Variational Data Assimilation with Deep Prior. Application to Geophysical Motion Estimation

Arthur Filoche

Thèse de doctorat de Sorbonne Université

Variational Data Assimilation with Deep Prior. Application to Geophysical Motion Estimation.

Arthur Filoche

Laboratoire d'Informatique de Sorbonne Université

Directeur de thèse: Dominique Béréziat
Co-encadrants: Anastase Charantonis & Julien Brajard
Rapporteurs: Marc Bocquet & Ronan Fablet
Examinateurs: Isabelle Bloch & Olivier Talagrand

Resumé

La récente résurgence de l'apprentissage profond a bouleversé l'état de l'art dans bon nombre de domaines scientifiques manipulant des données en grande dimension. En particulier, la disponibilité et la flexibilité des algorithmes ont permis d'automatiser la résolution de divers problèmes inverses, apprenant des estimateurs directement des données. Ce changement de paradigme n'a pas échappé à la recherche en prévision météorologique numérique. Cependant, les problématiques inhérentes aux géosciences comme l'imperfection des données et l'absence de vérité terrain compliquent l'application directe des méthodes d'apprentissage. Les algorithmes classiques d'assimilation de données, cadrant ces problèmes et permettant d'inclure des connaissances physiques, restent à l'heure actuelle les méthodes de choix dans les centres de prévision météorologique opérationnels.

Dans cette thèse, nous étudions expérimentalement l'hybridation d'algorithmes combinant apprentissage profond et assimilation de données, avec pour objectif de corriger des erreurs de prévisions dues à l'incomplétude des modèles physiques ou à la méconnaissance des conditions initiales. Premièrement, nous mettons en évidence les similitudes et nuances entre assimilation de données variationnelle et apprentissage profond. Suivant l'état de l'art, nous exploitons la complémentarité des deux approches dans un algorithme itératif pour ensuite proposer une méthode d'apprentissage de bout-en-bout. Dans un second temps, nous abordons le cœur de la thèse : l'assimilation de données variationnelle avec a priori profond, régularisant des estimateurs classiques avec des réseaux de neurones convolutionnels. L'idée est déclinée dans différents algorithmes incluant interpolation optimale, 4DVAR avec fortes et faibles contraintes, assimilation et super-résolution ou estimation d'incertitude simultanées. Nous concluons avec des perspectives sur les hybridations proposées.

Abstract

The recent revival of deep learning has impacted the state of the art in many scientific fields handling high-dimensional data. In particular, the availability and flexibility of algorithms have allowed the automation of inverse problem solving, learning estimators directly from data. This paradigm shift has also reached the research field of numerical weather prediction. However, the inherent issues in geo-sciences such as imperfect data and the lack of ground truth complicate the direct application of learning methods. Classical data assimilation algorithms, framing these issues and allowing the use of physics-based constraints, are currently the methods of choice in operational weather forecasting centers.

In this thesis, we experimentally study the hybridization of deep learning and data assimilation algorithms, with the objective of correcting forecast errors due to incomplete physical models or uncertain initial conditions. First, we highlight the similarities and nuances between variational data assimilation and deep learning. Following the state of the art, we exploit the complementarity of the two approaches in an iterative algorithm to then propose an end-to-end learning method. In a second part, we address the core of the thesis: variational data assimilation with deep prior, regularizing classical estimators with convolutional neural networks. The idea is declined in various algorithms including optimal interpolation, 4DVAR with strong and weak constraints, simultaneous assimilation, and super-resolution or uncertainty estimation. We conclude with perspectives on the proposed hybridization.

Remerciements

Achevant mon doctorat, je réalise le rôle essentiel de mon entourage dans cette expérience. Sans pouvoir être exhaustif, j'exprime ici ma sincère reconnaissance.

Je remercie d'abord les membres du jury pour avoir accepté d'évaluer mon travail : Isabelle Bloch, Olivier Talagrand, plus particulièrement Marc Bocquet et Ronan Fablet pour leurs rapports rendus dans de courts délais. Je tiens à exprimer toute ma reconnaissance à mes encadrants et directeur de thèse Anastase, Julien et Dominique ; merci à eux de m'avoir offert l'opportunité d'étudier à leurs côtés tout en me laissant une grande liberté.

Je remercie les scientifiques que j'ai pu côtoyer au cours de ces 3 années, ce fut un plaisir d'échanger avec vous. Merci à Fabienne de m'avoir offert la possibilité de présenter mon travail à l'étranger. Merci aux doctorants et post-doctorants Marie, Jorge, Rémi, Roméo, Quentin, Dimitri, Georgie et Jocelyn, le RU et le libanais du marché Monge vont me manquer. J'ai une pensée particulière pour Théo qui a su patiemment entendre mes idées et visions, je suis convaincu qu'il écrira une belle thèse au sein de notre équipe. Merci à Jean Feydy de m'avoir fait découvrir l'article "Deep Image Prior", me permettant de reconsidérer la nature des mes travaux.

En toute circonstance, j'ai eu la chance de pouvoir compter sur des amis merveilleux. Sans pouvoir tous les nommer, je remercie chaleureusement : Déborah d'avoir libéré Nicolas pour ces milliers de kilomètres à vélo, Maxime pour tout et surtout n'importe quoi, Camille et Lionel pour leur sens de la famille, Lucile et Thomas pour les bons produits, Julien et Marin pour les aventures, Thibault et Louis pour leur hospitalité, Guillaume et Justine pour les vendanges, Vincent pour le surf, Aude, Alexis et Louis pour leur noblesse, Move D pour son attitude et sa musique.

J'exprime mon infini gratitude à toute ma famille pour leur amour et leur éducation dont je suis fier. Merci à mon parrain pour son expérience. Merci à mes grands parents Catherine Filoche, Berthe Dupuis, Marcel Filoche et feu Roger Dupuis, je suis reconnaissant de leurs vies de labeur et m'estime privilégié d'avoir pu suivre des études. Merci à mon frère et mes parents que j'aime du fond du cœur, qui ont fait ce que je suis et à qui je dois tout.

Enfin, j'aimerais remercier à nouveau mon directeur de thèse Dominique Béréziat pour son soutient inconditionnel. Je souhaite à tout étudiant de pouvoir être accompagné par une personne aussi humaine et bienveillante.

Table of contents

List of Figures

List of Acronyms

AAE Average Angular Error

ADAM Adaptative Moment Estimation

BLUE Best Linear Unbiased Estimation

CFL Courant–Friedrichs–Lewy

DA Data Assimilation

DIP Deep Image prior

DL Deep Learning

DUACS Data Unification and Altimeter Combination System

L-BGFS Limited memory Assimilation Broyden-Fletcher-Goldfarb-Shanno algorithm

EPE End-Point Error

MAP Maximum A Posteriori

ML Machine Learning

MLE Maximum Likelihood Estimation

NWP Numerical Weather Prediction

OSSE Observing System Simulation Experiment

PDE Partial Differential Equation

PINN Physics-Informed Neural Network

RMSE Root Mean Square Error

SGLD Stochastic Gradient Langevin Descent

SSH Sea Surface Height

SSIM Structural Similarity

SST Sea Surface Temperature

3DVAR Three-Dimensional Variational Data Assimilation

4DVAR Four-Dimensional Variational Data Assimilation

CHAPTER 1

Introduction

1.1 Context

1.1.1 Statistical interpolation of Geo-scientific observation

Monitoring the atmosphere and the ocean has been a constant scientific concern, whether for global climate understanding or numerical weather prediction. And the quality and quantity of available data have shaped the methods of choice over time. Geo-scientific observations are inherently imperfect: variables of interest may be partially observed through a complex physical process up to the precision of the sensor. Estimating the state of a physical system from such data has then led to a great diversity of inverse problems and associated statistical interpolation methods.

Considering complex high-dimensional systems like the atmosphere and the ocean, it was unthinkable to reconstruct the physical state purely from observations. Advances in mechanistic modeling had to be leveraged leading to a new class of methods hoping to optimally combine observation and a physics-based dynamical model: Data Assimilation (DA) [1]. To this day, DA constitutes the state-of-the-art in numerical weather prediction, being used in all operational meteorological centers around the world, and is still an active field of research [2]. Best forecasts are usually obtained by merging variational [3, 4, 5] and ensemble methods [6, 7].

1.1.2 Variational Assimilation or Deep Learning ?

Nevertheless, the ever increasing volume, quality, and diversity of available observations and the maturity of algorithms have made purely data-driven modeling more appealing. Particularly, Machine Learning (ML) [8] methods leveraging large databases have proven to be an extremely useful tool to learn the complex relationships between variables. Such methods have already been applied in numerous Geo-scientific applications [9] and are still being investigated to enhance numerical weather prediction systems. Even though both ML and DA methods can be seen under the statistical interpolation umbrella [10], the strength of DA lies in its ability to deal with imperfect data in high-dimensional space.

Last decade, deep learning [11], a subset of machine learning, has experienced a revival achieving impressive results in diverse computational tasks involving complex high-dimensional signals like images, video, or text [12]. They are now considered state-of-the-art in applications involving spatio-temporal forecasting which makes them very appealing for numerical weather prediction [13]. Although they are not initially designed for the same purpose, deep learning and variational DA share many algorithmic aspects [14]. It has already been argued that both methods can benefit from each other [9, 10]. Data assimilation provides a rigorous framework to handle sparse data and physics-based knowledge while deep learning can leverage a collection of data extracting complex relationships from it. The question is now how to properly get the best of each method. This interrogation is at the heart of this thesis.

1.2 Subject of the thesis

1.2.1 Hybridizing machine learning and data assimilation

The difficulty arises when directly applying deep learning to highly-sparse earth observation. In the supervised learning approach, the ground truth is needed to train a model, which is usually not available in Earth sciences. On the other hand, Data Assimilation estimation can provide dense data. From this statement, approaches have naturally emerged in the data assimilation community, iterating data assimilation steps and machine learning steps for simultaneous state and parameters estimation [15, 16, 17, 18].

On the machine learning side, deep architectures constrained with a physics-based dynamical model in a 4DVAR fashion have appeared [19, 20], competing on the initial condition estimation task and confirming the statement made in [21] emphasizing expert-based bias for hybrid approaches. In [22, 23] the learning process is constrained to internally behave like a 4DVAR pushing the hybridization further.

1.2.2 Error correction in Data Assimilation

Errors in the data assimilation process arise from many sources. The mechanistic dynamical model, the observational system, the modeling [24] or even numerical sources like the discretization and the optimization algorithm can bias the estimation. Algorithms allowing model error correction have been developed [5] and studied [25], relying on a Gaussian model error hypothesis. Such a Gaussian hypothesis is practical to derive a variational formulation. However, it does not fully account for complex behaviors in geophysical dynamical models [26]. In addition, the assimilation algorithm relies on error covariance matrices as hyper-parameters which are hard to tune in high-dimension [27]. To handcraft such matrices combining expert knowledge and experiment, additional hypotheses have to be made such as locality assumptions [28, 29]. The ML tools are investigated toward more reliable model error correction [30, 31], learning the error after an assimilation step, then falling in the category of iterative methods in the particular case of doing one iteration.

1.2.3 Inductive bias and control parameters

All the modeling choices made in a statistical learning approach constitute inductive biases [32]. For instance, modeling Gaussian error or encoding spatial translational invariance in an operator are modeling biases [21]. The "Deep Image Prior" [33] work exhibits the mesmerizing effect of the bias induced by deep convolutional architecture, over-fitting a neural network on one image. Such architectures can represent complex correlations but also have a natural denoising effect [34, 35].

Coming back to similarities between deep learning and variational assimilation, they both shape a control parameters space and optimize on its backpropagating gradient through complex operators. Indeed, adjoint backpropagation through time steps or through hidden layers is equivalent [14]. Either initial condition constrained by a dynamical model or layered convolutional weights, the organization of the control parameters participates in such biases. The main subject of the thesis has been inspired by the "Deep Image Prior" work, adapting the idea in a variational assimilation framework.

1.2.4 Contributions

The main contributions of the thesis are methodological and empirical. We designed algorithms hybridizing variational assimilation and deep neural architectures, to then highlight their behaviors in assimilation experiments using two dynamical systems involving geophysical-like motion.

