}^{N} p_\nu(f(z_i;\theta) - y_i), \end{aligned} \tag{2.2.8}$$

where p_ν denotes the probability density function of ν. The loss function (2.2.4) is then

$$\ell(\theta) = \sum_{i=1}^{N} \log p_\nu\big(f(z_i;\theta) - y_i\big). \tag{2.2.9}$$

In the remainder, we will assume the noise ν to be normally distributed. As a consequence, the loss becomes a least-squares cost

$$\ell(\theta) = \sum_{i=1}^{N} \big(f(z_i;\theta) - y_i\big)^2, \tag{2.2.10}$$

and the maximum likelihood system identification problem (2.2.3) reduces to a least-squares regression:

$$\underset{\theta\in\mathbb{R}^n}{\text{minimize}} \sum_{i=1}^{N} \big(f(z_i;\theta) - y_i\big)^2. \tag{2.2.11}$$

Example 2.2.1 [Learning the electrostatic field from independent sensors] In Examples 2.1.1 and 2.1.2, the observation at the sensor position $z := u$ is modeled as $y = f(z;\theta) + \xi$, with the different sensors assumed to be independent and with Gaussian observational noise ξ. Hence, maximum likelihood estimation takes the form of (2.2.11).

Example 2.2.2 [Identification of dynamical systems] For dynamical systems, system identification consists in finding the transition model that best fits the trajectory observations $y_{0:T}$. Let $f(z;\theta)$ denote the model, with $z := (x, u)$ the state-action pair. Let us assume for simplicity that the states are fully observed: $y_t = x_t$, *i.e.* $h(x) = x$, with no observational noise $\xi = 0$, and scalar model error covariance $\eta_t \sim \mathcal{N}(0, \sigma^2 I_d)$. The collected observations at a given time t consist of $y_{0:t} = (x_0, \dots, x_t)$. Even though the trajectory observations are not mutually independent, we show here that system identification can be cast as a regression problem under our current assumptions. Since our physical model postulates independent additive Gaussian noise at each time step s, the distribution of a trajectory $x_{0:t}$ can be computed using the probability chain rule, as $x_{s+1}|x_s, u_s, \theta \sim f(z_s;\theta) + \eta_s$. We obtain

$$\begin{aligned} p(x_{0:t}|u_{0:t-1},\theta) &= \prod_{s=0}^{t-1} p(x_{s+1}|x_s,u_s,\theta) \\ &= \prod_{s=0}^{t-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}\left\|f(x_s,u_s;\theta) - x_{s+1}\right\|^2\right). \end{aligned} \tag{2.2.12}$$

Therefore, system identification here takes the form of an online regression problem, with the time-dependent loss function

$$\ell_t(\theta) = \sum_{s=0}^{t-1} \frac{1}{2\sigma^2}\left\|f(z_s;\theta) - x_{s+1}\right\|^2. \tag{2.2.13}$$

In the remainder of this thesis, we will adopt the notation $\hat{\theta}(z_{0:t})$ for the minimizer of (2.2.13) in the case of dynamical systems.

Minimization algorithm In this regression setting, it is a priori unclear whether a unique solution of (2.2.11) exists. In the next section, we will see that it is the case for linear models. For nonlinear models, we will discuss the popular gradient-based minimization algorithms, and how they find approximate solutions.

Model evaluation Once the parameters are estimated and $\hat{\theta}(y_{1:N})$ is found, the obtained model can be used to make predictions. We define the evaluation error of a parameter value θ as the prediction error of the model on a test dataset (z, y) as

$$\varepsilon(\theta) := \mathbb{E}\left[\|f(z;\theta) - y\|^2\right], \tag{2.2.14}$$

where (z, y) follows the data distribution and is drawn independently of the training data.

2.2.2 Linear system identification

We begin by describing the fundamental case of linear system identification, where the model is linear in its parameters: $f(z;\theta) = \theta^\top v(z)$, with $v(z)$ a known function. The least-squares loss (2.2.15) is then a quadratic cost:

$$\ell(\theta) = \sum_{i=1}^{N} \frac{1}{2}\left(\theta^\top v_i - y_i\right)^2, \tag{2.2.15}$$

where $v_i = v(z_i)$, leading to the linear regression problem

$$\underset{\theta\in\mathbb{R}^n}{\text{minimize}} \sum_{i=1}^{N}\left(\theta^\top v_i - y_i\right)^2. \tag{2.2.16}$$

Linear statistical models often arise when a certain structure of the problem is known a priori, and identifying the system comes down to estimating linear coefficients, as we illustrate in the two following examples.

Example 2.2.3 [Estimation of electric point charges] Following our state-space representation (2.1.2), we may derive the likelihood of our model for the point charge system of Example 2.1.1. We see from (2.1.2) that $y = f(u;\theta) + \xi$, with

$$f(z;\theta) = \sum_{j=1}^{n} \frac{\theta_j}{|z - \zeta_j|} \quad \in \mathbb{R}. \tag{2.2.17}$$

The regressor $f(z;\theta)$ is linear with respect to θ, and hence we may express it in the form

$$f(z;\theta) = \theta^\top v(z), \quad v(z) = (1/|z - \zeta_j|)_{1\leq j\leq n} \in \mathbb{R}^n. \tag{2.2.18}$$

Example 2.2.4 [Linear identification of dynamical systems] The dynamical system identification setting of Example 2.2.2 considerably simplifies in the case of a linear-Gaussian model, as introduced in Example 2.1.3. Recalling that $z = (x, u)$, a linear model takes the form $f(z;\theta) = \theta \times z$, with $\theta = (A, B) \in \mathbb{R}^{d\times(d+k)}$. The resulting loss takes the form

$$\ell_t(\theta) = \sum_{s=0}^{t-1} \frac{1}{2} \left\| \theta \times z_s - y_{s+1} \right\|^2. \tag{2.2.19}$$

Here again, we obtain the ordinary least squares cost (2.2.15) summed over the d rows of θ. Linear models are a popular choice for approximating dynamical systems, even though the dynamics are known to be nonlinear. System identification then consists in finding the model that best fits the observations among the class of linear models.

In the two previous examples, the linear model results in a quadratic cost function, whose minimization can be performed in closed form. In particular, the existence and uniqueness of a global minimizer are guaranteed by the convexity of the cost function. Note that such a guarantee is no longer valid when the model is not linear.

Proposition 2.1 [Ordinary least-squares estimator] Assuming that the v_i span $\mathbb{R}^n$, there exists a unique solution to linear regression problem (2.2.16). This solution is called the ordinary least squares estimator and admits the following expression

$$\hat{\theta}(y_{1:N}) := \Big(\sum_{i=1}^{N} v_i v_i^\top \Big)^{-1} \sum_{i=1}^{N} y_i v_i. \tag{2.2.20}$$

Assume that there exists $\theta_\star \in \mathbb{R}^n$ such that $y_i = {v_i}^\top \theta_\star + \xi_i$. The difference between the model parameters and the true parameters can be expressed as

$$\hat{\theta}(y_{1:N}) - \theta_\star = \Big(\sum_{i=1}^{N} v_i v_i^\top \Big)^{-1} \sum_{i=1}^{N} \xi_i v_i. \tag{2.2.21}$$

The ordinary least-squares estimator of Proposition 2.1 has a fundamental importance in statistical estimation and is used in countless scientific applications, from Gauss' and Legendre's orbit estimation problems to today's robots.

In the case where the observation distribution follows a linear model with a true parameter value $\theta_\star$, Proposition 2.1 also provides a formula comparing the least-squares estimator to $\theta_\star$. We may then examine the statistics of the estimator. In particular, the following result shows that the ordinary least-squares estimator is unbiased, and gives an expression of the parameter covariance.

Proposition 2.2 [Parameter covariance] Assume that the (z_i) are deterministic, that the observations (y_i) are distributed following $y_i = {v_i}^\top \theta_\star + \xi_i$ with $\xi \sim \mathcal{N}(0, \rho^2)$. Then $\hat{\theta}$ follows a Gaussian distribution. It is an unbiased estimator of $\theta_\star$, and the parameter covariance matrix is

$$\mathbb{E}\big[(\theta_N - \theta_\star)(\theta_N - \theta_\star)^\top\big] = \rho^2 \Big(\sum_{i=1}^N v_i v_i^\top\Big)^{-1} \quad \in \mathbb{R}^{n\times n}. \tag{2.2.22}$$

In particular,

$$\begin{aligned}\mathbb{E}[\|\theta_N - \theta_\star\|^2] &= \mathrm{Tr}\big(\mathbb{E}\big[(\theta_N - \theta_\star)(\theta_N - \theta_\star)^\top\big]\big) \\ &= \rho^2 \mathrm{Tr}\Big[\Big(\sum_{i=1}^N v_i v_i^\top\Big)^{-1}\Big].\end{aligned} \tag{2.2.23}$$

Proposition 2.2 gives a closed-form expression for the parameter error. For evaluating a linear model, we may wonder about the connection between the parameter error and the prediction error (2.2.14). The following result shows how these two mathematical quantities are related.

Proposition 2.3 [Evaluation error for a linear model] Assuming a linear model $f(z;\theta) = \theta^\top v(z)$ and an evaluation set with isotropic covariance matrix $\mathbb{E}[{z'}{z'}^\top] = \kappa^2 I_n$, the prediction error is an affine function of the parameter error:

$$\varepsilon(\theta) = \rho^2 + \kappa^2 \|\theta - \theta_\star\|^2. \tag{2.2.24}$$

Together, Propositions 2.2 and 2.3 give a closed-form formula for the prediction error (2.2.14) for linear models, thus showing that the accuracy of a linear model can be evaluated by computing the parameter error. Additionally, we will see in Section 2.3 that this formula allows us to analyze the optimal experimental design problem, that is the problem of choosing the inputs z_i that minimize the prediction error.

2.2.3 Nonlinear system identification and deep learning

While linear models guarantee a unique solution to the regression problem (2.2.11), this is not the case for more complex models. When f is no longer linear in θ, the cost function is generally non-convex, and the uniqueness of the solution no longer holds. Neural networks, in particular, are complex functions of their parameters, and it is well known that their loss landscapes are highly non-convex with numerous local minima.

Minimization algorithm In this non-convex minimization framework, it is common to use the gradient descent algorithm, or its variations to approach the minimum. Theoretical guarantees on minimization being out of reach, the practical performance of the algorithm becomes crucial for the success of system identification. In the deep learning community, stochastic versions of gradient descent are predominantly used, such as the well-known ADAM algorithm (Kingma and Ba, 2017). Interestingly, these methods can be seen as approximations of the Gauss-Newton minimization algorithm (Gauss, 1877), which was introduced in context of parameter estimation. They do so by approximating the full dataset with random samples or batch of samples at each iteration. We will not go into the details of these variations, it is important to note that they often succeed in approximating the minimization of a non-convex cost, provided the number of parameters is large enough (which

Algorithm 2.1 Gradient descent algorithm for loss minimization

input model $f(z;\theta)$, initial parameters $\theta \in \mathbb{R}^n$, learning rate γ, dataset D
output learned parameters $\bar{\theta}$
while not converged **do**
  compute $\ell(\theta; D)$
  update $\theta \leftarrow \theta - \gamma \nabla \ell(\theta)$
end while

requires a proportionally large number of data to avoid overfitting). We refer to the book of LeCun et al. (2015) for more details.

In the remainder of this thesis, we will refer to a neural network as any parametric function $f(z;\theta)$ that is differentiable with respect to its parameter vector θ and whose gradient can be computed efficiently, fitting within the gradient descent loss minimization framework of Algorithm 2.1.

2.2.4 Bayesian learning

In the previous section, we discussed the frequentist approach to system identification, where the goal is to estimate the parameters θ that best fit the observations y and input z, according to a pre-defined cost function. This approach, while efficient, provides point estimates of θ and does not directly quantify the uncertainty of these estimates.

The Bayesian approach, by contrast, treats the parameters θ as random variables and computes a posterior distribution over them based on the observations. In this framework, system identification involves both estimating the parameters and quantifying the uncertainty associated with these estimates. The prior distribution $p(\theta)$ encodes any previous knowledge or assumptions about the parameters, and the likelihood $p(y|\theta)$ describes how likely the observed data is given the parameters. The posterior distribution $p(\theta|y)$ is then obtained via Bayes' theorem:

$$p(\theta|y) = \frac{p(y|\theta)p(\theta)}{p(y)}. \tag{2.2.25}$$

Example 2.2.5 [Bayesian linear regression] For Bayesian linear regression, where the model is linear in parameters, closed-form solutions for the posterior exist under conjugate priors. Assume a Gaussian prior $\theta \sim \mathcal{N}(\mu_0, \Sigma_0)$. Adopting the notations of Section 2.2.2, and introducing

$$G_N := \sum_{i=1}^{N} v_i v_i^\top, \tag{2.2.26}$$

the parameter posterior is a Gaussian distribution

$$\theta|y_{1:N} \sim \mathcal{N}(\mu_N, \Sigma_N), \tag{2.2.27}$$

with

$$\Sigma_N = (\Sigma_0^{-1} + \frac{1}{\sigma^2} G_N)^{-1} \tag{2.2.28}$$

and

$$\mu_N = \frac{1}{\sigma^2}\Sigma_N \sum_{i=1}^{N} y_i v_i. \tag{2.2.29}$$

This approach provides not only a point estimate (typically the mean of the posterior) but also a measure of uncertainty (such as the posterior variance).

Bayesian methods are particularly useful for uncertainty quantification, as they allow for a probabilistic interpretation of the parameters, which is essential when dealing with noisy data or when the system's dynamics are not perfectly modeled. For complex physical models for which, unlike linear models, the Bayesian formulas cannot be computed in closed form, more advanced techniques exist for parameter inference such as simulation based inference (Cranmer et al., 2020).

In the context of physical systems, uncertainty quantification helps to ensure robustness and reliability in the presence of unmodeled dynamics and disturbances (Chowdhary et al., 2014). For instance, Bayesian methods can account for model mismatch and help in predicting the system's behavior even when the environment changes, as it provides an explicit measure of how confident we are in the identified parameters. A comprehensive introduction to these methods can be found in the work by Ljung (1998) on system identification, where Bayesian techniques are discussed alongside frequentist methods, and in (Peterka, 1981).

2.2.5 Challenges

The high cost of data collection on physical systems means that the amount of available data N is limited. Training neural network architectures with hundreds of millions of parameters $n \sim 10^8$, as is now common practice for language or images (Jiang et al., 2023), is then out of reach. One way of compensating for this lack of data is to incorporate prior physical knowledge into the model (Karniadakis et al., 2021).

Additionally, the interpretability of learning models and the quantification of uncertainty are crucial challenges when dealing with machine learning models. As we pointed out in Section 1, the mathematical value $f(z;\theta)$ predicted by a neural network does not include any indication of the model's certainty about the prediction. Furthermore, the neural network's parameter vector θ has no physical meaning. When dealing with surrogate models for physical systems, this can lead to a poor understanding of the network's predictions, which may be critical for downstream applications. Here again, incorporating physical knowledge and reducing the model complexity is key to improving interpretability (Ljung, 2010; Lovo et al., 2023).

Another important issue, which we will not study in this thesis, is the problem of simulation-based inference, where the physical model M is too complex for a maximum likelihood parameter to be found. This leads to challenging inverse problems, for which new numerical methods are required for inferring the model parameters (Cranmer et al., 2020).

2.3. Optimal experimental design

One of the subfields of system identification that will be of particular interest to us is that of experimental design, which aims to answer the following question.

Problem 2 [Optimal experimental design] "Given a parametric model $M(u;\theta)$ for the system's state, how to choose the inputs u that will produce the most informative data y for estimating the parameters θ?"

In the machine learning community, this problem is also referred to as active learning, active system identification or active data selection (Settles, 2009). First note that this is a decision problem rather than a learning problem: the learning model is assumed to be fixed, and the question concerns the choice of inputs u. Intuitively, measuring the system at randomly chosen points is not optimal for estimation. Instead, a strategy that samples the points where the model is uncertain should considerably improve estimation quality. The aim is to quantify the information provided by an experiment, then choose the input that maximizes this information under the constraints of the problem. In this section, we introduce the mathematical formalism of optimal experimental design, and discuss the challenges of optimal experimental design for complex learning models.

Motivation As we saw in Section 1, learning algorithms typically work in a low data regime because of the high cost of physical experiments. Therefore, it is crucial to optimize the dataset quality during the experiments. Several mathematical quantities come to mind for this purpose. In the deep learning community, model uncertainty is quantified using Bayesian methods, or by estimating model variability using ensemble techniques (Abdar et al., 2021). Although theoretically effective, these methods are computationally heavy, as they involve evaluating the model a large number of times to approximate parameter statistics. Incorporating uncertainty quantification into an experimental design optimization problem with a well-founded theoretical ground, while maintaining computational efficiency, is therefore a challenge.

We now turn to the mathematical formulation of the optimal experimental design problem.

2.3.1 Problem formulation

The goal is to find the experiments from which the trained model minimizes the evaluation error (2.2.14). We adopt a sequential point of view, where the experimenter runs a sequence of experiments, indexed by $1 \leq i \leq N$. For each experiment, an input u_i is chosen according to a policy $\Pi : y_{1:i-1}, \theta_{i-1} \mapsto u_i$ mapping past observations and the current parameter estimate to the next input. The policy models the experimenter's design strategy (Chernoff, 1972). We will assume that we can formulate the system identification problem as a regression, as in (2.2.11). The model parameters are estimated by maximum likelihood, and we assume that an algorithm is available to compute the maximum likelihood estimator $\hat{\theta} : (z_{1:i}, y_{1:i}) \mapsto \theta_i$. This estimator is a random variable, as a function of the experimental observations. Optimal experimental design aims to find the experiments providing the most informative experiments, yielding the best parameter value in terms of prediction error (2.2.14). Mathematically, this objective may be formulated as

$$\underset{\Pi \in \mathcal{P}}{\text{minimize}} \quad \mathbb{E}[\varepsilon(\theta_N)|\Pi], \tag{2.3.1}$$

where $\mathcal{P}$ is the set of possible policies, and where the expectation is an average over the experimental observations and the randomness induced by the policy. The sequential design procedure is summarized in Algorithm 2.2.

Of course, $\varepsilon(\theta)$ is an unknown function in general as it depends on a potentially complex parameterization $f(z;\theta)$ and on the true data distribution. The aim of optimal experimental design is to derive general mathematical criteria that quantify the informativeness of an experiment. As we will see in Section 2.3.3, this can be computed using the Fisher information matrix. Before turning to information-theoretic considerations, we first examine the question in the simplest learning example: linear regression.

Algorithm 2.2 Sequential experimental design

input model f, estimator $\hat{\theta}$, first estimate θ_0, policy Π, number of observations N
output estimate θ_N
for $1 \leq i \leq N$ **do**
  choose $u_i = \Pi(y_{1:i-1}|\theta_{i-1})$
  run an experiment with input u_i, observe y_i
  update $\theta_i = \hat{\theta}(z_{1:i}, y_{1:i})$
end for

2.3.2 Active linear regression

Let us consider a linear model $y = \theta^\top v(z) + \xi$, describing for example electrostatic field measurements from unknown point charges, as we have seen in Example 2.1.1. The experimenter selects N measurement points $u_1 = z_1, \ldots, u_N = z_N$, which should be chosen to make the estimation as accurate as possible, so as to minimize the final estimation error (2.3.1). We have seen in Proposition 2.3 that minimizing the evaluation error is in this case equivalent to minimizing the parameter error $\|\theta - \theta_\star\|^2$:

$$\underset{z_1,\ldots,z_N \in \mathcal{Z}}{\text{minimize}} \quad \mathbb{E}[\|\hat{\theta}(z_{1:N}, y_{1:N}) - \theta_\star\|^2] \tag{2.3.2}$$

where $\mathcal{Z}$ is the set of possible locations for the sensor. Furthermore, we see from Propositions 2.1 and 2.2 that this mean squared error may be computed in closed form for the least-squares estimator as a function of the inputs z_i. Therefore, minimizing the test error is equivalent to minimizing the variance of the maximum likelihood estimator, which may be computed from (2.2.22). Our optimal

experimental design objective (2.3.1) for linear regression then takes the following form

$$\begin{aligned} \underset{z_1,\dots,z_N \in \mathcal{Z}}{\text{maximize}} \quad & -\operatorname{Tr}\Big[\Big(\sum_{i=1}^{N} v_i v_i^\top\Big)^{-1}\Big] \\ \text{with} \quad & v_i = v(z_i). \end{aligned} \tag{2.3.3}$$

This is a standard, non-convex optimization problem, which may be solved using numerical methods (Fedorov, 2010). The optimal policy is then the policy returning the optimal inputs.

Quantifying the information with an exact formula in (2.3.3) is enabled by the properties of linear systems. Specifically, we used linearity to reduce the prediction squared error (2.2.14) to the parameter estimation error $\|\theta - \theta_\star\|^2$ in (2.3). We then used the closed forms of linear regression, and the assumption of deterministic z, to calculate the expected squared error (2.2.22) with Proposition 2.2. For a model where the regression variables z are not deterministic, Proposition 2.2 does not hold and deriving the expected parameter error may not be straightforward. For a nonlinear model, the very notion of parameter error does not necessarily make sense, as there is no guarantee that there is a "true" interpretable parameter vector $\theta_\star$ to compare with.

However, there is another interpretation to the criterion obtained in (2.3.3). The expected covariance matrix is also the inverse of the Hessian of the cost function (2.2.15), *i.e.* the negative Hessian of the log-likelihood. In principle, this quantity is easier to compute. In the next section, we will build on this observation and generalize the previous uncertainty measure to arbitrary models.

2.3.3 The Fisher information matrix

We want to measure model uncertainty in the case where the expected prediction error of (2.3.1) cannot be computed in simple form. This may happen for a linear model when the regression variables are not deterministic but have some randomness, as in the linear dynamical systems addressed in Chapter 4. In the most general scenario, no structural assumption is made on the system identification model, which may be highly nonlinear and as complex as a deep neural network. The ground-truth value of the parameter $\theta_\star$ is thus not well-defined, and computing the prediction mean squared error 2.2.14 is out of reach.

To study uncertainty and active learning in complex statistical models, we introduce some information-theoretic considerations that make no structural assumptions about the model and the data distribution. We refer to the work of MacKay (1992) for more details. Assuming a sequential decision-making algorithm as in Algorithm 2.2, the log-likelihood is approximated at each iteration i by its second-order expansion about the current maximum likelihood estimator θ_i, as

$$\log p(y|z,\theta) \simeq \log p(y|z,\theta_i) + (\theta-\theta_i)^\top \nabla_\theta \log p(y|z,\theta_i) + \frac{1}{2}(\theta-\theta_i)^\top \frac{\partial^2}{\partial\theta^2} \log p(y|z,\theta_i)(\theta-\theta_i), \tag{2.3.4}$$

where we have assumed that the likelihood function is twice differentiable. The estimator uncertainty is then measured by the local curvature of the likelihood function near the current estimate $\partial^2 p(y|z,\theta_i)/\partial\theta^2$, which is formally defined as the Fisher information matrix.

Definition 2.1 [Fisher information matrix] Given observations y and a parameter vector θ, the observed Fisher information matrix (Gelman et al., 2004) is defined as the following data-dependent matrix:

$$I(z,y;\theta) := -\frac{\partial^2}{\partial\theta^2} \log p(y|z,\theta) \quad \in \mathbb{R}^{n\times n}. \tag{2.3.5}$$

Note that $I(y;z,\theta)$ is a positive semi-definite matrix. Now, for a policy Π, the Fisher information is defined as the expected value of the observed Fisher information at the final estimate θ_N over the possible observations, assuming parameter θ and using policy Π:

$$\bar{I}(\theta,\Pi) := \mathbb{E}\left[I(z_{1:N}, y_{1:N}; \theta_N)|\theta,\Pi\right]. \tag{2.3.6}$$

Note that in the equation above, θ_N is a random variable of the observations $\theta_N = \hat{\theta}(z_{1:N}, y_{1:N})$, whose computation may be very difficult in general. In practice, this parameter value may be approximated with the current estimate θ_i at the i^{th} iteration of the sequential experimental design procedure.

Remark 2.1 [Bayesian interpretation] From a Bayesian point of view, (2.3.4) is a Gaussian approximation of the posterior distribution $p(\theta|z, y)$. The Fisher information may then be interpreted as the expected covariance matrix of this posterior distribution, whose inverse represents the posterior confidence intervals. Further details about Bayesian experimental design can be found in (Chaloner and Verdinelli, 1995).

In Figure 2.5, we provide an illustration of the observed Fisher information matrix at the maximum likelihood estimate θ_i, showing its interpretation in terms of curvature.

The Fisher information matrix gives us a natural criterion for our experimental design problem. The local curvature of the likelihood at parameter θ can be measured with a scalarization $\Phi : S_n(\mathbb{R}) \to \mathbb{R}_+$ of the Fisher information $\bar{I}$, which is a symmetric positive matrix. These scalarization may also be expressed in terms of the eigenvalues of $\bar{I}$. In optimal experimental design theory, these functions are referred to as optimality criteria. Some of these criteria are introduced in Table 2.1. We refer to the work of Pukelsheim (2006) for more details.

These information-theoretic quantities being defined, we may now state the sequential optimal experimental design problem as an information maximization. Given a parameter estimate θ, the policy is chosen to maximize the expected curvature of the likelihood function knowing θ:

$$\underset{\Pi \in \mathcal{P}}{\text{maximize}} \quad \Phi(\bar{I}(\theta, \Pi)), \tag{2.3.7}$$

where $\mathcal{P}$ is the set of possible policies. In our sequential experimental design framework of Algorithm 2.2, the model parameters are approximated with a sequence of iterates θ_i. Therefore, the policy may be refined at each iteration by solving a sequence of objectives

$$\Pi_i \in \underset{\Pi \in \mathcal{P}}{\text{argmax}} \quad \Phi(\bar{I}(\theta_i, \Pi)). \tag{2.3.8}$$

We will see applications of this sequential information-maximizing objective to controlled dynamical systems in Chapters 4 and 5.

Remark 2.2 Following the Bayesian interpretation of Remark 2.1, it can be shown that objective (2.3.7) is equivalent to maximizing the expected information gain in Bayesian optimal design (Lindley, 1956; MacKay, 1992).

Remark 2.3 From an optimization perspective, we note that the observed Fisher information (2.3.5) is also the Hessian of the training loss (2.2.4). Hence, our optimal experimental design objective also has the interpretation of seeking experiments that sharpen the loss landscape, confirming the intuition that an informative experiment discriminates between different plausible values of θ.

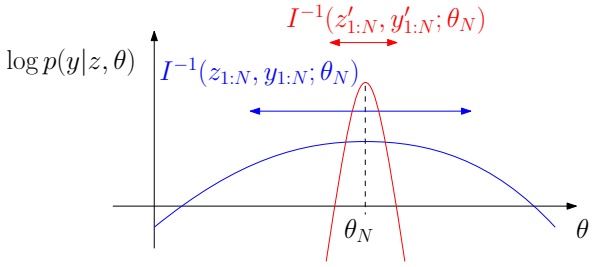

Figure 2.5. Curvature of the log-likelihood function around the maximum likelihood estimator θ_N for two sequence of experiments $(z_{1:N}, y_{1:N})$, in blue, and $(z'_{1:N}, y'_{1:N})$, in red. The second experiment yields a greater curvature of the log-likelihood, and hence is more informative for the estimation of θ.

Table 2.1. Alphabetical design criteria, seen as a function of the information matrix I, or as a function of the eigenvalues $\lambda_1, \dots, \lambda_n$ of I.

Criterion	$\Phi(I)$	$\Phi(\lambda_1, \dots, \lambda_n)$
A-optimality	$-\mathrm{Tr} I^{-1}$	$-\left(1/\lambda_1 + \dots + 1/\lambda_n\right)$
D-optimality	$\log\det I$	$\log\lambda_1 + \dots \log\lambda_n$
E-optimality	$\min\lambda(I)$	λ_1

The optimal experimental design problem (2.3.7) gives us an optimization objective that is valid for an arbitrary parametric model, and independent of the model and of any knowledge regarding the distribution of (z, y), unlike (2.3.1). We will see in the following example that this is a generalization of the linear experimental design problem (2.3.3) to nonlinear parameterizations.

Example 2.3.1 [Active linear regression] Back to our linear regression model, we recall that the log-likelihood of the observations is

$$\log p(y_{1:N}|z_{1:N}, \theta) = \sum_{i=1}^{N} \frac{1}{2\rho^2} \left(\theta^\top z_i - y_i\right)^2 - N\sqrt{2\pi\rho^2}, \tag{2.3.9}$$

from which we obtain

$$I(z_{1:N}, y_{1:N}; \theta) = \frac{1}{\rho^2} \sum_{i=1}^{N} z_i z_i^\top. \tag{2.3.10}$$

For a linear model, we note that the observed Fisher information depends neither on the parameter θ, nor on the observation values $y_{1:N}$. We also see that the information matrix coincides with the inverse of the parameter covariance matrix (2.2.22), *i.e.* the precision matrix. Therefore, the optimal experimental problem coincides with (2.3.3), with Φ the A-optimality criterion. This is hardly surprising, since in the case of an exactly linear model, approximation (2.3.4) becomes exact and the Fisher matrix is therefore exactly the posterior precision matrix. Importantly, (2.3.9) shows that we have a closed-form expression for our experimental design objective in the case of linear models, even in the case where the covariates z_i are random variables.

Finally, let us mention that, while computing the parameter covariance is in general out of reach for nonlinear models and the equality between Fisher information and the estimator covariance is no longer true, there is still an inequality between these two quantities, known as the Cramér-Rao bound (Cramér, 1999).

2.3.4 Bayesian experimental design

In the previous section, we introduced experimental design from a frequentist point of view, where the goal is to choose the experimental inputs u_i that maximize the Fisher Information Matrix in order to minimize the uncertainty in the parameter estimates θ. This approach relies on the assumption that we are designing the experiment to gather the most informative data under a given model, treating the parameters as fixed but unknown.

The Bayesian approach to experimental design offers a more flexible framework by incorporating prior knowledge about the parameters θ in the design process. In Bayesian experimental design, the goal is to optimize the choice of inputs not just to minimize the posterior uncertainty about the parameters, but to balance this uncertainty with prior beliefs about the parameters.

The central idea is to maximize the expected information gain, measured as the difference between the prior and the posterior distributions over θ, as opposed to relying solely on the Fisher information

matrix, which is based on point estimates. This allows for a more global exploration of the parameter space and incorporates uncertainty in both the data and the model.

More formally, in Bayesian experimental design, we aim to select the experimental conditions u that maximize the expected utility $\mathrm{EI}(u)$ with respect to the posterior distribution. A common utility function is the Kullback-Leibler divergence between the prior $p(\theta)$ and the posterior $p(\theta|y)$, which quantifies how much information the experiment is expected to add about θ. This can be expressed as:

$$\mathrm{EI}(u) = \mathbb{E}\left[D(p(\theta|y,u), p(\theta))\right], \tag{2.3.11}$$

where D is the Kullback-Leibler divergence (Kullback, 1997), and the expectation is taken over the possible observations y.

Example 2.3.2 [Bayesian experimental design for linear models] When the data model is linear, as in Example 2.2.5, we may compute the expected utility and find

$$\mathrm{EI}(u) = \frac{1}{2}(\log\det\Sigma_N - \log\det\Sigma_0). \tag{2.3.12}$$

Using the posterior formula for Σ_N, we observe that maximizing this quantity is equivalent to D-optimal design in the frequentist framework.

Bayesian experimental design can be interpreted as an extension of frequentist optimal experimental design, where the prior covariance is combined with the information gained from the experiment. Chaloner and Verdinelli (1995) provide a comprehensive introduction to Bayesian experimental design, including its theoretical foundations and applications.

2.3.5 Challenges

Modern methods for Bayesian optimal experimental design express the information gain brought by an experiment as an expectation over the possible outcomes. The resulting objective is approximated with nested Monte Carlo sampling techniques, presenting severe computational issues (Rainforth et al., 2024). The approach that we presented in this section is based on the Fisher information may be seen as a frequentist counterpart to Bayesian experimental design (Cox, 2006), and has the advantage of being more straightforward to compute.

Yet, computing and optimizing the Fisher information for arbitrary regression models is still challenging. Unlike linear regression, where the model uncertainty can be expressed in closed form, nonlinear models make the computation of (2.3.5) more difficult. Indeed, the latter requires second-order differentiation, which may be prohibitive for complex models such as neural networks. Moreover, the randomness induced by the physical system may also make the expectation of (2.3.6) non-trivial. Additionally, there is the question of solving the non-convex optimization problem (2.3.7), which can be computationally very costly. In physical systems, these challenges may be compounded by the small computational budget imposed by some applications, such as embedded systems, where all calculations of Algorithm 2.2 must be carried out in real time.

Given these computational constraints, it is crucial to find efficient and realistic approximations for calculating model uncertainty, and to obtain an efficient iterative experimental design policy following Algorithm 2.2. We will see applications to dynamical systems in Chapter 4 and in Chapter 5.

2.4. State estimation and data assimilation

Finally, the last statistical problem that we will consider is the state estimation problem.

Problem 3 [State estimation] "Given a model M encoding prior information of the system's physics, and partial and sparse observations y of an unknown state vector x, how to recover the most likely estimation of the state $\hat{x}(y)$?"

In geosciences, this problem is known as data assimilation (Bouttier and Courtier, 2002). In the control community and robotics, it is referred to as state estimation or filtering (Särkkä and Svensson, 2023). It is connected to many interesting mathematical problems, such as observability (Levin and Narendra, 1996; Bernard et al., 2022) and uncertainty quantification (D'Elia and Veneziani, 2013). In this section, we introduce a mathematical formulation of the data assimilation problem, which we illustrate with the simple example of Gaussian quadratic interpolation. We then discuss the applications and challenges of deep learning for data assimilation.

Motivation State estimation is of paramount importance for many physical applications. For any forecasting problem, such as numerical weather forecasting (Simmons et al., 1989) or model-based control (Tedrake, 2022), an accurate prediction requires a good estimate of the starting point: tomorrow's weather forecast is based on today's weather conditions, which should hence be estimated as accurately as possible. The state estimation problem is considerably hard to analyze for nonlinear models, and the numerical methods for solving it may be computationally heavy. As neural networks are becoming an increasingly popular tool to model physical systems with high generative capabilities (Ongie et al., 2020), they may become a powerful tool to recover a physical signal from sparse observations at low cost.

We now turn to the mathematical formulation of data assimilation problem.

2.4.1 Problem formulation

The aim of data assimilation is to reconstruct a state $x \in \mathbb{R}^d$ from partial noisy measurements $y \in \mathbb{R}^m$ of that state (Bouttier and Courtier, 2002; Bocquet et al., 2014). As the measurements may be sparse, we cannot generally hope to recover the state as a function of the data alone. Indeed, for a given observation vector y, a large number of states are compatible, making data assimilation an inverse problem. To reconstruct the state, we need to supplement the partial observations with another source of prior information on the state, which comes from our physical or statistical knowledge of the problem.