4DVAR software

A simple yet useful contribution of our work is a light 4DVAR Pytorch code, exhibiting the possibility to use powerful tools from the deep learning community based on automatic differentiation to implement variational assimilation algorithms usually less accessible.

Hybridizing machine learning and data assimilation

Simultaneous state and parameters estimation Participating in the development of algorithms iterating data assimilation and machine learning steps to correct model errors, we designed an experiment where we completed a physics-based model to finally retrieve a relevant part of the missing dynamics on an unobserved variable.

Learning variational assimilation directly from observations Hoping to circumvent iterative approaches, we designed a hybrid architecture bridging a neural network and a physics-based numerical scheme. The experiments show that the architecture was able to learn the variational inversion directly from observation.

Variational assimilation with deep prior

The heart of the thesis has been reshaping the control parameters space in vanilla 4DVAR, leveraging the inductive bias of deep convolutional architectures, largely inspired by the "Deep Image Prior" work. In all experiments, we noticed a strong regularizing effect of the method.

Strong-constraint 4DVAR with deep background prior We first adapted the strong-constraint 4DVAR, making the perfect model hypothesis. Forcing the initial condition to be generated by a deep architecture, we show progress toward circumventing the need for the Gaussian background hypothesis and the associated covariance matrix.

Simultaneous downscaling and assimilation Stressing the regularizing power of the method, we tested it on a more complex task, simultaneously assimilating and downscaling observation.

3DVAR with deep spatio-temporal prior We then extended the method toward spatio-temporal estimation and tested it in a 3DVAR setup. The experiment confirmed that a well-suited architecture can play the role of a background covariance matrix and also highlighted the importance of adding temporal convolution to the deep prior to account for temporal covariance.

Accounting for model error and uncertainty Finally, we give perspectives on how to adapt the method when the dynamical model is partially known and investigated a deep Bayesian architecture looking for uncertainty quantification.

1.3 Outline

This document is structured as follows: Chapter 2 introduces the necessary concepts and discusses the state of the art relevant to the thesis. Chapter 3 studies intricacies between variational assimilation and deep learning to later propose a learning-based algorithm. Chapter 4 introduces the concept of variational assimilation with deep prior, showcasing various experiments. Finally, Chapter 5 concludes the document by giving an overview and perspectives on this research.

CHAPTER 2

Background and Related Works

2.1 Bayesian Inversion

2.1.1 Inverse problem definition

An inverse problem [36] consists in determining causes knowing effects. Thus, this problem is the inverse of the so-called direct problem, consisting in deducing the effects, the causes being known. More formally, we define $\mathbf{X}$ as a system state of interest to be estimated from observation $\mathbf{Y}$. The relationship between $\mathbf{X}$ and $\mathbf{Y}$ is usually modeled as in Eq. 2.1, $\mathcal{F}$ being the forward operator describing the direct problem, and $\boldsymbol{\varepsilon}$ an additive noise representing measurement errors.

$$\mathbf{Y} = \mathcal{F}(\mathbf{X}) + \boldsymbol{\varepsilon} \tag{2.1}$$

The usual difficulties with inverse problems arise from the need to know the direct operator and the nature of the noise, which can be addressed by physical and mathematical modeling. According to the definition given by Hadamard [37], a problem is said well-posed if:

- a solution to this problem exists.
- this solution is unique.
- this solution depends on the data in a continuous manner.

On the contrary, if one of these conditions is not met, the problem is said ill-posed or ill-conditioned. A typical behavior of an ill-posed problem is the instability of the inversion, for instance when small disturbances in the data, like the noise, can have an arbitrarily large influence on the result. Regularization techniques exist to prevent such issues [38].

2.1.2 Bayesian point estimation

The Bayesian framework provides a simple yet powerful probabilistic method for inversion. The interest is to describe the so-called posterior probability $p(\mathbf{X} \mid \mathbf{Y})$ using the Bayes rule given in Eq. 2.2.

$$p(\mathbf{X} \mid \mathbf{Y}) = \frac{p(\mathbf{Y} \mid \mathbf{X})p(\mathbf{X})}{p(\mathbf{Y})} \tag{2.2}$$

It then requires a likelihood model $p(\mathbf{Y} \mid \mathbf{X})$ and a prior model $p(\mathbf{X})$, the marginal probability $p(\mathbf{Y})$ playing the passive role of a normalization constant. One can be interested in point estimate and then maximize the posterior over $\mathbf{X}$. If the modeling allows it, it can for instance be done in a variational manner by gradient descent, see Eq. 2.3. The likelihood term ensures consistency with data $\mathbf{Y}$ while the prior term emphasizes prior knowledge on $\mathbf{X}$.

$$\nabla_{\mathbf{X}} \log p(\mathbf{X}|\mathbf{Y}) = \nabla_{\mathbf{X}} \log p(\mathbf{Y}|\mathbf{X}) + \nabla_{\mathbf{X}} \log p(\mathbf{X}) \tag{2.3}$$

Maximizing the posterior probability $p(\mathbf{X} \mid \mathbf{Y})$ leads to the *Maximum A Posteriori* (MAP) estimation. In the event that the prior is a uniform distribution, maximizing the posterior probability is equivalent to maximizing the likelihood $p(\mathbf{Y} \mid \mathbf{X})$, which leads to the *Maximum Likelihood Estimation* (MLE).

2.1.3 Data Assimilation

Data Assimilation [39, 40] is a set of statistical methods solving particular inverse problem involving a partially known dynamical model $\mathbb{M}$ and incomplete data obtained through an observation operator $\mathbb{H}$. As described in Eqs. 2.4 and 2.5, the state $\mathbf{X}$ evolves according to the dynamics, and observations $\mathbf{Y}$ are available over a discrete time. The forward model $\mathcal{F}$ is then a combination of $\mathbb{M}$ and $\mathbb{H}$. Uncertainties about dynamics and observations are modeled by additive noises denoted ε_m and ε_R, respectively.

$$\text{Dynamics:} \qquad \mathbf{X}_{t+1} = \mathbb{M}_t(\mathbf{X}_t) + \varepsilon_m \tag{2.4}$$

$$\text{Observation:} \qquad \mathbf{Y}_t = \mathbb{H}_t(\mathbf{X}_t) + \varepsilon_{R_t} \tag{2.5}$$

The objective of DA is to provide an estimation of the system state $\mathbf{X}$ by optimally combining these two information sources. The estimation can later be used as an initial condition to produce a forecast. A famous method to solve this problem is the Kalman-Bucy filter [1]. For more details on Kalman filtering, please refer to [39].

Error sources in Data Assimilation

The forward model used in DA is often the result of many years of physical and sensor modeling, back-tested and tuned on observations. Still, these models are far from being perfect. At each modeling stage, errors are potentially introduced: the observation operator may not be perfectly known, underlying physics is probably not resolved at all scales, and boundary conditions and discretization schemes must be chosen. All of these error sources can be gathered in the so-called representation error [24]. Finally, the used inversion algorithm may not produce exact Bayesian inversion, which combined with the representation error, leads to bias in the estimation [41].

Data Assimilation in Geosciences

Geoscientific observations are imperfect by nature: either they are incomplete, noisy, or indirect. In addition, Earth system models are usually high-dimensional. Estimating the state of these physical systems using these data has led to a variety of complex inverse problems and DA has been handling them continuously [4, 2]. For instance, DA produces state-of-the-art results in various Numerical Weather Prediction (NWP) tasks and is mostly used in operational meteorological centers. To obtain their forecasting results, they tend to combine ensemble methods, such as the ensemble Kalman filter [6, 7], and the variational ones [3, 5] later detailed in Section 2.2.

2.1.4 Machine Learning

Machine Learning is a very large set of methods [8] solving computational tasks by leveraging a dataset denoted $\mathcal{D}$. The general idea is to select a class of parameterized models to represent the relationship between variables of interest, and then learn model parameters over $\mathcal{D}$. In the inverse problem context, an interesting function to learn would be the inversion operator. The first step is to choose a class of models $\widehat{\mathcal{F}}_\theta^{-1}$ parameterized by θ (see Eq. 2.6).

$$\begin{array}{rccc} \widehat{\mathcal{F}}_\theta^{-1} & : & \mathcal{Y} & \rightarrow & \mathcal{X} \\ & & \mathbf{Y} & \mapsto & \widehat{\mathcal{F}}_\theta^{-1}(\mathbf{Y}) \end{array} \tag{2.6}$$

The machine learning strategy is to define a risk $\mathcal{R}$ measuring the relevance of a model $\widehat{\mathcal{F}}_\theta^{-1}$ and optimizing it tweaking θ with the scheme of choice (see Eq. 2.7).

$$\star\widehat{\mathcal{F}}_\theta^{-1} = \arg\min_\theta \mathcal{R}(\theta) \tag{2.7}$$

In the so-called supervised approach, the dataset is constituted of observations paired with the associated ground truth, hence $\mathcal{D} = \{\mathbf{Y}^{(i)}, \mathbf{X}^{(i)}\}_{i=1}^N$. It is then common to define a loss function $\mathcal{L} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ evaluating the accuracy of an estimation $\widehat{\mathcal{F}}_\theta^{-1}(\mathbf{Y})$ compared to the ground truth $\mathbf{X}$ and the natural associated risk (see Eq. 2.8).

$$\mathcal{R}(\theta) = \mathbb{E}[\mathcal{L}(\widehat{\mathcal{F}}_\theta^{-1}(\mathbf{Y}), \mathbf{X})] \tag{2.8}$$

However, such risk assumes the knowledge of $p(\mathbf{X}, \mathbf{Y})$ while only $\mathcal{D}$ is available. A proxy is then to minimize the empirical risk $\mathcal{R}_\mathcal{D}$ defined in Eq. 2.9. There is usually no guarantee that parameters minimizing the empirical risk also minimize the actual risk $\mathcal{R}$. Such inconsistency can either lead to under-fitting, the estimator is biased, or over-fitting, the estimator has a high variance.

$$\mathcal{R}_\mathcal{D}(\theta) = \sum_i \mathcal{L}(\widehat{\mathcal{F}}_\theta^{-1}(\mathbf{Y}_i), \mathbf{X}_i) \tag{2.9}$$

In the unsupervised setting, the ground truth is not available, hence $\mathcal{D} = \{\mathbf{Y}^{(i)}\}_{i=1}^N$ and one has to come up with other strategies to perform the unsupervised inversion. For more details on ML please refer to [8].

2.1.5 Variational Inference

MLE and MAP estimations presented in Section 2.1.2 are useful tools but, as they are point estimation, they do not capture uncertainty and also are prone to data over-fitting. One may be interested in the full Bayesian estimation of the posterior $p(\mathbf{X} \mid \mathbf{Y})$. The exact estimation is often intractable for combinatorial reasons, trying to calculate the integral $p(\mathbf{Y}) = \int_\mathcal{X} p(\mathbf{X}', \mathbf{Y}) d\mathbf{X}'$. The idea behind variational inference, also known as variational Bayes, is to approximate the desired posterior distribution with a parameterized distribution $q_\theta(\mathbf{X}) \approx p(\mathbf{X} \mid \mathbf{Y})$, doing so by minimizing the Kullback–Leibler divergence between the two distribution $D_{\mathcal{KL}}(q_\theta(\mathbf{X}) \parallel p(\mathbf{X} \mid \mathbf{Y}))$, detailed in Eq. 2.10.

$$D_{\mathcal{KL}}(q_\theta(\mathbf{X}) \parallel p(\mathbf{X} \mid \mathbf{Y})) = \mathbb{E}_{q_\theta}[\log q_\theta(\mathbf{X})] - \mathbb{E}_{q_\theta}[\log p(\mathbf{X}, \mathbf{Y})] + \log p(\mathbf{Y}) \tag{2.10}$$

The log-evidence $\log p(\mathbf{Y})$ does not depend on θ so that minimizing $D_{\mathcal{KL}}(q_\theta(\mathbf{X}) \parallel p(\mathbf{X} \mid \mathbf{Y})$ is equivalent to maximizing the so-called Evidence Lower BOund $\mathcal{L}(\theta)$ given in Eq. 2.11. The success of such methods lies in the choice of q_θ but also in the optimization scheme. For more details, please look into this tutorial [42].

$$\mathcal{L}(\theta) = \mathbb{E}_{q_\theta}[\log p(\mathbf{X}, \mathbf{Y})] - \mathbb{E}_{q_\theta}[\log q_\theta(\mathbf{X})] \tag{2.11}$$

2.2 Variational Data Assimilation

In this section we develop on notion introduced section 2.1.3, describing the classical variational framework and the associated Bayesian model, leading to the 4DVAR algorithm [3, 4, 5]. Finally, we question the hypotheses made and acknowledge active areas for improvement.