The data assimilation problem is then as follows. Given partial observations y and prior information on the state, the aim is to estimate the most probable underlying state x. Using our state-space formulation, the Bayesian probabilistic framework lends itself well to the mathematical formalization of the problem. We assume that the theoretical information about the physics is captured by a prior distribution on the state $x \sim p(x)$, which me for example depend on the state model M introduced in Section 2.1. We recall that the noisy, partial observations of x are modeled as $y|x \sim h(x) + \xi$, with a known observation process h and a Gaussian additive noise $\xi \sim \mathcal{N}(0, R)$ independent of x. Then, data assimilation can be seen as the estimation of the state maximizing the state posterior distribution $p(x|y) = p(x)p(y|x)/p(y)$, that is

$$\underset{x \in \mathbb{R}^d}{\text{maximize}} \ \log p(x|y). \tag{2.4.1}$$

Let $U(x) = -\log p(x)$. Applying Bayes' formula and taking the negative logarithm, the maximum a posteriori estimation problem (2.4.1) takes the form

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \ U(x) + \frac{1}{2}\|h(x) - y\|^2_{R^{-1}}, \tag{2.4.2}$$

where we have adopted the notation $\|z\|_B = z^\top B^{-1} z$ for a positive definite matrix B.

2.4.2 Quadratic least-squares estimation

The first approach considered for data assimilation is naturally that of a linear-Gaussian model, where computations may be carried out in closed form. Assuming a Gaussian prior on the state $x \sim \mathcal{N}(\mu, P)$

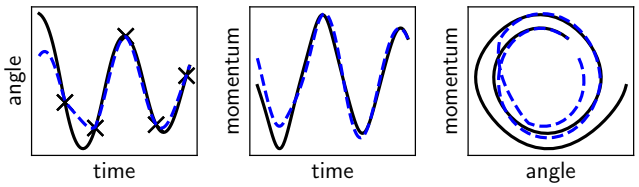

Figure 2.6. Nonlinear pendulum trajectory (black) reconstructed (blue) from sparse angular observations (×).

and a linear observation function $h(x) = Hx$, with $H \in \mathbb{R}^{m\times d}$, the variational Bayesian formulation for data assimilation (2.4.1) becomes a quadratic least-squares problem:

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad \frac{1}{2}\|x-\mu\|^2_{P^{-1}} + \frac{1}{2}\|Hx-y\|^2_{R^{-1}}, \tag{2.4.3}$$

whose solution may be computed in closed form as follows.

Proposition 2.4 [Optimal quadratic least-squares interpolation] The maximum a posteriori solution of (2.4.3) takes the form

$$\hat{x}(y) := \mu + K(y - H\mu), \tag{2.4.4}$$

with the Kalman gain

$$K = (P^{-1} + H^\top R^{-1} H)^{-1} H^\top R^{-1} \ \in \mathbb{R}^{m\times d}, \tag{2.4.5}$$

which is also equal to

$$K = PH^\top (HPH^\top + R)^{-1}. \tag{2.4.6}$$

The quadratic least-squares estimator formula (2.4.4) holds for a Gaussian prior distribution, which may seem restrictive from a deep learning point of view: unlike complex deep generative models, Gaussian distributions cannot capture complex multimodal signal distributions, such as natural image distributions (Ruderman and Bialek, 1993). However, Gaussian distributions can still capture quite complex behaviors in the case of dynamical systems, where the so-called linear Gaussian models gave rise to widely used state estimation algorithms such as the celebrated Kalman filter (Kalman and Bucy, 1961), with a wide spectrum of applications, ranging from robotics to meteorology (Tedrake, 2022; Bouttier and Courtier, 2002). For nonlinear systems, the linear-Gaussian framework may be applied sequentially by linearizing the model (Talagrand and Courtier, 1987; Tassa et al., 2014). Many examples could be presented to demonstrate the power of linear quadratic models. The following illustrates the smoothing problem for linear dynamical system.

Example 2.4.1 [Kalman smoothing] Let us assume a linear dynamical model as introduced in Example 2.1.3, with Gaussian model error $\eta_t \sim \mathcal{N}(0,\Sigma)$, with $x_0 = 0$ and $u_t = 0$ for simplicity, and with a fixed model parameter $\theta \in \mathbb{R}^{d\times d}$. Then the trajectory distribution $p(x_{0:T})$ is Gaussian. Indeed, by the probability chain rule,

$$p(x_{0:T}) = \prod_{t=0}^{T-1} \frac{1}{\sqrt{2\pi|\Sigma|}} \exp\left(-\frac{1}{2}\,\|\theta\times x_t - x_{t+1}\|^2_{\Sigma^{-1}}\right). \tag{2.4.7}$$

We see that the log-likelihood of the trajectory distribution $\log p(x_{0:T})$ is a quadratic form, hence showing that $x_{0:T}$ is normally distributed. Therefore, a temporal signal on a fixed time interval $x := x_{0:T}$ assumed to follow linear dynamics may be reconstructed from linear observations $y = Hx_{0:T}$ with the closed-form formula of Proposition (2.4). The state posterior estimation in this linear-quadratic dynamical setting is called Kalman smoothing (Särkkä and Svensson, 2023). Figure 2.6 shows the result of Kalman smoothing for a nonlinear pendulum for a full trajectory from sparse observations, with the Gaussian prior obtained from the linearized pendulum dynamics.

2.4.3 Bayesian state estimation

In classical state estimation, the focus is often on finding the most likely state x that minimizes a cost function based on the observations y, such as in maximum likelihood estimation. This approach results in a point estimate of the state, ignoring the uncertainty associated with the estimate.

In contrast, Bayesian state estimation provides a probabilistic framework where the goal is not to find a single best estimate, but rather to sample from the posterior distribution $p(x|y)$ which incorporates both the data and prior knowledge about the state. This approach yields a full distribution over possible states, allowing us to estimate not only the most probable state but also the uncertainty and credibility intervals around that state. For climate applications, this is particularly useful as the nonlinearity of the underlying chaotic systems yields complex multimodal posterior distributions (Epstein and Fleming, 1971).

For linear systems with Gaussian noise, the Kalman filter provides an optimal Bayesian estimate by producing a Gaussian posterior distribution for the state, as seen in Example 2.4.1. When the system is non-linear, the standard Kalman filter becomes inadequate. In such cases, extensions like the Ensemble Kalman Filter (Evensen, 2003) use an ensemble of state estimates to approximate the posterior distribution, making it a computationally efficient alternative for large-scale systems (Bocquet et al., 2014). However, these methods rely on Gaussian assumptions that may not hold for more complex systems. In such cases, particle filters or other sampling-based Bayesian methods offer more flexibility, as they make fewer assumptions about the distribution of the state and can more accurately capture non-linearities and non-Gaussian behaviors. For this framework, we refer to the comprehensive study of Särkkä and Svensson (2023)

2.4.4 Challenges

For geophysical applications, the state x represents the geophysical variables on a large spatial grid. It is hence a signal of high dimension with typically $d \sim 10^6$ or even $d \sim 10^9$. Additionally, the underlying physical equations are nonlinear. As a consequence, the prior regularization term $U(x)$ is highly non-convex, and costly to both evaluate and differentiate through, as it involves a computationally heavy physical simulator. Fore these reasons, efficiently finding a maximum likelihood solution to (2.4.3) is computationally challenging.

In the control community, the state estimation problem for nonlinear models is widely studied from a mathematical point of view, with the challenge of designing nonlinear observer (Bernard et al., 2022; Buisson-Fenet et al., 2023) with theoretical convergence guarantees.

Deep neural networks hold great promise for solving inverse problems (Bai et al., 2020), as they can help recover the corrupted signal by using the large amount of statistical information acquired on a training dataset. For the data assimilation problem in meteorology or oceanography, the ground truth signals x are not available as the geophysical systems are not fully observed. However, a promising research direction consists in training a deep neural network to learn a prior on high-resolution simulations, or on reanalysis datasets such as ERA5 (Muñoz-Sabater et al., 2021), like neural weather models (Ben Bouallègue et al., 2024), and then to use the learned prior on real observations in a “simulation-to-reality" fashion.

2.5. Summary of contributions

Now that we have introduced a statistical framework and various mathematical problems of interest, we provide an overview of our contributions to these problems, which are presented in the following chapters of this thesis.

2.5.1 Adaptation and multi-environment system identification

Chapter 3 addresses the system identification problem, Problem 1, as described in Section 2.2. We study the particular setting where observations have been collected under inhomogeneous experimental conditions. This situation frequently arises with physical systems, where certain physical parameters may vary from one experiment to another. As a consequence, the assumption of identically distributed data formulated in (2.2.7) must be challenged, and it is necessary to develop learning models that are robust to slight changes in the data distribution.

Setting Adopting a supervised learning framework as in (2.2.11), the observations $D = \{(z_i, y_i)\}_i$ come from different datasets $D_1, \dots, D_T$, each of which has been collected on the same system with the same underlying physics, but with different physical parameters $\varphi_1, \dots, \varphi_T$. The aim is to train a parametric model by taking advantage of all available data, and therefore to design an architecture that takes accounts for the variability of the physical system. Several approaches have been proposed in the deep learning community through the prism of meta-learning (Hospedales et al., 2021). The changes in experimental conditions are modeled as a two-fold parametric model $F(z; \theta, w)$, where w models the experimental variability. Extending the maximum likelihood estimation formulation of Section 2.2, system identification consists in minimizing the following training loss:

$$L(\theta) = \sum_{t=1}^{T} \sum_{(z,y) \in D_t} \big(F(z; \theta, w_t) - y\big)^2. \tag{2.5.1}$$

Challenge Black-box meta-learning approaches such as the MAML algorithm (Finn et al., 2017), and its variations applied to physical systems (Yin et al., 2021; Kirchmeyer et al., 2022; Park et al., 2023), proceed by modeling environment variations as an additive correction in the parameter space of a neural network $f(z; \theta)$, as $F(z; \theta, w) = f(z; \theta + w)$. Because the parameter space of a neural network is not interpretable, finding a connection between the learning parameters w and the physical parameters φ is challenging. Additionally, this model poses computational challenges at both training time and inference time, as the dependence of w as a function of θ may be complex.

Contribution Unlike black-box deep learning approaches that make no assumptions about the data distribution, we leverage the physical structure to learn the system with a simpler and interpretable model of the form $F(z; \theta, w) = w^\top v(z; \theta)$, where $v(z; \theta)$ is a neural network. We show that, by mimicking the structure of physical equations, this parameterization allows both for efficient learning and interpretability. We propose applications in parameter identification and adaptive control.

2.5.2 Active identification of linear dynamical systems

In Chapter 4, we study the optimal experimental design problem, Problem 2, as introduced in Section 2.3. We focus on the specific case of dynamical systems. In the reinforcement learning community, this problem is also referred to as the active exploration problem (Shyam et al., 2019). The aim is to design an experimental design exploration policy that drives the system towards highly informative states, so that the model parameters are estimated as fast as possible. Applications include aeronautics and robotics, where system parameters need to be identified in real time (Gupta et al., 1976; Spong et al., 2020).

Setting We focus on fully observed dynamical systems, learned with a linear dynamics model, as introduced in Example 2.1.3. As seen in Example 2.2.4, the transition function is modeled as $f(z; \theta) = \theta \times z$, with $z = (x, u)$ and $\theta = (A, B)$. For such a model, we have seen that the trajectory likelihood is

$$\log p(x_{0:t} | u_{0:t-1}, \theta) = -\frac{1}{2\sigma^2} \sum_{s=0}^{t-1} \|\theta \times z_s - x_{s+1}\|^2 - t \log(\sqrt{2\pi\sigma^2}). \tag{2.5.2}$$

In this dynamical setting, the experimental design policy of Algorithm 2.2 runs in real time: the model is learned online using the ordinary least squares estimator $\hat{\theta}$ of (2.2.20), and the control returned by the policy $u_t = \Pi(z_{0:t-1}|\theta_{t-1})$ is played at the next time step.

Challenge Two challenges arise for exploration in such dynamical systems. First, the interdependence between the trajectory states precludes explicit computation of the prediction error objective (2.3.1). Indeed, we see in (2.5.2) that the observations z_s are not fixed, as z_{s+1} is a function of the dynamics and the past trajectory $z_{0:s}$. Therefore, Proposition 2.2 does not apply and the linear optimal experimental design theory introduced in Section 2.3.2 may not be directly used. Second, the experimental design procedure of Algorithm 2.2 should run online at high frequency in real-life applications. Hence, the decision-making policy $\Pi : x_{0:t}, u_{0:t} \mapsto u_t$ is constrained to a small computational budget, making nonlinear optimization methods such as gradient-based approaches unlikely to fit. Several active system identification algorithms for dynamical systems have been recently proposed in the machine learning community (Wagenmaker et al., 2021; Mania et al., 2020). Although these algorithms are provided with optimality guarantees and theoretical bounds, their computational cost make them rather impractical for this online setting.

Contribution Resorting to information theory, we derive the Fisher information matrix and propose a theoretically grounded experimental design objective in this dynamical setting, in the form of (2.3.8). Mindful of the real-world constraints of physical systems, we compute a greedy approximation of the Fisher information and devise an information-maximizing algorithm that runs online with little computational burden. We compare its sample complexity and its computational cost with more elaborate gradient methods on simulated physical systems.

2.5.3 Exploration of nonlinear systems

Following on from Chapter 4, Chapter 5 studies Problem 2 for nonlinear models. This chapter aims to extend our adaptive exploration algorithm from linear dynamics models, where the model is linear in both the state and the parameter, to arbitrary nonlinear models. While linear dynamics can describe many systems locally and are computationally inexpensive, the number of systems that they can learn is limited: complex phenomena such as friction are known to be nonlinear (Marques et al., 2016). On the contrary, recent advances in deep learning have demonstrated that neural networks provide excellent approximations of nonlinear functions (Csáji et al., 2001; Robin et al., 2022), and can be trained efficiently. It is therefore natural to attempt to generalize the experimental design problem of Section 2.3 to nonlinear models.

Setting The problem is the same as in Chapter 4, generalized to arbitrary parametric models $f(z;\theta)$ for the dynamics. We have seen in Example 2.2.2 that the log-likelihood of a trajectory $z_{0:t}$ is

$$\log p(x_{0:t}|u_{0:t-1},\theta) = -\frac{1}{2\sigma^2}\sum_{s=0}^{t-1}\|f(z_s;\theta) - x_{s+1}\|^2 - t\log(\sqrt{2\pi\sigma^2}), \tag{2.5.3}$$

where f may be as complex as a deep neural network.

Challenge Unlike the linear dynamics setting of Chapter 4, we have seen in Section 2.3.3 that the Fisher information matrix depends on θ, and cannot be computed in closed form as a function of the trajectory. Deriving the information gain while maintaining low computational complexity therefore requires further approximations. Recent works from the statistical learning community have proposed bounds for the estimation error in this exploration problem (Mania et al., 2020), but with restrictive assumptions and little practical applicability. In the reinforcement learning community, parameter uncertainty for nonlinear models is either modeled by ensembling (Shyam et al., 2019) or by optimizing

an experimental design objective with nonlinear programming (Buisson-Fenet et al., 2020; Schultheis et al., 2020). These approaches assume extensive computational resources and may be too slow to run online in embedded systems.

Contribution Building on the work of MacKay (1992), we generalize the optimal experimental design approach of Chapter 4 to nonlinear exploration by considering the linearized parametric model. We thus generalize our online policy to nonlinear models, while maintaining a low computational cost. We compare our policy's performance with state-of-the art model-based and model-free exploration algorithms.

2.5.4 Neural data assimilation

Chapter 6 addresses the state estimation problem, Problem 3, as defined in Section 2.4. We propose a deep learning method to the data assimilation problem.

Setting In geophysics applications such as meteorology, the state is a spatio-temporal signal $x = x_{0:T}$. Assuming a parametric transition model $f(x;\theta)$ and Gaussian model noise, the prior distribution takes the form

$$p(x_{0:T}) = \prod_{t=0}^{T-1} \frac{1}{\sqrt{2\pi|\Sigma|}} \exp\left(-\frac{1}{2}\,\|f(x_t;\theta) - x_{t+1}\|^2_{\Sigma^{-1}}\right). \tag{2.5.4}$$

A maximum a posteriori state estimate is then computed by solving (2.4.2) numerically.

Challenge The state representing the spatio-temporal physical signal over a fixed time interval $x_{0:T} \in \mathbb{R}^d$ is of potentially high dimension: in real-world applications, d may be of order 10^9. Moreover, the model f typically corresponds to simulating the underlying physical equations (such as the Navier-Stokes equations), which are costly to compute and lead to a complex posterior distribution. State-of-the-art data assimilation methods such as 4D-Var (Le Dimet and Talagrand, 1986) proceed by solving (6.2.1) using gradient-based methods, hence requiring heavy computations of a high-dimensional state and differentiating through complex physical simulations. Recently, the success of diffusion models for imaging (Ho et al., 2020) has led to the development of so-called "plug and play" methods, where the neural network is trained to learn a prior (Laumont et al., 2022). Once trained, the neural prior can be used to solve a large number of inverse problems. In this line of work, Rozet and Louppe (2023) proposed a data assimilation method based on a diffusion model. Another type of approach called "end-to-end" aims to directly train a neural network to minimize the reconstruction error. They have the benefit of training the network directly on the task of interest, but the versatility of the trained model with respect to different observational processes is challenging.

Contribution Our aim is to develop an "end-to-end" neural data assimilation method where the expensive physical model is replaced with a trained neural network. Specifically, our goal is to learn the $y \mapsto \hat{x}(y)$ mapping from data. We train a neural model $A(y;\theta)$ with an objective of the form

$$\underset{\theta\in\mathbb{R}^n}{\text{minimize}} \quad \sum_{i=1}^{N} \frac{1}{2}\|A(y^{(i)};\theta) - x^{(i)}\|^2, \tag{2.5.5}$$

where $(x^{(i)})_i$ and $(y^{(i)})_i$ denote simulation data, and we explore how to model such an assimilation model to be versatile with respect to arbitrary observation processes H. We study how such an algorithm may improve the performance of 4D-Var while reducing its computational cost.

Chapter 3

Interpretable meta-learning of physical systems

Machine learning methods can be a valuable aid in the scientific process, but they need to face challenging system identification settings where data come from inhomogeneous experimental conditions. Recently, meta-learning approaches have made significant progress in system identification within a multi-task learning scenario, but they rely on black-box neural networks, resulting in high computational costs and limited interpretability. Leveraging the structure of the learning problem, we argue that multi-environment generalization can be achieved using a simpler learning model, with an affine structure with respect to the learning task. Crucially, we prove that this architecture can identify the physical parameters of the system, enabling interpretable learning. We demonstrate the competitive generalization performance and the low computational cost of our method by comparing it to state-of-the-art algorithms on physical systems, ranging from toy models to complex, non-analytical systems. The interpretability of our method is illustrated with original applications to physical-parameter-induced adaptation and to adaptive control.

Chapter organization This chapter is organized as follows. Section 3.1 presents the multi-environment learning problem and its challenges. Section 3.2 poses the statistical framework of this problem, along with examples. Section 3.3 introduces CAMEL, a meta-learning algorithm for the multi-environment learning problem. Section 3.4 studies the interpretability of CAMEL. Section 3.5 evaluates the performance of CAMEL with numerical experiments on various physical systems. Section 3.6 compares our contribution with related works. Section 3.7 summarizes our contributions and discusses its limitations and perspectives.

This chapter is based on the article *Interpretable Meta-Learning of Physical Systems* (Blanke and Lelarge, 2024), published in the Twelfth International Conference on Learning Representations (ICLR 2024).

3.1. Introduction

Learning physical systems is an essential application of artificial intelligence that can unlock significant technological and societal progress. Physical systems are inherently complex, making them difficult to learn Karniadakis et al. (2021). A particularly challenging and common scenario is multi-environment learning, where observations of a physical system are collected under inhomogeneous experimental conditions Caruana (1997). In such cases, the scarcity of training data necessitates the development of robust learning algorithms that can efficiently handle environmental changes and make use of all available data.

This multi-environment learning problem falls within the framework of multi-task learning, which has been widely studied in the field of statistics since the 1990s (Caruana, 1997). The aim is to exploit task diversity to learn a shared representation of the data and thus improve generalization. With the rise of deep learning, several meta-learning approaches have attempted in recent years to incorporate multi-task generalization into gradient-based training of deep neural networks. In the seminal paper by Finn et al. (2017), and several variants that followed (Zintgraf et al., 2019; Raghu et al., 2020), this is done by integrating an inner gradient loop in the training process. Alternatively, Bertinetto et al. (2019) proposed adapting the weights using a closed-form solver. As far as physical systems are concerned, the majority of the proposed methods have focused on specific architectures oriented towards trajectory prediction (Wang et al., 2022a; Kirchmeyer et al., 2022).

When learning a physical system from data, a critical yet often overlooked challenge is model interpretability (Lipton, 2018; Grojean et al., 2022). Interpreting the learned parameters in terms of the system's physical quantities is crucial to making the model more explainable, allowing for scientific discovery and downstream model-based applications such as control. In a multi-task learning setting, the diversity in the learning environments should enable the identification of the physical parameters that vary across the tasks.

The above approaches benefit from the expressiveness of deep learning, but are costly in terms of computational time, both for learning and for inference. Furthermore, the complexity and the black-box nature of neural networks hinder the interpretability of the learned parameters, even when the physical system is linearly parametrized. Recently, Wang et al. (2021) showed theoretically that the learning capabilities of gradient-based meta-learning algorithms could be matched by the simpler architecture of multi-task representation learning with hard parameter sharing, where the heads of a neural network are trained to adapt to multiple tasks (Caruana, 1997; Ruder, 2017). They also demonstrated empirically that this architecture is competitive against state-of-the-art gradient-based meta-learning algorithms for few-shot image classification. We propose to use multi-task representation learning for physical systems, and show how it can bridge the gap between the power of neural networks and the interpretability of the model, with minimal computational costs.

Contributions In this chapter, we study the problem of multi-environment learning of physical systems. We model the variability of physical systems with a multi-task representation learning architecture that is affine in task-specific parameters. By exploiting the structure of the learning problem, we show how this architecture lends itself to multi-environment generalization, with considerably lower cost than complex meta-learning methods. Additionally, we show that it enables identification of physical parameters for linearly parametrized systems, and local identification for arbitrary systems. Our method's generalization abilities and computational speed are experimentally validated on various physical systems and compared with the state of the art. The interpretability of our model is illustrated by applications to physical parameter-induced adaptation and to adaptive control.

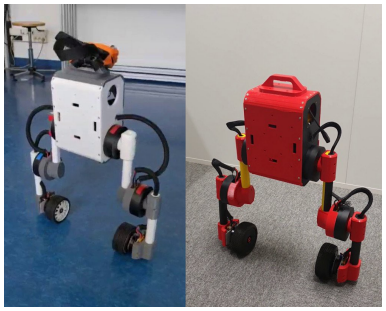

Figure 3.1. Upkie, a wheeled biped robot, under different experimental conditions. Credit: Stéphane Caron.

3.2. Learning from multiple physical environments

In this section, we present the problem of multi-task learning as it occurs in the physical sciences, and we summarize how it can be tackled with deep learning in a meta-learning framework.

3.2.1 The variability of physical systems

In general, a physical system is not fixed from one interaction to the next, as experimental conditions vary, whether in a controlled or uncontrolled way. From a learning perspective, we assume a meta-dataset $D := \cup_{t=1}^{T} D_t$ composed of T datasets, each dataset gathering observations of the physical system under specific experimental conditions. This setting challenges the assumption of identically distributed observations that is usually postulated in system identification (Ljung, 1986), hence it significantly deviates from the traditional regression problem as formulated in Section 2.2 of Chapter 2. The goal is to learn a predictor from D that is robust to task changes, in the sense that when presented a new task, it can learn the underlying function from a few samples (Hospedales et al., 2021).

For simplicity, we assume a classical supervised regression setting where $D_t := \{z_i^{(t)}, y_i^{(t)}\}_{1 \leq i \leq N_t}$ and the goal is to learn a $z \mapsto y$ predictor, although the approaches presented generalize to other settings such as trajectory prediction of dynamical systems. Note that in practice the number of tasks T is typically very limited, owing to the high cost of running physical experiments. We discuss two physical examples illustrating the need for multi-task learning algorithms, with different degrees of complexity.

Example 3.2.1 [Actuated pendulum] We begin with the simple pendulum, one of physics' most famous toy systems, already introduced in Example 2.1.4 of Section 2.1. Denoting its inertia and its weight by φ_2 and φ_2 and the applied torque by τ, the angle q obeys

$$\varphi_1 \ddot{q} + \varphi_2 \sin q = \tau. \tag{3.2.1}$$

In reinforcement learning and control, for example, we may want to learn the action $y = \tau$ as a function of coordinates $z = (q, \dot{q}, \ddot{q})$. In a data-driven framework, the trajectories collected may show variations in the pendulum parameters: the same equation (3.2.1) holds true, albeit with different parameters φ_1 and φ_2. This observation generalizes to more complex articulated systems, such as robots (see Figure 3.1), as we will see. Note that while the pendulum has well-understood physics, our goal is to model more advanced systems such as robots with complex effects (including friction), hence motivating a statistical approach and the use of deep neural networks.

A more complex, non-analytical example is that of learning the solution to a partial differential equation, which is rarely known in closed form and varies strongly according to the boundary conditions.

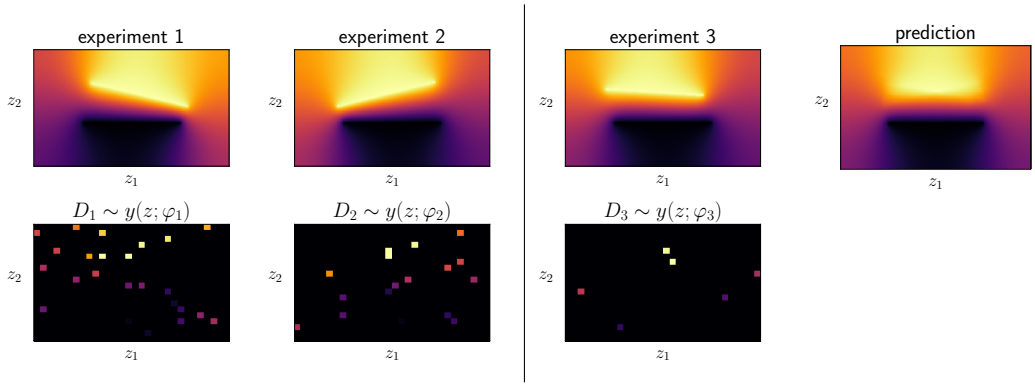

Figure 3.2. Challenges of modeling non-identically distributed data. **Left** Samples from a multi-environment dataset, from two different experimental conditions. **Right** The physical system at prediction time, with a few-shot dataset. The prediction on the right is obtained from a parametric model trained on the collection of all the datasets, without modeling their variability. This results in a blurry prediction blending the various environments, rather than capturing the changes in the system.

Example 3.2.2 [Electrostatic potential] As we saw in Example 2.1.2 of Chapter 2, the electrostatic potential $x(z)$ in a space Ω devoid of charges solves Laplace's equation

$$\Delta x = 0 \quad \text{on } \Omega, \qquad x(z) = b(z) \quad \text{on } \partial\Omega. \tag{3.2.2}$$

With the general aim of learning an emulator of the equation using a statistical approach, a parametric model is trained on data $y_i^{(t)} = x(z_i^{(t)})$ collected during different experiments t. In the data collection process, it may happen that the system slightly varies from one experiment to another. For example, the geometry of the boundary conditions may be perturbed, leading to changes in the data distribution parametrized by some physical parameter vector φ, as $\Omega = \Omega(\varphi)$, and $b(z) = b(z;\varphi)$. A robust data-driven solver should be able to generalize to (at least small) changes of $\partial\Omega(\varphi)$ and $b(z;\varphi)$.

The challenges of learning in such a multi-environment setting are illustrated in Figure 3.2, where we see that modeling the system's variability is necessary for accurately predicting its behavior under different experimental conditions.

3.2.2 Overview of multi-environment deep learning

Multi-task statistical learning has a long history, and several approaches to this problem have been proposed in the statistics community (Caruana, 1997). We will focus on the meta-learning paradigm (Hospedales et al., 2021), which has recently gained considerable importance and whose application to neural nets looks promising given the complexity of physical systems. We next describe the generic structure of meta-learning algorithms for multi-task generalization. The goal is to obtain a $z \mapsto y$ mapping in the form of a two-fold function $y \simeq g(z;w)$, where w is a tunable task-specific weight that models the environment variations.

Learning model Given the learning capabilities of neural networks, incorporating multi-task generalization into their gradient descent training algorithms is a major challenge. Since the seminal paper by Finn et al. (2017), several algorithms have been proposed for this purpose, with the common idea of finding a map adapting the weights of the neural network according to task data. A convenient point of view is to introduce a two-fold parameterization of a meta-model $F(z;\theta,w)$, with a task-agnostic parameter vector $\theta \in \mathbb{R}^n$ and task-specific weights w (also called learning contexts). For each task t, the task-specific weight is computed based on some trainable meta-parameters π and the task data currently

being processed as $w_t := A(\pi, D_t)$, according to an adaptation operation A that is differentiable with respect to π. The meta-parameters are trained to minimize the meta-loss function aggregated over the tasks, as we will see below. In this formalism, a meta-learning algorithm is determined by the meta-model $F(z; \theta, w)$ and the adaptation operation A.

We provide examples of recent architectures in Table 3.1. In MAML (Finn et al., 2017), the meta-parameter π is simply θ and the adaptation rule is computed as a gradient step in the direction of the task-specific loss improvement, in an inner gradient loop. The task variability is modeled as an additive correction in the parameter space of a neural network $f(z; \theta)$, as $F(z; \theta, w) = f(z; \theta + w)$. In CoDA (Kirchmeyer et al., 2022), the structure is the same but the meta-parameter π has a dimension growing with the number of tasks t and the adaptation operation is computed directly from the meta-parameters, with task-specific low-dimensional context vectors $\alpha_t \in \mathbb{R}^{d_\alpha}$ and a linear hypernetwork $\Theta \in \mathbb{R}^{n \times d_\alpha}$. Variants of MAML, CAVIA (Zintgraf et al., 2019) and ANIL (Raghu et al., 2020), fit into this scheme as well and correspond to the restriction of the adaptation inner gradient loop to a predetermined set of the network's weights. This framework also encompasses the CAMEL algorithm, which we introduce in Section 3.3.

Meta-training The training process is summarized in Algorithm 3.1. For each task t, the meta-learner computes a task-specific version of the model from the task dataset D_t, with the adapted weights $w_t := A(\pi, D_t)$. The error on the dataset D_t is measured by the task-specific loss

$$\ell(\theta, w; D_t) = \sum_{(z,y) \in D_t} \frac{1}{2} \big(F(z; \theta, w) - y\big)^2. \tag{3.2.3}$$

Parameters π are trained by gradient descent in order to minimize the regularized meta-loss defined as the aggregation of L_t and a regularization term $R(\pi)$:

$$L(\pi) := \sum_{t=1}^{T} \ell\big(\theta, w_t(\pi); D_t\big) + R(\pi). \tag{3.2.4}$$

Algorithm 3.1 Gradient-based meta-training

input meta-model $F(z; \theta, w)$, adaptation rule A, initial meta-parameters π, learning rate γ, task datasets $D_1, \dots D_T$
output learned meta-parameters $\bar{\pi}$
while not converged **do**
  for tasks $1 \leq t \leq T$ **do**
    compute θ from π
    adapt $w_t := A(\pi, D_t)$
    compute $\ell\big(\theta, w_t(\pi); D_t\big)$
  end for
  compute $L(\pi)$, as in (3.2.4)
  update $\pi \leftarrow \pi - \gamma \nabla L(\pi)$
end while

Table 3.1. Structure of various meta-learning models. Here $f(z; \theta) \in \mathbb{R}$ and $v(z; \theta) \in \mathbb{R}^r$ denote arbitrary parametric models, such as neural networks; "order" stands for differentiation order.

	MAML	CoDA	CAMEL
π	θ	$\theta, \Theta, \{\alpha_t\}$	$\theta, \{\omega_t\}$
$\dim(\pi)$	n	$n+n\times d_\alpha+d_\alpha\times T$	$n+r\times T$
$\dim(w)$	n		r
$A(\pi, D_t)$	$-\alpha\nabla_\theta L_t$	$\Theta\alpha_t$	ω_t
$F(x; \theta, w)$	$f(z; \theta + w)$		$w^\top v(z; \theta)$
training order	2	1	1
adaptation order	1	1	0

Prediction-time adaptation Once training is complete, the trained meta-parameters $\bar{\pi}$ define a tunable model $g(z; w) := F(z; \bar{\theta}, w)$, where $\bar{\theta}$ is the trained task-agnostic parameter vector. At test time, the trained meta-model is presented with a dataset D_{T+1} consisting of few samples (or shots) from a new task. Using this adaptation data, $\bar{\theta}$ is frozen and the task-specific weight w is tuned (possibly in a constrained set) by minimizing the prediction error on the adaptation dataset:

$$w_{T+1} \in \operatorname*{argmin}_w \ell\big(\bar{\theta}, w; D_{T+1}\big). \tag{3.2.5}$$

In all the above approaches, this minimization is performed by gradient descent. The resulting adapted predictor is defined as $F(z; \bar{\theta}, w_{T+1})$. The meta-learning algorithm is then evaluated by the performance of the adapted predictor on new samples from task $T+1$.

Computational cost The inner-loop gradient-based adaptation used in MAML and its variants suffers from the computational cost of second-order optimization, since Hessian-vector products are computed in numbers proportional to the number of tasks. Furthermore, the cost of gradient-based adaptation at test time can also be crucial, especially for real-time applications where the trained model must be adapted at high frequency.

3.3. Context-Affine Multi-Environment Learning

Physical systems often have a particular structure in the form of mathematical models and equations. The general idea behind model-based machine learning is to exploit the available structure to increase learning performance and minimize computational costs (Karniadakis et al., 2021). With this in mind, we adopt in this section a simpler architecture than those shown above, and show how it lends itself particularly well to learning physical systems.

Problem structure We note that many equations in physics exhibit an affine task dependence, since the varying physical parameters often are linear coefficients (as we see in Example 3.2.1, and we shall further explain in Section 3.4). By incorporating this same structure and hence mimicking physical equations, the model should be well-suited for learning them and for interpreting the physical parameters. Following these intuitions, we propose to learn multi-environment physical systems with affine task-specific context parameters.

Definition 3.1 [Context-affine multi-task learning] The prediction is modeled as an affine function of low-dimensional task-specific weights $w \in \mathbb{R}^r$ with a task-agnostic feature map $v(z;\theta) \in \mathbb{R}^r$ and a task-agnostic bias $c(z;\theta) \in \mathbb{R}$:

$$F(z;\theta,w) = c(z;\theta) + w^\top v(z;\theta). \tag{3.3.1}$$

The dimension r of the task weight must be chosen carefully. It must be larger than the estimated number of physical parameters varying from task to task but smaller than the number of training tasks, so as to observe the function v projected over a sufficient number of directions. During training, the task-specific weights are directly trained as meta-parameters along with the shared parameter vector: $\pi = (\theta, \omega_1 \ldots, \omega_T)$ and $w_t = A(\pi, D_t) = \omega_t$. The meta-parameters are jointly trained by gradient descent as in Algorithm 3.1. At test time, the minimization problem of adaptation (3.2.5) reduces to ordinary least squares.

The architecture introduced in Definition 3.1 is equivalent to multi-task representation learning with hard parameter sharing (Ruder, 2017) and is proposed as a meta-learning algorithm in (Wang et al., 2021) We will refer to it in our physical system framework as Context-Affine Multi-Environment Learning (CAMEL). In this work, we show that CAMEL is particularly relevant for learning physical systems. Table 3.1 compares CAMEL with the meta-learning algorithms described above.

Computational benefits As the task weights $(\omega_t)_{t=1}^T$ are kept in memory during training instead of being computed in an inner loop, CAMEL can be trained at minimal computational cost. In particular, it does not need to compute Hessian-vector products as in MAML, or to propagate gradients through matrix inversions as in (Bertinetto et al., 2019). The latter operations can be prohibitively costly in our physical modeling framework, where the number of data points N_t is large (it is typically the size of a high-resolution sampling grid, or the number of samples in a trajectory). Adaptation at test time is also computationally inexpensive since ordinary least squares guarantees a unique solution in closed form, as long as the number of samples exceeds the dimension r of the task weight. For

real-time applications, the online least-squares formula (Kushner and Yin, 2003) ensures adaptation with minimal memory and compute requirements, whereas gradient-based adaptation (as in CoDA or in MAML) can be excessively slow.

Applicability The meta-learning models described in Section 3.2.2 seek to learn multi-task data from a complex parametric model (typically a neural network), making the structural assumption that the weights vary slightly around a central value θ in parameter space: $F(z;\theta,w_t) = f(z;\theta+\delta\theta(w_t))$, where $\delta\theta$ is a function of w_t and $\|\delta\theta_t\| \ll \|\theta\|$. Extending this reasoning, the model should be close to its linear approximation:

$$f(z;\theta+\delta\theta_t) \simeq f(z;\theta) + {\delta\theta_t}^\top \nabla f(z;\theta), \tag{3.3.2}$$

where we observe that the output is an affine function of the task-specific component $\delta\theta_t$. We believe that (3.3.2) explains the observation that MAML mainly adapts the last layer of the neural network (Raghu et al., 2020). In Definition 3.1, v and c are arbitrary parametric models, which can be as complex as a deep neural network and are trained to learn a representation that is linear in the task weights. Following (3.3.2), we expect CAMEL's expressivity to be of the same order as that of more complex architectures, with $c(z;\theta)$, w_t and $v(z;\theta)$ playing the roles of $f(z;\theta)$, $\delta\theta_t$ and $\nabla f(z;\theta)$ respectively. Another key advantage of CAMEL is the interpretability of the model, which we describe next.

3.4. Interpretability and system identification

The observations of a physical system are often known to depend on certain well-identified physical quantities that may be of critical importance in the scientific process. When modeling the system in a data-driven approach, it is desirable for the trained model parameters to be interpretable in terms of these physical quantities (Karniadakis et al., 2021), thus ensuring controlled and explainable learning (Linardatos et al., 2021). We here focus on the identification of task-varying physical parameters, which raises the question of the identifiability of the learned task-specific weights. System identification and model identifiability are key issues when learning a system (Ljung, 1998). Although deep neural networks are becoming increasingly popular for modeling physical systems, their complex structure makes them impractical for parameter identification in general (Nelles, 2001).

Physical context identification In mathematical terms, the observed output y is considered as samples of an unknown state function $x(z;\varphi)$ of the input and a physical context vector $\varphi \in \mathbb{R}^\kappa$, gathering the parameters of the system. In our multi-environment setting, the observations from each task are defined by a vector φ_t as $y_i^{(t)} = x(z_i,\varphi_t)$, up to some observation noise. At test time, a new environment corresponds to an unknown underlying physical context φ_{T+1}. While adaptation consists in minimizing the prediction error on the data as in (3.2.5), the interpretation goes further and seeks to identify φ_{T+1}. This means mapping the learned task-specific weights w to the physical contexts φ, *i.e.* learning an estimator $\hat{\varphi} : w \mapsto \varphi$ using the training data and the trained model. Assuming that the physical parameters of the training data $\{\varphi_t\}$ are known,

this can be viewed as a regression problem with T samples, where $\hat{\varphi}$ is trained to predict φ_t from weights w_t learned on the training meta-dataset.

3.4.1 Linearly parametrized systems

We are primarily interested in the case where the physical parameters are known to be linear in the system equation, as

$$x(z;\varphi) := \mu(z) + \varphi^\top \nu(z), \quad \varphi,\nu(z) \in \mathbb{R}^\kappa. \tag{3.4.1}$$

This class of systems is of crucial importance: although simple, it covers a large number of problems of interest, as the following examples illustrate. Furthermore, it can apply locally to more general system, as we shall see later.

Example 3.4.1 [Electric point charges] Point charges are a particular case of Example 3.2.2 with point boundary conditions, as we saw in Example 2.1.1 of Chapter 2. We assume that κ point charges $(\varphi^{(1)}, \dots, \varphi^{(\kappa)}) := \varphi \in \mathbb{R}^\kappa$ are located at some known positions $\zeta^{(1)}, \dots, \zeta^{(\kappa)} \in \Omega$. The resulting field can be computed using Coulomb's law as a function of the position z, and is found to be proportional to these charges: $x(z;\varphi) = \varphi^\top \nu(z)$, with $\nu(z) \propto (1/\|z - \zeta^{(j)}\|)_j$. Although the solution is known in closed form, this example can illustrate more complex problems where an analytical solution is out of reach (and hence ν is unknown) but the linear dependence on certain well-identified parameters is postulated or known.

Example 3.4.2 [Inverse dynamics in robotics] The Euler-Lagrange formulation for the rigid body dynamics has the form

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + \gamma(q) = \tau, \tag{3.4.2}$$

where q is the generalized coordinate vector, M is the mass matrix, C is the Coriolis force matrix, $\gamma(q)$ is the gravity vector and τ is the input (Tedrake, 2022). It can be shown that (3.4.2) is linear with respect to the system's dynamic parameters (Nguyen-Tuong and Peters, 2010), and hence takes the form of (3.4.1) for scalar controls. A simple, yet illustrative system with this structure is the actuated pendulum (3.2.1), where it is clear that the equation is linear in the inertial parameters φ_1 and φ_2. The inverse dynamics equation can be used for trajectory tracking (Spong et al., 2020), as it predicts u from a target trajectory $\{q(s)\}$ (see Appendix 3.B.3).

3.4.2 Locally linear physical contexts

In the absence of prior knowledge about the system under study, the most reasonable structural assumption for multi-task data is to postulate small variations in the system parameter: $\varphi = \varphi_0 + \delta\varphi$. The learned function can then be expanded and found to be locally linear in physical contexts:

$$x(z;\varphi) \simeq x(z;\varphi_0) + \delta\varphi^\top \nabla x(z;\varphi_0), \tag{3.4.3}$$

where the gradient is taken with respect to φ. Equation (3.4.3) has the form (3.4.1) with $\mu(z) = x(z;\varphi_0)$ and $\nu(z) = \nabla x(z;\varphi_0)$.

Example 3.4.3 [Linearization of boundary perturbations] For a general boundary value problem such as (3.2.2), we may assume that the boundary conditions $\partial\Omega(\varphi)$ and $b(z,\varphi)$ vary smoothly according to parameters φ (such as angles or displacements). If these variations are small and the problem is sufficiently regular, the resulting solution $x(z;\varphi)$ can be reasonably approximated by (3.4.3).

3.4.3 Parameter identification with CAMEL

We now study the problem of system identification under the assumption of parameter linearity (3.4.1) using the CAMEL metamodel (3.3.1). We study the identifiability of the model and therefore investigate the vanishing training loss limit, with $c = \mu = 0$ for simplicity, yielding

$$\omega_t{}^\top v(z_i^{(t)}) = \varphi_t{}^\top \nu(z_i^{(t)}) \quad \text{for all} \quad 1 \le t \le T,\ 1 \le i \le N_t. \tag{3.4.4}$$

Identifiability Posed as it is, we can easily see that the physical parameters φ_t are not directly identifiable. Indeed, for any $P \in \mathrm{GL}_r(\mathbb{R})$, the weights ω and the feature vector v produce the same data as the weights $\omega' := P^\top \omega$ and the feature map $v' = P^{-1}v$, since $\omega^\top v = \omega^\top P P^{-1} v$. This problem is related to that of identification in matrix factorization (see for example Fu et al. (2018)). Now that we have recognized this symmetry of the problem, we can ask whether it characterizes the solutions found by CAMEL. The following result provides a positive answer.

Proposition 3.1 [Model identifiability] Assume that the training points are uniform across tasks: $z_i^{(t)} = z_i$, and $N_t = N$ for all $1 \leq t \leq T$ and $1 \leq i \leq N$, with $\kappa \leq r < N, T$. Assume that both sets $\{\nu(z_i)\}$ and $\{\varphi_t\}$ span $\mathbb{R}^\kappa$. In the limit of a vanishing training loss $L(\pi) = 0$, the trained meta-parameters recover the parameters of the system up to a linear transform: there exist $P, Q \in \mathbb{R}^{\kappa \times r}$ such that $\varphi_t = P\omega_t$ for all training task t and $\nu(z_i) = Qv(z_i)$ for all $1 \leq i \leq N$. Additionally, $QP^\top = I_r$.

A proof is provided in Appendix 3.A, along with the case $c \neq \mu$. Proposition 3.1 shows that CAMEL learns a meaningful representation of the system's features instead of overfitting the examples from the training tasks. Remarkably, the relationship between the learned weights and the system parameters is linear and can be estimated using ordinary least squares

$$\hat{\varphi}(\omega) = \hat{P}\omega, \quad \hat{P} = \underset{P \in \mathbb{R}^{\kappa \times r}}{\operatorname{argmin}} \frac{1}{2} \sum_{t=1}^{T} \|P\omega_t - \varphi_t\|_2^2. \tag{3.4.5}$$

Although the relationship between the model and the system, in general, is likely to be complex, especially when deep neural networks are used, the structure of our model and the linear physical contexts enable the derivation of the problem symmetries and the computation of an estimator of the physical parameters. For black-box meta-learning architectures, exhibiting the symmetries in model parameters and computing an identification map seems out of reach, as the number of available tasks T can be very limited in practice (Pourzanjani et al., 2017).

Zero-shot adaptation Looking at the problem from another angle, Proposition 3.1 also shows that ω can be estimated linearly as a function of φ, at least when $r = \kappa$ (which ensures that P is nonsingular). Computing an estimator of ω as a function of φ with the inverse regression to (3.4.5) enables a zero-shot (or physical parameter-induced) adaptation scenario: when an estimate of the physical parameters of the new environment is known a priori, a value for the model weights can be inferred.

3.5. Experimenting on physical systems

The architecture that we have presented is expected to adapt efficiently to the prediction of new environments, and identify (locally or globally) their physical parameters, as shown in Section 3.4. In this section, we validate these statements experimentally on various physical systems: Sections 3.5.1 and 3.5.2 deal with systems with linear parameters (as in (3.4.1)), on which we evaluate the interpretability of the algorithms. We then examine a non-analytical, general system in Section 3.5.3. We compare the performances of CAMEL and its zero-shot adaptation version φ-CAMEL introduced in Section 3.4.3 with state-of-the-art meta-learning algorithms. Our code and demonstration material are available at `https://github.com/MB-29/CAMEL`.

Baselines We have implemented the MAML algorithm of Finn et al. (2017), and its ANIL variant (Raghu et al., 2020), which is computationally lighter and more suitable for learning linearly parametrized systems (according to observation (3.3.2)). We have also adapted the ℓ_1-CoDA architecture of Kirchmeyer et al. (2022) for supervised learning (originally designed for time series prediction). In all our experiments, the different meta-models share the same underlying neural network architecture, with the last layer of size $r \gtrsim \kappa$. Additional details can be found in Appendix 3.B. The linear regressor computed for CAMEL in (3.4.5) is computed after training for all architectures with their trained weights w_t, and is available at test time for identification.

3.5.1 Interpretable learning of an electric point charge system

As a first illustration of multi-environment learning, we are interested in a data-driven approach to electrostatics, where the experimenter has no knowledge of the theoretical laws (Maxwell's equations, as in Example 3.2.2) of the system under study. The electrostatic potential is measured at various points in space, under different experimental conditions. The observations collected are then used to train a meta-learning model to predict the electrostatic field from new experiments, based on very limited data. We start with the toy system described in Example 3.4.1, which provides a qualitative illustration of the behavior of various learning algorithms: $\kappa = 3$ point charges placed in the plane at fixed locations. This experiment is repeated with varying charges $\varphi \in \mathbb{R}^3$.

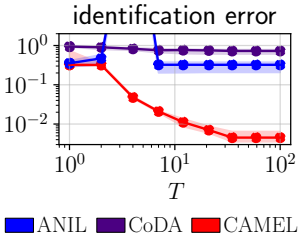

Figure 3.3. Average relative error for the point charge identification.

Results For this system with linear physical parameters, CAMEL outperforms other baselines and can predict the electrostatic field with few shots, as shown in Figure 3.7 and Table 3.2 (5-shot adaptation). Figure 3.3 shows the identification error over 30 random test environments with standard deviations, as a function of the number of training tasks. Thanks to the sample complexity of linear regression, CAMEL accurately identifies system charges, achieving less than 1% relative error with 10 training tasks.

3.5.2 Multi-task reinforcement learning and online system identification

Another scientific field in which our theoretical framework can be applied is multi-task reinforcement learning, in which a control policy is learned using data from multiple environments of one system (Vithayathil Varghese and Mahmoud, 2020). We saw in Example 3.4.2 that robot joints obey the inverse dynamics equation, which turns out to be linear in the robot's inertial parameters. Consequently, our architecture lends itself well to the statistical learning of this equation from multiple environment data, as well as to the identification of the dynamic parameters. We may then exploit the learned model of the dynamics to perform adaptive inverse dynamics control (see Appendix 3.B.4) of robots with unknown parameters, and learn the parameters simultaneously.

Systems We experiment with systems of increasing complexity, starting with 2D simulated systems: cartpole and acrobot. To make them more realistic, we add friction in their dynamics. The analytical equation (3.4.2) is hence inaccurate, which motivates the use of a data-driven learning method. We then experiment on the simulated 6-degree-of-freedom robot Upkie (Figure 3.4), for which (3.4.2) is unknown and the wheel torque is learned from the ground position and the joint angles.

Experimental setup Learning algorithms are trained on trajectories (a more challenging setting than uniformly spaced data) obtained from multiple system environments. At test time, a new environment is instantiated and the model is adapted from a trajectory of few observations. The resulting adapted model is then used to predict control values for the rest of the trajectory. For the carptole and the robot arm, the predicted values are used to track a reference trajectory using inverse dynamics control. For Upkie, we could not directly use the predicted controls for actuation, but we compare the open-loop predictions with the executed control law. The target motions are swing-up trajectories for the cartpole and the arm, and a 0.5m displacement for Upkie. Since Upkie is a very unstable system, it is controlled in a 200Hz model predictive control loop (Rawlings, 2000).

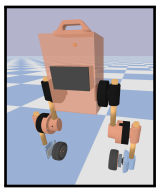

Figure 3.4. Upkie.

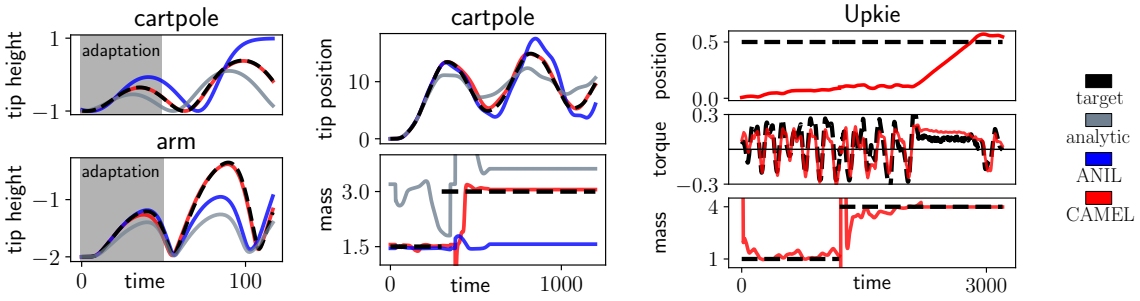

Figure 3.5. Tracking of a reference trajectory using the learned inverse dynamics controller. **Left** 50-shot adaptation. **Center and right** The model and the controller are adapted online.

Online adaptive control We also investigate a challenging time-varying dynamics setting where the inertial parameters of the system change abruptly at a given time. This scenario is very common in real life and requires the development of control algorithms robust to these changes and fast enough to be adaptive (Åström and Wittenmark, 2013). In our case, we double the mass of the cart in the cartpole system, and we quadruple the mass of Upkie's torso. The learning models adapt their task weights online and adjust their control prediction. In an application to parameter identification, we also compute the estimated values of the varying parameter over time.

Results The 100-shot adaptation error of the control values is reported in Table 3.2. The trajectories obtained with inverse dynamics control adapted from 50 shots are plotted in Figure 3.5 for CAMEL and for the best-performing baseline, ANIL, along with the analytical solution. Only CAMEL adapts well enough to track the target trajectory. The analytic solution underestimates the control as it does not account for friction, resulting in inaccurate tracking. In the adaptive control setting, the variation in the mass of the cart leads to a deviation from the target trajectory but CAMEL is able to adapt quickly to the new environment and identifies the new mass, unlike ANIL. Experimentation on Upkie shows that the computational time of adaptation can be crucial, as we found that the gradient-based adaptation of ANIL and CoDA was too slow to run in the 200Hz model predictive control loop. On the other hand, CAMEL's gradient-free adaptation and interpretability allow it to track and identify changes in system dynamics, and to correctly predict the stabilizing control law.

3.5.3 Beyond context-linear systems

In order to evaluate our method on general systems with no known parametric structure, we consider the following non-analytical electrostatic problem of the form shown in Example 3.2.2. The field is created by a capacitor formed by two electrodes that are not exactly parallel. The variability of the different experiments stems from the misalignment $\delta\varphi \in \mathbb{R}^2$, in angle and position, of the upper electrode. We apply the same methodology as described in Section 3.5.1. The whole multi-environment learning experiment is repeated several times with varying magnitudes of misalignment, by replacing $\delta\varphi$ with $\varepsilon\,\delta\varphi$ for different values of $\varepsilon \in [0, 1]$. This parameterization allows us to move gradually from local perturbations

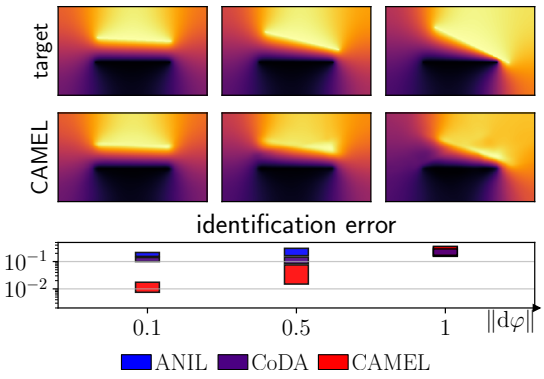

Figure 3.6. Adaptation and relative identification error for the ε-capacitor, with increasing ε.

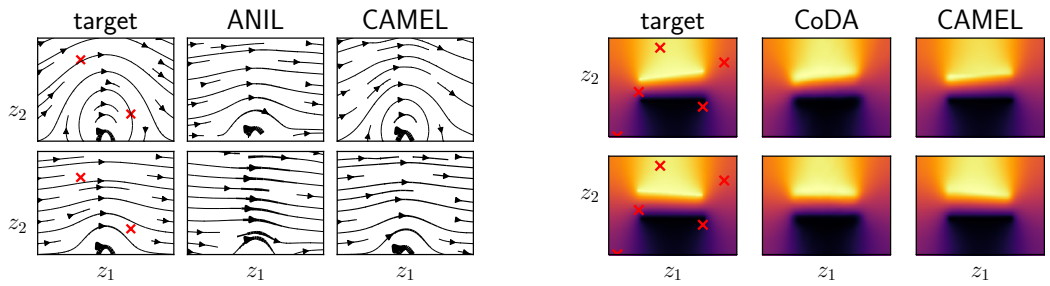

Figure 3.7. Few-shot adaptation on two out-of-domain environments of the point charge system in a dipolar setting (left) and the capacitor (right). The adaptation points are represented by the × symbols. The vector fields are derived from the learned potential fields using automatic differentiation.

when $\varepsilon \ll 1$ (as in Example 3.4.3) to arbitrary variations in the environment.

Results The 40-shot adaptation error for the ε-capacitor is reported in Table 3.2, with perturbation of full magnitude $\varepsilon = 1$ and with $\varepsilon = 0.1$. We also show the 5-shot adaptation of CAMEL and the best performing baseline, CoDA, for $\varepsilon = 0.2$ in Figure 3.7. When the system parameters are fully nonlinear, CAMEL and the baselines perform similarly, but CAMEL is much faster. In the second case, CAMEL outperforms them by an order of magnitude and accurately predicts the electrostatic field, whereas CoDA's exhibits lower precision. Predictions and average identification error (with standard deviations) are plotted as a function of ε in Figure 3.6. For small ε, the system parameter perturbation is well identified, enabling a zero-shot adaptation. Remarkably, Figure 3.7 suggests that the zero-shot model φ-CAMEL performs as well as its few-shot counterpart in this regime, demonstrating the effectiveness of interpretability.

Table 3.2. Average adaptation mean squared error (left) and computational time (right).

System	Charges	Capacitor	ε-Capacitor	Cartpole	Arm	Upkie	Training	Adaptation
MAML	1.6E-1	N/A	N/A	1.8E0	8.1E-1	1.5E-2	30	10
ANIL	9.2E-4	3.6E-2	1.1E-3	2.5E-2	7.5E-1	1.9E-2	10	3
CoDA	8.2E-2	2.6E-2	1.0E-3	8.1E-1	9.3E-1	2.1E-2	2	8
R2-D2	1.2E-4	3.1E-4	4.2E-4	8.5E-3	3.5E-1	2.3E-2	20	1
CAMEL	1.0E-4	2.6E-2	1.9E-4	3.1E-3	2.4E-1	8.2E-3	1	1

3.6. Related work

Multi-task meta-learning Meta-learning algorithms for multi-task generalization have gained popularity (Hospedales et al., 2021), with the MAML algorithm of Finn et al. (2017) playing a fundamental role in this area. Based on the same principle, the variants ANIL (Raghu et al., 2020) and CAVIA (Zintgraf et al., 2019) have been proposed to mitigate training costs and reduce overfitting. Interpretability is addressed in the latter work, using a large number of training tasks. In a different line of work, Bertinetto et al. (2019) proposed the R2-D2 architecture where the heads of the network are adapted using the closed-form formula of Ridge regression. The similarities between multi-task representation learning and gradient-based learning are studied in (Wang et al., 2021) from a theoretical point of view, in the limit of a large number of tasks. Unlike our method, the approaches above rely on the assumption that the number of training tasks is large (in few-shot image classification for example, where it can be in the millions (Wang et al., 2021; Hospedales et al., 2021)) and the number of data points per task is limited. For physical systems, in contrast, since experimenting is often costly, the number of tasks available at training is typically very limited, but the number of points for each task can be large. The

assumption of limited allows the task-specific weightsto be stored in the meta-parameter vector instead of being computed at each training step.

Meta-learning physical systems Meta-learning has been applied to multi-environment data for physical systems, with a focus on dynamical systems, where the target function is the flow of a differential equation. Recent algorithms include LEADS (Yin et al., 2021), in which the task dependence is additive in the output space and CoDA (Kirchmeyer et al., 2022), where parameter identification is addressed briefly, but under strong assumptions of input linearity. Wang et al. (2022b) propose physical-context-based learning, but context supervision is required for training. From a broader point of view, the interpretability of the statistical model can be imposed by adding physical constraints to the loss function (Raissi et al., 2019).

Multi-task reinforcement learning Meta-learning has given rise to a number of fruitful new approaches in the field of reinforcement learning. Sodhani et al. (2021) and Clavera et al. (2019) propose multi-task deep learning algorithms, but no structure is assumed on the dynamics and the learned weights can be interpreted only statistically, in the parameter space of a large black-box neural network. Multi-task learning of inverse dynamics with varying inertial parameters is studied in (Williams et al., 2008) using Gaussian processes, but parameter identification is not addressed.

3.7. Conclusion

We introduced CAMEL, a simple multi-task learning algorithm designed for multi-environment learning of physical systems. For general and complex physical systems, we demonstrated that our method performs as well as the state-of-the-art, at a much lower computational cost. Moreover, when the learned system exhibits a linear structure in its physical parameters, our architecture is particularly effective, and enables the identification of these parameters with little supervision, independently of training. The effectiveness of our approach for parameter identification is demonstrated in our experiments.

We proposed a particular application in the field of robotics where our data-driven method enables concurrent adaptive control and system identification. We believe that enforcing more physical structure in the meta-model, using for example Lagrangian neural networks (Lutter et al., 2019), can improve its sample efficiency and extend its applicability to more complex robots.

While we focused on classical regression tasks, our framework can be generalized to predict dynamical systems by combining it with a differentiable solver (Chen et al., 2018). Another interesting avenue for future research is the use of active learning, to make the most at out the available training resource and enhance the efficiency of multi-task learning for static and dynamic systems (Wang et al., 2023; Blanke and Lelarge, 2023).

Chapter 4

Online active identification of linear dynamical systems

This work addresses the problem of exploration in an unknown environment. For multi-input multi-output linear time-invariant dynamical systems, we use an optimal experimental design framework and introduce an online greedy policy where the control maximizes the information of the next step. In a setting with a limited number of observations, our algorithm has low complexity and shows experimentally competitive performances compared to more elaborate gradient-based methods.

Chapter organization This chapter is organized as follows. Section 4.1 presents the active system identification problem and its challenges. In Section 4.2, we pose the mathematical framework of this sequential decision-making problem. Section 4.3 introduces our online greedy information-maximizing policy. Section 4.4 evaluates the performance of this policy with numerical experiments. In Section 4.5, we compare our contribution with related works. In Section 4.6, we summarize our contributions, and we discuss its limitations and perspectives.

This chapter is based on the article *Online greedy identification of linear dynamical systems* (Blanke and Lelarge, 2022), published in the Proceedings of the IEEE 61st Conference on Decision and Control (CDC 2022).

4.1. Introduction

System identification is a primary field in control theory (Ljung, 1998) and has applications in many fields such as econometrics, robotics, aeronautics, mechanical engineering or reinforcement learning (Natke, 1992; Goodwin and Payne, 1977; Gupta et al., 1976; Moerland et al., 2021). The task consists in estimating the parameters of an unknown system by sampling data from it. This work is concerned with the identification of dynamical systems, which find applications in aeronautics (Gupta et al., 1976) or in robotics (Spong et al., 2020) for instance. As a general rule, collecting data from a physical system comes at a high cost. In aeronautics or robotics, for example, carrying out an experiment costs equipment, personnel, time and the risk of damage. Consequently, it is always of interest to be able to estimate these parameters using as little data as possible. This observation raises the following question: if the experimenter can choose his inputs when interacting with the system, how can he choose the inputs that will produce the most informative data for parameter estimation? The problem thus defined is called optimal experimental design, or active system identification, and it is the issue that we will be tackling in this work.

In this chapter, we will be focusing on a simple, yet rich class of models: multivariate linear dynamical models. These models can describe simple systems, where the dynamics are known to be approximately linear. However, they can also describe more complex systems. Indeed, nonlinear dynamics can always be linearized around an equilibrium. For example, the angular response of an aviation system around a reference angle can be well described by a linear system. A linear model can even describe a nonlinear system globally, when implemented in a high-frequency updated control loop: the model's error is compensated for by feedback from the control loop. In the control community, these systems are referred to as multi-input multi-output (MIMO) linear time-invariant (LTI) dynamical systems.

The active identification of dynamical systems has been the subject of much work in the control community since the 1970s, in the framework of optimal experimental design (Fedorov et al., 1972; Pukelsheim, 2006). Most of this work focuses on a frequency-domain approach, which assumes that the system has reached a quasi-stationary regime (Gevers et al., 2011). We argue that the validity of this assumption is questionable in many applications, where time scales are too short for a stationary regime to be considered. In robotics, for example, the robot's environment can change rapidly, and the corresponding linear model must be able to be estimated just as quickly. Recently, the active dynamical system identification problem has received significant interest from the machine learning community for its connections with reinforcement learning (Wagenmaker and Jamieson, 2020; Wagenmaker et al., 2021; Mania et al., 2020). These works propose a theoretical approach to active learning, where the emphasis is on the computation of estimatoin bounds. The resulting algorithms have theoretical guarantees, but are only asymptotically valid in time. Moreover, the computational cost of these algorithms can be significant. For these reasons, we explore the active identification problem with a particular eye on these practical constraints, and propose an algorithm that is efficient in a regime where both the number of observations and computational resources are small.

Contributions In this work, we explore a setting for linear system identification with hard limitations on the number of interactions with the real system and on the computing resources used for planning and estimation. To the best of our knowledge, finite-time system identification guarantees are only available in the long time limit which makes the hypothesis of linear time-invariant dynamics quite unlikely. Using a framework based on optimal experimental design, we propose a greedy online algorithm requiring minimal computing resources. The resulting policy gives a control that maximizes the amount of information collected at the next step. We show empirically that for short interactions with the system, this simple approach can actually outperform more sophisticated gradient-based methods. We study the computational complexity of our algorithms and compare their performance against each other and against an oracle that we design, both on average and on real-life dynamical systems.

4.2. The active system identification problem

In this section, we provide a mathematical formulation of the active system identification, following up on the framework introduced in Chapter 2. Let $A \in \mathbb{R}^{d\times d}$ and $B \in \mathbb{R}^{d\times k}$ be two matrices; we consider the following discrete-time dynamics model:

$$\begin{aligned} x_0 &= 0, \\ x_{t+1} &= Ax_t + Bu_t + \eta_t, \quad 0 \leq t \leq T-1 \end{aligned} \tag{4.2.1}$$

where $x_t \in \mathbb{R}^d$ is the state, $\eta_t \sim \mathcal{N}(0, \sigma^2 I_d)$ is a normally distributed isotropic noise with known variance σ^2 and the control variables $u_t \in \mathbb{R}^k$ are chosen by the controller with the following power constraint:

$$\frac{1}{T}\sum_{t=0}^{T-1} \|u_t\|^2 \leq \beta^2. \tag{4.2.2}$$

Assumptions and notations Since the scope of this work is exploration in state space, we assume perfect state observations $y_t = x_t$. The parameters of the model are $(A\,B) := \theta \in \mathbb{R}^{d\times(d+k)}$. We let $n = d \times (d+k)$ be the number of scalar parameters. We assume that the system can be described by a true parameter value $(A_\star\, B_\star) := \theta_\star$ generating the observations according to (4.2.1), according to the independent Gaussian noise assumption. This parameter value is unknown initially and is to be identified. We denote by $z_t := (x_t, u_t)$ the state-action pair and by $z_{0:t} := (x_{0:t}, u_{0:t-1})$ a trajectory in state-action space. At each time step, an estimator $\hat{\theta} : z_{0:t} \mapsto \hat{\theta}(z_{0:t}) \in \mathbb{R}^n$ yields an estimate $\theta_t = \hat{\theta}(z_{0:t})$ of the parameters from the past trajectory. We define a policy $\Pi : z_{0:t} \to u_t$ as a mapping from the past trajectory to future input. The set of policies meeting the power constraint (4.2.2) is noted $\mathcal{P}_\beta$. The choice of inputs depends on the current estimate of the system's dynamics, and hence on the estimate of the system parameters, which we will emphasize by writing by $\Pi(z_{0:t}) = \Pi(z_{0:t}|\theta_t)$. Note that it may happen that $B_\star$ is known advance, in which case only $A_\star$ is to be estimated and $\theta = A$, $n = d \times d$.

Objective Our goal is to find a policy $\Pi \in \mathcal{P}_\beta$ under which the resulting trajectory $z_{0:T}$ gives the best possible estimation $\theta_T := \hat{\theta}(z_{0:T})$ for $\theta_\star$. As we saw in Proposition 2.3 of Chapter 2, the learning performance for linear models is accurately measured by the parameter error, which we here denote by

$$\varepsilon(\theta) := \|\theta - \theta_\star\|^2. \tag{4.2.3}$$

The goal of active system identification is to find a policy Π minimizing the expected error, which may be formulated as

$$\underset{\Pi\in\mathcal{P}_\beta}{\text{minimize}} \quad \mathbb{E}[\varepsilon(\theta_N)|\Pi], \tag{4.2.4}$$

where the expectation is taken with respect to the data-generating distribution, that is the dynamics (4.2.1) of the true parameters $\theta_\star$, under policy Π. The sought policy should not only optimize this mathematical objective, but also meet the practical constraints mentioned in Section 4.1.

4.2.1 Sequential identification

Since objective (4.2.4) depends on the true, unknown system parameters, it cannot be directly computed by the controller and minimized with respect to Π to find an optimal policy. We therefore need to find a way of quantifying the informativeness of the controller's inputs, in order to deduce an effective exploration policy. Once a policy is found, the system's parameters are sequentially estimated, and the successive iterates θ_t are used to improve the dynamical model for planning the next inputs with $\Pi(z_{0:t}|\theta_t)$. This scheme is summarized in Algorithm 4.1, which may be seen as an adaptation of the sequential experimental design algorithm of Algorithm 2.2 to the setting of linear dynamical systems.

Algorithm 4.1 Sequential system identification

input learning model f, time horizon T, policy $\Pi \in \mathcal{P}_\beta$, estimator $\hat{\theta}$
output parameter estimate θ_T
for $0 \leq t \leq T-1$ **do**
  choose $u_t = \Pi_t(x_{0:t}, u_{0:t-1}|\theta_t)$
  observe $x_{t+1} = A_\star x_t + B_\star u_t + \eta_t$
  update $\theta_{t+1} = \hat{\theta}(x_{0:t+1}, u_{0:t})$
end for

Example 4.2.1 [Random policy] A naive strategy for system identification consists in playing random inputs with maximal energy at each time step: $\Pi(z_{0:t}) = u_t \sim \mathcal{N}\big(0, \frac{\beta^2}{k} I_k\big)$ for all t.

Example 4.2.2 [Task-optimal pure exploration] In a theoretical approach to active system identification, Wagenmaker et al. (2021) derive a cost function that provably approximates (4.2.4) in the long time limit $T \to +\infty$ at an optimal rate. Their policy selects inputs minimizing this cost, over planning intervals of exponentially growing length $t_i = 2^i \times H$ for some initial time horizon H. Therefore, the policy updates $\Pi(.|\theta_t)$ is constant with respect to θ_t over these exponentially long planning intervals.