2.2.1 Classic modeling framework

We consider a system state trajectory over a discrete time stamps $\mathbf{X} = [\mathbf{X}_0, \ldots, \mathbf{X}_T]$ that we want to estimate from the observations $\mathbf{Y} = [\mathbf{Y}_0, \ldots, \mathbf{Y}_T]$ knowing a dynamical model $\mathbb{M}$. Solving this inverse problem requires modeling choices inducing biases [32]. In this section we detail such choices made in the classic assimilation framework, hoping to challenge them later in the thesis.

Observations

Partial and noisy observations $\mathbf{Y}_t$ are available through an observation operator $\mathbb{H}_t$, see Eq. 2.12. Uncertainties at date t are modeled by an **additive Gaussian white noise**, denoted ε_{R_t}, of **known covariance matrix $\mathbf{R}_t$**. The noises are supposed **uncorrelated in time and independent from other error sources** and from $\mathbf{X}_t$.

$$\text{Observational model} \quad \begin{cases} \mathbf{Y}_t = \mathbb{H}_t(\mathbf{X}_t) + \varepsilon_{R_t} \\ \varepsilon_{R_t} \sim \mathcal{N}(0, \mathbf{R}_t) \end{cases} \tag{2.12}$$

Background

A background $\mathbf{X}_B$ gives prior information about the initial state $\mathbf{X}_0$, see Eq. 2.13. Uncertainty is modeled by an **additive Gaussian white noise**, denoted ε_B, of **known covariance matrix B**. This noise is supposed **independent from other error sources and** from $\mathbf{X}_0$.

$$\text{Background model} \quad \begin{cases} \mathbf{X}_0 = \mathbf{X}_B + \varepsilon_B \\ \varepsilon_B \sim \mathcal{N}(0, \mathbf{B}) \end{cases} \tag{2.13}$$

Dynamics

The physical state $\mathbf{X}_t$ evolves according to a **partially-known Markovian dynamics** $\mathbb{M}_t$, see Eq. 2.14. Uncertainties about the dynamics at date t are modeled by an **additive Gaussian white noise**, denoted ε_{m_t}, of **known covariance matrix** $\mathbf{Q}_t$. The noises are supposed **uncorrelated in time and independent from other error sources** and from $\mathbf{X}_t$.

$$\text{Evolution model} \quad \begin{cases} \mathbf{X}_{t+1} = \mathbb{M}_t(\mathbf{X}_t) + \varepsilon_{m_t} \\ \varepsilon_{m_t} \sim \mathcal{N}(0, \mathbf{Q}_t) \end{cases} \tag{2.14}$$

2.2.2 Bayesian derivation of the 4DVAR cost function

Using all the hypotheses given in the modeling stage, it is possible to derive a loss function associated with a maximum a posteriori (MAP) estimation using the Bayes rule. We note $d^2_M(x,y) = \|x-y\|^2_A = \langle (x-y) | \mathbf{A}^{-1}(x-y) \rangle$ the Mahalanobis distance associated with an invertible matrix $\mathbf{A}$.

Likelihood model

The likelihood model is defined by $p(\mathbf{Y} \mid \mathbf{X})$. Independence between $\mathbf{X}_t$ and ε_{R_t} implies that $p(\mathbf{Y}_t \mid \mathbf{X}_t) = p(\varepsilon_{R_t})$ while non-correlation in time gives $p(\mathbf{Y} \mid \mathbf{X}) = \prod_{t=0}^{T} p(\mathbf{Y}_t \mid \mathbf{X}_t)$. Combining these last two equations, we have the likelihood formulation given in Eq. 2.15, where K is a constant term.

$$p(\mathbf{Y} \mid \mathbf{X}) = K \prod_{t=0}^{T} \exp\left(-\frac{1}{2}\|\varepsilon_{R_t}\|^2_{\mathbf{R}_t}\right) = K \exp\left(-\frac{1}{2}\sum_{t=0}^{T} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t}\right) \tag{2.15}$$

Prior model

The prior model is defined by $p(\mathbf{X}) = p(\mathbf{X}_0, \ldots, \mathbf{X}_T)$. The Markovian dynamics hypothesis gives $p(\mathbf{X}) = p(\mathbf{X}_0) \prod_{t=1}^{T} p(\mathbf{X}_t \mid \mathbf{X}_{t-1})$, by marginalizing over previous states. But, from the background modeling, we have $p(\mathbf{X}_0) = K' \exp(-\frac{1}{2}\|\varepsilon_B\|^2_{\mathbf{B}})$, K' being a constant. Finally, using the independence between $\mathbf{X}_t$ and ε_{m_t} we obtain Eq. 2.16, where K'' is a constant (for more details please refer to [39]).

$$p(\mathbf{X}) = K'' \exp\left(-\frac{1}{2}\|\varepsilon_B\|^2_{\mathbf{B}} - \frac{1}{2}\sum_{t=0}^{T-1} \|\varepsilon_{m_t}\|^2_{\mathbf{Q}_t}\right) \tag{2.16}$$

A posteriori estimation

Looking for a point estimate, it is common to maximize the posterior probability $p(\mathbf{X} \mid \mathbf{Y})$ over $\mathbf{X}$. Taking the logarithmic version of the Bayes rule gives the Eq. 2.17.

$$\log p(\mathbf{X} \mid \mathbf{Y}) = \log p(\mathbf{Y} \mid \mathbf{X}) + \log p(\mathbf{X}) - \log p(\mathbf{Y}) + \log K'' \tag{2.17}$$

The constant K'' and the marginal distribution $p(\mathbf{Y})$ do not intervene as they do not depend on $\mathbf{X}$. The maximum a posteriori estimation can then be obtained by solving the model-constrained optimization problem described in Eq. 2.18, where $\mathcal{J}_{4DVAR}$ denotes the cost function to be optimized in the 4DVAR algorithm.

$$\begin{aligned} \min_{\varepsilon_B, \varepsilon_{m_t}} \quad & \mathcal{J}_{4DVAR}(\varepsilon_B, \varepsilon_{m_t}) = \frac{1}{2}\|\varepsilon_B\|^2_{\mathbf{B}} + \frac{1}{2}\sum_{t=0}^{T} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} + \frac{1}{2}\sum_{t=0}^{T-1} \|\varepsilon_{m_t}\|^2_{\mathbf{Q}_t} \\ \text{s.t.} \quad & \mathbf{X}_{t+1} = \mathbb{M}_t(\mathbf{X}_t) + \varepsilon_{m_t} \end{aligned} \tag{2.18}$$

$\mathcal{J}_{4DVAR}$ is composed of the background error, the observational error, and the model error terms, each weighted by their covariance matrices, respectively. It is to be noted that optimizing over controls $(\varepsilon_B, \varepsilon_{m_t})$ is equivalent to optimizing over the state $\mathbf{X}$. Also, it is common to group the background error with the first model and observational error in the condensed version detailed in Eq. 2.19, $\mathbf{Q}_0$ being modified adequately.

$$\mathcal{J}_{4DVAR}(\mathbf{X}) = \frac{1}{2}\sum_{t=1}^{T} \|\mathbb{H}_t(\mathbf{X}_t) - \mathbf{Y}_t\|^2_{\mathbf{R}_t} + \frac{1}{2}\sum_{t=0}^{T-1} \|\mathbb{M}_t(\mathbf{X}_t) - \mathbf{X}_{t+1}\|^2_{\mathbf{Q}_t} \tag{2.19}$$

2.2.3 The 4DVAR optimization algorithm

The assimilation problem has been shaped into an optimal control problem, where the estimated system trajectory should be close to the observation but respecting the evolution equation. This balance depends on covariance matrices, hence their importance in the modeling process. The 4DVAR algorithm simply optimizes $\mathcal{J}_{4DVAR}$ numerically, usually with the L-BFGS quasi-Newton algorithm [43]. However, descent algorithms require gradients with respect to control parameters (ε_B, ε_{m_t}, $t \in [1 : T-1]$), and this is non-trivial to obtain. By denoting multiple model integration between two times $\mathbb{M}_{t_1 \to t_2}$ and using the adjoint state method [39, 44], it is possible to derive the analytical expression given in Eq. 2.21. Obviously, the operator $\mathbb{M}_{t_1 \to t_2}$ depends on the dynamical models $\mathbb{M}_t$ between those two times but it is important to note that they also depends on ε_{m_t} which plays a role in the evolution.

$$\nabla_{\varepsilon_b} \mathcal{J} = \mathbf{B}^{-1} \varepsilon_b - \sum_{t=0}^{T} \left[\frac{\partial (\mathbb{H}_t \circ \mathbb{M}_{0 \to t})}{\partial \mathbf{X}} \right]^{\top} {\mathbf{R}_t}^{-1} \varepsilon_{R_t} \tag{2.20}$$

$$\nabla_{\varepsilon_{m_t}} \mathcal{J} = \mathbf{Q}^{-1} \varepsilon_{m_t} - \sum_{t'=t+1}^{T-1} \left[\frac{\partial (\mathbb{H}_{t'} \circ \mathbb{M}_{t \to t'})}{\partial \mathbf{X}} \right]^{\top} \mathbf{R}_{t'}^{-1} \varepsilon_{R_{t'}} \tag{2.21}$$

Computing those gradients then implies having at disposal an adjoint code of the tangent linear model. This can be done using software based on automatic differentiation, in an offline manner for instance using Tapenade [45] or in an online manner with Autograd [46]. The 4DVAR descent algorithm then consists of alternating forward integrations with the dynamics and backward ones with the adjoint. After convergence, the whole system state is estimated and the end of the temporal window can be used as initial conditions to produce a forecast. We give a schematic view of the forward integration operator in Figure 2.1 and the algorithmic details in Algorithm 1.

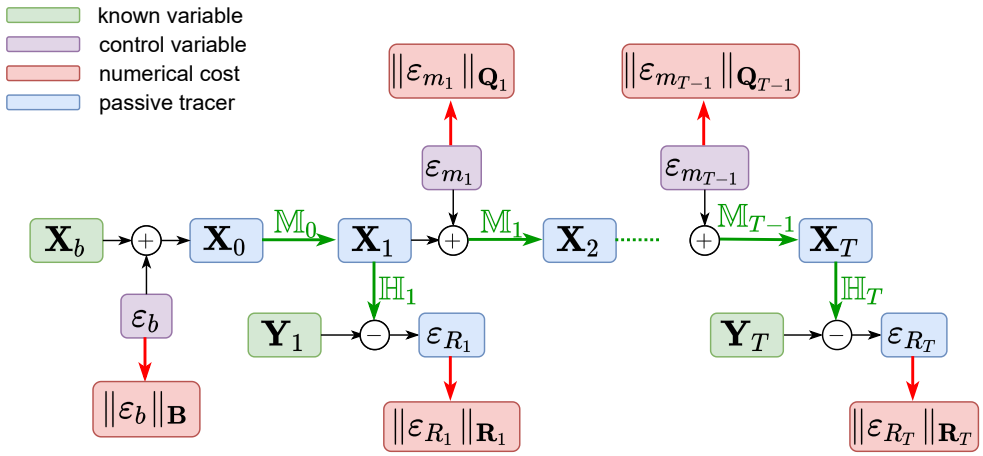


Figure 2.1: Computational graph of the forward model and associated control in the weak-constraint 4DVAR algorithm calculation