Example 4.2.3 [Oracle] An oracle is a controller that is assumed to choose its policy with the knowledge of the true parameter $\theta_\star$. It can hence optimize objective (4.2.4) in open loop with respect to the inputs $u_{0:T-1}$. By definition, the inputs played by the oracle are the optimal inputs for our problem of mean squared error system identification.

4.2.2 Ordinary least-squares estimation

Before considering the decision-making problem of finding an informative policy, we turn to the learning problem of parameter estimation. Given a trajectory $z_{0:t}$, a natural estimator for the matrix parameters $\theta_\star$ is the maximum likelihood estimator, which as we shall see coincides with the ordinary least squares estimator. In the remainder of this chapter, we will denote $p(z_{0:t}|\theta)$ rather than $p(x_{0:t}|u_{0:t-1}, \theta)$.

The data-generating distribution given the parameter θ can be computed using the probability chain rule with the dynamics (4.2.1):

$$p(z_{0:t}|\theta) = \frac{1}{(\sqrt{2\pi\sigma^2})^t} \exp\left[-\frac{1}{2\sigma^2}\sum_{s=0}^{t-1} \|Ax_s + Bu_s - x_{s+1}\|_2^2\right]. \tag{4.2.5}$$

The log-likelihood of the observation is

$$\log p(z_{0:t}|\theta) = -\frac{1}{2\sigma^2}\sum_{s=0}^{t-1} \|Ax_s + Bu_s - x_{s+1}\|_2^2 - t\log(\sqrt{2\pi\sigma^2}) \tag{4.2.6}$$

As we saw in Chapter 2, we can compute the maximum likelihood estimator in closed form. The following result summarizes the formulae and adapts them in an online setting.

Proposition 4.1 [Maximum likelihood esimtator] Given a trajectory $z_{0:t}$, the maximum likelihood estimator for θ is obtained as

$$\hat{\theta}(z_{0:t}) = \Big(G_t^{-1}\sum_{s=0}^{t-1} z_s x_{s+1}^\top\Big)^\top \quad \in \mathbb{R}^{(d+k)\times d}, \tag{4.2.7}$$

where

$$G_t := \sum_{s=0}^{t-1} z_s z_s^\top \quad \in \mathbb{R}^{(d+k)\times(d+k)} \tag{4.2.8}$$

is the Gram matrix of the trajectory. The Gram matrix and the maximum likelihood estimator satisfy the following recursions (Kushner and Yin, 2003):

$$\begin{aligned} G_{t+1} &= G_t + z_{t+1} z_{t+1}^\top \\ \theta_{t+1}^\top &= G_{t+1}^{-1}(G_t \theta_t^\top + z_t x_{t+1}^\top). \end{aligned} \tag{4.2.9}$$

Remark 4.1 [System controllability] We assume that the Gram matrix is invertible. This is related to the controllability of our dynamical system (4.2.1) and can be quantified in terms of the rank of a controllability matrix that may be computed with $A_\star$ and $B_\star$. We refer to (Kirk, 1970) for more details and we will assume controllability of the system in the remained of this work.

In the remainder of this work, we will use the maximum likelihood estimator for system identification (4.2.7). Owing to the linearity of the model in its parameters, we have been able to calculate this estimator explicitly as a function of the trajectory. In the absence of dynamics, for a static linear model, we have seen in Proposition 2.2 of Chapter 2 that this estimator follows a Gaussian distribution, whose moments can be easily calculated as a function of the regression variables. Here, we will see that the dynamics of the system make this calculation far more complex.

Proposition 4.2 The gap between the likelihood estimator (4.2.7) and $\theta_\star$ is

$$\hat{\theta}(z_{0:t}) - \theta_\star = \left(G_t^{-1} \sum_{s=0}^{t-1} z_s \eta_s^\top\right)^\top. \tag{4.2.10}$$

Because the z_s are random and mutually correlated, we see from (4.2.10) that the distribution of $\hat{\theta}$ around $\theta_\star$ is potentially far from being Gaussian, and that we cannot easily guarantee that this estimator is unbiased. Unlike the case of linear regression where the z_s are fixed, the stochasticity of $z_{0:t}$ and their correlation with the past noise η_s make the computation of the first-order and second-order moments nontrivial. As a consequence, we cannot quantify the estimator uncertainty directly from its second-order moments, as we did in Section 2.2.2, and we will hence resort to the Fisher information matrix and optimal experimental design theory.

4.3. Online D-optimal identification

In this section, we use information theory to quantify the quality of a trajectory for system identification. We derive an optimization problem consisting in maximizing information, and propose an approximate solution to this problem to obtain an online exploration policy.

4.3.1 Optimal experimental design

In this section, we apply the theory of optimal experimental design (Fedorov et al., 1972; Steinberg and Hunter, 1984) to quantify the informativeness of a trajectory. In our framework, the system parameters are estimated by maximum likelihood (4.2.7) from a full trajectory $z_{0:T}$ of length T. Among all the possible trajectories, we want to measure which ones make this estimation the most accurate. As we have seen in Section 4.2.2, computing the parameter covariance matrix is out of reach because of the nontrivial correlations between the trajectory and the noise. Therefore, we leverage information theory and we compute the the Fisher information matrix, the definition of which is recalled next in the specific context of dynamical system identification.

Table 4.1. Alphabetical optimal design criteria.

Criterion	$\Phi(I)$	$\Phi(\lambda_1, \dots, \lambda_n)$
A-optimality	$-\mathrm{Tr} I^{-1}$	$-\left(1/\lambda_1 + \cdots + 1/\lambda_n\right)$
D-optimality	$\log\det I$	$\log\lambda_1 + \dots \log\lambda_n$
E-optimality	$\min\lambda(I)$	λ_1

Definition 4.1 [Fisher information matrix] The observed Fisher information matrix of a trajectory $z_{0:T}$ at a parameter value θ is defined as

$$I(z_{0:T};\theta) := -\frac{\partial^2}{\partial\theta^2}\log p(z_{0:T}|\theta) \quad \in \mathbb{R}^{n\times n}. \tag{4.3.1}$$

Recall that this matrix as defined is a random variable. The (expected) Fisher information matrix at parameter θ and a policy Π is defined as the expected value of I under θ and Π:

$$\bar{I}(\theta,\Pi) := \mathbb{E}\left[I(z_{0:T};\theta)|\Pi;\theta\right] \tag{4.3.2}$$

The Fisher information matrix measures the curvature of the likelihood function around the maximum likelihood estimator. The larger the curvature, the sharper the maximum of the likelihood function, and therefore the smaller the uncertainty at this maximum. Hence, the information brought by $z_{0:T}$ is measured by the size of the Fisher information matrix.

Proposition 4.3 [Observed Fisher information] The observed Fisher information matrix of our model is

$$I(z_{0:T};\theta) = \frac{1}{\sigma^2}\mathrm{diag}(G_T, \dots, G_T), \tag{4.3.3}$$

the number of blocks being d.

Remarkably, the observed Fisher information does not explicitly depend on the parameter θ, but only on the input and the observations $z_{0:T}$. As in the case of classical linear regression (see Section 2.3.2), this is due to the linearity of the model, leading to a quadratic log-likelihood in the parameter. However, unlike classical linear regression, the expected information $\bar{I}(\theta,\Pi)$ still implicitly depends on θ, the trajectory is not fixed, but generated from the dynamical model (4.2.1).

This matrix being positive, its size may be measured by several scalar functions. In optimal experimental design theory, these functions are referred to as optimality criteria, as we saw in Section 2.3.3 of Chapter 2. We next recall their definition.

Definition 4.2 [Optimality criteria] The size of the information matrix is measured by some criterion $\Phi : \mathbb{S}_n^+(\mathbb{R}) \to \mathbb{R}_+$, which is a functional of I, or of its positive eigenvalues $\lambda_1, \dots, \lambda_n \geq 0$. The quantity $\Phi(\bar{I})$ is called information gain; it represents the amount of information brought by the experiment and should be maximized with respect to the problem decision variables. Some usual criteria are presented in Table 4.1. The optimal design criteria are required to have properties such as homogeneity, monotonicity and concavity in the sense of the Loewner ordering, which can be interpreted in terms of information theory: monotonicity means that a larger information matrix brings a greater amount of information, concavity means that information cannot be increased by interpolation between experiments. We refer to the work of Pukelsheim (2006) for more details.

The information-theoretic formalism introduced above allow us to formulate a maximal information objective defined as the information gain of the Fisher information matrix, as we defined in (2.3.7).

Gathering this objective with the problem's dynamical constraints, we obtain the following stochastic optimal control problem:

$$\begin{aligned} \underset{(z_t)}{\text{maximize}} \quad & \Phi\Big(\mathbb{E}\Big[\sum_{t=0}^{T-1} z_t {z_t}^\top\Big]\Big) \\ \text{subject to} \quad & x_{t+1} = A_\star x_t + B_\star u_t + \eta_t \quad 0 \leq t \leq T-1, \\ & \|u_t\|_2 \leq \beta^2, \end{aligned} \tag{4.3.4}$$

where we note that the dynamical constraint are stochastic and unknown, as they depend on the system dynamics.

4.3.2 One-step-ahead objective

The optimal control objecitve (4.3.4) describes how the system should be excited for the parameter estimation to be maximally accurate. Although theoretically sound, it cannot be solved directly, as the dynamic constraints are unknown. While these constraints can be sequentially approximated by the current estimate of the dynamics, the accuracy of an erroneous model diverges in the long term (*i.e.* for long planning horizons) due to the accumulation of errors. Moreover, the numerical resolution of (4.3.4) is challenging due to the non-convexity of the objective function.

In the work of Wagenmaker et al. (2021), the system identification policy optimizes an objective that is closely related to (4.3.4) with the A-optimal criterion. To make the optimization problem tractable, a stationary regime is assumed as the inputs are restricted to periodic signals and are optimized in the frequency domain over an exponentially long planning horizons. On the contrary, we are interested in a scenario where computations must be fast enough for the policy to run in real time, and must be able to adapt quickly to each new observation. To achieve this, we will resort to a greedy approximation of (4.3.4).

Greedy approximation Note that, for each t, the Gram matrix can be split into two terms representing the past trajectory and the future trajectory respectively:

$$G_T = \sum_{s=0}^{t-1} z_s {z_s}^\top + \sum_{s=t}^{T-1} z_s {z_s}^\top, \tag{4.3.5}$$

where the first sum is fixed by the past observations, and the second sum is a random variable depending on the next inputs. We argue that long-term estimates of the states are likely to be inaccurate, and that problem (4.3.4) is too complex to be optimized at high frequency anyway. Therefore, we propose an approximation that favors computational simplicity and adaptability over extensiveness, and we replace the information matrix by its truncation at the next time step in the information gain:

$$\Phi(G_T) \simeq \Phi(G_t + z_{t+1} z_{t+1}^\top). \tag{4.3.6}$$

Using this approximation, we can then choose each input u_t at time $t$by maximizing a one-step-ahead optimization problem in $\mathbb{R}^k$, by computing and maximizing the greedy approximation (4.3.6) of the information gain. More precisely, our policy computes u_t at time t by solving the following optimization problem

$$\begin{aligned} \underset{u \in \mathbb{R}^k}{\text{maximize}} \quad & \Phi\left(G_t + z z^\top\right) \\ \text{subject to} \quad & z = (A_t x_t + B_t u, 0), \quad \|u\|^2 \leq \beta^2. \end{aligned} \tag{4.3.7}$$

The optimization objective of problem (4.3.7) is defined in terms of G_t, x_t and θ_t, so we may express the solution as the output of a policy $u_t = \Pi(z_{0:t}|\theta_t)$.

Remark 4.2 With this greedy policy, the energy constraint imposed for one input ensures that the global power constraint (4.2.2) is met. Furthermore, this policy does not require the knowledge of the time horizon T in advance.

We will see in the next section that problem (4.3.7) can be solved accurately and at a cheap cost for the D-optimality criterion. Moreover, our policy has the advantage of being updated each time a new observation is collected, thus ensuring that all the available information is used for planning. This way, the bias affecting planning due to the estimation error about the parameters is minimized. When planning is performed over larger time sequences, this bias could impair the identification, as the controller could spend much time exploring directions given by the wrong dynamics.

Algorithm 4.2 Greedy system identification

inputs initial guess θ_0, power β^2
output final estimate θ_T
for $0 \leq t \leq T-1$ **do**
  $u_t \in \underset{\|u\|_2^2=\beta^2}{\operatorname{argmax}} \Phi\big(M_t + z(u)z(u)^\top\big)$
  play u_t, observe x_{t+1}
  $G_{t+1} = G_t + z_t z_t^\top$
  $\theta_{t+1}^\top = G_{t+1}^{-1}\big(G_t\theta_t^\top + z_t x_{t+1}^\top\big)$
end for

4.3.3 Solving the one-step optimal design problem

Adopting D-optimality as an informativeness measure, we show how the one-step-ahead input design problem that we proposed in Section 4.3.2 can be solved efficiently.

Proposition 4.4 For the D-optimality crtierion, there exists a postive definite symmetric matrix $Q \in \mathbb{R}^{k\times k}$ and $b \in \mathbb{R}^k$ such that the problem (4.3.7) is equivalent to

$$\begin{aligned} \underset{u\in\mathbb{R}^k}{\text{maximize}} \quad & u^\top Q u - 2b^\top u \\ \text{subject to} \quad & \|u\|_2^2 \leq \beta^2. \end{aligned} \tag{4.3.8}$$

We now characterize the minimizers of problem (4.3.8). Since Q is positive definite, the quadratic form increases with the value of $\|u\|$, hence any maximizer lies on the sphere of radius β. We may thus equivalently consider the equality constrained problem

$$\begin{aligned} \underset{u\in\mathbb{R}^k}{\text{maximize}} \quad & u^\top Q u - 2b^\top u \\ \text{subject to} \quad & \|u\|_2^2 = \beta^2. \end{aligned} \tag{4.3.9}$$

Proposition 4.5 Note $\{\alpha_i\}$ the eigenvalues of Q, and u_i and b_i the coordinates of u_* and b in a corresponding orthonormal basis. Then a solution u_* of (4.3.9) satisfies the following equations for some nonzero scalar μ:

$$u_i = b_i/(\alpha_i + \mu) \quad \text{and} \quad \sum_i \frac{{b_i}^2}{(\alpha_i+\mu)^2} = \beta^2. \tag{4.3.10}$$

Proof. By the Lagrange multiplier theorem there exists a nonzero scalar μ such that $Qu_* - b = -\mu\, u_*$, where μ can be scaled such that $Q + \mu I_k$ is nonsingular. Inverting the optimal condition and expanding the equality constraint gives the two stated conditions.

□

Problem (4.3.8) can hence be solved at the cost of a scalar root-finding and an eigenvalue decomposition. In (Hager, 2001), bounds are provided so as to initialize the root-finding search efficiently. Our greedy active system identification algorithm is summarized in Algorithm 4.2, from which we can see that it is designed to run online.

4.4. Performance study

We run experiments to validate the performance of our exploration algorithm and compare it to gradient-based experimental design algorithms. Our code is available online at `https://github.com/MB-29/greedy-identification`.

4.4.1 Complexity analysis

Definition 4.3 [Performance] The performance of policy Π is measured by the parameter error $\varepsilon(\theta_t)$ averaged over the experiments on the true system. We study the performance of our algorithms as a function of the number of observations T and C the computational cost. We also introduce the computational rate $c = C/T$.

As a baseline, we introduce a gradient-based exploration algorithm that optimizes (4.3.4) using stochastic gradient descent, with Φ the A-optimality functional, with a batch size of $b = 100$ and $\{t_i\} = \{0, 10, T/2, T\}$.

Algorithm 4.2 and the gradient-based identification algorithms have linear time complexity. We see in the example of static linear regression (see Proposition 2.2) that the squared error should scale like $1/T$, which is verified experimentally in the dynamical setting. Given the previous observations, we postulate that the performance of our algorithms takes the form

$$\varepsilon(C, T) = \alpha(c)/T. \tag{4.4.1}$$

We build an experimental diagram where we plot the average estimation error for $\theta_\star = A_\star$ as a function of the two types of resource T and C for the gradient algorithm. Increasing C allows for more gradient steps. We run trials with random matrices $A_\star$ of size $d = 4$, with $B = I_d$. We set $\beta = 1$, $\sigma = 10^{-2}$, $T \in [60, 220]$. The obtained performances are compared with those of the greedy algorithm, which has a fixed, small computational rate c. Our diagrams are plotted on Fig. 4.1.

Our diagrams show that the greedy algorithm is preferable in a phase of low computational rate: $C < c \times T$, as suggested by (4.4.1). The phase separation corresponds to a relatively high number of gradient steps. Indeed, the iso-performance along this line are almost vertical, meaning that the gradient descent has almost converged. Furthermore, the maximum performance gain of the gradient algorithm relatively to the greedy algorithm is of 10%.

4.4.2 Average estimation error

We now test the performances of our algorithms on random matrices, with the same settings as in the previous experiment. For each matrix $A_\star$, we also compute an oracle optimal control optimizing (4.2.4) by stochastic gradient descent, with a batch size of $b = 100$, and run a random input baseline (see Example 4.2.1), and the TOPLE algorithm of Wagenmaker et al. (2021).

Both the gradient algorithm and the greedy algorithm closely approach the oracle. The former performs slightly better than the latter on average. However, the computational cost of the gradient algorithm is far larger, as Table 4.2 shows. Indeed, the number of gradient steps to reach convergence in this setting is found to be of order 100.

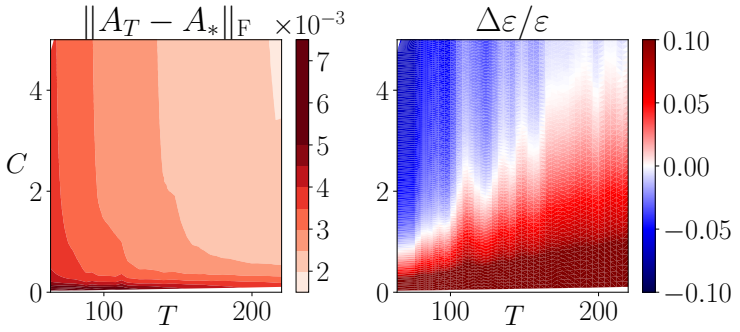

Figure 4.1. Experimental (T, C) diagram. **Left** Performance of the gradient algorithm, with varying T and C (varying number of gradient steps). **Right** Relative performance of the gradient algorithm with respect to the greedy algorithm: positive means that greedy performs better.

Table 4.2. Average computational rate for the different algorithms.

	Random	**TOPLE**	**Gradient**	**Greedy**
c	1	20	50	2.36

4.4.3 Identification of an aircraft system

We now study a more realistic setting from the field of aeronautics: we apply system identification to an aircraft system. We use the numerical values issued in a report from the NASA (Gupta et al., 1976). The lateral motion of a Lockheed Jet star is described by the slideslip and roll angles and the roll and yaw rates. The control variables are the aileron and rudder angles $(\delta_{\mathrm{a}}, \delta_{\mathrm{r}}) := u$. The linear dynamics for an aircraft flying at 573.7 meters/sec at 6.096 meters are given by the following matrix, obtained after discretization and normalization of the continuous-time system (Gupta et al., 1976):

$$A_\star = \begin{pmatrix} .955 & -.0113 & 0 & -.0284 \\ 0 & 1 & .0568 & 0 \\ -.25 & 0 & -.963 & .00496 \\ .168 & 0 & -.00476 & -.993 \end{pmatrix}, \quad B_\star = 0.1 \times \begin{pmatrix} 0 & 0.0116 \\ 0 & 0 \\ 1.62 & .789 \\ 0 & -.87 \end{pmatrix}, \tag{4.4.2}$$

and $\sigma = 1$, $\beta \simeq 4$ deg.

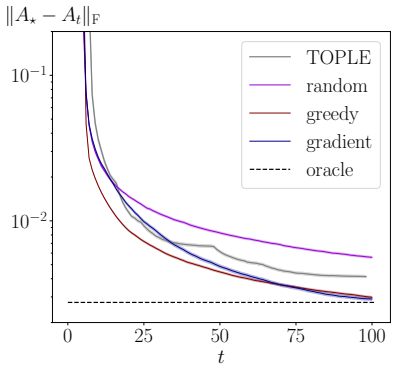

Figure 4.2. Identification error for random $A_\star$ averaged over 1000 samples.

Table 4.3. Frobenius error for $A_\star$ in the lateral system of the aircraft, $T = 150$. Our oracle algorithm reaches an error of 8.0×10^{-2}. The computational time is expressed in an arbitrary unit.

	Random	**TOPLE**	**Gradient**	**Greedy**
Error	1.1×10^{-1}	8.6×10^{-2}	8.3×10^{-2}	8.2×10^{-2}
Time	1	55.7	25	1.13

We apply our algorithms to this LTI system. Our results are summarized in Table 4.3.

As we can see, the greedy algorithm outperforms the gradient-based algorithms, both in performance and in computational cost. This could be explained by the fact that the signal-to-noise ratio in this system is high, leading to high planning uncertainty. It is hence more effective to plan one-step-ahead than to do planning over large epochs. We obtain similar results for the longitudinal system of a C-8 Buffalo aircraft (Gupta et al., 1976).

4.5. Related work

System identification has long been studied in the control community, and several works have been dedicated to LTI systems. Optimal experimental design approaches have been applied to single-input single-output (SISO) systems (Mehra, 1974; Keviczky, 1975; Goodwin and Payne, 1977) or MIMO systems in the frequency domain or with randomized time-domain inputs (Mehra, 1976). More recently, system identification received considerable attention in the machine learning community, with the aim of obtaining finite-time bounds on the parameter estimation error (Simchowitz et al., 2018; Sarkar and Rakhlin, 2019; Tsiamis and Pappas, 2019; Jedra and Proutiere, 2020). The question of designing optimal inputs is tackled in (Wagenmaker and Jamieson, 2020; Wagenmaker et al., 2021). The authors derive an asymptotically optimal algorithm by computing the control in the frequency domain to maximize an optimal design objective, with theoretical estimation rate guarantees. Unlike our online policy, the proposed input design algorithm plans over exponentially large epochs, which is computationally prohibitive for real-time applications.

4.6. Conclusion

In this work, we explored a setting for optimal experimental design with hard constraints on the number of interactions with the real system and on the computing resources used for planning and estimation. We introduced a greedy online algorithm requiring minimal computing resources and showed empirically that for small values of interactions with the system, it can actually outperform more sophisticated gradient-based methods.

Our framework is simple and contains strong assumptions, the aim being to study specifically the problem of system identification and design of experiments. In a more realistic framework, several of these assumptions need to be reviewed. Firstly, dynamical systems are rarely fully observed: we typically only have access to state estimates, potentially sparsely sampled in time. Likelihood calculations would therefore have to be adapted, and state values would have to be replaced by their estimates (Bar-Shalom, 1972). Secondly, in real systems, there are more mathematical constraints on both input and state. In robotics, for example, these constraints ensure the robot's equilibrium or safety (Tedrake, 2022). For a more operational description, these additional constraints should be included in optimization problems. However, the main conclusion of our work remains: an efficient exploration policy can be obtained by considerably simplifying a complex informativity criterion, which is in particular a non-convex function of states. This is of key importance in robotics, where model predictive control programs run at high frequencies, and can therefore support limited computations.

However, convex programs such as quadratic programs are solved much faster than nonlinear programs thanks to advanced optimization tools (Bambade et al., 2023).

Finally, we may wonder whether our exploration policy may be generalized to nonlinear dynamical systems. We study this matter in Chapter 5.

Chapter 5

Adaptive exploration of nonlinear dynamical systems

Model-based reinforcement learning is a powerful tool, but collecting data to fit an accurate model of the system can be costly. Exploring an unknown environment in a sample-efficient manner is hence of great importance. However, the complexity of dynamics and the computational limitations of real systems make this task challenging. In this chapter, we introduce FLEX, an exploration algorithm for nonlinear dynamics based on optimal experimental design. Our policy maximizes the information of the next step and results in an adaptive exploration algorithm, compatible with generic parametric learning models and requiring minimal resources. We test our method on a number of nonlinear environments covering different settings, including time-varying dynamics. Keeping in mind that exploration is intended to serve an exploitation objective, we also test our algorithm on downstream model-based classical control tasks and compare it to other state-of-the-art model-based and model-free approaches. The performance achieved by FLEX is competitive, and its computational cost is low.

Chapter organization This chapter is organized as follows. In Section 5.1, we provide the context and motivations for studying the nonlinear exploration problem. We introduce the underlying mathematical formalism in Section 5.2. In Section 5.3, we study an information-theoretic criterion for linearly parameterized models. In Section 5.4, we introduce FLEX, our exploration algorithm that adaptively maximizes this objective online. Section 5.5 extends our approach to generic, nonlinear models. We validate our method experimentally in Section 5.6. Section 5.7 compares our contribution with related works. Section 5.8 summarizes our contributions and discusses its limitations and perspectives.

This chapter is based on the article *FLEX: an Adaptive Exploration Algorithm for Nonlinear Systems* (Blanke and Lelarge, 2023), published in the Proceedings of the 40th International Conference on Machine Learning (ICML 2023).

5.1. Introduction

Control theory and model-based reinforcement learning have had a range of achievements in various fields including aeronautics, robotics and energy systems (Kirk, 1970; Sutton and Barto, 2018). For the agent to find an effective control policy, the mathematical model of the environment must faithfully capture the dynamics of the system and thus must be fit with data. However, collecting observations can be expensive: consider for example an aircraft system, for which running experiments costs a lot of energy and time (Gupta et al., 1976). In this regard, active exploration (or system identification) aims to excite the system in order to collect informative data and learn the system globally in a sample-efficient manner (Yang et al., 2021), independent of any control task. Once this task agnostic exploration phase is completed, the learned model can be exploited to solve multiple downstream tasks. In computer science and machine learning, exploration is also at the core of various online learning problems such as the multi-armed bandit problem (Soare et al., 2014) and tree exploration (Cosson and Massoulié, 2023), where the goal is to learn the environment as efficiently as possible.

For actuated systems such as robots, dynamics may be complex and generally take the form of a nonlinear function of the state. An example would be air friction, which is essential to consider for an accurate control law, and yet difficult to model from physical principles (Faessler et al., 2018; De Simone et al., 2015). While exploration in linear systems is well understood (Goodwin and Payne, 1977), efficiently learning nonlinear dynamics is far more challenging. For realistic applications, this is compounded by the hard limitations of memory and computational resources of embedded systems (Tassa et al., 2012). Furthermore, the dynamics or the agent's model of it may vary over time, and the exploration policy must adapt as the data stream is collected. Therefore, it is critical that the algorithm works adaptively, with limited memory storage and that it runs fast enough to be implementable in a real system, while being flexible enough to learn complex dynamics with potentially sophisticated models.

Different approaches have been proposed recently for exploring nonlinear environments, and have proceeded by maximizing an information gain (or uncertainty) on the parameters for specific classes of learning models. An exact computation can be derived for models with linear parameterizations (Schultheis et al., 2020), which are however of limited expressivity. Uncertainty can also be computed with Gaussian processes (Buisson-Fenet et al., 2020) or approximated using ensembles of neural networks (Shyam et al., 2019; Sekar et al., 2020) but these approaches suffer from a quadratic memory complexity and an important computational cost respectively, making them unlikely to be implementable in real systems. In all the above approaches, the inputs are planned episodically by solving a non-convex optimization problem where the nonlinear dynamics are simulated over a potentially large time horizon. Planning then relies on nonlinear solvers, which may be too slow to run in real time (Kleff et al., 2021). Furthermore, planning over large time horizons renders the algorithm unable to adapt to new observations as they are collected. As a result, the agent may spend a long time trusting a wrong model and exploring uninformative states. In Chapter 4, we have focused on deriving a fast and adaptive exploration policy for a linear dynamics model. However, maintaining such guarantees while exploring substantially more complex systems with a nonlinear model remains a significant challenge and an open area of research.

Contributions This chapter examines the problem of active exploration of nonlinear environments, with great importance attached to the constraints imposed by real systems. Based on information theory and optimal experimental design, we define an exploration objective that is valid for generic parametric learning models, encompassing linear models and neural networks. We derive an online approximation of this objective and introduce FLEX, a fast and adaptive exploration algorithm. The sample-efficiency and the adaptivity of our method are demonstrated with experiments on various nonlinear systems including a time-varying environment and the performance of FLEX is compared to several baselines. We further evaluate our exploration method on downstream exploitation tasks and

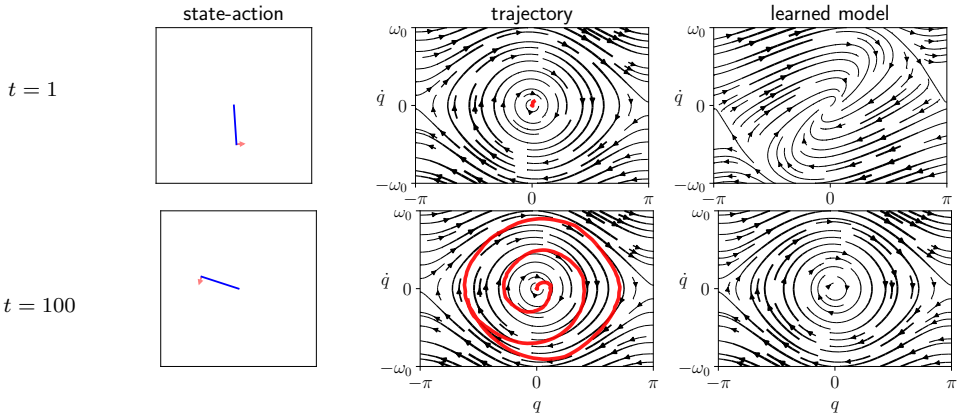

Figure 5.1. Illustration of active exploration for the dynamics of the pendulum. At each time step, an action is chosen (left). The decision is taken by using the past trajectory (middle) to learn a model f of the dynamics (right) and hence a prediction of the next states.

compare FLEX to model-based and model-free exploration approaches.

5.2. Exploring a nonlinear environment

In nonlinear dynamical systems, the state $x \in \mathbb{R}^d$ and the input $u \in \mathbb{R}^k$ are governed by an equation of the form

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f_\star(x, u), \tag{5.2.1}$$

where $f_\star$ is a nonlinear function modeling the dynamics. This function is typically unknown or partially unknown, and our objective is to learn it from data, with as few samples as possible. Our system is modeled as discrete states of the dynamics (5.2.1) with Gaussian model error:

$$x_{t+1} = x_t + \mathrm{d}t f(x_t, u_t; \theta) + \eta_t, \quad 0 \leq t \leq T-1, \tag{5.2.2}$$

where $x_t \in \mathbb{R}^d$ is the state vector, $\eta_t \sim \mathcal{N}(0, \sigma^2 I_d)$ is a normally distributed isotropic noise with known variance σ^2, $\mathrm{d}t$ is a known time step, $f(x, y; \theta)$ is our learning model with parameter vector $\theta \in \mathbb{R}^n$, T is the number of observations, and the control variables $u_t \in \mathbb{R}^k$ are chosen by the agent with the constraint $\|u_t\|_2 \leq \beta$. Since the scope of this work is exploration in state space, we assume perfect state observations $y_t := x_t$, as in Chapter 4. We assume that f is a differentiable function, and we write indifferently $f(x, u; \theta)$ or $f(z; \theta)$ where $z = (x\ u) \in \mathbb{R}^{d+k}$ is the state-action pair. Note that our problem could be formulated in the framework of continuous Markov decision processes (Sutton and Barto, 2018), but we find (5.2.2) more suitable for our approach.

Sequential learning The dynamics function $f_\star$ is learned from past observations with a generic parametric model $f(z, \theta)$. At each time t, the observed trajectory yields an estimate $\theta_t = \hat{\theta}(x_{0:t+1}, u_{0:t})$ following an estimator $\hat{\theta}$, such as the maximum likelihood estimator introduced in Chapter 2. At the end of the exploration, the agent returns a final value θ_T. Although this learning problem is rich and of an independent interest, we will adopt simple, bounded-memory, online learning rules and focus in this work on the following decision-making process.

Sequential decision-making As we saw in Chapters 2 and 4, the agent's decision takes the form of a policy $\Pi : (x_{0:t}, u_{0:t-1} | \theta_t) \mapsto u_t$, mapping the past trajectory to the future input, knowing the current

Algorithm 5.1 active exploration

input learning model f, time horizon T, time step $\mathrm{d}t$, policy $\Pi \in \mathcal{P}_\beta$, estimator $\hat{\theta}$
output parameter estimate θ_T
for $0 \leq t \leq T-1$ **do**
  choose $u_t = \Pi(x_{0:t}, u_{0:t-1}|\theta_t)$
  observe $x_{t+1} = x_t + \mathrm{d}t f_\star(x_t, u_t)$
  update $\theta_{t+1} = \hat{\theta}(x_{0:t+1}, u_{0:t+1})$
end for

parameter θ_t. We denote by $\mathcal{P}_\beta$ the set of policies satisfying the constraint of amplitude β. The sequential decision-making process is summarized in Algorithm 5.1 and illustrated in Figure 5.1. The goal of active exploration is to choose inputs that make the trajectory as informative as possible for the estimation of $f_\star$ with f, as stated below.

The problem For an arbitrary learning model of the dynamics f provided with an estimator $\hat{\theta}$ and for a fixed number of observations T, the goal is to find an exploration policy Π for which the learned model is as close to $f_\star$ as possible at the end of exploration. Formally, we adapt the definition the estimation error of parameter θ, introduced in Section 2.2 of Chapter 2:

$$\varepsilon(\theta) = \mathbb{E}\big[\|f(z,\theta) - f_\star(z)\|^2\big], \tag{5.2.3}$$

with z follows the data distribution . We look for a policy yielding the best estimate of $f_\star$ at time T:

$$\underset{\Pi \in \mathcal{P}_\beta}{\text{minimize}} \quad \mathbb{E}[\varepsilon(\theta_T)|\Pi], \tag{5.2.4}$$

where the expectation is taken over the stochastic dynamics (5.2.1) and possibly the randomness induced by the policy.

Practical considerations In addition to the sample efficiency objective (5.2.4), we attach great importance to the three following practical points that emerge from the observations of Section 5.1. *Adaptivity*: as opposed to episodic planning for which $\Pi(.|\theta)$ remains constant with respect to θ throughout long time intervals, we want our policy to accommodate to new observations at each time step. *Computational efficiency*: evaluating the policy Π should require limited computational resources. *Flexibility*: our policy should be valid for a broad class of models f, as we will see with the following examples.

Example 5.2.1 [Damped pendulum] The angle q of a pendulum driven by a torque u satisfies the following nonlinear differential equation:

$$\ddot{q} + \alpha\dot{q} + \omega_0^2 \sin q = bu. \tag{5.2.5}$$

This second-order system can be described by the bidimensional state variable $x = (q, \dot{q}) \in \mathbb{R}^2$ and the nonlinear map $f_\star(q, \dot{q}, u) = (\dot{q}, -\alpha\dot{q} - \omega_0^2 \sin q + bu)$.

Example 5.2.2 [Nonlinear friction] Using the same notations, the dynamics of a mass subject to a control force and friction is given (in the case of one-dimensional system for simplicity) by $f_\star(q, \dot{q}, u) = (\dot{q}, \mathrm{friction}(q, \dot{q}) + bu)^\top$. The friction force is notoriously difficult to model. One possibility would be the nonlinear function $\mathrm{friction}(q, \dot{q}) = -\alpha|\dot{q}|q$ (Zhang et al., 2014).

The choice of model is crucial and depends on prior knowledge about the system. In Example 5.2.1, if the system is known down to its scalar parameters, learning the dynamics reduces to linear regression. In the opposite case where little or nothing is known about the structure, as in Example 5.2.2, it would be desirable to fit more sophisticated parametric models. For instance, the recent successes of

neural networks and their ability to express complex functions and to be trained online make them a promising option for our exploration task (Goodfellow et al., 2016).

Importantly, the exploration policy should depend on the learned model and drive the system to regions with high uncertainty. In contrast, random exploration fails to explore nonlinear systems globally: typically, friction forces pull the system toward a fixed point, while learning the environment requires large amplitude trajectories and hence temporal coherence in the excitation.

5.3. Information-theoretic view of exploration

Given a model of the dynamics, how do we choose inputs that efficiently navigate in phase space towards informative states? This choice should be guided by some measure of the information that the trajectory provides about our learning model. In this section, we turn to information theory and study the simplest form of models: linear models provide us not only with a natural learning rule but also with an information-theoretic measure of exploration. We then leverage this criterion to define an optimization objective.

5.3.1 Linear models

An important class of models is the class of functions with an affine dependence on the parameters. In the remainder of this work, we use the word "linear" for an affine dependence. Note that the term "linear" refers to the parameter dependence of a model, but the dependence in the state-action pair z is still assumed to be nonlinear in general.

Definition 5.1 [Linear model] The most general form for a linear dependence of f in its parameters is

$$f(z,\theta) = V(z) \times \theta + c(z), \tag{5.3.1}$$

where the feature matrix $V(z) \in \mathbb{R}^{d\times n}$ and $c(z) \in \mathbb{R}^d$ are independent of θ. We denote by $v^{(j)} \in \mathbb{R}^n, 1 \leq j \leq d$ the rows of V, and we define $V_t := V(z_t)$.

The class of linear models is critically important for several reasons. First, the dynamics can very often be formulated as an affine function of some well-defined scalar parameters, hence allowing for efficient estimation by ordinary least squares (Tedrake, 2022). Besides, a natural mathematical exploration objective from optimal experimental design theory can be derived for linear models (see Section 5.3.2). Finally, the computations allowed for linear models can be generalized to nonlinear models, as we will see in Section 5.5.

Example 5.3.1 [Learning the pendulum] When the pendulum of Example 5.2.1 is known up to the parameters $\theta = (\omega_0^2, \alpha, b)^\top$ then the dynamics can be learned with a linear model as defined in (5.3.1) with $n = 3$ by defining

$$V(z) = \begin{pmatrix} 0 & 0 & 0 \\ -\sin\phi & -\dot{\phi} & u \end{pmatrix}, \quad c(z) = (\dot{\phi}, 0)^\top. \tag{5.3.2}$$

The real dynamics function is $f_\star(z) = V(z) \times \theta_\star + c(z)$ with $\theta_\star$ the true parameters.

An important case of linear models is when the parameters are separated in a matrix form according to row-wise dependence, as follows.

Definition 5.2 [Matrix parameterization] We call a matrix parameterization a model of the form

$$f(z,\Theta) = \Theta \times \phi(z), \tag{5.3.3}$$

where $\phi : \mathbb{R}^{d+k} \to \mathbb{R}^r$ is a feature map and $\Theta \in \mathbb{R}^{d\times r}$ is a parameter matrix. The structure (5.3.1) is recovered by defining θ as the vectorization of Θ of size $n = d \times r$, and the feature matrix as the block matrix $V = \mathrm{diag}(\phi^\top, \dots, \phi^\top) \in \mathbb{R}^{d\times n}$.

Example 5.3.2 [Linear dynamics] Consider the case of a linear time-invariant model for the dynamics, as we saw in Chapter 4: $f(z, \Theta) = \Theta \times z$, with the parameters $\Theta = (A\, B) \in \mathbb{R}^{d\times(d+k)}$. This falls into the category a matrix parameterization (5.3.3) with $\phi(z) = z$ and $r = d + k$.

Example 5.3.3 [Random Fourier Features] (Schultheis et al., 2020) propose to model nonlinear dynamics with Random Fourier Features, which take the form (5.3.3), with $\phi(z)$ a random feature.

5.3.2 Optimal experimental design

In this section, we recall the notions of information theory introduced in Section 2.3 of Chapter 2 that will allow us to formulate exploration as a mathematical problem. Let y be an observation whose distribution $p(y|z, \theta)$ depends on a parameter $\theta \in \mathbb{R}^n$ and a decision variable, or design, denoted z, that is chosen following a policy Π. Experimental design theory provides a quantitative answer to the question: how informative are the observations y for estimating θ when using design z? We answered this question in Section 2.3.3 of Chapter 2, and we recall here how the information may be defined as a scalar functions of the Fisher information matrix.

Definition 5.3 [Fisher information and information gain] The observed Fisher information matrix (Gelman et al., 2004) of observation y at some parameter value θ is

$$I(z, y; \theta) = -\frac{\partial^2 \log p(y|z, \theta)}{\partial \theta^2} \quad \in \mathbb{R}^{n\times n}. \tag{5.3.4}$$

The D-optimal information gain for policy Π is defined as

$$g(\Pi|\theta) = \log\det\left(\mathbb{E}\left[I(z, y; \theta)\middle|\Pi, \theta\right]\right). \tag{5.3.5}$$

This information gain quantifies the information about θ provided by the observations with policy Π, and it may be interpreted as the volume of the confidence ellipsoid for the parameter vector θ. Several other functionals can be used instead of $\log\det$, leading to other optimality criteria. The D-optimality criterion benefits from a property of scale invariance (Pukelsheim, 2006) and is suitable for our online setting because it provides a simple rank-one update formula, as we will see in Section 5.4. A D-optimal policy is a maximizer of the information gain:

$$\underset{\Pi}{\text{maximize}} \quad g(\Pi|\theta). \tag{5.3.6}$$

From a Bayesian perspective, D-optimality minimizes the entropy of the expected posterior on θ, or equivalently the mutual information between the current prior and the expected posterior (Chaloner and Verdinelli, 1995).

5.3.3 Optimally informative inputs for exploration

In this section, we apply the optimal experimental design framework to our dynamical setting, where the Gaussian noise assumption and the linear structure of the model allow for exact computations. The observations are the state-action trajectory $z_{0:t} = (x_{0:t}, u_{0:t-1})$ and the policy Π chooses the inputs $u_{0:t-1}$.

Definition 5.4 [Gram matrix] An important quantity is the Gram matrix of the features, defined as

$$G_t = \sum_{s=0}^{t-1} {V_s}^\top V_s \quad \in \mathbb{R}^{n\times n}. \tag{5.3.7}$$

Proposition 5.1 Assume a linear model (5.3.1) for the data-generating distribution of the trajectory, with $c = 0$ for simplicity. Then, the maximum likelihood estimate of θ is

$$\hat{\theta}(z_{0:t}) = G_t^{-1} \sum_{s=0}^{t-1} {V_s}^\top x_{s+1} \tag{5.3.8}$$

and the observed Fisher information matrix is

$$I(z_{0;t}; \theta) = \frac{1}{\sigma^2} G_t. \tag{5.3.9}$$

Note that, as we saw in Chapter 4 for linear dynamics, I does not depend explicitly on θ but only on the observations for linear models.

It follows from (5.3.6) and Proposition 5.1 that, in our setting, D-optimal inputs solve the following optimal control problem:

$$\begin{aligned} \underset{(z_t)}{\text{maximize}} \quad & \log\det\Big(\sum_{t=0}^{T-1} {V_t}^\top V_t\Big) \\ \text{subject to} \quad & x_{t+1} = x_t + \mathrm{d}t f(x_t, u_t), \quad 0 \le t \le T-1, \\ & \|u_t\|^2 \le \beta^2, \end{aligned} \tag{5.3.10}$$

where we neglect the noise in the dynamics for simplicity, and recall that $V_t = V(z_t)$.

Remark 5.1 [Optimal design for matrix models] For models of the form (5.3.3), one can readily show that an equivalent objective is obtained by defining $V_t := \phi(z_t)^\top$ instead of $V_t = V(z_t)$.

5.3.4 Sequential learning

Linear models can be learned online with the recursive least squares formula for the estimator (5.3.8). Assuming for simplicity $d = 1$, $c = 0$ and denoting $v_t = V(z_t)^\top \in \mathbb{R}^n$, online learning takes the form

$$\theta_{t+1} = \theta_t - G_t^{-1} v_t ({v_t}^\top \theta_t - x_{t+1}) \tag{5.3.11a}$$

$$= \theta_t - \Gamma_t \nabla \ell_t(\theta_t). \tag{5.3.11b}$$

with the squared error loss $\ell_t(\theta) = \frac{1}{2} \times \|{v_t}^\top \theta - x_{t+1}\|_2^2$ and the matrix learning rate $\Gamma_t := G_t^{-1}$. Equation (5.3.11b) makes it transparent that recursive least squares is an online gradient descent step. The memory cost of learning is $\mathcal{O}(n^2)$ for the storage of G_t and θ_t. For $d > 1$, there is one update for each row, hence d updates per time step t.

5.4. Adaptive D-optimal exploration with FLEX

We adopt D-optimality (5.3.10) as an objective for our exploration policy. However, this problem is non-convex and providing a numerical solution is computationally challenging. Furthermore, the dynamics constraint is unknown and can only be approximated with the current knowledge of the dynamics, which is improved at each time step. Although previous approaches have opted for an episodic non-convex optimization with large time horizons (Schultheis et al., 2020; Wagenmaker et al., 2021), it is desirable to quickly update the choice of inputs at the same frequency as they are collected. In this section, we introduce FLEX (Algorithm 5.2), an adaptive D-optimal exploration algorithm with low computational complexity. In this section, we use the notations of linear models as defined in Section 5.3. We will show how this formalism extends to arbitrary, nonlinear models in Section 5.5.

5.4.1 One-step-ahead information gain

Since we seek minimal complexity and adaptivity, we choose to devote the computational effort at time t to the choice of the next input u_t only. We want to define an informativeness measure F_t for input u_t and solve a sequence of problems of the form

$$\begin{aligned} u_t \in \quad & \underset{u \in \mathbb{R}^k}{\operatorname{argmax}} \quad F_t(u) \\ & \text{subject to} \quad \|u\|^2 \leq \beta^2. \end{aligned} \tag{5.4.1}$$

As stated before, we also attach great importance to the computational time of solving (5.4.1).

The function F_t should quantify the information brought by u_t, for which we derived a mathematical expression in our information-theoretic considerations of Section 5.3. Therefore, we want the problem sequence (5.4.1) to be an approximation of the problem (5.3.10). As in Chapter 4, we propose a greedy approximation, which can be derived as follows. At time t, the past trajectory $z_{0:t}$ is known and the choice of u_t immediately determines the next state x_{t+1}. We define F_t as the predicted information gain truncated at $t+1$.

Definition 5.5 [Predicted information gain] Letting $x(u) := x_t + \mathrm{d}t f(x_t, u, \theta_t)$ be the one-step-ahead state prediction and $z(u) := (x(u), 0)$, we define

$$F_t(u) = U(z(u)) \tag{5.4.2}$$

with

$$U(z) = \log\det(G_t + V(z)^\top V(z)). \tag{5.4.3}$$

When V is of rank one, a simpler formula can be derived.

Lemma 5.1 [Determinant Lemma] By choosing a row $v := v^{(i)}$ for $1 \leq i \leq d \in \mathbb{R}^n$ of the feature matrix V and approximating $V^\top V \simeq vv^\top$, the expression for the information gain can be simplified as follows:

$$U(z) = \log\det G_t + v^\top(z) G_t^{-1} v(z). \tag{5.4.4}$$

We adopt the rank-one approximation of Lemma 5.1 in the remainder of this work. Note that the index i of row v can be either drawn randomly or chosen using prior knowledge: the i-th row of V is informative if the i-th component of the model is sensitive with respect to the parameters. One could also design a numerical criterion for this choice.

5.4.2 Computing D-optimal inputs

For linear systems, we have shown in Chapter 4 that a greedy D-optimal policy yields good exploration performance. However for nonlinear systems, the nonlinearity of $v(z)$ makes the maximization problem (5.4.1) challenging. To obtain a simpler optimization problem, we linearize the model with respect to the state. Intuitively, since we are planning between t and $t + \mathrm{d}t$, the corresponding change in the state is small so it is reasonable to use a linear approximation to the mapping $z \mapsto v(z)$.

Proposition 5.2 Linearizing our objective (5.4.2) to first order in $\mathrm{d}t$ yields the following approximation to the optimization problem (5.4.1):

$$\begin{aligned} \underset{u \in \mathbb{R}^m}{\text{maximize}} \quad & u^\top Q u - 2b^\top u \\ \text{subject to} \quad & \|u\|_2^2 \leq \beta^2, \end{aligned} \tag{5.4.5}$$

where Q and b are computed in terms of the Gram matrix $G := G_t \in \mathbb{R}^{n\times n}$ and the vector v and the derivatives $D := \partial v/\partial x \in \mathbb{R}^{n\times d}$, and $B := \mathrm{d}t\, \partial f/\partial u \in \mathbb{R}^{d\times m}$ evaluated at $\bar{z} := z(u=0)$, as follows

$$\begin{aligned} Q &= B^\top D^\top G^{-1} D B \quad \in \mathbb{R}^{k\times k}, \\ b &= -B^\top D^\top G^{-1} v \quad \in \mathbb{R}^k. \end{aligned} \tag{5.4.6}$$

Algorithm 5.2 Fast Linearized EXploration (FLEX)

input model f, horizon T, time step $\mathrm{d}t$, first estimate θ_0
output parameter estimate θ_T
for $0 \leq t \leq T-1$ **do**
  compute Q_t, b_t from (5.4.6)
  choose $u_t \in \underset{u^\top u \leq \beta^2}{\operatorname{argmax}}\, u^\top Q_t u - 2b_t^\top u$ (Proposition 5.2)
  observe $x_{t+1} = x_t + \mathrm{d}t\, f_\star(x_t, u_t) + \eta_t$
  compute $\ell_t(\theta) = \frac{1}{2}\|f(x_t, u_t, \theta) - (x_{t+1} - x_t)/\mathrm{d}t\|_2^2$
  update $\theta_{t+1} = \theta_t - \Gamma_t \nabla \ell_t(\theta_t)$ as in (5.5.6)
end for

Remark 5.2 [Linear dynamics] When the dynamics are linear as in Example 5.3.2, it follows from Example 5.3.2 and Remark 5.1 that $v(z) = z$, hence $\partial v/\partial x = I_{n,d}$ and $G_t = \sum_{s=0}^{t-1} z_s z_s{}^\top \in \mathbb{R}^{d\times d}$. Then, applying Proposition 5.2 yields $Q = B^\top G_t B$ and $b = B^\top G_t(I_d + \mathrm{d}tA)x_t$ and we find exactly the greedy optimal design algorithm introduced in Chapter 4.

Our exploration policy is summarized in Algorithm 5.2. Note that since it optimizes a greedy objective at each time step, it is adaptive by nature and it does not require the knowledge of the time horizon T. The following result shows that solving (5.4.5) can be achieved at low cost, ensuring that our policy is computationally efficient.

Proposition 5.3 Problem (5.4.5) can be solved numerically at the cost of a scalar root-finding and a $m \times m$ matrix eigenvalue decomposition.

5.5. From linear models to nonlinear models

For the cases when no prior information is available about the structure of the dynamics (see Example 5.2.2), it is desirable to generalize the policy derived in Section 5.4 to more complex models that are not linear in the parameters. In this section, we extend Algorithm 5.2 to generic, nonlinear parametric models. We assume that f is doubly differentiable with respect to z and θ.

5.5.1 Linearized model

The developments of Section 5.4 are based on the linear dependence of the model on the parameters. For nonlinear models, a natural idea is to make a linear expansion of the model: assuming that the parameter vector is close to a convergence value $\theta_\star$, we can linearize f to first order in $\theta - \theta_\star$:

$$f(z, \theta) \simeq f(z, \theta_\star) + \frac{\partial f}{\partial \theta}(z, \theta_\star) \times (\theta - \theta_\star). \tag{5.5.1}$$

In the limiting regime where this linear approximation would hold, f would be a linear model with features

$$V(z) = \frac{\partial f}{\partial \theta}(z, \theta_\star) \quad \in \mathbb{R}^{d\times n}. \tag{5.5.2}$$

Considering this analogy, we can generalize D-optimal experimental design (MacKay, 1992), and we aim to extend the results of Section 5.4 to nonlinear models.

5.5.2 Online exploration with nonlinear models

In our dynamical framework, we expect the approximation (5.5.1) to be increasingly accurate as more observations are collected, hence motivating the generalization of the exploration strategy developed

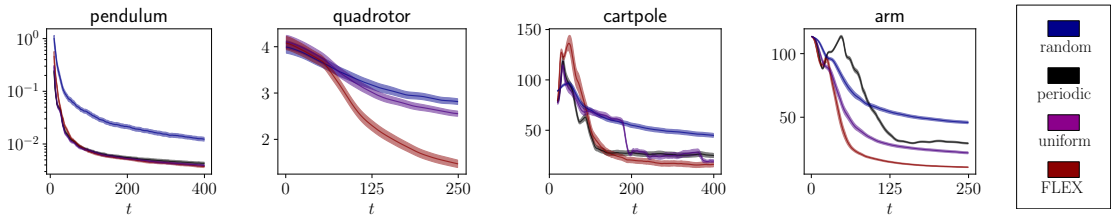

Figure 5.2. Evaluation error over time for different environments as a function of time t averaged over 100 trials.

in Section 5.4 to nonlinear models. The features of the linearized model (5.5.1) are unknown because the Jacobian (5.5.2) depends on the unknown parameter $\theta_\star$ in general. However, we can approximate $\theta_\star$ by the current estimate θ_t at each time step along the trajectory. By defining

$$V_t := \frac{\partial f}{\partial \theta}(z_t, \theta_t), \tag{5.5.3}$$

we extend the notion of the Gram matrix in Definition 5.4 as well as the optimal control problem (5.3.10) to nonlinear models. Note that since V_t depends only on (z_t, θ_t), the Gram matrix can still be computed online along the trajectory. Similarly, we extend the approach developed in Section 5.4 by defining

$$v(z) := \nabla_\theta f^{(i)}(z, \theta_t) \tag{5.5.4}$$

with $1 \leq i \leq d$ chosen as in Lemma 5.1. With these quantities defined, Algorithm 5.2 is extended to arbitrary models.

Remark 5.3 [Consistency with linear models] When the model is exactly linear in the parameter as in (5.3.1), the Jacobian is $\partial f / \partial \theta = V(z)$ so (5.5.3) is consistent with the definition of V_t in Definition 5.1. Therefore, (5.5.3) can be viewed as a generalization of the optimal experimental design exploration of Section 5.4 to nonlinear models.

5.5.3 Computational perspective

In Algorithm 5.2, we need to compute $D = \partial v / \partial x$, which amounts to computing the derivatives of f in both θ and z:

$$\frac{\partial v}{\partial x} = \left(\frac{\partial^2 f^{(i)}}{\partial x \partial \theta} \right) \in \mathbb{R}^{n \times d}. \tag{5.5.5}$$

For neural networks, this matrix can be computed by automatic differentiation. The cost of solving (5.4.5) does not depend on n so the computation of D becomes the computational bottleneck for large models. The complexity of the latter operation is $\mathcal{O}(nd)$ with automatic differentiation.

5.5.4 Sequential learning

We train nonlinear models using online gradient descent:

$$\theta_{t+1} = \theta_t - \Gamma_t \nabla \ell_t(\theta_t), \tag{5.5.6}$$

which extends (5.3.11b). For neural networks, online learning with an adaptive learning rate, corresponding to scalar $\Gamma_t = \gamma_t \in \mathbb{R}$, is known to be effective (Bottou, 2012; Kingma and Ba, 2015). The gradient step can be averaged over a batch for smoother learning, at the cost of storing a small amount of data points.

5.6. Experiments

We run several experiments to validate our method. Our code and a demonstration video are available at https://github.com/MB-29/exploration. More details about the experiments can be found in Appendix C.

5.6.1 Exploration benchmark

We first test our policy on various nonlinear environments from classical control, covering different values for d and m. We compare its performance in terms of sample efficiency to that of various baselines. The agents have the same learning model, but different exploration policies. Random exploration draws inputs at random. For pendulum-like environments, a periodic oracle baseline excites the system at an eigenmode, yielding resonant trajectories of large amplitude. A baseline called "uniform" maximizes the distance of the trajectory points in the state space: it optimizes objective (5.4.2) with $U(x) = \frac{1}{2}\sum_{s=0}^{t} \|x - x_s\|^2$.

Experimental setup The learning models include various degrees of prior knowledge on the dynamics. A linear model is used for the pendulum, and neural networks are used for the other environments. The Jacobians of Proposition 5.2 are computed using automatic differentiation. At each time step, the model is evaluated with (5.2.3) computed over a fixed grid.

Results The results are presented in Figure 5.2. Our algorithm is sample-efficient and it outperforms the baselines in all the environments. We also display the trajectories obtained with our policy in phase space in Figure 5.3. Not only does FLEX produce informative trajectories of large amplitude, but more specifically it devotes energy so as to explore regions with higher uncertainty, unlike the baselines. Although the policy optimizes the information in a greedy fashion, it interestingly produces inputs with long-term temporal coherence. This is illustrated in our demonstration video. High-dimensional exploration is tackled in Appendix B.

5.6.2 Tracking of time-varying dynamics

In real systems, dynamics may vary over time and an adaptive exploration is crucial for accommodating to changes in the environment. We test the adaptivity of our method by exploring a time-varying system, and compare to an episodic agent. The system is a central, repulsive force field centered on a star, moving around a circle uniformly at a period T. The agent is a spaceship and the control variable is the acceleration. The force field varies significantly only in the vicinity of the star: the spaceship learns information about the dynamics only when it is close, and hence needs to track the star.

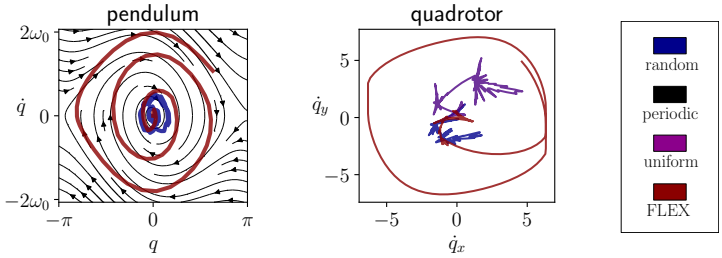

Figure 5.3. Trajectories in phase space.

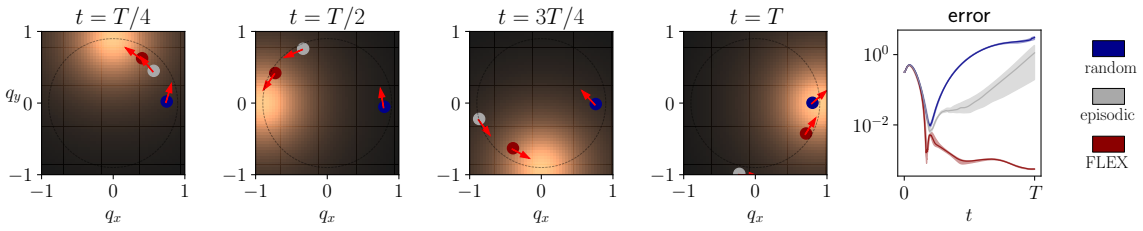

Figure 5.4. Time-varying force field system. **Left** Trajectories of the agents in the position space (q_x, q_y). The color gradient represents the force field and the red arrows represent the actions of the agents. **Right** The error curves over time, averaged over 100 trials.

Experimental setup The model learns the center and the radius of the force field, yielding a nonlinear parameterization (see Appendix C.2). With this model, three agents learn the dynamics: a random policy, FLEX, and an episodic agent that plans D-optimal inputs by solving (5.3.10) over a time horizon of $T/20 = 50$ repeatedly. At each time step, the model is evaluated in the parameter space with the distance to the real system parameters at the current time.

Results The trajectories and the error curves are presented in Figure 5.4. Although the episodic agent initially follows the star, it eventually loses track of the dynamics because of the delay induced by planning over a time interval. Our adaptive policy, on the other hand, successfully explores the dynamics. Even though the planning of inputs has linear time complexity in T in both cases, we observe a slowdown by a factor 100 for the episodic agent. We believe that this experiment on this toy model illustrates the relevance of an adaptive exploration policy in realistic settings.

5.6.3 From exploration to exploitation

The goal of exploration is ultimately to obtain an accurate model for the system for model-based control. In order to validate the relevance of our approach to this framework, we evaluate the model learned during exploration on model-based control tasks. We compare it to the recent active exploration algorithms RHC and MAX (Schultheis et al., 2020; Shyam et al., 2019) and to the model-free reinforcement learning algorithm SAC (Haarnoja et al., 2018).

Experimental setup We experiment on the pendulum and the cartpole of the DeepMind control suite (Tunyasuvunakool et al., 2020), for which we added noise. Throughout exploration, the learned model is evaluated using both (5.2.3) and with the exploitation cost achieved by a model-based control algorithm on the swingup task. The experimental details can be found in Appendix C.3 and in (Schultheis et al., 2020) along with the performance of the algorithms used for comparison. The control task is considered solved when the cost is lower than a value that we have chosen arbitrarily based on simulations.

Results Our results are presented in Table 5.1 and in Figure 5.4. Our algorithm is sample-efficient in terms of exploitation, as it allows for a model-based control algorithm to solve the task faster than the other baselines. Its computational cost is low.

5.6.4 Exploration in a high-dimensional environment

We propose an additional experiment showing the behaviour of our algorithm in high dimension. The nonlinear system we consider is a chain of n coupled damped pendulums, with unknown friction. Each pendulum is coupled with its two nearest neighbors. Only the first pendulum is actuated and the

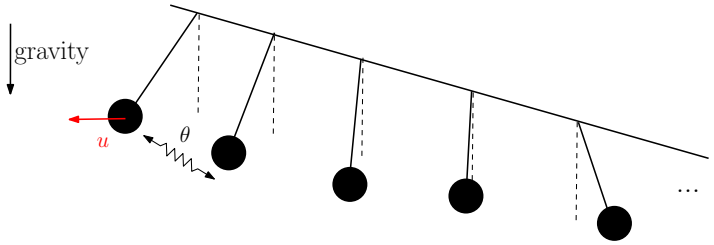

Figure 5.5. Chain of n coupled pendulums.

motion of the rest of the pendululms is due to the successive coupling (Bitar et al., 2017). Exploration of the system consists in finding the friction forces on the pendulums. The system is illustrated in Figure 5.5. We believe that this system is illustrative for the typical setup of system identification. In robotics for example, the experiment seeks to measure the parameters of a humanoid constituted of large number of joints. Furthermore, the system we propose poses both challenges of large dimension and underactuation. Indeed, the dimension of the state space is $d(n) = 2n$, and only one of the n pendulums is actuated, the motion being propagated from neighbor to neighbor. This experiment allows us to monitor both the sample efficiency and the computational cost of FLEX in a challenging setting of large dimension.

Experimental setup The system is modeled with a linear model with unknown friction coefficients $\theta \in \mathbb{R}^n$ from observations $x_t \in \mathbb{R}^{2n}$. We define the sample complexity as the number of samples required to obtain a 10^{-2} parameter error. For different values of n, we measure the sample complexity and the computational time of Random and FLEX.

Results We provide our results in Table 5.3. These results capture the behavior of the algorithm when the dimension d grows. Our algorithm FLEX seems to reach a linear sample complexity with respect to d, with reasonable computational time, whereas random exploration fails to explore and yields super-linear sample complexity. Our results suggest that despite larger computational cost, FLEX remains sample efficient and competitive in high-dimensional environments.

5.7. Related work

Active linear system identification Exploration of linear systems has been studied extensively in the control community (Goodwin and Payne, 1977) and more recently in the machine learning community (Simchowitz et al., 2018; Jedra and Proutiere, 2020; Wagenmaker et al., 2021; Even, 2023). From the practical point of view, an online and fast exploration policy is introduced in (Blanke and Lelarge, 2022). Although linear models may describe nonlinear systems locally, they are not sufficient for designing complex model-based control laws (Tassa et al., 2014).

Table 5.1. Number of observations required to solve the swingup task (first row) and average computation time per observation (second row) for the pendulum (top rows) and the cartpole (bottom rows).

Method	RAND	MAX	SAC	RHC	FLEX
samples	> 2k	2000	> 2k	500	50
compute	1	100	2	8	4
samples	> 2k	> 2k	> 2k	600	300
compute	1	20	1.5	2	1.6

Nonlinear exploration Several methods have been proposed to learn nonlinear dynamics and to model uncertainty, including model-error control synthesis (Crassidis, 1999), Gaussian processes (Buisson-Fenet et al., 2020), Random Fourier Features (Schultheis et al., 2020), and neural networks ensembles (Shyam et al., 2019; Sekar et al., 2020). The latter approaches are computationally heavy, which is an issue slow for real-time applications. From a theoretical point of view, exploration with nonlinear models is addressed by Mania et al. (2020) and Wagenmaker et al. (2023). These works introduce theoretical bounds for the parameter estimation error, and derive asymptotically optimal algorithms. Unlike our online policy, the proposed input design algorithm plans over exponentially large epochs, which is computationally prohibitive for real-time applications.

Experimental design for neural networks The extension of optimal experimental design to neural networks is proposed in (MacKay, 1992) for static systems, and in (Cohn, 1993) for dynamical systems with an offline algorithm and a focus on G-optimal designs.

5.8. Conclusion

We proposed an exploration algorithm based on D-optimal design, running online and adaptively. Our experiments demonstrate its sample efficiency both in terms of exploration and exploitation, its low computational cost and its ability to track time-varying dynamics. These results are encouraging for applications on real systems.

Although it is not the focus of our work, the online learning rule conditions the quality of exploration. In particular, the agent should be able to learn with bounded memory to meet the computational requirements of embedded systems. While we used a rather naive online learning algorithm, recent advances in the communities of machine learning and control are promising for learning dynamics online (Min et al., 2022). The computational cost of our method is dominated by the calculation of derivatives. Automatic differentiation is an active research field and we believe that progress in that direction can be made to reduce this cost.

It would be interesting to generalize FLEX to the more realistic setting of a partially observed state model (Goodwin and Payne, 1977). Another research direction is to use our exploration objective as an exploration bonus in the exploration-exploitation trade-off.

Interestingly, the information-theoretic considerations of active learning can also be adapted for the inverse problem of unlearning, where the goal is to forget about certain data points rather that to actively select new data. This subject has recently sparked interest in the deep learning community (Bourtoule et al., 2021; Fraboni et al., 2024).

Table 5.2. Performance of algorithms Random and FLEX for the exploration of the n coupled pendulums.

n	2	5	10	20	50
d	4	10	20	40	100
Random					
sample complexity	20	100	200	500	> 1000
computational time	1	2	6	60	250
FLEX					
sample complexity	10	50	100	200	500
computational time	3	5	10	80	350

Chapter 6

Incremental neural data assimilation

Data assimilation is a central problem in many geophysical applications, such as weather forecasting. It aims to estimate the state of a potentially large system, such as the atmosphere, from sparse observations, supplemented by prior physical knowledge. The size of the systems involved and the complexity of the underlying physical equations make it a challenging task from a computational point of view. Neural networks represent a promising method of emulating the physics at low cost, and therefore have the potential to considerably improve and accelerate data assimilation. This chapter presents preliminary results of a neural approach to data assimilation. We introduce a deep learning framework where the physical system is modeled as a sequence of coarse-to-fine Gaussian prior distributions parametrized by a neural network. This allows us to define an assimilation operator, which is trained in an end-to-end fashion to minimize the reconstruction error on a dataset with different observation processes. We illustrate our approach on chaotic dynamical physical systems with sparse observations, and compare it to traditional variational data assimilation methods.

Chapter organization This chapter is organized as follows. Section 6.1 presents the data assimilation problem and its challenges. Section 6.2 poses a Bayesian statistical framework of this problem, and introduces the Gaussian least-squares interpolation method. In Section 6.3, we present a neural architecture for data assimilation. We experiment on simulated physical systems in Section 6.4. Section 6.5 compares our contribution with related works. Section 6.6 summarizes our contributions and discusses its limitations and perspectives.

This chapter is based on the article *Incremental Neural Data Assimilation* (Blanke et al., 2024), accepted at the ICML 2024 AI for Science workshop (ICML 2024 AI4Science).

6.1. Introduction

Artificial intelligence is transforming many fields, and has a growing number of applications in industry. In the sciences, it has the potential to considerably accelerate the scientific process. Geophysics and weather forecasting are areas where deep learning is particularly active, with recent months seeing an explosion in the number of large neural models for the weather forecasting problem (Pathak et al., 2022; Lam et al., 2022; Hoyer et al., 2023), building on reanalysis datasets such as ERA5 (Muñoz-Sabater et al., 2021) for training. In this work, we focus on the data assimilation problem that underpins weather forecasting: tomorrow's weather forecast is based on today's weather conditions, which are not directly measured, but are estimated from few observations. Data assimilation is the inverse problem of estimating the geophysical state of the globe on the basis of these sparse observations and of prior knowledge of the physics. The estimated state then serves as the starting point for forecasting. While deep learning models are revolutionizing the forecasting problem, they have yet to be applied operationally to data assimilation.

The application of neural networks to inverse problems is an active area of research. The general idea consists in training a neural network to reconstruct a signal, using for training examples a dataset of simulated physical states serving as ground truth. For the data assimilation problem, several approaches have been proposed to incorporate a deep learning in the loop. Arcucci et al. (2021) propose a sequential scheme where a neural network is trained at regular time steps to combine data assimilation and the forecasting model. Recently, the success of diffusion models for imaging (Ho et al., 2020) has led to the development of so-called "plug and play" methods, where the neural network is trained to learn a prior (Laumont et al., 2022). Once trained, the neural prior can be used to solve a large number of inverse problems. In this line of work, Rozet and Louppe (2023) proposed a data assimilation method based on a diffusion model. Another type of approaches called "end-to-end" aim at directly training a neural network to minimize the reconstruction error. They have the benefit of training the network directly on the task of interest, but the versatility of the trained model with respect to the different observational processes is challenging. An end-to-end neural reconstruction algorithm is proposed in (Fablet et al., 2021b), and aims to learn the prior distribution of the signal by defining the reconstruction as a maximum a posterior estimate, leading to a bi-level optimization problem. However, the complex prior induced by the neural network may hamper the convergence of this estimate, as it relies on non-convex optimization. Instead, we explore a model where the prior has a sufficiently simple structure to guarantee a convex posterior distribution.