Algorithm 1 – Weak-constraint 4DVAR

Initialize control variable $\boldsymbol{\varepsilon}_b$, $\boldsymbol{\varepsilon}_m$

forward: integrate $\mathbf{X}$ and compute J
backward: automatic differentiation returns ∇J

while stop criterion **do**
  update control variables:
  $\boldsymbol{\varepsilon}_b$, $\boldsymbol{\varepsilon}_m$ = optimizer$(\boldsymbol{\varepsilon}_b, \boldsymbol{\varepsilon}_m, J, \nabla J)$

  forward: integrate $\mathbf{X}$ and compute J
  backward: automatic differentiation returns ∇J

end while
return X

Particular case #1: Strong-constraint 4DVAR

It can be relevant to assume the dynamics perfect on the temporal scale of the assimilation window, which means vanishing model errors $\boldsymbol{\varepsilon}_m$. The prior becomes $p(\mathbf{X}) = p(\mathbf{X}_0)$ and estimating the full system state trajectory is then equivalent to estimating the initial condition. The assimilation is said with strong-constraint as there is no degree of freedom for potential model errors. From a numerical optimization perspective, it suffices to specify "$\mathbf{Q}^{-1} = 0$" implying no numerical cost so no gradient back-propagation.

$$\mathcal{J}_{s4DVAR}(\mathbf{X}) = \|\mathbf{X}_B - \mathbf{X}_0\|^2_{\mathbf{B}} + \frac{1}{2}\sum_{t=0}^{T} \|\mathbb{H}_t \circ \mathbb{M}_{0\to t}(\mathbf{X}_0) - \mathbf{Y}_t\|^2_{\mathbf{R}_t} \tag{2.22}$$

Particular case #2: 3DVAR

When a dynamical model is not available, it is still possible to perform spatio-temporal interpolation using best linear unbiased estimation (BLUE or 3DVAR) [47, 48]. In this case, the prior then only relies on the knowledge of a background matrix $\mathbf{B}$, representing spatio-temporal covariance. The produced estimation has a closed form $\widehat{\mathbf{X}}_{blue} = \mathbf{B}\mathbb{H}^T(\mathbb{H}\mathbf{B}\mathbb{H}^T + \mathbf{R})^{-1}$ and can be achieved equivalently in a variational manner [49], minimizing the energy function detailed in Eq. 2.23.

$$\begin{aligned} \mathcal{J}_{3DVAR}(\mathbf{X}) &= (\mathbf{Y} - \mathbb{H}\mathbf{X})^T\mathbf{R}^{-1}(\mathbf{Y} - \mathbb{H}\mathbf{X}) + \mathbf{X}^T\mathbf{B}^{-1}\mathbf{X} && (2.23)\\ &= \|\mathbf{Y} - \mathbb{H}\mathbf{X}\|^2_{\mathbf{R}} + \|\mathbf{X}\|^2_{\mathbf{B}} && (2.24) \end{aligned}$$

2.2.4 Discussion on Gaussian modeling

All the hypotheses made in the classical modeling framework constitute inductive biases in the sense that they will emphasize solutions. Even though such biases are necessary to solve excessively ill-posed assimilation problems, they are questionable. Making the Gaussian errors hypothesis is indeed convenient to derive the least-square associated variational problem. However, it is acknowledged that Gaussian-based modeling does not fully account for complex behaviors in geophysical dynamical models [26]. This approximation corresponds to a second-order statistics truncation then allowing tractability of the Bayesian estimation.

Maximum a posteriori estimation

Whether when the observation operator or the dynamical model is non-linear, the $\mathcal{J}_{4DVAR}$ loss function becomes non-convex, making it impossible to guarantee the Bayesian estimation. It is often the case, and MAP estimation is never reached, still, the loss function is used in an operational context and provides good results.

Hyper parameters tuning

The 4DVAR algorithm relies on covariance matrices $\mathbf{B}$ and $\mathbf{Q}$ as hyper-parameters weighting regularizing terms [50] in the cost function. Tuning these hyper-parameters is critical for the accuracy of the analysis as choosing particular matrices will promote a particular set of solutions. But such statistics are usually impossible to know due to the high dimensionality of studied systems. As discussed with the original solution in [5], estimating those covariances from data is not an option. Still referencing [26], even with the Gaussian hypothesis, additional hypotheses reducing the dimensionality of the covariance matrix subspace have to be made to ensure the tractability of the estimation. Handcrafting these matrices has become a full-fledged research field and tools to diagnose the consistency of designed matrices have been developed [28]. In [29], they try an automatic approach of grid-searching the background matrix parameters using MLE with an additional locality hypothesis.

Model errors

As studied in [51], model errors come from many sources such as discretization, unresolved-scale or approximate parameters tuning. And the relevance of uncorrelated in times additive Gaussian model errors has been questioned as it has not initially improved assimilation and forecast performances [27]. Further design based on expert knowledge was needed. In [52], they investigate the impact of length-scale model errors and show a positive impact but augment computational complexity. Machine learning can be a tool

of interest to tackle this issue [30, 31], and this will be further discussed in the following sections.

2.3 Deep Learning

Deep Learning (DL) [11] is a subset of ML using artificial neural network models. Deep neural networks are composed of a succession of linear and non-linear parameterized operators, called layers, allowing them to learn complex functions. Last decade, deep learning has undergone a revival at the confluence of several factors including increasing data availability, computing power, and algorithm maturity. Their success resides in their ability to learn representation from raw high-dimensional data and then remove the need for expert-engineered data processing. They now achieve state-of-the-art performances in a countless number of tasks from diverse areas including image processing, natural language processing, speech processing, or time-series forecasting [53].

2.3.1 Architecture inductive bias

Even though neural networks learn directly from data, deep architectures have required intensive layer design and organization, constituting induced biases, meaning they will promote particular solutions [32]. In [21] they study the graphical nature of deep networks emphasizing relational inductive biases. For instance, convolution layers encode spatial translational invariance while recurrent layers [54] encode temporal invariance.

2.3.2 Convolutional neural network

The revival of interest in these methods has to some extent started with a convolutional neural network winning an image classification competition [55]. Since then, architectures have dramatically improved [56, 57, 58] and they have broadly been used to solve various imaging inverse problems [59]. Application-specific architectures have also appeared, for instance for motion estimation [60, 61, 62] incorporating advection-based prior knowledge. In [34, 35], they put efforts into digging the induced bias by deep convolutional neural networks and exhibiting their ability to represent correlation and show that such architecture has a low-pass filtering effect, respectively.

2.3.3 Deep Image prior

We here take a paragraph to discuss the "Deep Image Prior" computer vision paper [33] as it largely inspired Chapter 4. Authors use a deep convolutional neural architecture to solve various imaging inverse problems such as denoising, inpainting, or super-resolution.

To do so, the architecture is overfitted to the image to be restored, starting from an arbitrary noise. Using the inverse problem in imaging classification provided in [59], variational data assimilation and deep image prior both belong to the unlearned methods class, in the sense that they perform inversion using only one observation. Results are stunning and show that the architecture itself acts as a strong regularizer, even competing with supervised-learning methods for some tasks. The method has led to a variety of adaptations on different problems for instance involving temporal data [63] or for surface reconstruction from cloud points [64]. The main criticism of the method is that it is hard to know when to stop the optimization without accessing the ground truth. At some point, the network is able to overfit any noise, even though they show in the original paper that it really takes a lot of optimization for a convolutional architecture to fit a white noise. However, the issue is being addressed: the early stopping method used in the original paper is refined in [65], [66] uses additional total variation regularization, "Deep Decoder" [67] employs an under-parametrized architecture, and [68] proposes a Bayesian version.

A basic application of the deep image prior idea to sea surface height data for super-resolution has been the subject of communication at ICLR 2022, workshop AI4Earth [69].

2.3.4 Physics and Deep Learning

Physics-constrained Learning

Variational data assimilation has a pioneering expertise in PDE-constrained optimization [3], making use of automatic differentiation to retro-propagate gradients through the dynamical system. In [19, 20] the output of a neural network is used as input in a dynamical model, and architectures are trained with such gradients, in a supervised and adversarial manner, respectively. Physically-consistent architectures are also developed and [70, 71] propose a general framework to enforce conservation of desired quantity by neural architectures. Finally, Physics Informed Neural Networks (PINNs) [72] have encountered phenomenal successes leading to literally thousands of papers [73]. They can be used to "solve forward and inverse problems involving nonlinear partial differential equations" without needing a numerical scheme. The idea is to represent state variables by a neural network depending on spatio-temporal coordinates. Then such networks are differentiated with automatic differentiation and constrained with the known PDE system.

Model emulation and discovery

Deep learning has demonstrated great abilities to represent complex spatio-temporal relationships, and it can be used to emulate dynamical models by learning physics-based numerical simulation exclusively from data [74] or using physics constraints [75], gaining

orders of magnitude in terms of computational cost. Also, methods inferring governing dynamics directly from data are being developed, [76] performs sparse regression over a dictionary of differential operators, in [77] the learned sub-grid parametrization and PINNs can also be used to discover dynamics [78].

Simulation Based Inference

Mechanistic physics modeling of natural phenomena usually leads to a complex parameterized numerical simulator of a high-dimensional system. A key challenge is then to constrain simulation parameters with observations. The classical Bayesian approach would be to look into the likelihood of observed data given parameters, but the complexity of the system typically makes it intractable. Simulation-based inference (SBI) [79] tackles this problem by using neural density estimators like normalizing flows [80]. Even though those inferences appear over-confident at the moment [81], the issue is being addressed [82] and there already are impressive applications, for instance in dark matter astrophysics [83] and particle physics [84].

2.4 Geoscientific observation

2.4.1 Earth data science

Earth sciences have a long-standing experience in managing geophysical data. The NWP community has been handling associated engineering and theoretical challenges. Those data represent large volumes increasing by the day (terabytes per day [9]), they are diverse and they contain measurement uncertainties. Machine learning provides great tools to learn complex relationships between remote-sensed and target variables, for instance using tree-based regression and classification methods [85, 86]. Also, practices from the Machine Learning community are being adopted such as benchmark datasets to evaluate and compare data-driven approaches [87]. Regarding spatio-temporal state estimation tasks, it is hard for a fully data-driven approach to compete with well-established data assimilation leveraging years of mechanistic modeling. On the other hand, deep learning has become highly efficient at automatically extracting relevant features from high-dimensional spatio-temporal data. State-of-the-art spatio-temporal architectures design have even emerged from Earth science application [88]. So, Data Assimilation or Deep Learning? In [10] they discuss similarities of both Bayesian inversion methods and argue that they could benefit each other, which is confirmed in [9]. The question is not whether but how to hybridize these methods? This interrogation is at the heart of the thesis.

A deep learning-based rain nowcasting application of ours has been the subject of a communication at Remote Sensing, MDPI, 2021 [89].

2.4.2 Synthetic Datasets for motion estimation

Framework

To evaluate various algorithms in the geophysical motion estimation task, we will use two different simulated geophysical dynamics. Both evolution systems are based on discretized partial differential equations. Synthetic data are generated by integrating dynamics over several initial conditions and constitute a ground truth dataset of multiple system trajectories.

State-space system The system evolves over a discrete space-time domain $\Omega \times [0 : T]$ where Ω is a bounded domain of $\mathbb{Z}^2$. At each time, the system state $\mathbf{X}_t = (\mathbf{w}_t^\top \quad I_t)^\top$ is composed of a physical variable tracer I_t and the associated motion field $\mathbf{w}_t$. The numerical schemes used are implemented with the native differentiable software Pytorch [46] and can be found in the thesis-associated GitHub. The main goal of our assimilation processes will be to estimate $\mathbf{w}_t$ hence the motion estimation naming. We will refer to such experiments as twin experiments, meaning that the ground truth is available to quantify the quality of the estimation.

Observation For both systems we will simulate observations similarly: I will be sparse in time and available at regular dates but the $\mathbf{w}$ component will never be observed. At observational date t, I_t is fully observed, the observation operator is a linear projection such that $\mathbb{H}_t \mathbf{X}_t = I_t$. Gaussian white noise ε_{R_t} is then added to I_t. The variance of this noise is expressed as a percentage of the peak amplitude (half peak-to-peak amplitude) of the original signal. If not precise, this variance is fixed to 1.5%, roughly matching the confidence of altimeter sensors [90]. The covariance matrix $\mathbf{R}_t$ of this noise is then naturally diagonal. From a computational standpoint, we will treat such tensor as mask sharing the state dimension, as displayed in Figure 2.2.

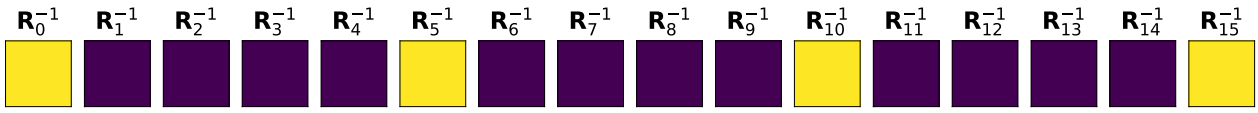


Figure 2.2: Example of simulated observational variance error matrix

Shallow water model

The first studied dynamical model corresponds to the shallow water equations system described in Eq. 2.25. State variables of the considered system are $I = \eta$, the height deviation of the horizontal pressure surface from its mean height, and $\mathbf{w}$, the associated velocity field. $\mathbf{w}$ can be decomposed in u and v, the zonal and meridional velocity,

respectively. H represents the mean height of the horizontal pressure surface and g the acceleration due to gravity.

$$\begin{cases} \dfrac{\partial \eta}{\partial t} + \dfrac{\partial (\eta + H)u}{\partial x} + \dfrac{\partial (\eta + H)v}{\partial y} = 0 \\ \dfrac{\partial u}{\partial t} + g\dfrac{\partial \eta}{\partial x} = 0 \\ \dfrac{\partial v}{\partial t} + g\dfrac{\partial \eta}{\partial y} = 0 \end{cases} \tag{2.25}$$

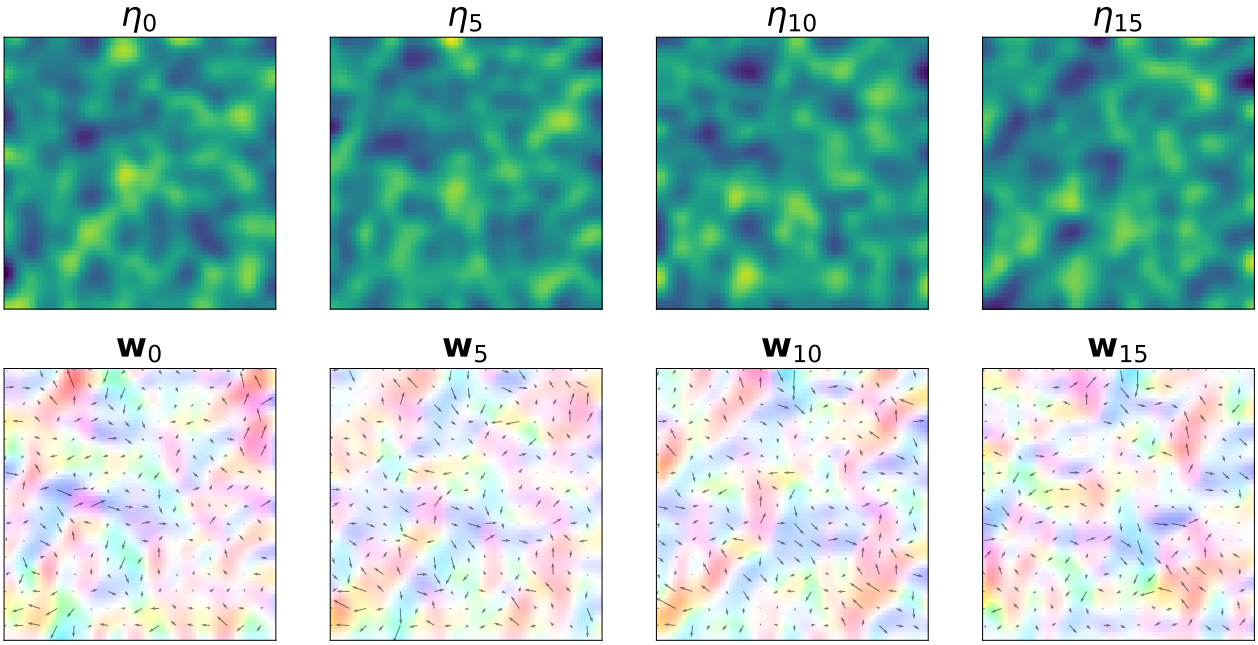


Figure 2.3: Example of simulated trajectory with the shallow water dynamics. The 2D-field of arrows represents the direction and the intensity of the velocity, the colormap provides the same information but helps visualization.

Numerical details After reaching an equilibrium starting from Gaussian random initial conditions, system trajectories are simulated as shown in Figure 2.3. At each time step, images represent a square area of $10^5 \times 10^5\,\mathrm{m}^2$ so that each pixel corresponds to a square of side $dx = dy \approx 1500\,\mathrm{m}$ mimicking the scale of high-resolution ocean simulation [91]. H and g are fixed to $100\,\mathrm{m}$ and $9.81\,\mathrm{m.s}^{-2}$, respectively. The equations are discretized using first-order upwind numerical schemes and the integration time step dt is defined as $dt = \min(dx, dy)/2\sqrt{gH}$ for numerical stability.

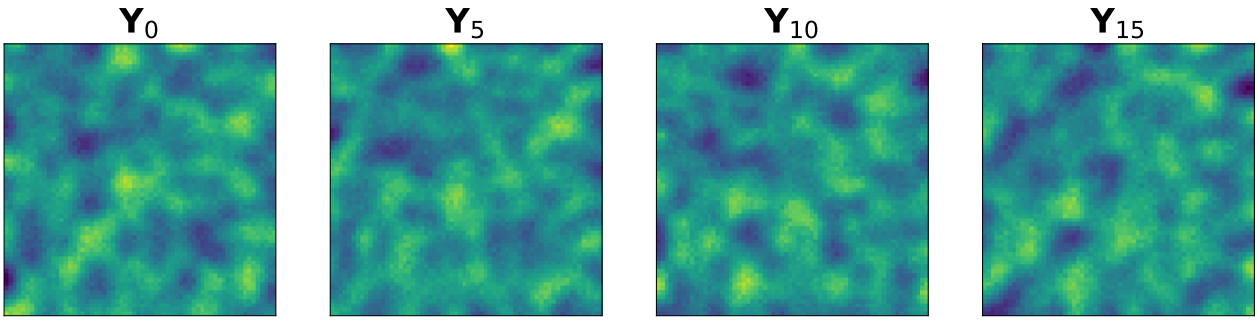
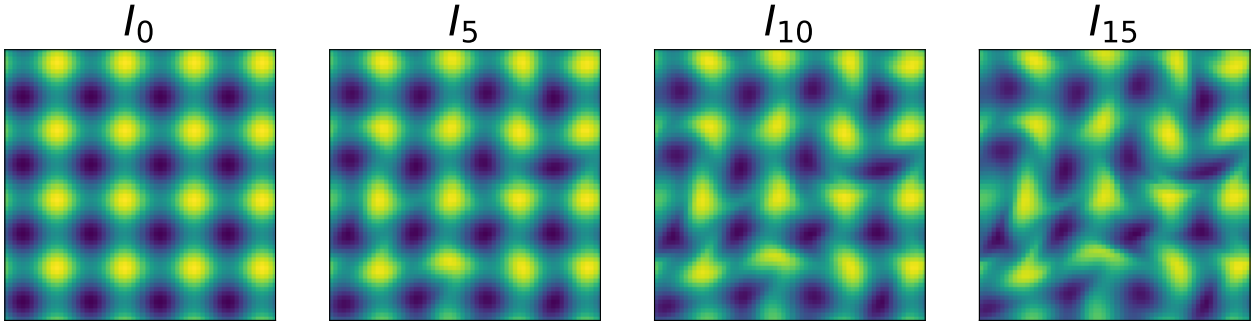


Figure 2.4: Example of simulated observations with the shallow water dynamics.

Non-linear advection dynamics

It is frequent to encounter advection-based dynamics that characterize transport in the atmosphere or the ocean. It is for instance used in state-of-the-art rain nowcasting methods [92, 93]. We decided to study the non-linear advection dynamical systems described in Eq. 2.26, following the work of [94, 95]. Respectively, these equations represent: linear advection of the I tracer by the velocity field, non-linear advection of the velocity by itself, vanishing Neumann boundary conditions for I and vanishing Dirichlet boundary conditions for $\mathbf{w}$. The motion field $\mathbf{w}$ transports the passive tracer I and also itself. We use a semi-Lagrangian numerical scheme [96] to discretize the advection operator. The semi-Lagrangian scheme is an implicit scheme, and not subject to CFL conditions. It implies that the time step can be chosen independently of the space step and velocity magnitude compared to explicit schemes.

$$\begin{cases} \dfrac{\partial I}{\partial t} + \mathbf{w}.\nabla I = 0 \\ \dfrac{\partial \mathbf{w}}{\partial t} + (\mathbf{w}.\nabla)\mathbf{w} = 0 \\ \nabla I = 0, \quad \partial\Omega \\ \mathbf{w} = 0, \quad \partial\Omega \end{cases} \tag{2.26}$$



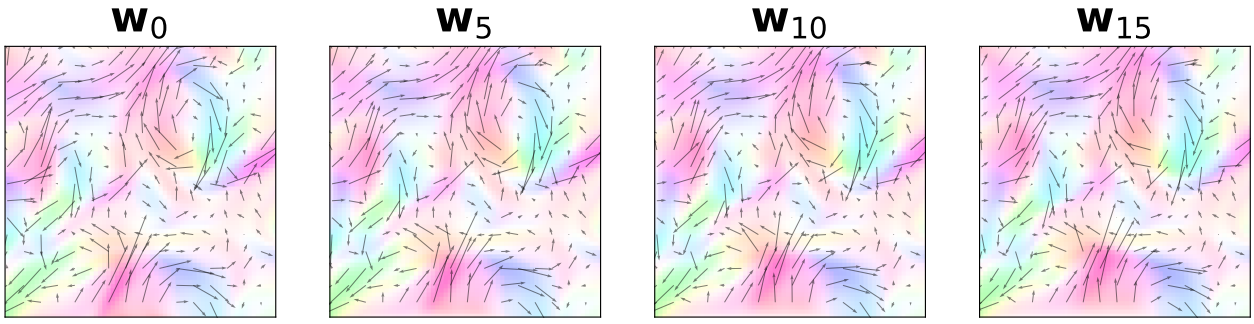


Figure 2.5: Example of simulated trajectory with the advection dynamics. The 2D-field of arrows represents the direction and the intensity of the velocity, the colormap provides the same information but helps visualization.

Numerical details Initial motion fields come from an ocean circulation model reanalysis in different areas of the North Atlantic, expressed in $\mathrm{m\,s^{-1}}$. Different used zones, exposed in Figure 2.6, constitute the diversity of the dataset. I_0 is always the same 2D-sine as displayed in Figure 2.5. The space-time domain is discretized over a grid of parameter $dx = dy = 10000$ and $dt = 8640$ which means, in relation with measurement unit of the velocity field, that each pixel covers an area of $10\,\mathrm{km}^2$ and each time step represents $8640\,\mathrm{s} = 0.1$ day.