Contributions In this chapter, we present a neural method for data assimilation. We introduce a data assimilation operator parametrized by a neural Gaussian prior, that is designed to locally improve the likelihood of an estimate. Our model is trained to minimize the reconstruction error in an end-to-end fashion. We show how this operator may be iterated to reconstruct complex signals. The effectiveness of our method is demonstrated on simulated nonlinear physical systems. We also show how our method may be used to enhance traditional data assimilation methods.

6.2. The data assimilation inverse problem

In this section, we recall the mathematical framework of data assimilation that we introduced in Chapter 2, Section 2.4.

The aim of data assimilation is to reconstruct a state $x \in \mathbb{R}^d$ from partial noisy measurements $y \in \mathbb{R}^m$ of that state (Bouttier and Courtier, 2002; Bocquet et al., 2014). For meteorological applications, for instance, the state x represents the physical quantities on a grid representing the globe, and the observations y are partial measurements, from different sources: in situ measurements, weather balloons, satellites, ... These measurements may be very sparse, with an observation rate m/d that may be of the order 1%, so we cannot generally hope to recover the state as a function of the observations

alone. Indeed, for a given observation vector y, a large number of states are compatible, making data assimilation an inverse problem. To reconstruct the state, we need to supplement the partial observations with another source of prior information on the state, which comes from our physical or statistical knowledge of the problem.

The data assimilation problem is then as follows. Given partial observations y and prior information on the state, the aim is to estimate the most probable underlying state x. The Bayesian probabilistic framework lends itself well to the mathematical formalization of the problem : the theoretical information about the state physics is captured by a prior distribution $x \sim p(x)$, and the noisy, partial observations of x can be modeled as $y|x \sim h(x) + \xi$, with h the observation process, and an unbiased additive noise that is typically assumed to be Gaussian $\xi \sim \mathcal{N}(0, R)$ and independent of x. Then, data assimilation can be seen as the estimation of the state maximizing the state posterior distribution $p(x|y) = p(x)p(y|x)/p(y)$. Under the assumption of Gaussian observational noise, this can be formulated as the following minimization problem

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad U(x) + \frac{1}{2}\|h(x) - y\|^2_{R^{-1}}, \tag{6.2.1}$$

with $U(x) = -\log p(x)$, and where we have adopted the notation $\|z\|_C = \sqrt{z^\top C z}$ for a positive definite matrix C. We assume for simplicity that the observation function h is known, although it may be only partially known in some cases, such as remote sensing (Liang, 2005) or medical imaging (Rangayyan and Krishnan, 2024).

Problem size For weather prediction, the state x represents the geophysical variables on a large spatial grid. It is hence a signal of very high dimension with typically $d \sim 10^6$ or even $d \sim 10^9$. The size of the data assimilation problem makes the computations and memory costs very heavy, severely limiting the computational budget of any numerical method. In the development of new learning-based methods, it is essential to keep this computational constraint in mind if we hope to scale up to real-size systems.

6.2.1 Least-squares Gaussian interpolation

The first approach considered for data assimilation is naturally that of a linear-Gaussian model, as we saw in Section 2.4.2 of Chapter 2. Assuming a Gaussian a priori on the state $x \sim \mathcal{N}(\mu, P)$ and a linear observation function $h(x) = Hx$, with $H \in \mathbb{R}^{m\times d}$, the variational Bayesian formulation for data assimilation (6.2.1) becomes a quadratic least-squares problem:

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad \frac{1}{2}\|x - \mu\|^2_{P^{-1}} + \frac{1}{2}\|Hx - y\|^2_{R^{-1}} \tag{6.2.2}$$

whose maximum a posteriori solution takes the form

$$x_{\text{MAP}}(y; \mu, P) := \mu + K(y - H\mu), \tag{6.2.3}$$

with the H-dependent Kalman gain

$$K = PH^\top(HPH^\top + R)^{-1} \ \in \mathbb{R}^{m\times d}. \tag{6.2.4}$$

In the remainder of this work, the dependence with respect to H is implicitly assumed in all quantities that depend on the observation vector y.

For meteorological applications, the state x that is optimized for is a snapshot of the set of geophysical variables at a given time, when the observations have been collected. The background term μ is the forecast of this state from the past observations.

Computational cost For large-scale applications, solving (6.2.2) by computing the closed-form expression (6.2.3) yields a $\mathcal{O}(m^3 + dm)$ complexity in general, as it involves solving a $m \times m$ linear system and computing a matrix-vector products of size $d \times m$. In operational geophysical applications, this cost may be a bottleneck as d and m may reach prohibitively large values. To avoid such costs, (6.2.2) is solved by such as conjugate gradient (Fletcher and Reeves, 1964). In the data assimilation community, this variational approach for the estimation of a large-scale geophysical spatial state is called 3D-Var (Courtier et al., 1998).

6.2.2 Spatio-temporal data assimilation

So far, the prior knowledge of the state has taken the form of a Gaussian distribution, which can capture the proximity of the searched state to an estimate, and the correlations of one state variable to another. Least squares interpolation then searches for the state most faithful to the data, within a fluctuation zone around the estimate. Although simple and analytically solvable, this approach does not use signal physics equations as prior information.

In the 1990s, the quality of data assimilation analyses improved significantly by incorporating a physical model to the reconstruction prior, leading to the state-of-the-art variational assimilation algorithm 4D-Var (Le Dimet and Talagrand, 1986). This algorithm is a generalization of 3D-Var to time-distributed observations, where the estimated signal x is a temporal sequence of the spatial geophysical state on a time window, *i.e.* a trajectory, rather than one single snapshot. The temporal dimension allows formulating the system's dynamical equations as a constraint for the signal. The reconstruction algorithm is applied sequentially on a sliding time window, in combination with a forecasting model, to produce regularly updated estimates of the meteorological variables. Alongside 4D-Var, other algorithms exist for data assimilation of dynamical systems, including sequential methods such as the celebrated Kalman filter, and its extensions to nonlinear models (Jazwinski, 2007), and to ensembling (Evensen, 2003). In this work, we focus on the so-called weak-constraint 4D-Var algorithm (Trémolet, 2007; Fisher et al., 2012), which we briefly explain next. Weak-constraint 4D-Var has the advantage being naturally related to the Bayesian formulation (6.2.1), and is used in operational systems.

For simplicity, we abstract from the time dimension in our mathematical formalism, and still denote the spatio-temporal signal as $x \in \mathbb{R}^d$. The knowledge of a physical dynamical model materializes as knowledge of a prior distribution $U(x)$ in (6.2.1), which can be computed and differentiated through with respect to x. In geophysics, this model is typically a fluid dynamics simulator, and its gradients are computed using the adjoint method (Talagrand and Courtier, 1987). Hence, the resulting $U(x)$ is more complex and more informative than a Gaussian prior, but comes with heavy computational costs. In the remained of this work, we assume that the observational processes are linear: $h(x) = Hx$. In practice, h is nonlinear and is sequentially approximated by its linear approximation. We argue that linearizing the physical model is computationally far more expensive than linearizing the observational process, and hence that considering only linear observations does not severely restrict the problem generality.

The weak-constraint 4D-Var algorithm aims to minimizing (6.2.1) by a Gauss-Newton descent algorithm (Gauss, 1877), with line-serach correction (Nocedal and Wright, 1999). More precisely, a sequence of estimates $\{z_k, 1 \leq k \leq L\}$ approximating the reconstruction signal is iteratively computed. At each iteration k, the objective function is approximated by its quadratic expansion in the vicinity of $z_k \in \mathbb{R}^d$. Specifically, the prior term is approximated as

$$U(x) \simeq U(z) + \nabla U(z)^\top (x - z) + \frac{1}{2}(x - z)^\top \nabla^2 U(z)(x - z). \tag{6.2.5}$$

We may express expansion (6.2.5) as a Gaussian log-likelihood:

$$U(x) \simeq \frac{1}{2}\|x - \mu(z)\|^2_{P(z)^{-1}}, \tag{6.2.6}$$

with

$$P(z) \simeq \nabla^2 U(z)^{-1}, \tag{6.2.7a}$$

$$\mu(z) = z - P(z)^{-1} \nabla U(z), \tag{6.2.7b}$$

the approximation above referring to the gradient-Hessian approximation.

Weak-constraint 4D-Var is described in Algorithm 6.1. We see that the sequence of estimates (z_k) is iterated with a recursion of the form

$$x_k = A(z_k, y) \tag{6.2.8a}$$

$$z_{k+1} = z_k + \alpha_k (x_k - z_k), \tag{6.2.8b}$$

where assimilation operator A improves the current estimate z using the observations and the local approximation of the model, by performing a local optimal interpolation:

$$A(z, y) = x_{\mathrm{MAP}}(y; \mu(z), P(z)). \tag{6.2.9}$$

Limitations The 4D-Var algorithm represents the state of the art for data assimilation in geophysics, and is deployed in operational meteorological centers. Its main limitation is the high computational cost of simulating and differentiating through the physical model. In Algorithm 6.1, each computation of P_k and μ_k comes with a large cost in addition to the cost of computing (6.2.9), hence limiting the method's accuracy. Note that this method may also be viewed as an application of the iterative Kalman smoother (Bell and Cathey, 1993; Ménard and Daley, 1996; Fisher et al., 2005; Mandel et al., 2013). As is well known, an additional limitation of this method is that the non-convexity of U may lead to a complex minimization landscape, making the descent algorithm likely to be stuck in local minima (Gratton et al., 2007; Mandel et al., 2013). In the next section, we propose to overcome these limitations by learning operator A from data.

6.3. Neural data assimilation

Deep neural networks hold great promise for solving inverse problems (Bai et al., 2020), as they can help recover the corrupted signal by using the large amount statistical information acquired on a training dataset. For the data assimilation problem in meteorology or oceanography, the ground truth signals x are not available as the geophysical systems are not observed. However, a promising research direction consists in training a deep neural network to learn a prior on high-resolution simulations, or on the reanalysis datasets such as ERA5, like neural weather models (Ben Bouallègue et al., 2024).

Deep learning approaches to inverse problems may be separated in two categories (Mukherjee et al., 2021). A first category of algorithms aims to learn a prior $U(x)$ from a training dataset, using a neural network, independently of the inverse problem. Once trained, the learned prior can be adapted to a reconstruction algorithm to reconstruct the signal. These algorithms are often called "plug-and-play", as the trained neural prior can be used for any downstream inverse problem. In a second category of algorithms, referred to as "end-to-end" learning algorithms, the neural network is explicitly trained to solve the inverse problem. In this case, the training consists of minimizing the neural network's reconstruction error, based on a dataset of state and observations pairs $(x^{(i)}, y^{(i)})$.

One challenge in training end-to-end algorithms is the multiplicity of possible observation processes: the trained neural network must be compatible with all possible (x, y), and hence with varying observation processes H, with different dimensions m for the observations. It should therefore model only the prior distribution $U(x)$, and not depend directly on the observation process H.

6.3.1 Neural assimilation operator

We adopt an end-to-end learning approach, and we aim at learning a neural assimilation algorithm by minimizing a reconstruction error. We observe that, unlike other inverse problems such as image inpainting, data assimilation often starts with a first physically plausible estimate z of the unknown state. Therefore, rather than learning to interpolate the observations from scratch, we train a neural network to improve the state estimate given z. Drawing inspiration from the 4D-Var algorithm, we learn an assimilation operator $A(z, y; \theta)$, where θ denotes the parameter vector of a neural network. As in (6.2.6), we model the local prior distribution conditioned on z as a Gaussian prior

$$x|z \sim \mathcal{N}(\mu(z;\theta), P(z;\theta)), \tag{6.3.1}$$

where $\mu(z;\theta)$ and $P(z;\theta)$ are trainable neural networks. Given this Gaussian prior, the observations are incorporated by solving the least-squares interpolation (6.2.2):

$$A(z, y; \theta) = x_{\mathrm{MAP}}(y; \mu(z;\theta), P(z;\theta)). \tag{6.3.2}$$

Versatility As we pointed out, the trained neural network should be compatible with arbitrary observation processes. By formulating it as the solution of a y-dependent interpolation problem, our assimilation operator (6.3.2) is defined for any observation process (H, y), although the underlying neural networks models only the prior distribution. In particular, the neural networks involved depend neither on y, nor on H: the neural assimilation operator (6.3.2) combines the observations with a neural prior (6.3.1) of the state through the computation of a maximum likelihood estimator, and this computation is valid for any (H, y) pair for the same neural network. At prediction time, the trained neural networks $\mu(z;\theta), P(z;\theta)$ may be used to assimilate a new observation y obtained from an arbitrary observation process H by solving (6.3.2).

Training Given a dataset $(x^{(i)}, y^{(i)}, z_0^{(i)})$ consisting of signals $x^{(i)}$ and partial observations $y^{(i)}$ obtained from different observation processes $H^{(i)}$, supplemented with coarse estimates $z^{(i)}$ of the signal, the neural prior (6.3.1) is trained to minimize the reconstruction error with the following objective:

$$\begin{aligned} \underset{\theta\in\mathbb{R}^n}{\text{minimize}} \quad & \sum_{i=1}^{N} \|A(z^{(i)}, y^{(i)}; \theta) - x^{(i)}\|^2 \\ \text{with} \quad & A(z, y; \theta) = x_{\mathrm{MAP}}(y; \mu(z;\theta), P(z;\theta)). \end{aligned} \tag{6.3.3}$$

We train our model by minimizing (6.3.3) using stochastic gradient descent, with the ADAM optimizer (Kingma and Ba, 2015). This training objective takes the form of a bi-level optimization problem. Solving the inner optimization problem involves computing the optimal interpolation (6.2.3), which is computed solving a linear system of size m. We need to propagate the gradients with respect to θ through this no-trivial operation during training. This may be handled by implicit differentiation, allowing to compute the gradients of the solution with respect to θ, without explicitly inverting the system's matrices (Johnson, 2012).

This training objective is similar to that of (Fablet et al., 2021b), where a neural interpolator called 4DVarNet is used to learn both the global prior $U(x)$ and the minimization algorithm of (6.2.1), rather than a local operator $A(z, y) \mapsto x$. In our case, however, the inner optimization problem (6.2.1) can be solved explicitly because the cost is quadratic. In contrast, it is only approximately solved in the case of 4DVarNet, due to the non-convexity of the inner cost.

Computational cost As we mentioned, the large size of the targeted physical systems requires carefully considering the computational cost of the data assimilation methods. We model P as a band matrix, hence limiting both the memory storage to a $\mathcal{O}(d)$ cost and the computational complexity of solving the linear system in (6.2.3) using the Thomas algorithm (Datta, 2010). This structure also imposes a temporal structure in the signal.

6.3.2 Incremental neural data assimilation

Since our assimilation operator is trained to reconstruct the signal from a coarse approximation, a one-shot reconstruction is likely to yield blurry results. To improve reconstruction, we may iterate this operator, with the aim of progressively improving the reconstruction signal. Building on the recent advances of cold diffusion (Bansal et al., 2024), we propose an iterative strategy aiming at reconstructing the signal in a coarse-to-fine fashion. We introduce a scalar temperature parameter $0 \leq s \leq 1$ modeling the coarseness of the reconstruction, and we allow our neural prior to depend on s as $\mu(z;\theta,s)$, $P(z;\theta,s)$. Intuitively, the prior should be coarser for larger values of s, and become sharper and more local as $s \to 0$. We provide estimates z_k at different temperature levels $\{s_1 \geq \cdots \geq s_L\}$ as linear interpolations between z_0 and $z_L := x$:

$$z_k^{(i)} = s_k z_0^{(i)} + (1 - s_k) z_L^{(i)}. \tag{6.3.4}$$

Our training objective is adapted as

$$\underset{\theta \in \mathbb{R}^n}{\text{minimize}} \quad \sum_{k=1}^{L} \sum_{i=1}^{N} \|A(z_k^{(i)}, y^{(i)}; \theta, s_k) - x^{(i)}\|^2. \tag{6.3.5}$$

At prediction time, the signal is reconstructed by iteratively applying $A(z, y; \theta, s)$ following the sampling algorithm introduced in (Bansal et al., 2024). We provide a detailed description of our iterative reconstruction method in Algorithm 6.2.

Algorithm 6.1 Incremental weak-constraint 4D-Var

input observation vector $y \in \mathbb{R}^m$, observation matrix H, iteration number L, initial estimate z_0, tangent linear physical model μ, P
output state estimation z_L
initialize $z_0 := x_0$
for $0 \leq k \leq L-1$ **do**
  compute $P_k := P(z_k), \quad \mu_k = \mu(z_k)$
  estimate $x_k = x_{\text{MAP}}(y; \mu_k, P_k)$
  compute line search parameter α_k
  update $z_{k+1} = z_k + \alpha_k(x_k - z_k)$
end for

Algorithm 6.2 Incremental neural data assimilation

input observation vector $y \in \mathbb{R}^m$, observation matrix H, iteration number L, initial estimate z_0, neural models μ, P, trained weights θ
output state estimation z_L
initialize $z_0 := x_0$
for $0 \leq k \leq L-1$ **do**
  compute $P_k := P(z_k; \theta, s_k)$, $\mu_k = \mu(z_k; \theta, s_k)$
  estimate $x_k = x_{\text{MAP}}(y; \mu_k, P_k)$
  compute temperature parameter s_k
  update $z_{k+1} = z_k + s_k(x_k - z_0)$
end for

6.4. Experiments on physical systems

In order to evaluate the performances of our data assimilation algorithm, we experiment on two simulated dynamical systems: the pendulum and the Lorenz 63 dynamical systems. We train our neural model on a dataset generated from the dynamical system with different trajectories x sampled from random initial conditions, and different observation processes, leading to various (x, y) pairs for the same x. Our JAX implementation of our neural assimilation algorithm is available online at `https://github.com/MB-29/assimilation`.

Architecture We take for $\mu(z;\theta,s)$ and $P(z;\theta,s)$ two fully-connected neural networks of depth 4 and width 32. The dependence with respect to s is implemented as a positional embedding. The $d \times d$ matrix P is modeled as a band matrix with bandwidth $b = 2\varphi$, with φ the phase space dimension.

Baselines We compare our neural assimilation algorithm with various baseline. Each method starts from a first guess estimate z_0 of the signal, computed by performing a Gaussian interpolation from

the observations (see below). We implement the weak-constraint 4D-Var algorithm as a Levenberg-Marquardt Gauss-Newton Algorithm using the JAXopt implementation (Blondel et al., 2021), and the Diffrax library for differentiating through differential equation solvers (Kidger, 2021). As an ablation, an "unconditional" cold diffusion model is trained to restore the signal by minimizing objective (6.3.5) without the information provided by the observations. It is then applied following Algorithm 6.2 just as our neural assimilation algorithm, without using y. The resulting reconstructed signal depends on the observations only through the first estimate z_0, which is computed to match y, but the neural network is trained to compute the next iterates by increasing only the prior term $U(x)$ in (6.2.1), not the observation likelihood.

6.4.1 Pendulum

We start with the pendulum, which is arguably one of the simplest nonlinear physical systems. Importantly, the pendulum is simple enough to be decently approximated by linear dynamics. It can be shown that a linear dynamical model with Gaussian model noise yields a Gaussian prior distribution for the trajectory x. Therefore, a natural first guess for the pendulum consists in the quadratic least-squares estimator $z_0 := x_{\mathrm{MAP}}(y; \mu_0, P_0)$, where μ_0 and P_0 can be computed analytically as a function of the initial condition distribution and the pendulum's linear model. Starting from this estimate, we run the baselines and our neural assimilation algorithm.

Data We generate discrete trajectories $x^{(i)}$ of $T = 100$ time steps from the nonlinear pendulum dynamics with random initial conditions sampled in phase space, which is of dimension 2, hence $d = 2 \times 100 = 200$. The observations are generated by observing the pendulum's position at sparse time steps, with Gaussian observation noise $\xi \sim \mathcal{N}(0, \rho^2 I_m)$, with $\rho = 0.01$.

Experimental setup We train an adaptation operator to reconstruct the signal in one shot from z_0, following (6.3.3). At prediction time, we apply the trained neural assimilation map $A(z; y; \theta)$ to z_0 on a separate independent dataset.

Results Reconstruction samples are presented in Figure 6.1. While the linear model fails at reconstructing the trajectories outside the linearization zone (angle and momentum close to 0), one application of our neural assimilation operator accurately reconstructs the signal. The performances of the various methods are shown in Table 6.1. Although the pendulum is simple enough for all the methods to accurately reconstruct the signal, we see that the computational gain offered by a train neural network is considerable with respect to computing the physical model.

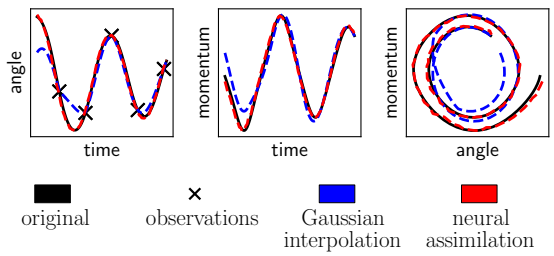

Figure 6.1. Reconstructed trajectories for the pendulum.

6.4.2 Lorenz 63

We now turn to a more complex system. The Lorenz system is a simplified physical model for atmospheric convection (Lorenz, 1963). Three variables are governed by the following set of coupled

nonlinear ordinary differential equations:

$$\begin{aligned} \frac{\mathrm{d}u_1}{\mathrm{d}t} &= \sigma(u_2 - u_1) \\ \frac{\mathrm{d}u_2}{\mathrm{d}t} &= \rho u_1 - u_2 - u_1 u_3 \\ \frac{\mathrm{d}u_3}{\mathrm{d}t} &= u_1 u_2 - \beta u_3. \end{aligned} \tag{6.4.1}$$

We set $\sigma = 10$, $\rho = 28$ and $\beta = 8/3$, values for which the system is known to exhibit chaotic solutions. We sample the initial conditions in the system's stationary distribution, following the experimental setup of (Rozet and Louppe, 2023).

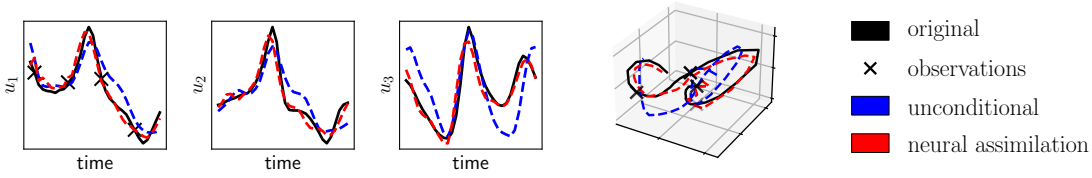

Figure 6.2. Reconstructed trajectories for the Lorenz 63 system.

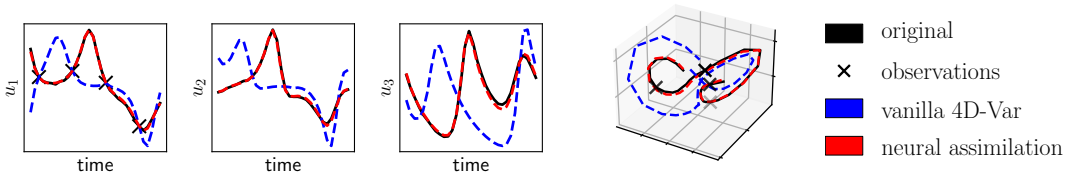

Figure 6.3. Output of 4D-Var from various initializations.

Data We generate datasets of trajectories by integrating (6.4.1) between time steps of length $\mathrm{d}t = 0.025$, and adding a small amount of Gaussian noise $\eta \sim \mathcal{N}(0, \mathrm{d}t I_3)$ at each time step. The number of time steps is $T = 32$, hence $d = 96$. We normalize each component of the trajectory to have zero mean and unit variance. The observations are sparse samples from the first component u_1 only, with observation noise of size 0.05. We take for the initial state estimate $z^{(i)}$ the maximum likelihood interpolation of $y^{(i)}$ under the moment-matching Gaussian distribution of $x^{(i)}$, which is the coarse Gaussian approximate of $p(x)$. More precisely, $z_0^{(i)} = x_{\mathrm{MAP}}(y^{(i)}; \hat{\mu}, \hat{P})$, with $\hat{\mu}$ and $\hat{P}$ the empirical mean and the empirical covariance of $\{x^{(i)}\}$. We define $\{z_k^{(i)}\}$ as in (6.3.4) with regular spacing $s_k = 1 - k/(L+1)$. We take $L = 5$.

Experimental setup We train our neural assimilation operator to reconstruct the signal at different temperatures following (6.3.5). At prediction time, we apply Algorithm 6.2 for the neural methods, along with the 4D-Var algorithm (Algorithm 6.1). Furthermore, in order to establish a link between our new neural method and traditional assimilation methods, we investigate how the output of the neural method, which is a priori not interpretable, may be transformed into a plausible physical signal. To do this, we correct these estimates with several iterations of 4D-Var on top of the neural estimate of the signal, with a fixed maximal number of 25 iterations. As a result, the new output is constrained to

Table 6.1. Performances of the various approaches. The computational time unit is the run-time of the fastest of the algorithms at prediction time.

Method	4D-Var	Cold diffusion	Neural assimilation
Pendulum error	0.1	0.15	0.1
Lorenz 63 error	0.9	1.1	0.5
Computational time	10	1	2

satisfy the physical model, but potentially at a lower cost than if we had started from scratch because the initialization that we provided is already close to the true signal.

Results Figure 6.2 shows reconstruction samples from the baselines and from our method, and Table 6.1 shows the average reconstruction error for the various methods. We can see that our neural data assimilation algorithm can reconstruct the signal while staying close to the observations. In contrast, the unconditional baseline cannot efficiently improve both the signal likelihood and the data fidelity. Compared to 4D-Var, our neural approach offers considerable computational gains, and good accuracy in these experiments. Further, we compare the reconstructed signals corrected by 4D-Var for an observation sample in Figure 6.3, where a fixed number of 4D-Var iterations are applied to two different initializations: the Gaussian first-guess and the neural reconstruction of our algorithm. The initialization provided by our method allows to recover the original signal with very high accuracy by running few steps of 4D-Var on top of the neural estimate, while the 4D-Var algorithm with Gaussian initialization ("vanilla") leads to an inaccurate local minimum. Importantly, the improvement with respect to a Gaussian initialization is significant, both in terms of reconstruction error and in terms of number of iterations, as the 4D-var algorithm converged after 4 iterations from the neural initialization and 23 iterations from the Gaussian initialization. We further discuss the comparison between deep learning data assimilation approaches and 4D-Var in Section 6.6.

6.5. Related work

The state of the art methods for data assimilation are the 4D-Var algorithm (Le Dimet and Talagrand, 1986; Trémolet, 2007) and the ensemble Kalman filter (Evensen, 2003; Bocquet et al., 2014). The statistical component in these approaches lies in the definition of covariance matrices for the background state estimates, for the model and for the observations. The numerical cost of computing the physical model and its linear tangent local approximation may be considerable for large systems.

In recent years, several deep learning algorithms have been proposed for the data assimilation problem. Building on diffusion models (Ho et al., 2020), Rozet and Louppe (2023) propose a data assimilation method based on score-based diffusion. This approach proceeds in a plug-and-play fashion, and sampling from the posterior distribution relies on an approximation that is computed on the trained model. Among "end-to-end" deep learning approaches for data assimilation, the one that is closest related to ours is the 4DVarNet algorithm of Fablet et al. (2021b), which aims to directly train a neural network to minimize the reconstruction error. The complex prior modeled by the neural network is non-Gaussian, and estimating the maximum a posteriori reconstruction in this framework relies on non-convex optimization.

6.6. Conclusion

In this work, we have shown how deep learning methods may be applied to the data assimilation problem. Our neural method models the signal in a coarse-to-fine fashion and is trained to minimize the reconstruction error. Importantly, we have shown how such a deep learning method may be used

in combination with a traditional data assimilation method to enhance the reconstruction accuracy and reduce the computational time.

We believe that deep learning methods alone might not be accurate enough to completely outperform traditional physics-based approaches such as 4D-Var. While our neural approach had good reconstruction results on the presented simulated physical systems, it should be noted that the small size of these systems allows for the neural network to learn the stationary distribution from a reasonably small dataset. For real-life systems, it is unlikely that a neural network can accurately generalize the learned signal outside a training dataset, where the physics may be complex and fairly different from what the model has seen. In contrast, physics-based approaches are far more general, as the simulated physical laws are accurate everywhere in the state space. Therefore, using a deep learning algorithm to provide an approximate solution, and using it as an input to 4D-Var to reduce the number of iterations seems like a good trade-off benefitting the best of both words.

In future work, it would be interesting to apply our method to physical systems of larger scale, and to explore how the computational burden of data assimilation may be further reduced on such high-dimensional systems. Another important aspect that is crucial for data assimilation is uncertainty quantification, for which there has been recent progress in the deep learning community (Arcucci et al., 2021; Corso et al., 2022).

Chapter 7

Conclusion

The final chapter of this thesis summarizes our contributions and discusses several limitations, some possible extensions and the open problems of the research area.

Chapter organization This chapter is organized as follows. Section 7.1 summarizes our contributions. Section 7.2 highlights the key messages of this thesis. Section 7.3 discusses perspectives and future work.

7.1. Thesis summary

In this thesis, we studied several applications of deep learning to physical systems. In Chapter 1, we highlighted the potential for deep learning applications in the sciences, and discussed the specific challenges posed by physical systems. Chapter 2 introduced key statistical problems studied in the remainder of this thesis: system identification, optimal experimental design and data assimilation. We first presented the basic formulations of these problems as quadratic least squares regressions. Then, we explored how these problems may generalize to advanced learning models such as neural networks.

In Chapter 3, we addressed the problem of physical system identification through the prism of deep learning, in the challenging situation of non-identically distributed observations. This framework is common in the sciences, and it significantly adds complexity to the learning problem. By casting it as a multi-task learning problem, we demonstrated how deep meta-learning can incorporate this dimension into neural networks in a black-box fashion. Despite the generality of this approach, we highlighted its limitations, both from a computational point of view and in terms of interpretability. To overcome these two limitations, we proposed a more physically consistent model which, in contrast to black-box modeling, uses the inherent structure of the problem to reduce complexity and increase interpretability.

In Chapters 4 and 5, we examined the problem of active learning. Not only did we identify that this problem raises questions of computational cost in general, but we also focused on controlled dynamic systems, where this question becomes a critical constraint for applications on embedded systems. We studied the experimental design problem through the prism of information theory and derived an information-maximizing optimization problem to approximate this problem with limited computational resources. We developed an online algorithm for linear dynamics in Chapter 4 and then extended this algorithm to complex learning models, such as neural networks, in Chapter 5.

Finally, in Chapter 6, we explored the data assimilation problem, where the complexity of the underlying physics and the size of the system pose a major computational challenge. By learning the physical signal on a simulated dataset, we showed that deep learning algorithms could provide an approximate solution at very low computational cost, which would considerably improve the performance of traditional algorithms based on physical models.

7.2. Key messages

In the course of this thesis, we studied various statistical problems with a view to integrating a deep learning model into them, along with some of the challenges that this integration poses. The various approaches that we proposed have the common goal of leveraging the power of deep neural networks in scientific applications while overcoming these challenges. In this section, we extract the key messages that have emerged from our study.

Linear-Gaussian theory In Chapter 2, we presented the problems of system identification, exploration and state estimation by introducing the simplest statistical models based on linear estimators: the linear regression for system identification problem (2.2.16), the active linear regression problem (2.3.3) for input design, and the quadratic interpolation problem (2.4.3) for state estimation. Because of their simplicity, these different mathematical questions are well-posed and can be analyzed effectively. Moreover, they can be solved efficiently with low computational cost. This well-established linear-Gaussian framework underpins many applications in physical systems (Särkkä and Svensson, 2023), with its simplicity compensated by its computational efficiency. We argue that this framework provides a solid foundation for developing deep learning approaches, and that interpreting complex models as perturbations of well-understood linear models offers valuable theoretical insights.

Beyond linear models We demonstrated throughout our experimental results that neural networks have powerful approximation capabilities for physical systems. However, their complex parameterization deviates significantly from linearity. As a result, the closed-form expression of least-squares linear models do not hold when integrating deep learning. In this thesis, we sought to incorporate deep learning models while maintaining some computational advantages of linear models, by linearizing the neural network parameterization. In Chapter 3, rather than using a black-box multi-task learning meta-model, we expressed an affine relationship in the context parameter. In Chapters 4 and 5, rather than expressing the information gain through a complex Bayesian expectation, we linearized the parametric models, to derive a tractable expression for the Fisher information. In Chapter 6, rather than representing the signal with a complex distribution, we sequentially approximated it by a local quadratic expansion, ensuring a Gaussian prior distribution with the neural network defining its parameters. As opposed to black-box deep learning, following this approach allowed us to maintain a certain proximity to the original linear theory. For example, parameter adaptation in Chapter 3 is reduced to linear regression. In Chapters 4 and 5, the learning model is considered linear in its parameters, hence allowing for classical information-theoretic derivations. In Chapter 6, although parameterized by a neural network, the prior distribution is a Gaussian distribution, hence connecting our approach to the 3D-Var and 4D-Var algorithms. Note that linearizing a model has always been a standard method in science to analyze complex behaviors, and has underpinned countless applications, from Gauss (1809) and the least squares method, to differential dynamic programming in nonlinear optimal control for robotics (Tassa et al., 2014; Jallet et al., 2022), and 4D-Var in operational data assimilation (Janisková and Lopez, 2013).

Practical applicability Throughout this thesis, we emphasized the practical applicability of our methods under the physical constraints of real-world systems. By treating complex learning models as locally linear Gaussian, we designed algorithms with lower computational costs compared to black-box approaches. For example, our affine multi-task learning model in Chapter 3 allows tuning the model with the ordinary least squares formula. In Chapters 4 and 5, our greedy approximation of the Fisher information enabled an online exploration policy with limited computational resources. In Chapter 6, modeling the state prior distribution as a Gaussian distribution ensured that the reconstruction was the unique solution to a quadratic least-squares cost minimization problem. Importantly, these contributions significantly differ from the black-box approaches by replacing non-convex optimization at prediction time with quadratic problems with closed-form solutions, thereby saving considerable computational resources and offering convergence guarantees. This computational gain is crucial in fields like robotics, where gradient-based methods are often too slow for real-time application (see *e.g.*, the experiments on the wheeled robot Upkie in Chapter 3). Moreover, our methods may enhance the interpretability of complex neural network algorithms: Chapter 3 interprets meta-learning models in terms of physical parameters, Chapters 4 and 5 offer a simple mathematical interpretation of the information gain, and Chapter 6 couples neural networks with 4D-Var for accurate, physically-plausible reconstructions at low cost. Given the black-box nature of neural networks, interpretability is key for the deployment of deep learning in scientific applications (Carter et al., 2023).

7.3. Limitations, open problems and perspectives

While our algorithms have demonstrated promising results in simulated physical environments, one of the significant limitations of our work lies in their application to real-world systems. Implementing these algorithms on more realistic systems would provide valuable insights into their practical utility and robustness. For instance, conducting system identification experiments on the Upkie robot (Caron, 2023) would allow us to assess the performance and adaptability of our methods in a dynamic robotic environment. Similarly, applying our data assimilation techniques to oceanographic and atmospheric reanalyses would help evaluate their effectiveness in complex geophysical contexts, providing a clearer picture of their scalability and reliability in real-world applications.