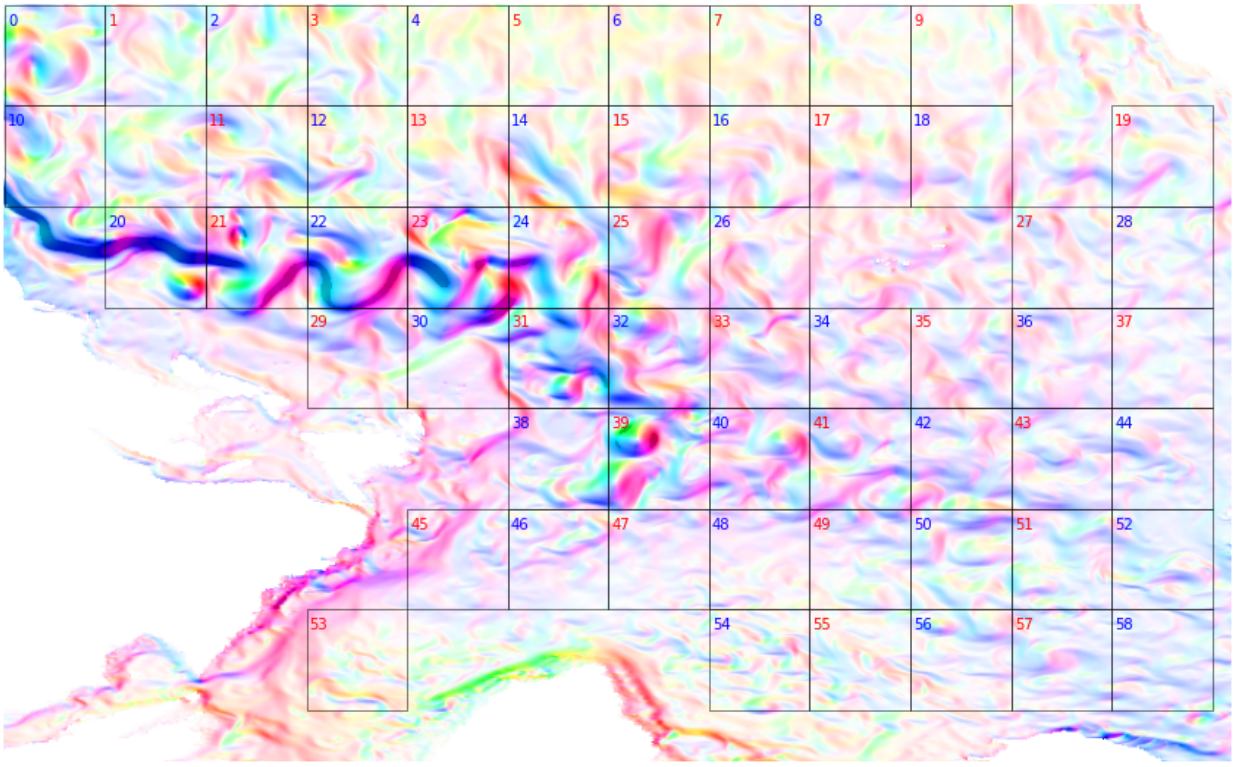


Figure 2.6: Initial motion fields by zones.

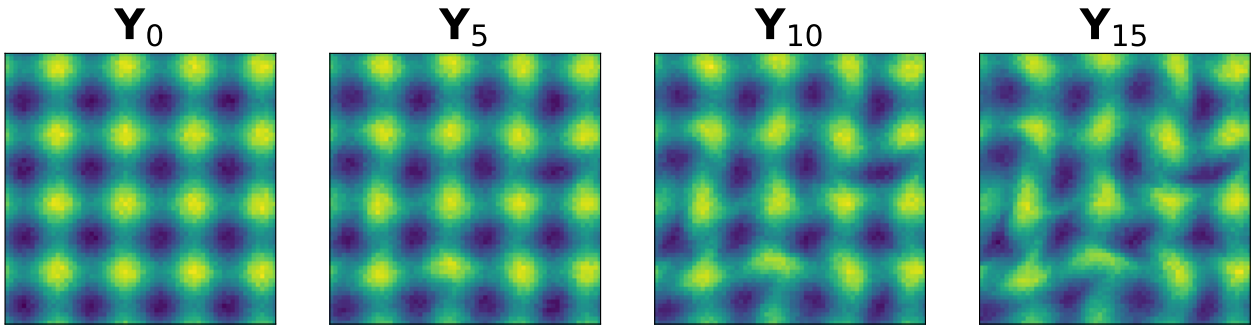


Figure 2.7: Example of simulated observations with the advection dynamics

Motion Estimation

Background prior regularization Motion estimation using image assimilation is a topic closely related to variational optical flow estimation [97, 98]. This advection system not being invertible, this particular problem is then ill-conditioned, even using noiseless observation. We can note here that the shallow water system is on the contrary invertible, but the inherent presence of noise still makes the motion estimation problem ill-posed. To overcome those issues we employ classical optical flow regularization handcrafted by experts [99]. As in [94, 100] the estimated motion field will be forced to be smooth by constraining $\|\nabla \mathbf{w}_0\|_2^2$ and $\|\nabla . \mathbf{w}_0\|_2^2$ to be small. As proved in [101], these terms can directly be included in the background error using a particular matrix $\mathbf{B}$ such that $\alpha\|\nabla \mathbf{w}_0\|_2^2 + \beta\|\nabla . \mathbf{w}_0\|_2^2 = \|\mathbf{X}_0 - \mathbf{X}_b\|^2_{B_{\alpha,\beta}}$ This is done by using a Toeplitz matrix for $\mathbf{B}^{-1}$ where descending diagonals are filled with regularization parameters. Parameters α and β are usually to be tuned.

Score metrics To evaluate the quality of the motion estimation $\widehat{\mathbf{w}}_t$, we will use two score metrics. The endpoint error $\|\widehat{\mathbf{w}}_t - \mathbf{w}_t\|_2$ and the angular error $\arccos(\widehat{\mathbf{w}}_t, \mathbf{w}_t)$ are classical optical flow scores, they calculate the Euclidean distance and the average angular deviation between the estimation and the ground truth, respectively. Once calculated on the whole image, we average pixel by pixel to obtain the average Endpoint Error (EPE) and average angular error (AAE). We will also monitor statistics characterizing smoothness and physical properties of the estimation such as $\|\nabla \mathbf{w}_t\|_2$, $\|\nabla . \mathbf{w}_t\|_2$ and $\|\Delta \mathbf{w}_t\|_2$.

$$\text{Average Endpoint Error:} \quad \frac{1}{N} \sum_{i \in \Omega} \frac{||\mathbf{w}^i - \widehat{\mathbf{w}}^i||_2}{2 \max\left(||\mathbf{w}^i||_2, ||\widehat{\mathbf{w}}^i||_2\right)} \tag{2.27}$$

$$\text{Average Angular Error:} \quad \frac{180}{\pi N} \sum_{i \in \Omega} \arccos\left(\frac{\mathbf{w}^i \cdot \widehat{\mathbf{w}}^i}{||\mathbf{w}^i||_2 ||\widehat{\mathbf{w}}^i||_2}\right) \tag{2.28}$$

2.5 Hybrid ML and DA

2.5.1 Machine Learning for Data Assimilation

Machine learning applications for Numerical Weather Forecasting being part of the next-10-year-road-map of ECMWF are exposed in [102]. Task automation using ML can be thought of at all different levels in the operational setting: going from the observation pre-processing to forecast post-processing and of course through the DA processes. They identify key applications including the one we are particularly interested in: "Bias correction and model learning" and "Mapping of non-Gaussian to Gaussian distribution for data assimilation". There already are promising results regarding off-line learning of model error to use it in an operational 4DVAR [103]. Once the set-up for model error learning is established, it can be used inside the data assimilation framework to refine the state estimation, as well-described [31, 16], opening the door for iterative methods. The same authors are also investigating online learning versions [104].

2.5.2 Iterative ML-DA methods for simultaneous state and parameters estimation

Simultaneous state and parameter estimation is an active research field in the DA community [105] including likelihood-based methods, which are closely related to machine learning. In [106] they already propose an iterative scheme based on the Expectation-maximization algorithm to identify stochastic parametrization. While the idea is old [107], similar approaches have emerged in the hope to learn a dynamical model directly from sparse and noisy data using a data assimilation scheme [16, 18, 17], allowing, in the end, to jointly estimate state and model parameters. They exploit the synergy between DA and ML (see Figure 2.8), providing either dense data in the "expectation" or a powerful learnable class of models in the "maximization" step, respectively. The proper Bayesian framework for such methods was later described in [15].

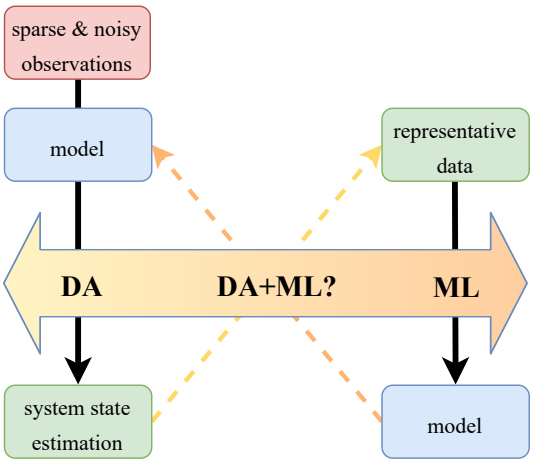


Figure 2.8: Schematic view of the potential synergy between DA and ML

2.5.3 Data Assimilation for Machine Learning

The other way around is to start the hybridization from the ML side inducing bias inspired by data assimilation [9]. As advocated in [21], combining the flexibility of deep learning with expert-based bias is a powerful complementary approach. In [108] the learning process is constrained in a Runge-Kutta scheme, while in [23, 109, 22], they introduced "4DVAR-Net" architecture enforcing their network to internally behave like a 4DVAR. Dynamics-constrained learning approaches [19, 20, 75] are really similar to 4DVAR in the sense that they use gradients back-propagated through PDE-based dynamical models to update their control parameters. Finally, as pointed out in [72], PINNs can be used for inverse problems, particularly in a data assimilation mindset.

2.6 Original Experiment and critical discussion

The work presented in this section has partially been the subject of a communication at NeurIPS 2020, workshop AI4Earth [110].

In this work, we propose to use an iterative ML-DA method, for simultaneous state and parameters estimation, on a relatively high-dimensional system where the physics is partially known. We aim at completing this base model by learning the dynamics of a fully-unobserved variable through data assimilation. Also, we benefit from powerful and flexible tools provided by the deep learning community based on automatic differentiation that is clearly suitable for variational data assimilation, avoiding explicit adjoint modeling.

We introduce a hybrid model with learnable components. To train it, we leverage DA ability to learn from sparse and noisy observations with the help of deep learning tools

based on automatic differentiation. Finally, we test it in a twin experiment and succeed in partially retrieving a missing dynamics of a fully-unobserved variable in a relatively high-dimensional system.

2.6.1 Framework

The control problem

A system state $\mathbf{X}$ evolves over discrete time t according to a parameterized dynamical Markov model $\mathbb{M}_\theta$. Partial and noisy observations $\mathbf{Y}$ are available through an observation operator $\mathbb{H}$ as represented in Figure 2.9. The system state trajectory $[\mathbf{X}_0, \ldots, \mathbf{X}_T]$ and the model parameters $\boldsymbol{\theta}$ are the quantities to be estimated.

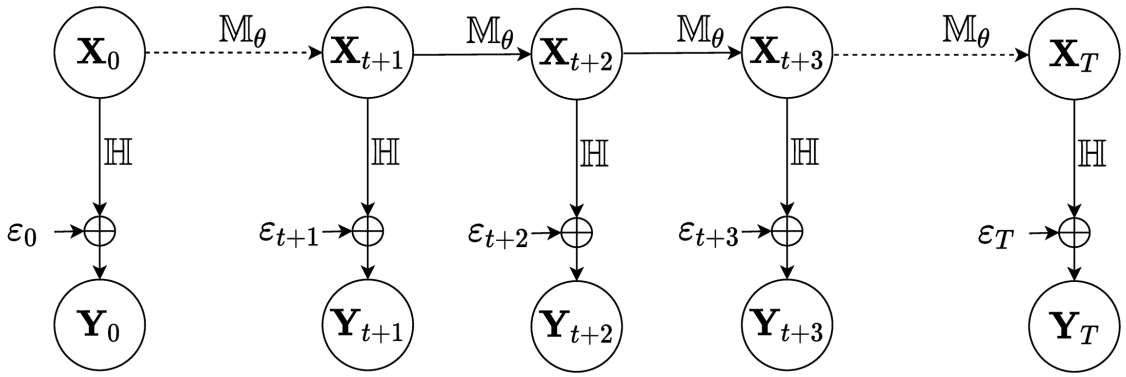


Figure 2.9: System representation as a parametric hidden Markov model

In the variational formalism this can be expressed through the minimization of an energy function of the general form $J(\mathbf{X}, \boldsymbol{\theta}) = \sum_t \|\mathbf{Y}_t - \mathbb{H}\mathbf{X}_t\| + \sum_t \|\mathbf{X}_{t+1} - \mathbb{M}_\theta(\mathbf{X}_t)\|$, where the first term is the observation error and the second the model error. One major difficulty of such an optimization problem arises from the fact that the control variables are of different natures. Usually, system states are estimated over a temporal window which may not include enough examples to train a machine learning model and more particularly a neural network. We may consider several trajectories $\left[\mathbf{X}_0^i, \ldots, \mathbf{X}_T^i\right]$ even though $\boldsymbol{\theta}$ should not depend on a particular one. Ultimately, we choose to avoid the excessively ill-posed joint optimization problem.

Coordinate descent: Alternate DA and ML

When $\boldsymbol{\theta}$ is known, DA methods combining $\mathbb{M}_\theta$ and $\mathbf{Y}$ can produce state estimation $\widehat{\mathbf{X}}$. On the flip side, when the system is fully observed with total confidence, $\boldsymbol{\theta}$ can be learned by regression. Alternating DA and ML steps, we successively optimize along $\mathbf{X}$ and $\boldsymbol{\theta}$. This is why we refer to the algorithm we use as *coordinate descent* (see Algorithm 2).

Algorithm 2 Coordinate descent

Initialize $\boldsymbol{\theta}$
while Convergence **do**
  for every trajectory i **do**
    States estimation $\widehat{\mathbf{X}}^i$ minimizing $\sum_t \|\mathbf{Y}_t^i - \mathbb{H}\mathbf{X}_t^i\|$
  end for
  Learning $\boldsymbol{\theta}$ minimizing $\sum_{i,t} \|\mathbf{X}_{t+1}^i - \mathbb{M}_\theta(\mathbf{X}_t^i)\|$
end while

2.6.2 Experiment

Available information We used the advection-based synthetic dataset introduced in Section 2.4.2 using partial and noisy observations. Only the passive tracer I is observed with an additive noise: $\mathbf{Y}_t = I_t = \mathbb{H}\mathbf{X}_t + \varepsilon_{R_t}$ where ε_{R_t} is a Gaussian white noise of known covariance matrix $\mathbf{R}_t$. Available information about the underlying physics is also limited: we assume to know the model except for the evolution equation on the motion field (non-linear advection).

Introducing our hybrid model Our main goal is therefore to recover this missing dynamics of the motion field which is never observed. To do so we introduce in Eq. 2.29 a parameterized hybrid dynamics $\mathbb{M}_\theta$ that combines a numerical scheme [96] representing the known physics $\mathbb{M}_P$ with a fully convolutional neural network $\mathbb{M}_L(\boldsymbol{\theta})$ to be trained to represent the unresolved part of the ground truth dynamics. In the perfect case, $\boldsymbol{\theta}^*$ is such that $f_{\theta^*}(\mathbf{w}) = \mathbf{w}.\nabla\mathbf{w}$.

$$\mathbb{M}_P + \mathbb{M}_L(\boldsymbol{\theta}) = \mathbb{M}_\theta \quad \text{the resolvent of the following PDE-system} \quad \begin{cases} \dfrac{\partial I}{\partial t} + \mathbf{w}.\nabla I = 0 \\ \dfrac{\partial \mathbf{w}}{\partial t} + f_\theta(\mathbf{w}) = 0 \\ \nabla I = 0, \quad \partial\Omega \\ \mathbf{w} = 0, \quad \partial\Omega \end{cases} \tag{2.29}$$

Learning scheme As depicted before and schematized in Figure 2.10, we will use a coordinate descent approach alternating assimilation and learning steps in order to ultimately train the hybrid model. To build a consequent enough training set, several trajectories are assimilated. During assimilation steps, $\boldsymbol{\theta}$ is fixed while during learning steps $\mathbf{X}$ is fixed. We initialize the procedure with the incomplete physics-based model which means that the first assimilation step is equivalent to a well-known variational optical flow estimation [97]. The next assimilation steps are performed with an evolution model of the motion field as in [95].

Data Assimilation step: strong-constraint 4DVAR States estimation is achieved on a sliding time window of size w with the 4DVAR algorithm which aims at solving the PDE-constrained optimization problem described in Eq. 2.30. While weak-constraint 4DVAR, allowing model errors, generally produces better results, we intentionally chose to use the strong version, assuming a perfect model, as we represent this model error with a parameterized dynamics model. Regularization parameters α and β are tuned using sequential model-based optimization [111] over forecast performance after assimilation.

$$\begin{aligned} \min_{\mathbf{X}_t} \quad & J_{DA}(\mathbf{X}_t;\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=t}^{t+w-1} \|\mathbf{Y}_i - \mathbb{H}\mathbf{X}_i\|^2_{R_i} + \frac{\alpha}{2}\|\nabla \mathbf{w}_t\|_2^2 + \frac{\beta}{2}\|\nabla.\mathbf{w}_t\|_2^2 \\ \text{s.t.} \quad & \mathbf{X}_{i+1} = \mathbb{M}_\theta(\mathbf{X}_i) \end{aligned} \tag{2.30}$$

Machine Learning step The ML step is a regression on estimated states minimizing $J_{ML}(\widehat{\mathbf{X}}_t, \widehat{\mathbf{X}}_{t+1}) = \|\widehat{\mathbf{w}}_{t+1} - (\widehat{\mathbf{w}_t} + \int_t^{t+1} f_\theta(\widehat{\mathbf{w}_t}))\|_2^2$. The neural network is trained by stochastic gradient descent and employs batch normalization, 3×3 kernel convolution and ReLu activation.

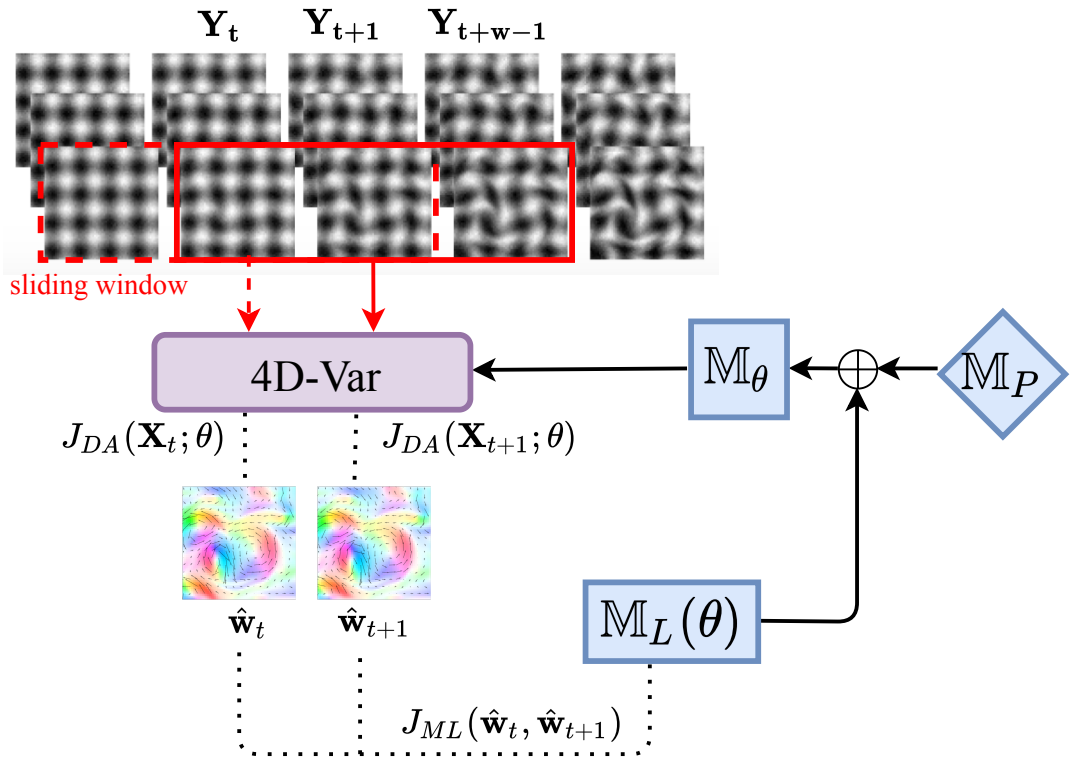


Figure 2.10: Schematic view of the learning scheme

Preliminary results - Forecast skill To evaluate the trained hybrid model $\mathbb{M}_\theta$, we produce forecasts over multiple initial conditions which were not used during the training

and compare them with ground truth trajectories by calculating RMSE on the tracer I. We benchmark the obtained hybrid model against the incomplete physics-based model $\mathbb{M}_P$ and a hybrid model trained on perfect motion field data $\mathbb{M}_\theta^p$ from ground truth simulation usually unknown. In Figure 2.11, we see that $\mathbb{M}_\theta$ outperforms the physics-based model $\mathbb{M}_P$ and is relatively close to $\mathbb{M}_\theta^p$ whose performance is only limited by the network architecture choice. Also, reduced uncertainty indicates the ability to generalize particularly in areas with intense motion. In conclusion, we succeeded in partially retrieving a missing dynamics of a fully-unobserved variable in a relatively high-dimensional system.

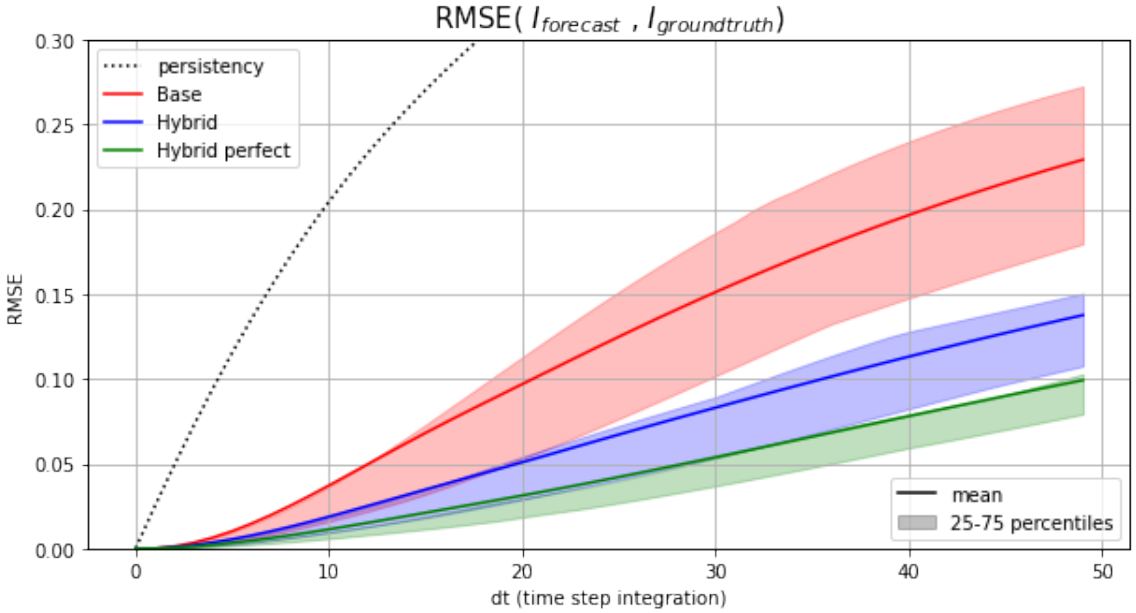


Figure 2.11: Forecast skills of the completed dynamics

2.6.3 Discussion

Engineering problem

While a numerical scheme can be used at the desired temporal scale, the hybrid model we introduced fixes the time step. A partial solution would be to fix it at the smallest possible scale and then iterate when a long-period integration is needed, then take the risk of errors accumulating. But the main engineering issues come from the iterative algorithm. At each iteration, the whole database has to be assimilated and then a deep neural network has to be trained to make each iteration computationally intensive. That being said, we noticed that results after 1 iteration were already relevant even if the best models were obtained after 3 iterations. There is also the question of the weights initialization at each iteration which, with bad luck, can make the algorithm diverge. Finally, a broader computational issue that will not be addressed in the thesis is the cohabitation of numerical schemes, usually optimized on CPU, and neural networks,

optimized on GPU.

Philosophical problem

The proposed experiment may have some interest but was not realistic in the sense that we knew what we were looking for. Modeling the model error with a Markovian dynamics, we induced the perfect bias. Is it adequate to make this modeling choice when errors could come from the observation operator, unresolved processes, discretization, boundary conditions, or even from bias in the optimization algorithm? If yes, should not the model error covariance $\mathbf{Q}$ evolve with the model, and how should we update it? Aside from these considerations, the used sliding window is a numerical trick we could not justify properly with the Bayesian framework.