Another limitation of our research is the lack of a comprehensive theoretical understanding of our methods. Despite their empirical success, our algorithms rely on deep neural networks, and hence lack rigorous convergence guarantees. Ensuring the stability of these algorithms is essential to prevent potentially costly errors and enhance the reliability of the model. Such an understanding could build on recent advances in the theory of deep learning for training convergence beyond the overparameterized regime (Robin et al., 2022), and in online learning (Ghai et al., 2022) and meta-learning (Wang et al., 2021).

In the field of deep learning applied to physical systems, several open problems remain critical for advancing both machine learning and scientific discovery. One of the foremost challenges is the interpretability of deep learning models, which are often treated as "black boxes" lacking transparency. In scientific applications, it is essential to understand how models arrive at their predictions, as this ensures reliability and trust in high-stakes domains such as climate modeling or robotics. A first step towards addressing this challenge is provided in this thesis, where we propose interpretable deep learning methods for system identification and data assimilation, demonstrating that models can be constrained to respect physical laws while maintaining predictive accuracy. This line of research can be further advanced by exploring frameworks like physics-informed neural networks, which embed physical constraints into learning models (Raissi et al., 2019).

Another open problem is data efficiency. Physical systems often suffer from data scarcity, as collecting data from real-world experiments can be expensive or impractical. Most deep learning methods rely on large datasets to perform well, and this presents a significant hurdle in fields like oceanographic and atmospheric reanalysis. One promising avenue for reducing the reliance on large datasets is the incorporation of physical knowledge directly into the learning model. Physical signals often adhere to well-known physical laws or constraints, such as conservation of energy or mass, which can be exploited to guide the learning process. By constraining the model to search only among physically-plausible signals, we can significantly reduce the sample complexity. This approach leverages the inherent structure of the problem, thereby allowing the model to generalize more effectively from limited data. In this thesis, we take initial steps in this direction through our work on interpretable meta-learning, where we incorporate the physical structure of the system into the learning process. Though many recent studies focus on integrating physical knowledge into neural networks (Liu et al., 2021; Richter-Powell et al., 2022; Hansen et al., 2023), the full extent of how this approach can accelerate learning processes in more complex tasks remains an open problem.

Moreover, an important challenge is the computational cost associated with processing large physical signals, such as those involved in weather or climate models. As we have seen, the size of these signals can be prohibitively large, leading to substantial resource demands when training machine learning models. To mitigate this, it would be highly beneficial to develop methods for efficiently compressing these signals while maintaining the integrity of the physical information. However, it remains an open question whether this can be achieved by directly incorporating physical knowledge, given that physical laws themselves are often costly to evaluate. A promising research direction involves combining dynamical system theory with data-driven methods to compress the signals. Techniques such as Dynamic Mode Decomposition (Tu, 2013) and the Koopman operator (Brunton et al., 2021) allow for the decomposition of complex systems into meaningful modes, which can be leveraged for efficient compression. Recent advances in Koopman operator learning (Lusch et al., 2018) and physical signal compression using deep learning (Glaws et al., 2020) could offer new solutions for reducing the computational burden while preserving essential dynamics. These methods hold potential for large-scale applications, particularly in the context of data assimilation and forecasting for climate and weather systems, where computational efficiency is paramount.

Lastly, the integration of generative models – particularly for uncertainty quantification, be it in state estimation or in system identification – is a rapidly evolving field. One promising area is the exploration of deep generative models, such as diffusion models (Ho et al., 2020), which have shown great potential in image generation. These models target a distribution rather than a numerical output, offering a

natural uncertainty quantification method. Recent works applied such generative models to scientific problems in drug discovery (Corso et al., 2022), numerical weather forecast (Lam et al., 2022), and data assimilation (Rozet and Louppe, 2023; Huang et al., 2024) with convincing results. However, substantial work remains to be done towards end-to-end training of these architectures, reducing their computational cost, and scaling them up to large real-world systems. Another interesting challenge in using generative models for science is managing the perception-distortion trade-off (Blau and Michaeli, 2018). Unlike imaging applications, where the goal is to generate visually plausible signals, deep learning models for scientific applications should prioritize minimal distortion to ensure accurate and reliable results.

By addressing these challenges and, future research can aim to develop more robust, interpretable, and efficient deep learning models for physical systems.

Appendix of Chapter 2

Proof of Proposition 2.2. Let $V := (v_1, \dots, v_N)^\top \in \mathbb{R}^{N \times n}$, $E := (\xi_1, \dots, \xi_N)^\top \in \mathbb{R}^{N \times n}$. Then from Proposition 2.1

$$\hat{\theta}(y_{1:N}) - \theta_\star = (V^\top V)^{-1} V^\top E \tag{2..1}$$

Then,

$$\begin{aligned} \mathbb{E}\big[(\hat{\theta}(y_{1:N}) - \theta_\star)(\hat{\theta}(y_{1:N}) - \theta_\star)^\top\big] &= \mathbb{E}\big[(V^\top V)^{-1} V^\top E E^\top V (V^\top V)^{-1}\big] \\ &= (V^\top V)^{-1} V^\top \mathbb{E}[EE^\top] V (V^\top V)^{-1} \\ &= \rho^2 (V^\top V)^{-1} \end{aligned} \tag{2..2}$$

Where the second equality comes from the non-randomness of the v_i, and the third equality comes from the independence of the ξ_i. □

Appendix of Chapter 3

3.A. Proofs

Lemma 3.1 Let $v_1, \dots, v_N$, and $w_1, \dots, w_T \in \mathbb{R}^r$, and let $\kappa \leq r$ and $v'_1, \dots, v'_N$, and $w'_1, \dots, w'_T \in \mathbb{R}^\kappa$ be two sets of vector of full rank, satisfying $\forall i, t, {w_t}^\top v_i = {w'_t}^\top v'_i$. Then there exist $P, Q \in \mathbb{R}^{\kappa \times r}$ such that $w'_t = P w_t$ and $v'_i = Q v_i$. Furthermore, $QP^\top = I_\kappa$.

Proof of Lemma 3.1. Denoting by $V \in \mathbb{R}^{N \times r}$, $V' \in \mathbb{R}^{N \times \kappa}$, $W \in \mathbb{R}^{T \times r}$ and $W' \in \mathbb{R}^{T \times \kappa}$ the matrix representations of the vectors, the scalar equalities $\forall i, t, {w_t}^\top v_i = {w'_t}^\top v'_i$ take the matrix form

$$VW^\top = V'{W'}^\top. \tag{3.A.1}$$

Since V' is of full rank, the matrix $V'^+ := (V'{V'}^\top)^{-1}{V'}^\top \in \mathbb{R}^{\kappa \times N}$ is well defined and is a left inverse of V'. Multiplying (3.A.1) by V'^+ yields

$$W' = WP^\top \quad \text{with} \quad P := V'^+ V \ \in \mathbb{R}^{\kappa \times r}. \tag{3.A.2}$$

Similarly,

$$V' = VQ^\top \quad \text{with} \quad Q := W'^+ W \ \in \mathbb{R}^{\kappa \times r}. \tag{3.A.3}$$

We also obtain $QP^\top = W'^+ W P^\top = W'^+ W' = I_\kappa$

□

Proof of Proposition 3.1. Applying Lemma 3.1 to $v'_i := \nu(z_i)$, $v_i := v(z_i)$, and $w_t := \omega_t$, $w'_t := \varphi_t$ yields the stated result. □

The case where $c, \mu \neq 0$ can be handled as follows. We augment φ and ν, and ω and v with an additional dimension, with the last components of φ and ω equal to 1 and the last components of ν and v equal to μ and c respectively. The augmented vectors satisfy the assumptions of Proposition 3.1 provided the augmented v'_i and w'_t span $\mathbb{R}^{\kappa+1}$. The proposition then applies, and implies that the physical parameters φ_t can be recovered with an affine transform. This case is tackled experimentally in the capacitor experiment (Section 3.5.3), where $\mu \neq 0$ *a fortiori* since the electrostatic field is linearized around a nonzero value. The physical parameters are identified using an affine regression.

3.B. Experimental details

3.B.1 Architectures

All neural networks are trained with the ADAM optimizer Kingma and Ba (2015). For CoDA, we set $d_\xi = k$, chosen according to the system learned. For all the baselines, the adaptation minimization problem (3.2.5) is optimized with at least 10 gradient steps, until convergence.

For training, the number of inner gradient steps of MAML and ANIL is chosen to be 1, to reduce the computational time. We have also experimented with larger numbers of inner gradient steps. This improved the stability of training, but at the cost of greater training time.

3.B.2 Systems

We provide further details about the physical systems on which the experiments of Section **??** are performed.

3.B.2.1 Point charges

The κ charges are placed at fixed locations in the plane at fixed location. The training inputs are located in $\Omega = [-1, 1] \times [0, 1]$ which is discretized into a 20×20 grid and the ground truth potential field is computed using Coulomb's law.

The training data is generated by changing each charge's value in $\{1, \dots, 5\}^\kappa$, hence $T = 5^\kappa$. We have experimented on different settings with various numbers of charges, and various locations. In Section 3.5.1, a dipolar configuration is investigated, where $\kappa = 3$, and one of the charges is far away on the left and two other charges of opposite sign are located near $\zeta_2 = 0$. Gaussian noise of size $\sigma = 0.1$ is added to the field values revealed to the learner in the test dataset.

The system is learned with a neural network of 4 hidden layers of width 16, with the last layer of size $r = \kappa$.

For evaluation, the test data is generated with random charges drawn from a uniform distribution in $[1, \dots, 5]^\kappa$ and the data points are drawn uniformly in Ω

3.B.2.2 Capacitor

The space is discretized into a 200×300 grid. The training environments are generated with 10 values of the physical context $\varphi := (\varphi_1, \varphi_2) \in [0, 0.5] \times [-0.5, 0.5]$ containing the angular and the positional perturbation of the second plate, drawn uniformly. The ground truth electrostatic field is computed with the Poisson equation solver of Zaman (2022). For evaluation, 5 new environments are drawn with the same distribution.

The system is learned with a neural network of 4 hidden layers of width 64, with the last layer of size $r = \kappa + 1 = 3$.

3.B.2.3 Cartpole and arm

We have implemented the manipulator equations for the cartpole and the arm (or acrobot), following Tedrake (2022), and have added friction. The training data is generated by actuating the robots with sinusoidal inputs, with for each environment 8 trajectories of 200 points and random initial conditions and periods. At test time, the trajectories are generated with sinusoidal inputs for evalutation, and with swing-up inputs for trajectory tracking.

Cartpole The pole's length is set to 1, the varying physical parameters are the masses of the cart and of the pole: $\varphi_t \in \{1, 2\} \times \{0.2, 0.5\}$, so $T = 4$. For evalutation, the masses are drawn uniformly around $(2, 0.3)$, with an amplitude of $(1, 0.2)$. The system is learned with a neural network of 3 hidden layers of width 16, with the last layer of size $r = \kappa + 2 = 4$.

Arm The arm's length are set to 1, the varying physical parameters are the inertia and the mass of the second arm: $\varphi_t \in \{0.25, 0.3, 0.4\} \times \{0.9, 1.0, 1.3\}$, so $T = 9$. For evalutation, the inertial parameters are drawn uniformly around $(0.5, 1)$, with an amplitude of $(0.2, 0.3)$. The system is learned with a neural network of 4 hidden layers of width 64, with the last layer of size $r = \kappa + 2 = 4$.

3.B.2.4 Upkie

Information about the open-source robot Upkie can be found at https://github.com/tasts-robots/upkie.

We trained the meta-learning algorithm on balancing trajectories of 1000 observations, with 10 different values for Upkie's torso, ranging from 0.5 to 10 kilograms. For evaluation, the mass is sampled in the same interval.

The system is learned with a neural network of 4 hidden layers of width 64, with the last layer of size $r = \kappa + 2 = 3$.

3.B.3 Inverse dynamics control

Inverse dynamics control is a nonlinear control technique that aims at computing the control inputs of a system given a target trajectory $\{\bar{q}(s)\}$ Spong et al. (2020). Using a model $\hat{\mathrm{ID}}$ for the inverse dynamics equation (3.4.2), the feedforward predicted control signal $\hat{\tau} = \hat{\mathrm{ID}}(\bar{q}, \dot{\bar{q}}, \ddot{\bar{q}})$. These feedforward control values can then be combined with a low gain feedback controller to ensure stability, as

$$\tau = \hat{\tau} + K(\bar{q} - q) + K'(\dot{\bar{q}} - \dot{q}). \tag{3.B.1}$$

For the cartpole, we used $K = K' = 0.5$. For the robot arm, we used $K = K' = 1$.

3.B.4 Adaptive control

In a time-varying dynamics scenario, CAMEL can be used for adaptive control and system identification. Given a target trajectory, the task-agnostic component v of the model predictions can be computed offline. In the control loop, the task-specific component ω is updated with the online least squares formula. The control loop is summarized in Algorithm 3.1, where we have assumed $c = 0$ for simplicity. The estimated inertial parameters are deduced from the task-specific weights with the identification matrix (3.4.5).

Algorithm 3.1 Adaptive trajectory tracking

input trained feature map $v(x)$, target trajectory $s \mapsto \bar{q}_s$
Offline control
for timestep $0 \leq s \leq H - 1$ **do**
  compute $\bar{z}_s = (\bar{q}_s, \dot{\bar{q}}_s, \ddot{\bar{q}}_s)$
  compute features $\bar{v}_s := v(\bar{z}_s)$
end for
Control loop
Initialize $M_0 = I_r, \quad \omega_0 = (0, \dots, 0)$
for time step $1 \leq s \leq H$ **do**
  compute $\hat{\tau}_s = \omega_s^\top \bar{v}_s$
  compute $e_s = q_s - \bar{q}_s$
  play $\tau_s := \hat{\tau}_s + K e_s$
  observe q_{s+1}, $\dot{q}_{s+1}$
  compute $v_s := v(x_s)$
  update $M_{s+1} = M_s - \frac{M_s v_s (M_s v_s)^\top}{1 + v_s^\top M_s v_s}$
  update $\omega_{s+1} = \omega_s - (v_s{}^\top \omega_s - u_s) M_{s+1} v_s$
end for

3.B.5 Additional numerical results

We provide details concerning Table 3.2.

Computational time For the computational times of Table 3.2, we arbitrarily chose the shortest time as the time unit, for a clearer comparison among the baselines. The computational times were measured and averaged over each experiment, with equal numbers of batch sizes and gradient steps across the different architectures. For training, the time was divided by the number of gradient steps.

Table 3.1. Adaptation performances with standard deviations.

System	Charges, 30 trials		Capacitor, 5 trials	
	3-shot	10-shot	5-shot	40-shot
MAML	4.1E-0 ± 2E-0	1.6E-1 ± 5E-2	N/A	N/A
ANIL	3.5E0 ± 5E-1	9.2E-4 ± 5E-4	4.4E-2 ± 2E-2	3.6E-2± 1E-2
CoDA	1.0E-1 ± 9E-2	8.2E-2 ± 3E-2	4.7E-2 ± 5E-5	2.6E-2± 1E-2
CAMEL	2.0E-4 ± 1E-4	1.0E-4 ± 5E-5	3.6E-2 ± 2E-2	2.6E-2 ± 1E-2

System	ε-Capacitor, $\varepsilon = 0.1$, 5 trials	
	3-shot	30-shot
MAML	N/A	N/A
ANIL	1.1E-3 ± 5E-5	1.1E-3 ± 5E-5
CoDA	1.2E-3 ± 5E-4	1.0E-3 ± 5E-4
CAMEL	4.2E-4 ± 1E-4	1.9E-4 ± 2E-5

System	Cartpole, 50 trials		Arm, 50 trials	
	50-shot	100-shot	50-shot	100-shot
MAML	4.3E0 ± 7E-1	3.5E0 ± 6E-1	1.0E0 ± 1E-1	8.1E-1 ± 5E-2
ANIL	3.8E-1 ± 1E-1	2.5E-2 ± 9E-2	8.5E-1 ± 1E-1	7.5E-1 ± 4E-2
CoDA	3.8E-1 ± 9E-3	8.1E-1 ± 1E-1	9.5E-1 ± 9E-2	9.3E-1 ± 6E-2
CAMEL	4.8E-2 ± 1E-2	3.1E-3 ± 5E-4	3.1E-1 ± 5E-2	2.4E-1 ± 1E-2

System	Upkie, 15 trials
MAML	1.5E-2 ± 7E-3
ANIL	1.9E-2 ± 6E-3
CoDA	2.1E-2 ± 3E-3
CAMEL	8.2E-3 ± 5E-3

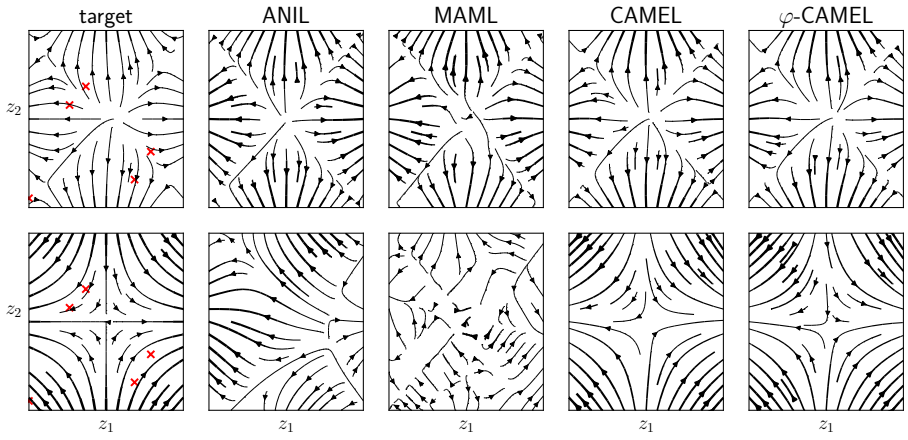

Figure 3.1. 5-shot adaptation for the 4 point charge system. **Top.** The four charges are positive, as in the training meta-dataset. **Bottom** Two of the four charges are negative.

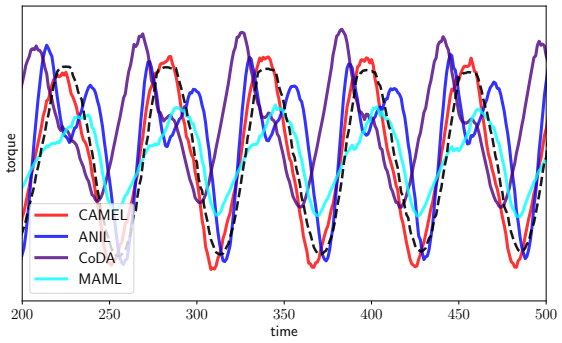

Figure 3.2. Upkie torque prediction, 100-shot adaptation.

Appendix of Chapter 4

Proof of Proposition 4.2. Using the expression (4.2.7) for $\hat{\theta}(z_{0:t})$, we may compute

$$\begin{aligned}
\hat{\theta}(z_{0:t})^\top &= M_t^{-1} \sum_{s=0}^{t-1} z_s x_{s+1}^\top \\
&= M_t^{-1} \sum_{s=0}^{t-1} z_s (\theta_\star z_s + \eta_s)^\top \\
&= M_t^{-1} \sum_{s=0}^{t-1} z_s z_s^\top \theta_\star^\top + M_t^{-1} \sum_{s=0}^{t-1} z_s \eta_s^\top \\
&= \theta_\star^\top + M_t^{-1} \sum_{s=0}^{t-1} z_s \eta_s^\top .
\end{aligned} \tag{4..2}$$

□

Proof of Proposition 4.3. We saw that the log-likelihood of a trajectory $z_{0:T}$ is

$$\log p(z_{0:T}|\theta) = -\frac{1}{2\sigma^2} \sum_{t=0}^{T-1} \left\| {z_t}^\top \theta - x_{t+1} \right\|_2^2 - T \log(\sqrt{2\pi\sigma^2}) \tag{4..3}$$

We note that

$$\|Ax_t + Bu_t - x_{t+1}\|_2^2 = \sum_{i=1}^{d} \left(z_t^\top \theta^{(i)} - x_{t+1}^{(i)} \right)^2 \tag{4..4}$$

is a sum of d squares, each term corresponding to one row $\theta^{(i)}$ of θ. Therefore, there are no cross products between the $\theta^{(i)}$ in (4..5). Hence, $\partial^2 \log p(z_{0:T}|\theta)/\partial\theta^2$ is a block-diagonal matrix by seeing θ as a vector where the rows are concatenated. The i-th block matrix equals

$$\frac{\partial^2 \log p(z_{0:T}|\theta)}{\partial {\theta^{(i)}}^2} = -\frac{1}{2\sigma^2} \sum_{t=0}^{T-1} z_t {z_t}^\top . \tag{4..5}$$

The result of the proposition follows. □

Proof of Proposition 4.4. From the matrix determinant lemma,

$$\det \left(M_t + zz^\top \right) = \det M_t \times \left(1 + z^\top {M_t}^{-1} z \right). \tag{4..6}$$

By taking the logarithm,

$$\log \det \left(M_t + zz^\top \right) = 1 + \log \det M_t + z^\top {M_t}^{-1} z. \tag{4..7}$$

Now recall that $z = (A_t x_t + B_t u, 0)$, and let $P := (M_t^{-1})_{1 \leq i,j \leq d}$ denote the first diagonal submatrix of M_t^{-1} of size $d \times d$. Then

$$z^\top {M_t}^{-1} z = x_t^\top A_t^\top P A_t x_t + 2 x_t^\top A_t^\top P B_t u + u^\top B_t^\top P B_t u \tag{4..8}$$

It follows that

$$\log\det\left(M_t + z z^\top\right) = u^\top Q q - 2 b^\top u \tag{4..9}$$

with

$$Q = B_t^\top P B_t, \quad \text{and} \quad b = -B_t^\top P A_t x_t. \tag{4..10}$$

□

Appendix of Chapter 5

A. Key definitions and approximations

We summarize step by step the approximations and the definitions pertaining FLEX from linear models to nonlinear models.

Table 5.2. Recap of our the important quantities we defined and their approximations for linear and nonlinear models.

	Linear model	Nonlinear model
Aassumption	$f(z,\theta) = V(z) \times \theta + c(z)$ as in (5.3.1)	$f(z,\theta)$ differentiable
Learning	online least squares = maximum likelihood estimator $\theta_{t+1} = \theta_t - \Gamma_t \nabla \ell_t(\theta_t)$ as in (5.3.11b)	online gradient descent $\simeq$ maximum likelihood estimator $\theta_{t+1} = \theta_t - \gamma_t \nabla \ell_t(\theta_t)$ as in (5.5.6)
Feature map	$V(z)$ as in Definition 5.1	$V(z) := \dfrac{\partial f}{\partial \theta}(z, \theta_\star)$ as in (5.5.2) $\theta_\star \simeq \theta_t$
	$V_t := V(z_t)$ as in Definition 5.1	$V_t := \dfrac{\partial f}{\partial \theta}(z_t, \theta_t)$ as in (5.5.3)
	$v := V^{(k)}$ as in Lemma 5.1	
Gram matrix	$M_t := \sum_{s=0}^{t-1} {V_s}^\top V_s$ as in (5.3.7)	
Information matrix	$I = M_t$ as in (5.3.9)	$I \simeq M_t$ as in 5.5.2
D-optimal information gain	$G(z) := \log\det(M_t + V(z)^\top V(z))$ as in (5.4.3)	
Rank-one approximation	$G(z) \simeq \log\det M_t + v^\top(z) M_t^{-1} v(z)$ as in (5.4.4)	
Matrices	$M := M_t$ $D := \partial v / \partial x$ $B := \mathrm{d}t\, \partial f / \partial u$ $Q := B^\top D^\top M^{-1} D B$ $b := -B^\top D^\top M^{-1} v$ as in (5.5.3)	,

Algorithm 5.2 Fast Linearized EXploration (FLEX) for nonlinear models

input nonlinear model f, horizon T, time step $\mathrm{d}t$, first estimate θ_0
output parameter estimate θ_T
for $0 \le t \le T-1$ **do**
  compute $V_t := \dfrac{\partial f}{\partial \theta}(z_t, \theta_t)$ and as in (5.5.3) and (5.5.4)
  compute Q_t, b_t with V_t as in (5.4.6)
  choose $u_t \in \underset{u^\top u \le \beta^2}{\operatorname{argmax}}\, u^\top Q_t u - 2 b_t^\top u$ (Proposition 5.2)
  observe $x_{t+1} = x_t + \mathrm{d}t\, f(x_t, u_t) + \eta_t$
  compute $\ell_t(\theta) = \frac{1}{2} \| f(x_t, u_t, \theta) - (x_{t+1} - x_t)/\mathrm{d}t \|_2^2$
  update $\theta_{t+1} = \theta_t - \gamma \nabla \ell_t(\theta_t)$ as in (5.5.6)
end for

B. Exploration in high-dimensional environments

We propose an additional experiment showing the behaviour of our algorithm in high dimension. We will add this experiment to our submission.

Table 5.3. Performance of algorithms FLEX and Random for the exploration of the n coupled pendulums.

n	2	5	10	20	50
d	4	10	20	40	100
sample complexity, Random	20	100	200	500	> 1000
compute, Random	1	2	6	60	250
sample complexity, FLEX	10	50	100	200	500
compute, FLEX	3	5	10	80	350

The nonlinear system we consider is a chain of n coupled damped pendulums, with unknown friction. Each pendulum is coupled with its two nearest neighbors. Only the first pendulum is actuated an the motion of the rest of the pendululms is due to the successive coupling (Bitar et al., 2017). Exploration of the system consists in finding the friction forces on the pendulums.

We believe that this system is illustrative for the typical setup of system identification. In robotics for example, the experiment seeks to measure the parameters of a humanoid constituted of large number of joints. Furthermore, the system we propose poses both challenges of large dimension and underactuation. Indeed, the dimension of the state space is $d(n) = 2n$, and only one of the n pendulums is actuated, the motion being propagated from neighbor to neighbor. This experiment allows us to monitor both the sample efficiency and the computational cost of FLEX in a challenging setting of large dimension.

Setup The system is modeled with a linear model with unknown friction coefficients $\theta \in \mathbb{R}^n$ from observations $x_t \in \mathbb{R}^{2n}$. We define the sample complexity as the number of samples required to obtain a 10^{-2} parameter error. For different values of n, we measure the sample complexity and the computational time of FLEX and Random.

Results We provide our resultsin Table 5.3. These results capture the behaviour of the algorithm when the dimension d grows. Our algorithm FLEX seems to reach a linear sample complexity with respect to d, with reasonable computational time, whereas random exploration fails to explore and yields super-linear sample complexity. Our results suggest that despite larger computational cost, FLEX remains sample efficient and competitive in high-dimensional environments.

C. Experimental details

C.1 Exploration benchmark

C.1.1 Additional results

The temporal coherence of the inputs generated by FLEX is illustrated by Figure 5.3. The spectral density shows that the energy is peaked on a frequency, hence implying that there is a temporal structure exciting the cartole near resonance, and thereby yielding informative trajectories.

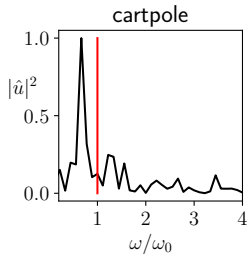

Figure 5.3. Spectral density of the inputs generated by FLEX in the cartpole environment.

C.1.2 Environments

We provide additional details on the environments and the learning models used. We denote the translation variables by q_x and q_y and the angle variables by q_ϕ.

Environment 1 (Pendulum, $d = 2, k = 1$)**.** The dynamics are given by (5.2.5).

Environment 2 (Quadrotor, $d = 6$, $k = 2$)**.** The planar quadrotor with nonlinear friction follows the following equations (Zhang et al., 2014)

$$\begin{aligned} \varphi_1 \ddot{q_x} &= -(u_1 + u_2) \sin q_\phi - \alpha q_x |\dot{q_x}| \dot{q_x} \\ \varphi_1 \ddot{q_y} &= (u_1 + u_2) \cos q_\phi - \alpha q_y |\dot{q_y}| \dot{q_y} - \varphi_2 \\ \varphi_3 \ddot{q_\phi} &= \rho(u_1 - u_2) \end{aligned} \tag{C.1}$$

with φ_1 the mass, φ_2 the weight, φ_3 the moment of inertia, ρ the distance to the base and α a friction coefficient.

Environment 3 (Cartpole, $d = 4, k = 1$)**.** We implement the dynamics provided in (Barto et al., 1983).

Environment 4 (Robot arm / double pendulum, $d = 6$, $k = 2$)**.** Equations available in (Chen, 2008).[$d = 4$, $k = 2$]

C.1.3 Baselines

The random baseline returns $u_t \sim \frac{\beta}{\sqrt{k}} \mathcal{U}([-1, 1]^\beta)$. The uniform policy maximizes the uniformity objective by gradient descent, with 100 gradient steps at each time step. The periodic policy returns inputs of the for $u_t = \beta \sin(\omega_0 t)$, with ω_0 an eigenmode of the system.

C.1.4 Models

We use the following learning models in Section 5.6.1. of width 8 with one hidden layer and tanh nonlinearity trained using ADAM optimizer (Kingma and Ba, 2015) with a batch size of 100.

Pendulum We use the linear model of Example 5.2.1 and learn it by ordinary least squares.

Quadrotor We learn the friction force with a neural net, and a learning rate of $\gamma = 0.02$.

Cartpole We parametrize $f(z, \theta) = a_\theta(\xi) + u \times b_\theta(\xi)$ with the observations $\xi = (q_x, \dot{q}_x, \cos q_\phi, \sin q_\phi, \dot{q}_\phi)$, and a_θ and b_θ given by a neural network, trained with a learning rate of $\gamma = 0.1$.

Arm We use a neural network to learn $f_\star(., u = 0)$ as a function of $\xi = (\cos q_{\phi_1}, \sin q_{\phi_1}, \dot{q}_{\phi_1}, \cos q_{\phi_2}, \sin q_{\phi_2}, \dot{q}_{\phi_2})$ and a learning rate of $\gamma = 0.05$.

C.2 Tracking of time-varying dynamics

The state of the spaceship in the plane is denoted $x = (q_x\, \dot{q}_x\, q_y\, \dot{q}_y)^\top$. The center of the star has time-varying coordinates

$$\Big(\kappa_x(t), \kappa_y(t)\Big) = \Big(\cos(2\pi t/T), \sin(2\pi t/T)\Big). \tag{C.2}$$

and the dynamics take the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix}\dot{q}_x \\ \dot{q}_y\end{pmatrix} = -\frac{1}{1+\frac{1}{\rho^2}\Big((q_x-\kappa_x)^2+(q_y-\kappa_y)^2\Big)}\frac{q}{\|q\|_2} \tag{C.3}$$

The agents know the dynamics down to the parameters κ_x, κ_y and ρ, which they learn by online gradient descent, with optimier ADAM and a learning rate of $\gamma = 0.01$.

C.3 From exploration to exploitation

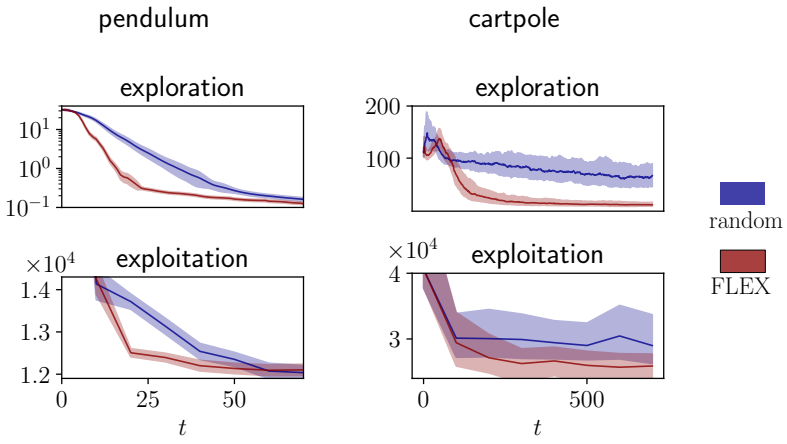

Figure 5.4. Performance of our FLEX and random exploration evaluated on downstream model-based control tasks in pendulum and carptole.

We add noise in the dynamics: $\sigma = 0.001$ for the pendulum and $\sigma = 0.05$ for the cartpole.

We use a linear model for the pendulum, and a neural network model with the same architecture as those of Section C.1 for the cartpole. The dynamics are learned as a function of $\cos q_\phi$ and $\sin q_\phi$, and q_x and $\dot{q}_x$ for the cartpole.

Since we implemented our models with Pytorch (Paszke et al., 2017), we used we use the `mpc` package and the iLQR algorithm for exploitation (Amos et al., 2018). The quadratic costs are

$$C = 100(1-\cos q_\phi)^2 + 0.1\sin^2 q_\phi + 0.1\dot{q}_\phi^2 + 0.001u^2 \tag{C.4}$$

for the pendulum and

$$C = 100q_x^2 + 100(1-\cos q_\phi)^2 + 0.1\sin^2 q_\phi^2 + 0.1\dot{q}_x^2 + 0.1\dot{q}_\phi + 0.001u^2 \tag{C.5}$$

for the cartpole. When comparing the cost values to competitors, only the order of magnitude matters since the control algorithm used for exploitation are different from an experiments to another.

We measured the computational time on a laptop and averaged it over 100 runs for FLEX and the random policy, then compared with the values of (Schultheis et al., 2020) and by setting the commputational time of the random policy to 1. Here again, only the orders of magnitude matter.

D. Proofs

D.1 Proof of Proposition 5.1

Proof. The data-generating distribution knowing the parameter θ can be computed using the probability chain rule:

$$p(y|\theta) = (\frac{1}{\sqrt{2\pi\sigma^2}})^t \exp\left(-\frac{1}{2\sigma^2}\sum_{s=0}^{t-1}\|V(z_s)\times\theta - x_{s+1}\|_2^2\right). \tag{D.1}$$

which yields

$$\begin{aligned}\log p(y|\theta) &= -\frac{1}{2\sigma^2}\sum_{s=0}^{t-1}\|V(z_t)\times\theta - x_{t+1}\|_2^2 - t\log(\sqrt{2\pi\sigma^2}) \\ &= -\frac{1}{2\sigma^2}\sum_{i=1}^{d}\sum_{s=0}^{t-1}\left(\theta^\top v^{(i)}(z_t) - x_{t+1}^{(i)}\right)_2^2 - t\log(\sqrt{2\pi\sigma^2}).\end{aligned} \tag{D.2}$$

Differentiating twice yields and taking the opposite yields

$$\begin{aligned}I(y,\theta) &= \frac{1}{\sigma^2}\sum_{i=1}^{d}\sum_{s=0}^{t-1} v^{(i)}(z_t)v^{(i)}(z_t)^\top \\ &= \frac{1}{\sigma^2}\sum_{s=0}^{t-1} V(z_s)^\top V(z_s) \\ &= \frac{1}{\sigma^2}M_t.\end{aligned} \tag{D.3}$$

□

D.2 Proof of Proposition 5.2

Proof. At first order in the neighborhood of $\bar{x} := x_t + \mathrm{d}t\, f(x_t, 0)$,

$$v(x) = v(\bar{x}) + \frac{\partial v}{\partial x}(\bar{x})\times(x-\bar{x}) + \mathcal{O}\left(\|x-\bar{x}\|^2\right), \tag{D.4}$$

Moreover, the first-order expansion of $x(u)$ near $u = 0$ is

$$x(u) = \bar{x} + \mathrm{d}tBu + o\left(\mathrm{d}t\right) \tag{D.5}$$

where u is of order β. Substituting expansion (D.5) into expansion (D.4) yields the first-order expansion of $v(z(u))$. Omitting the additive constants, we obtain the stated result. □

D.3 Proof of Proposition 5.3

Proof. See Chapter 4. Since exploration aims at producing trajectories of large amplitude, we focus on inputs with maximal norm and assume an equality constraint in the optimization problem. The strict inequality can be handled readily with unconstrained minimization. Let us denote by $u_\star$ a minimizer of (5.4.5), $\{\alpha_i\}$ the eigenvalues of Q, and $u_\star^{(i)}$ and b_i the coordinates of $u_\star$ and b in a corresponding orthonormal basis. By the Lagrange multiplier theorem, there exists a nonzero scalar μ

such that $Qu_\star - b = -\mu\, u_\star$, where μ can be scaled such that $Q + \mu I_m$ is nonsingular. The inversion of the optimal condition and the expansion of the equality constraint yield

$$u_\star^{(i)} = b_i/(\alpha_i + \mu) \tag{D.6a}$$

$$\sum_i \frac{{b_i}^2}{(\alpha_i + \mu)^2} = \beta^2. \tag{D.6b}$$

The minimizer $u_\star$ is determined by solving (D.6b) for μ. □

Bibliography

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information fusion*, 76:243–297, 2021.

Brandon Amos, Ivan Jimenez, Jacob Sacks, Byron Boots, and J. Zico Kolter. Differentiable MPC for End-to-end Planning and Control. 2018.

Rossella Arcucci, Jiangcheng Zhu, Shuang Hu, and Yi-Ke Guo. Deep data assimilation: integrating deep learning with data assimilation. *Applied Sciences*, 11(3):1114, 2021.

Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013.

Yanna Bai, Wei Chen, Jie Chen, and Weisi Guo. Deep learning methods for solving linear inverse problems: Research directions and paradigms. *Signal Processing*, 177:107729, 2020.

Antoine Bambade, Fabian Schramm, Sarah El Kazdadi, Stéphane Caron, Adrien Taylor, and Justin Carpentier. Proxqp: an efficient and versatile quadratic programming solver for real-time robotics applications and beyond. 2023.

Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold diffusion: Inverting arbitrary image transforms without noise. *Advances in Neural Information Processing Systems*, 36, 2024.

Yaakov Bar-Shalom. Optimal simultaneous state estimation and parameter identification in linear discrete-time systems. *IEEE Transactions on Automatic control*, 17(3):308–319, 1972.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5): 834–846, 1983.

M. Beauchamp, Q. Febvre, H. Georgenthum, and R. Fablet. 4dvarnet-ssh: end-to-end learning of variational interpolation schemes for nadir and wide-swath satellite altimetry. *Geoscientific Model Development*, 16(8):2119–2147, 2023. doi: 10.5194/gmd-16-2119-2023. URL https://gmd.copernicus.org/articles/16/2119/2023/.

B.M. Bell and F.W. Cathey. The iterated kalman filter update as a gauss-newton method. *IEEE Transactions on Automatic Control*, 38(2):294–297, 1993. doi: 10.1109/9.250476.

Zied Ben Bouallègue, Mariana CA Clare, Linus Magnusson, Estibaliz Gascon, Michael Maier-Gerber, Martin Janoušek, Mark Rodwell, Florian Pinault, Jesper S Dramsch, Simon TK Lang, et al. The rise of data-driven weather forecasting: A first statistical assessment of machine learning-based weather forecasts in an operational-like context. *Bulletin of the American Meteorological Society*, 2024.

Pauline Bernard, Vincent Andrieu, and Daniele Astolfi. Observer design for continuous-time dynamical systems. *Annual Reviews in Control*, 53:224–248, 2022.

L Bertinetto, J Henriques, P Torr, and A Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations (ICLR), 2019*. International Conference on Learning Representations, 2019.

Diala Bitar, Najib Kacem, and Noureddine Bouhaddi. Investigation of modal interactions and their effects on the nonlinear dynamics of a periodic coupled pendulums chain. *International Journal of Mechanical Sciences*, 127:130–141, 2017. ISSN 0020-7403. doi: https://doi.org/10.1016/j.ijmecsci.2016.11.030. URL https://www.sciencedirect.com/science/article/pii/S0020740316309614. Special Issue from International Conference on Engineering Vibration - ICoEV 2015.

Matthieu Blanke and Marc Lelarge. Online greedy identification of linear dynamical systems. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 5363–5368, 2022. doi: 10.1109/CDC51059.2022.9993030.

Matthieu Blanke and Marc Lelarge. FLEX: an Adaptive Exploration Algorithm for Nonlinear Systems. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 2577–2591. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/blanke23a.html.

Matthieu Blanke and Marc Lelarge. Interpretable Meta-Learning of Physical Systems. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=nnicaG5xiH.

Matthieu Blanke, Ronan Fablet, and Marc Lelarge. Neural Incremental Data Assimilation. In *ICML 2024 AI for Science workshop*, 2024. URL https://arxiv.org/abs/2406.15076.

Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6228–6237, 2018.

Mathieu Blondel, Quentin Berthet, Marco Cuturi, Roy Frostig, Stephan Hoyer, Felipe Llinares-López, Fabian Pedregosa, and Jean-Philippe Vert. Efficient and modular implicit differentiation. *arXiv preprint arXiv:2105.15183*, 2021.

Marc Bocquet et al. Introduction to the principles and methods of data assimilation in geosciences. *Notes de cours, École des Ponts ParisTech*, 2014.

Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436. Springer, 2012.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

F Bouttier and P Courtier. Data assimilation concepts and methods march 1999. *Meteorological training course lecture series. ECMWF*, 718:59, 2002.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

Steven L Brunton, Marko Budišić, Eurika Kaiser, and J Nathan Kutz. Modern koopman theory for dynamical systems. *arXiv preprint arXiv:2102.12086*, 2021.

Mona Buisson-Fenet, Friedrich Solowjow, and Sebastian Trimpe. Actively learning gaussian process dynamics. In *Learning for dynamics and control*, pages 5–15. PMLR, 2020.

Mona Buisson-Fenet, Lukas Bahr, Valery Morgenthaler, and Florent Di Meglio. Towards gain tuning for numerical KKL observers. *IFAC-PapersOnLine*, 56(2):4061–4067, 2023.

James A Carlson, Arthur Jaffe, and Andrew Wiles. *The millennium prize problems*. American Mathematical Soc., 2006.

Stéphane Caron. Reinforcement learning for legged robots. 2023.

A Carter, S Imtiaz, and GF Naterer. Review of interpretable machine learning for process industries. *Process Safety and Environmental Protection*, 170:647–659, 2023.

Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

Supriyo Chakraborty, Richard Tomsett, Ramya Raghavendra, Daniel Harborne, Moustafa Alzantot, Federico Cerutti, Mani Srivastava, Alun Preece, Simon Julier, Raghuveer M Rao, et al. Interpretability of deep learning models: A survey of results. In *2017 IEEE smartworld, ubiquitous intelligence & computing, advanced & trusted computed, scalable computing & communications, cloud & big data computing, Internet of people and smart city innovation (smartworld/SCALCOM/UIC/ATC/CBDcom/IOP/SCI)*, pages 1–6. IEEE, 2017.

Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995. ISSN 08834237. URL http://www.jstor.org/stable/2246015.

Joe Chen. Chaos from simplicity: an introduction to the double pendulum. 2008.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Herman Chernoff. *Sequential analysis and optimal design*. SIAM, 1972.

Girish Chowdhary, Hassan A Kingravi, Jonathan P How, and Patricio A Vela. Bayesian nonparametric adaptive control using gaussian processes. *IEEE transactions on neural networks and learning systems*, 26(3):537–550, 2014.

Ignasi Clavera, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyztsoC5Y7.

David Cohn. Neural network exploration using optimal experiment design. *Advances in neural information processing systems*, 6, 1993.

Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776*, 2022.

Romain Cosson and Laurent Massoulié. Collective tree exploration via potential function method. *arXiv preprint arXiv:2311.01354*, 2023.

Philippe Courtier, E Andersson, W Heckley, D Vasiljevic, M Hamrud, A Hollingsworth, F Rabier, M Fisher, and J Pailleux. The ecmwf implementation of three-dimensional variational assimilation (3d-var). i: Formulation. *Quarterly Journal of the Royal Meteorological Society*, 124(550):1783–1807, 1998.

David Roxbee Cox. *Principles of statistical inference*. Cambridge university press, 2006.

Harald Cramér. *Mathematical methods of statistics*, volume 26. Princeton university press, 1999.

Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020.

John L Crassidis. Robust control of nonlinear systems using model-error control synthesis. *Journal of guidance, control, and dynamics*, 22(4):595–601, 1999.

Balázs Csanád Csáji et al. Approximation with artificial neural networks. *Faculty of Sciences, Etvs Lornd University, Hungary*, 24(48):7, 2001.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

Biswa Nath Datta. *Numerical linear algebra and applications*. SIAM, 2010.

MARCO C De Simone, SERENA Russo, and ALESSANDRO Ruggiero. Influence of aerodynamics on quadrotor dynamics. *Recent Researches in Mechanical and Transportation Systems Influence*, pages 111–118, 2015.

Jacob D Durrant and J Andrew McCammon. Molecular dynamics simulations and drug discovery. *BMC biology*, 9:1–9, 2011.

Marta D'Elia and Alessandro Veneziani. Uncertainty quantification for data assimilation in a steady incompressible navier-stokes problem. *ESAIM: Mathematical Modelling and Numerical Analysis*, 47 (4):1037–1057, 2013.

Edward S Epstein and Rex J Fleming. Depicting stochastic dynamic forecasts. *Journal of Atmospheric Sciences*, 28(4):500–511, 1971.

Mathieu Even. Stochastic gradient descent under Markovian sampling schemes. In *International Conference on Machine Learning*, pages 9412–9439. PMLR, 2023.

Geir Evensen. The ensemble Kalman filter: Theoretical formulation and practical implementation. *Ocean dynamics*, 53:343–367, 2003.

Ronan Fablet, M. Amar, Q. Febvre, M. Beauchamp, and B. Chapron. END-TO-END PHYSICS-INFORMED REPRESENTATION LEARNING FOR SATELLITE OCEAN REMOTE SENSING DATA: APPLICATIONS TO SATELLITE ALTIMETRY AND SEA SURFACE CURRENTS. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-3-2021: 295–302, 2021a. doi: 10.5194/isprs-annals-V-3-2021-295-2021. URL https://imt-atlantique.hal.science/hal-03349917.

Ronan Fablet, Bertrand Chapron, Lucas Drumetz, Etienne Mémin, Olivier Pannekoucke, and François Rousseau. Learning variational data assimilation models and solvers. *Journal of Advances in Modeling Earth Systems*, 13(10):e2021MS002572, 2021b.

Matthias Faessler, Antonio Franchi, and Davide Scaramuzza. Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robotics and Automation Letters*, 3(2):620–626, 2018. doi: 10.1109/LRA.2017.2776353.

Valerii Fedorov. Optimal experimental design. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(5):581–589, 2010.

V.V. Fedorov, V.V. Fedorov, W.J. Studden, E.M. Klimko, and Academic Press (Londyn). *Theory of Optimal Experiments*. Cellular Neurobiology. Academic Press, 1972. ISBN 9780122507502. URL https://books.google.fr/books?id=v6vTAvqGny4C.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

Michael Fisher, Mand Leutbecher, and GA Kelly. On the equivalence between kalman smoothing and weak-constraint four-dimensional variational data assimilation. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 131(613):3235–3246, 2005.

Mike Fisher, Yannick Trémolet, Harri Auvinen, David Tan, and Paul Poli. *Weak-constraint and long-window 4D-Var*. ECMWF Reading, UK, 2012.

Reeves Fletcher and Colin M Reeves. Function minimization by conjugate gradients. *The computer journal*, 7(2):149–154, 1964.

Yann Fraboni, Martin Van Waerebeke, Kevin Scaman, Richard Vidal, Laetitia Kameni, and Marco Lorenzi. SIFU: Sequential informed federated unlearning for efficient and provable client unlearning in federated optimization. In Sanjoy Dasgupta, Stephan Mandt, and Yingzhen Li, editors, *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 3457–3465. PMLR, 02–04 May 2024. URL https://proceedings.mlr.press/v238/fraboni24a.html.

Bernard Friedland. *Control system design: an introduction to state-space methods*. Courier Corporation, 2012.

Xiao Fu, Kejun Huang, and Nicholas D Sidiropoulos. On identifiability of nonnegative matrix factorization. *IEEE Signal Processing Letters*, 25(3):328–332, 2018.

Junpeng Gao, Mike Yan Michelis, Andrew Spielberg, and Robert K Katzschmann. Sim-to-real of soft robots with learned residual physics. *arXiv preprint arXiv:2402.01086*, 2024.

Carl Friedrich Gauss. *Theoria motus corporum coelestium in sectionibus conicis solem ambientium auctore Carolo Friderico Gauss*. sumtibus Frid. Perthes et IH Besser, 1809.

Carl Friedrich Gauss. *Theoria motus corporum coelestium in sectionibus conicis solem ambientium*, volume 7. FA Perthes, 1877.

Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd ed. edition, 2004.

Michel Gevers, Xavier Bombois, Roland Hildebrand, and Gabriel Solari. Optimal Experiment Design for Open and Closed-loop System Identification. *Communications in Information and Systems (CIS)*, 11(3):197–224, 2011. URL https://hal.archives-ouvertes.fr/hal-00765659. Special Issue Dedicated to Brian Anderson on the Occasion of His 70th Birthday: Part II.

Udaya Ghai, Zhou Lu, and Elad Hazan. Non-convex online learning via algorithmic equivalence. *Advances in Neural Information Processing Systems*, 35:22161–22172, 2022.

M Ghil, Pascal Yiou, Stéphane Hallegatte, BD Malamud, P Naveau, A Soloviev, P Friederichs, V Keilis-Borok, D Kondrashov, V Kossobokov, et al. Extreme events: dynamics, statistics and prediction. *Nonlinear Processes in Geophysics*, 18(3):295–350, 2011.

Andrew Glaws, Ryan King, and Michael Sprague. Deep learning for in situ data compression of large turbulent flow simulations. *Physical Review Fluids*, 5(11):114602, 2020.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

G.C. Goodwin and R.L. Payne. *Dynamic System Identification: Experiment Design and Data Analysis*. Developmental Psychology Series. Academic Press, 1977. ISBN 9780122897504. URL https://books.google.fr/books?id=tu5QAAAAMAAJ.

Serge Gratton, Amos S Lawless, and Nancy K Nichols. Approximate gauss–newton methods for nonlinear least squares problems. *SIAM Journal on Optimization*, 18(1):106–132, 2007.

Christophe Grojean, Ayan Paul, Zhuoni Qian, and Inga Strümke. Lessons on interpretable machine learning from particle physics. *Nature Reviews Physics*, 4(5):284–286, 2022.

NK Gupta, RK Mehra, and WE Hall Jr. Application of optimal input synthesis to aircraft parameter identification, 1976.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

William W Hager. Minimizing a quadratic over a sphere. *SIAM Journal on Optimization*, 12(1): 188–208, 2001.

Ehsan Haghighat, Maziar Raissi, Adrian Moure, Hector Gomez, and Ruben Juanes. A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, 379:113741, 2021.

Derek Hansen, Danielle C Maddix, Shima Alizadeh, Gaurav Gupta, and Michael W Mahoney. Learning physical models that can respect conservation laws. In *International Conference on Machine Learning*, pages 12469–12510. PMLR, 2023.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

Stephan Hoyer, Janni Yuval, Dmitrii Kochkov, Ian Langmore, Peter Norgaard, Griffin Mooers, and Michael P Brenner. Neural general circulation models for weather and climate. *AGU23*, 2023.

Langwen Huang, Lukas Gianinazzi, Yuejiang Yu, Peter D Dueben, and Torsten Hoefler. Diffda: a diffusion model for weather-scale data assimilation. *arXiv preprint arXiv:2401.05932*, 2024.

Wilson Jallet, Antoine Bambade, Nicolas Mansard, and Justin Carpentier. Constrained differential dynamic programming: A primal-dual augmented lagrangian approach. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13371–13378. IEEE, 2022.

Marta Janisková and Philippe Lopez. Linearized physics for data assimilation at ecmwf. In *Data assimilation for atmospheric, oceanic and hydrologic applications (vol. ii)*, pages 251–286. Springer, 2013.

Andrew H Jazwinski. *Stochastic processes and filtering theory*. Courier Corporation, 2007.

Yassir Jedra and Alexandre Proutiere. Finite-time identification of stable linear systems optimality of the least-squares estimator. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 996–1001. IEEE, 2020.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Steven G Johnson. Notes on adjoint methods for 18.335. *Introduction to Numerical Methods*, 2012.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Rudolph E Kalman and Richard S Bucy. New results in linear filtering and prediction theory. 1961.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

L. Keviczky. "design of experiments" for the identification of linear dynamic systems. *Technometrics*, 17(3):303–308, 1975. ISSN 00401706. URL http://www.jstor.org/stable/1268065.

Patrick Kidger. *On Neural Differential Equations*. PhD thesis, University of Oxford, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL http://dblp.uni-trier.de/db/conf/iclr/iclr2015.html#KingmaB14.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Matthieu Kirchmeyer, Yuan Yin, Jeremie Dona, Nicolas Baskiotis, Alain Rakotomamonjy, and Patrick Gallinari. Generalizing to new physical systems via context-informed dynamics model. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 11283–11301. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/kirchmeyer22a.html.

Donald E Kirk. *Optimal control theory: an introduction*. Springer, 1970.

Sébastien Kleff, Avadesh Meduri, Rohan Budhiraja, Nicolas Mansard, and Ludovic Righetti. High-frequency nonlinear model predictive control of a manipulator. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7330–7336. IEEE, 2021.

Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

Thorsten Kurth, Shashank Subramanian, Peter Harrington, Jaideep Pathak, Morteza Mardani, David Hall, Andrea Miele, Karthik Kashinath, and Anima Anandkumar. Fourcastnet: Accelerating global high-resolution weather forecasting using adaptive fourier neural operators. In *Proceedings of the platform for advanced scientific computing conference*, pages 1–11, 2023.

H. Kushner and G.G. Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Stochastic Modelling and Applied Probability. Springer New York, 2003. ISBN 9780387008943. URL https://books.google.fr/books?id=_0bIieuUJGkC.

Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Ferran Alet, Suman Ravuri, Timo Ewalds, Zach Eaton-Rosen, Weihua Hu, et al. Graphcast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794*, 2022.

Rémi Laumont, Valentin De Bortoli, Andrés Almansa, Julie Delon, Alain Durmus, and Marcelo Pereyra. Bayesian imaging using plug & play priors: when langevin meets tweedie. *SIAM Journal on Imaging Sciences*, 15(2):701–737, 2022.

François-Xavier Le Dimet and Olivier Talagrand. Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects. *Tellus A: Dynamic Meteorology and Oceanography*, 38(2):97–110, 1986.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Adrien Marie Legendre. *Nouvelles méthodes pour la détermination des orbites des comètes: avec un supplément contenant divers perfectionnemens de ces méthodes et leur application aux deux comètes de 1805*. Courcier, 1806.

Ronan Legin, Alexandre Adam, Yashar Hezaveh, and Laurence Perreault-Levasseur. Beyond gaussian noise: A generalized approach to likelihood analysis with non-gaussian noise. *The Astrophysical Journal Letters*, 949(2):L41, 2023.

Pierre Gilles Lemarié-Rieusset. *The Navier-Stokes problem in the 21st century*. Chapman and Hall/CRC, 2018.

Asriel U Levin and Kumpati S Narendra. Control of nonlinear dynamical systems using neural networks. ii. observability, identification, and control. *IEEE transactions on neural networks*, 7(1):30–42, 1996.

Shouhong Li, Yazhe Sun, Weiwei Yang, Bingzhen Wang, Changlei Ma, and Zheng Tian. Research on the average daily operating cost of oceanographic research vessels of both china and the united states and its application analysis. *Advances in Economics and Management Research*, 6(1):47–47, 2023.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Shunlin Liang. *Quantitative remote sensing of land surfaces*. John Wiley & Sons, 2005.

Pantelis Linardatos, Vasilis Papastefanopoulos, and Sotiris Kotsiantis. Explainable ai: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021. ISSN 1099-4300. doi: 10.3390/e23010018. URL https://www.mdpi.com/1099-4300/23/1/18.

Dennis V Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, pages 986–1005, 1956.

Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

Kefu Liu. Identification of linear time-varying systems. *Journal of Sound and Vibration*, 206(4): 487–505, 1997.

Ziming Liu, Yunyue Chen, Yuanqi Du, and Max Tegmark. Physics-augmented learning: A new paradigm beyond physics-informed learning. *arXiv preprint arXiv:2109.13901*, 2021.

Lennart Ljung. *System Identification: Theory for the User*. Prentice-Hall, Inc., USA, 1986. ISBN 0138816409.

Lennart Ljung. System identification. In *Signal analysis and prediction*, pages 163–173. Springer, 1998.

Lennart Ljung. Perspectives on system identification. *Annual Reviews in Control*, 34(1):1–12, 2010. ISSN 1367-5788. doi: https://doi.org/10.1016/j.arcontrol.2009.12.001. URL https://www.sciencedirect.com/science/article/pii/S1367578810000027.

Lennart Ljung, Carl Andersson, Koen Tiels, and Thomas B Schön. Deep learning and system identification. *IFAC-PapersOnLine*, 53(2):1175–1181, 2020.

Edward N Lorenz. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.

Alessandro Lovo, Corentin Herbert, and Freddy Bouchet. Interpretable probabilistic forecast of extreme heat waves. In *EGU General Assembly Conference Abstracts*, pages EGU–14493, 2023.

Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

Michael Lutter, Christian Ritter, and Jan Peters. Deep lagrangian networks: Using physics as model prior for deep learning. *arXiv preprint arXiv:1907.04490*, 2019.

David JC MacKay. Information-based objective functions for active data selection. *Neural computation*, 4(4):590–604, 1992.

Stéphane Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.

Jan Mandel, Elhoucine Bergou, and Serge Gratton. 4DVAR by ensemble Kalman smoother. *arXiv preprint arXiv:1304.5271*, 2013.

Horia Mania, Michael I. Jordan, and Benjamin Recht. Active learning for nonlinear system identification with guarantees, 2020.

Filipe Marques, Paulo Flores, JC Pimenta Claro, and Hamid M Lankarani. A survey and comparison of several friction force models for dynamic analysis of multibody mechanical systems. *Nonlinear Dynamics*, 86:1407–1443, 2016.

R. Mehra. Optimal inputs for linear system identification. *IEEE Transactions on Automatic Control*, 19(3):192–200, 1974. doi: 10.1109/TAC.1974.1100554.

Raman K. Mehra. Synthesis of optimal inputs for multiinput-multioutput (mimo) systems with process noise part i: Frequenc y-domain synthesis part ii: Time-domain synthesis. In Raman K. Mehra and Dimitri G. Lainiotis, editors, *System Identification Advances and Case Studies*, volume 126 of *Mathematics in Science and Engineering*, pages 211–249. Elsevier, 1976. doi: https://doi.org/10.1016/S0076-5392(08)60873-5. URL https://www.sciencedirect.com/science/article/pii/S0076539208608735.

Richard Ménard and Roger Daley. The application of kalman smoother theory to the estimation of 4dvar error statistics. *Tellus A*, 48(2):221–237, 1996.

Youngjae Min, Kwangjun Ahn, and Navid Azizan. One-Pass Learning via Bridging Orthogonal Gradient Descent and Recursive Least-Squares. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 4720–4725, 2022. doi: 10.1109/CDC51059.2022.9992939.

Thomas M. Moerland, Joost Broekens, and Catholijn M. Jonker. Model-based reinforcement learning: A survey, 2021.

Subhadip Mukherjee, Marcello Carioni, Ozan Öktem, and Carola-Bibiane Schönlieb. End-to-end reconstruction meets data-driven regularization for inverse problems. *Advances in Neural Information Processing Systems*, 34:21413–21425, 2021.

Joaquín Muñoz-Sabater, Emanuel Dutra, Anna Agustí-Panareda, Clément Albergel, Gabriele Arduini, Gianpaolo Balsamo, Souhail Boussetta, Margarita Choulga, Shaun Harrigan, Hans Hersbach, et al. Era5-land: A state-of-the-art global reanalysis dataset for land applications. *Earth system science data*, 13(9):4349–4383, 2021.

Kenneth R Muske and James B Rawlings. Model predictive control with linear models. *AIChE Journal*, 39(2):262–287, 1993.

Aditya Nandy, Chenru Duan, and Heather J Kulik. Audacity of huge: overcoming challenges of data scarcity and data quality for machine learning in computational materials discovery. *Current Opinion in Chemical Engineering*, 36:100778, 2022. ISSN 2211-3398. doi: https://doi.org/10.1016/j.coche.2021.100778. URL https://www.sciencedirect.com/science/article/pii/S2211339821001106.

H.G. Natke. System identification: Torsten söderström and petre stoica. *Automatica*, 28(5):1069–1071, 1992. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(92)90167-E. URL https://www.sciencedirect.com/science/article/pii/000510989290167E.

O. Nelles. *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Engineering online library. Springer, 2001. ISBN 9783540673699. URL https://books.google.fr/books?id=7qHDgwMRqM4C.

Duy Nguyen-Tuong and Jan Peters. Using model knowledge for learning inverse dynamics. In *2010 IEEE international conference on robotics and automation*, pages 2677–2682. IEEE, 2010.

Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer, 1999.

Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

TN Palmer. Towards the probabilistic earth-system simulator: A vision for the future of climate and weather prediction. *Quarterly Journal of the Royal Meteorological Society*, 138(665):841–861, 2012.

Junyoung Park, Federico Berto, Arec Jamgochian, Mykel Kochenderfer, and Jinkyoo Park. First-order context-based adaptation for generalizing to new dynamical systems, 2023. URL https://openreview.net/forum?id=AW0i0lOhzqJ.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. *arXiv preprint arXiv:2202.11214*, 2022.

Eva S Peper, Sebastian Kozerke, and Pim van Ooij. Magnetic resonance imaging-based 4d flow: The role of artificial intelligence. In *Artificial Intelligence in Cardiothoracic Imaging*, pages 333–348. Springer, 2022.

Václav Peterka. Bayesian approach to system identification. In *Trends and Progress in System identification*, pages 239–304. Elsevier, 1981.

Arya A Pourzanjani, Richard M Jiang, and Linda R Petzold. Improving the identifiability of neural networks for bayesian inference. In *NIPS Workshop on Bayesian Deep Learning*, volume 4, page 31, 2017.

Emerson M. Pugh and George H. Winslow. *The Analysis of Physical Measurements*. Series in Physics. Addison-Wesley, Reading, Massachusetts, 1966.

Friedrich Pukelsheim. *Optimal design of experiments*. SIAM, 2006.

S Joe Qin and Thomas A Badgwell. An overview of nonlinear model predictive control applications. *Nonlinear model predictive control*, pages 369–392, 2000.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rkgMkCEtPB.

Tom Rainforth, Adam Foster, Desi R Ivanova, and Freddie Bickford Smith. Modern bayesian experimental design. *Statistical Science*, 39(1):100–114, 2024.

Maziar Raissi, Paris Perdikaris, and George 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*, 378:686–707, 2019.

Rangaraj M Rangayyan and Sridhar Krishnan. *Biomedical signal analysis*. John Wiley & Sons, 2024.

James B Rawlings. Tutorial overview of model predictive control. *IEEE control systems magazine*, 20 (3):38–52, 2000.

Alexander Reutlinger, Dominik Hangleiter, and Stephan Hartmann. Understanding (with) toy models. *The British Journal for the Philosophy of Science*, 2018.

Jack Richter-Powell, Yaron Lipman, and Ricky TQ Chen. Neural conservation laws: A divergence-free perspective. *Advances in Neural Information Processing Systems*, 35:38075–38088, 2022.

David AR Robin, Kevin Scaman, et al. Convergence beyond the over-parameterized regime using rayleigh quotients. *Advances in Neural Information Processing Systems*, 35:10725–10736, 2022.

François Rozet and Gilles Louppe. Score-based data assimilation. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 40521–40541. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/7f7fa581cc8a1970a4332920cdf87395-Paper-Conference.pdf.

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*, 2017.

Daniel Ruderman and William Bialek. Statistics of natural images: Scaling in the woods. *Advances in neural information processing systems*, 6, 1993.

Tuhin Sarkar and Alexander Rakhlin. Near optimal finite time identification of arbitrary linear dynamical systems, 2019.

Simo Särkkä and Lennart Svensson. *Bayesian filtering and smoothing*, volume 17. Cambridge university press, 2023.

Matthias Schultheis, Boris Belousov, Hany Abdulsamad, and Jan Peters. Receding horizon curiosity. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1278–1288. PMLR, 30 Oct–01 Nov 2020. URL https://proceedings.mlr.press/v100/schultheis20a.html.

Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pages 8583–8592. PMLR, 2020.

Burr Settles. Active learning literature survey. 2009.

Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5779–5788. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/shyam19a.html.

Max Simchowitz, Horia Mania, Stephen Tu, Michael I. Jordan, and Benjamin Recht. Learning without mixing: Towards a sharp analysis of linear system identification, 2018.

AJ Simmons, DM Burridge, M Jarraud, C Girard, and W Wergen. The ecmwf medium-range prediction models development of the numerical formulations and the impact of increased resolution. *Meteorology and atmospheric physics*, 40:28–60, 1989.

Marta Soare, Alessandro Lazaric, and Rémi Munos. Best-arm identification in linear bandits. *Advances in Neural Information Processing Systems*, 27, 2014.

Shagun Sodhani, Amy Zhang, and Joelle Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pages 9767–9779. PMLR, 2021.

Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. John Wiley & Sons, 2020.

David M Steinberg and William G Hunter. Experimental design: review and comment. *Technometrics*, 26(2):71–97, 1984.

Stephen M. Stigler. Gauss and the invention of least squares. *The Annals of Statistics*, 9(3):465–474, 1981. ISSN 00905364. URL http://www.jstor.org/stable/2240811.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13693–13696, 2020.

Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael W Mahoney, and Amir Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior. *Advances in Neural Information Processing Systems*, 36, 2024.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Richard Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.

Olivier Talagrand and Philippe Courtier. Variational assimilation of meteorological observations with the adjoint vorticity equation. i: Theory. *Quarterly Journal of the Royal Meteorological Society*, 113 (478):1311–1328, 1987.

Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4906–4913. IEEE, 2012.

Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175. IEEE, 2014.

Russ Tedrake. *Underactuated Robotics*. 2022. URL https://underactuated.csail.mit.edu.

Nils Thuerey, Konstantin Weißenow, Lukas Prantl, and Xiangyu Hu. Deep learning methods for reynolds-averaged navier–stokes simulations of airfoil flows. *AIAA Journal*, 58(1):25–36, 2020.

Yannick Trémolet. Model-error estimation in 4d-var. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 133 (626):1267–1280, 2007.

Anastasios Tsiamis and George J. Pappas. Finite sample analysis of stochastic system identification, 2019.

Jonathan H Tu. *Dynamic mode decomposition: Theory and applications*. PhD thesis, Princeton University, 2013.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638. doi: https://doi.org/10.1016/j.simpa.2020.100022. URL https://www.sciencedirect.com/science/article/pii/S2665963820300099.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Nelson Vithayathil Varghese and Qusay H. Mahmoud. A Survey of Multi-Task Deep Reinforcement Learning. *Electronics*, 9(9), 2020. ISSN 2079-9292. doi: 10.3390/electronics9091363. URL https://www.mdpi.com/2079-9292/9/9/1363.

Andrew Wagenmaker and Kevin Jamieson. Active learning for identification of linear dynamical systems, 2020.

Andrew Wagenmaker, Guanya Shi, and Kevin G Jamieson. Optimal exploration for model-based rl in nonlinear systems. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 15406–15455. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/31e018f43ab9c7065c058cc2c5848128-Paper-Conference.pdf.

Andrew J Wagenmaker, Max Simchowitz, and Kevin Jamieson. Task-optimal exploration in linear dynamical systems, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/wagenmaker21a.html.

Haoxiang Wang, Han Zhao, and Bo Li. Bridging multi-task learning and meta-learning: Towards efficient training and effective adaptation. In *International conference on machine learning*, pages 10991–11002. PMLR, 2021.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 2022a.

Liuping Wang. *Model predictive control system design and implementation using MATLAB*, volume 3. Springer, 2009.

Rui Wang, Robin Walters, and Rose Yu. Meta-learning dynamics forecasting using task inference. *Advances in Neural Information Processing Systems*, 35:21640–21653, 2022b.

Yiping Wang, Yifang Chen, Kevin Jamieson, and Simon Shaolei Du. Improved active multi-task representation learning via lasso. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 35548–35578. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/wang23b.html.

Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919*, 1(1):1–34, 2020.

Christopher Williams, Stefan Klanke, Sethu Vijayakumar, and Kian Chai. Multi-task gaussian process learning of robot inverse dynamics. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2008. URL https://proceedings.neurips.cc/paper_files/paper/2008/file/15d4e891d784977cacbfcbb00c48f133-Paper.pdf.

Tianpei Yang, Hongyao Tang, Chenjia Bai, Jinyi Liu, Jianye Hao, Zhaopeng Meng, Peng Liu, and Zhen Wang. Exploration in deep reinforcement learning: a comprehensive survey. *arXiv preprint arXiv:2109.06668*, 2021.

Yuan Yin, Ibrahim Ayed, Emmanuel de Bézenac, Nicolas Baskiotis, and Patrick Gallinari. Leads: Learning dynamical systems that generalize across environments. *Advances in Neural Information Processing Systems*, 34:7561–7573, 2021.

Yuan Yin, Matthieu Kirchmeyer, Jean-Yves Franceschi, Alain Rakotomamonjy, and patrick gallinari. Continuous PDE dynamics forecasting with implicit neural representations. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=B73niNjbPs.

Mohammad Asif Zaman. Numerical solution of the poisson equation using finite difference matrix operators. *Electronics*, 11(15), 2022. ISSN 2079-9292. doi: 10.3390/electronics11152365. URL https://www.mdpi.com/2079-9292/11/15/2365.

Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and Thomas Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *IEEE Transactions on Robotics*, 36(4): 1307–1319, 2020.

Dewei Zhang, Hui Qi, Xiande Wu, Yaen Xie, and Jiangtao Xu. The quadrotor dynamic modeling and indoor target tracking control method. *Mathematical problems in engineering*, 2014, 2014.

Luisa Zintgraf, Kyriacos Shiarli, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *International Conference on Machine Learning*, pages 7693–7702. PMLR, 2019.