Moving forward

The engineering complication coupled with the doubts about the modeling framework pushed us to explore new ways to hybridize the methods with the hope of avoiding iterating algorithms. The thought was: if the classical variational assimilation framework is too rigid to welcome a neural network, maybe we should try the other way around and optimize the neural network with assimilation-based constraints. Having a 4DVAR coded in Pytorch was the first piece and how to integrate it into a neural architecture is addressed in the next chapters.

CHAPTER 3

A Deep Learning view of strong-constraint 4DVAR

3.1 4DVAR in the Deep Learning framework

3.1.1 Similarities and nuances between 4DVAR and Deep Learning

From a high-enough point of view, data assimilation and machine learning can be described by the same Bayesian network as well detailed in [10]. Variational assimilation and DL share even more, we discuss below similarities and nuances between 4DVAR and DL.

Model and Data The model in DA usually refers to a discretized PDE system derived from physical laws, while in DL it refers to the neural architecture. Even though deep architecture can be used to learn a dynamical system, the classical way is to directly learn the inversion of interest, in our case the system state estimation. DA focuses on one observational window which would constitute only one sample from the ML point of view. A classical inversion algorithm like 4DVAR would then be classified in the unlearned methods category [59], in the sense that no database is used. On the contrary, ML and particularly deep neural architectures are usually trained with numerous samples.

Non-linearity and non-convex optimization Non-linearities arises in both cases, weither it is from the underlying physics of the studied system, or explicitly programmed to widen the range of learnable functions. The difficulty of variational DA and DL opti-

mization problems directly comes from those non-linearities, rendering the loss function non-convex. Variational DA usually uses a quasi-Newton optimizer such as BFGS [43] which is a deterministic gradient descent. When dealing with a large database, the gradient becomes too costly to evaluate at each iteration. This is why DL architectures are trained with mini-batch stochastic gradient descent. Such optimization schemes have proved to be very useful allowing to escape sharp minima and having a regularizing effect [112]. Also, optimization of highly-non-convex landscape related to deep architecture has progressed dramatically thanks to momentum and adaptive learning rate strategies [113], making DL methods more accessible and then more appealing.

Control parameters Strong-constraint variational assimilation is an initial value optimal control problem so that 4DVAR optimizes the initial condition directly on the discretizing grid. Control parameters in DL are the weights of the network and they are usually organized with care. As discussed in Section 2.3.1, architecture design encodes inductive bias allowing to deal with specific data and emphasizing particular estimation [21]. For instance architectures with stacked convolutional layers are great to deal with high-dimensional gridded data escaping the curse of dimensionality. We would like to emphasize that the organization of control parameters is of great interest.

Adjoint state and the backpropagation algorithm Even though control parameters have a different role in both optimization, an adjoint backpropagation through time steps or through hidden layers behaves in a similar manner, as pointed out in [14]. The gradient back-propagation algorithm [114] is literally the computational equivalent of the adjoint state method [36, 39, 44]. Such methods are implemented using automatic differentiation.

Software Even though all these analogies summed up in Table 3.1 are debatable, the possibility to solve variational problems using flexible tools from the deep learning community like Autograd [46] is real. Usually, solving variational assimilation problems implies having at disposal a discrete adjoint of the numerical model, using dedicated software like Tapenade [45], which is non-trivial and known to be an operational issue [2]. When designing a neural network architecture, only choices regarding the forward model are made. During the optimization, the adjoint network derived from the computational graph is used to calculate gradients. This part is fully managed by the software.

	Data Assimilation	Deep Learning
Data	one window	training set
Model	dynamics	inversion operator
Non-linearity	physics	activation function
Control parameters	state	weights
Cost function	non-convex	non-convex
Optimization	deterministic	stochastic
Automatic differentiation	adjoint state method	backpropagation algorithm

Table 3.1: Variational Data Assimilation versus Deep Learning

4DVAR Implementation in Pytorch

Re-writing and using 4DVAR in a deep learning friendly language was the first step toward developing hybrid data-driven and knowledge-driven inversion methods. Forward computational graph of strong-constraint 4DVAR and the associated algorithm can be found in Figure 3.1 and Algorithm 3, respectively.

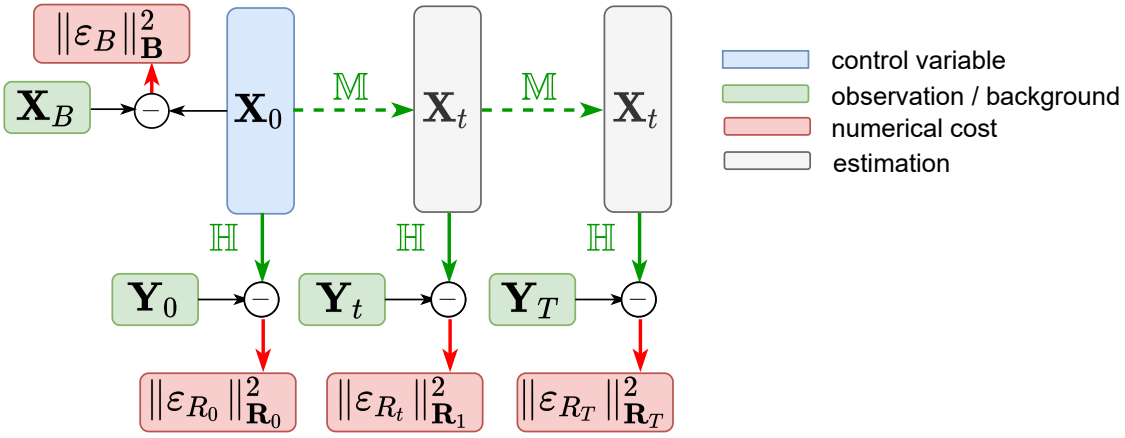


Figure 3.1: Forward computational graph of strong-constraint 4DVAR

Algorithm 3 –PyTorch 4DVAR

Initialize control variables $\mathbf{X}_0$
while stop criterion **do**
  forward: integrate $\mathbb{M}_{0\to T}(\mathbf{X}_0)$ and compute $\mathcal{J} = \|\varepsilon_{\mathbb{B}}\|^2_{\mathbf{B}} + \sum_t \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t}$
  backward: automatic differentiation returns $\nabla_{\mathbf{X}_0}\mathcal{J}$
  update: $\mathbf{X}_0 = \text{optimizer}(\mathbf{X}_0, \mathcal{J}, \nabla_{\mathbf{X}_0}\mathcal{J})$
end while
return $\mathbf{X}_0$

3.1.2 Comparison between C and Pytorch implementations

The work presented in this section has been the subject of a communication at ORASIS 2021 [115]. The code associated with the conducted experiments can be found at this GitHub address [1].

We aim at testing our Pytorch version of 4DVAR. To do so we will assimilate two sets of images, one from rain radar, and one giving sea surface temperatures. The considered physical processes can partially be described by an advection model hence the use of dynamics detailed in the Eqs. 2.26. We then compare the results with a reliable code developed in [95, 93] and show that there is no significant difference in terms of performance. However, we found that this simplicity and flexibility come at a computational cost.

Images assimilation experiments

Rain radar images In Figure 3.2, we display four consecutive radar rain maps obtained from MeteoNet network [116]. These images have been acquired over the region of Brest during an extremely rainy episode that occurred on January 3, 2018. The time step is 5 minutes, and the spatial resolution is approximately 1 square kilometer. We propose to estimate the velocity map from 4 consecutive acquisitions. As we do not have ground truth, the fourth acquisition will be extrapolated in time, using the estimated velocity map, to provide forecasts at various short time horizons. This is called *Rain Nowcasting*. Forecast images will be compared to Radar rain map acquisitions to provide relevant performance statistics.

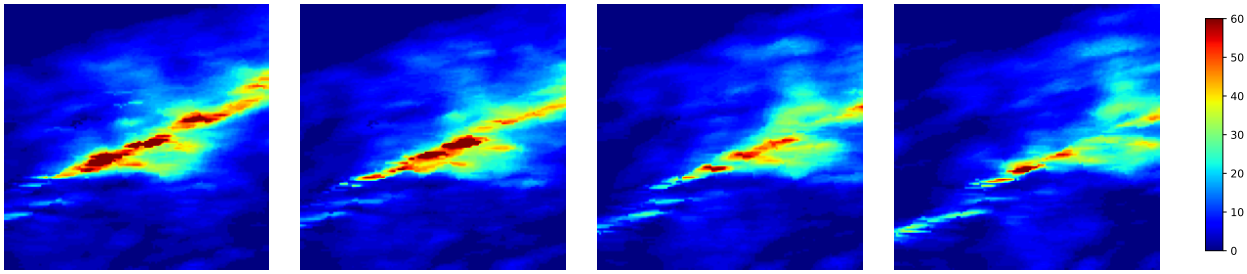


Figure 3.2: Four successive acquisitions of rainfall maps. Intensity unit is millimeter per hour

Sea surface temperature images Figure 3.3 shows four sea surface temperature maps (SST) over a small area of the North Atlantic Ocean. Data were obtained from the Marine Copernicus Service[2] and are the reanalyses of satellite observations through a physical model of the ocean. The time step is of 1 day, and the spatial resolution of 10

[1] `https://github.com/ArFiloche/Py4DVar`

[2] `https://resources.marine.copernicus.eu/?option=com_csw&task=results`

square kilometers. Similarly to rain maps, we propose to estimate velocity maps from 4 consecutive SST maps. Sea surface circulation data are available in Copernicus and used as a reference.

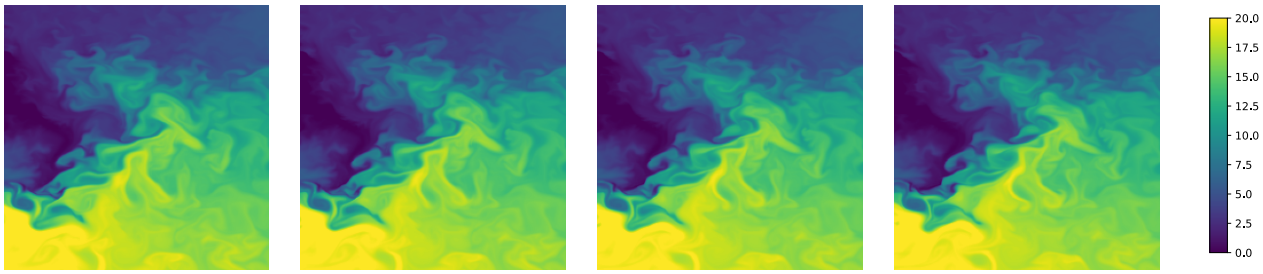


Figure 3.3: Four successive SST acquisitions of North Atlantic. Intensity unit is Celsius degree

The use of this model is physically justified in both our geophysics experiments. For rain nowcasting on a short time window, it describes well the transport of storm cells by wind, cells formation being neglected. The perfect model hypothesis is then acceptable and the strong-constraint 4DVAR is sufficient to estimate cells' velocities (motion fields). For the estimation of sea surface circulation (motion fields) from SST images, Eq. (2.26) constitutes an approximation of Navier-Stokes equations used in the reanalysis; therefore we use the weak-constraint 4DVAR.

Forecast

At the end of the assimilation process, an estimation of the full system state trajectory is available. This means that at each time t we have an estimation $\widehat{\mathbf{X}}_t = \begin{pmatrix} \widehat{\mathbf{w}}_t^\top & \widehat{I}_t \end{pmatrix}^\top$. To produce a forecast at a given time horizon, $T+k$, we simply integrate the evolution model using the end of the assimilation window as the initial condition: $\widehat{\mathbf{X}}_{T+k} = \mathbb{M}_{T \to T+k}(\widehat{\mathbf{X}}_T)$. In the forecast setup, model errors $\boldsymbol{\varepsilon}_m$ cannot be estimated and therefore are not considered.

3.1.3 Results

In this section, we present experimental results obtained on both datasets. We compare our newly developed Pytorch code with a C code, already used in several experiments [95, 93] where the adjoint dynamics is obtained with Tapenade.

Nowcasting on rain radar images

After strong 4DVAR assimilation of 4 rain radar images (Figure 3.2) on a window representing a 15-minute period, we integrate the estimated state further in time to produce forecast at 10, 20, and 30 minutes horizons. As depicted in Figure 3.4 both versions of

the algorithm behave in the same manner. Also, we note a smoothing effect which is an issue of the semi-Lagrangian scheme and the counterpart of an implicit scheme.

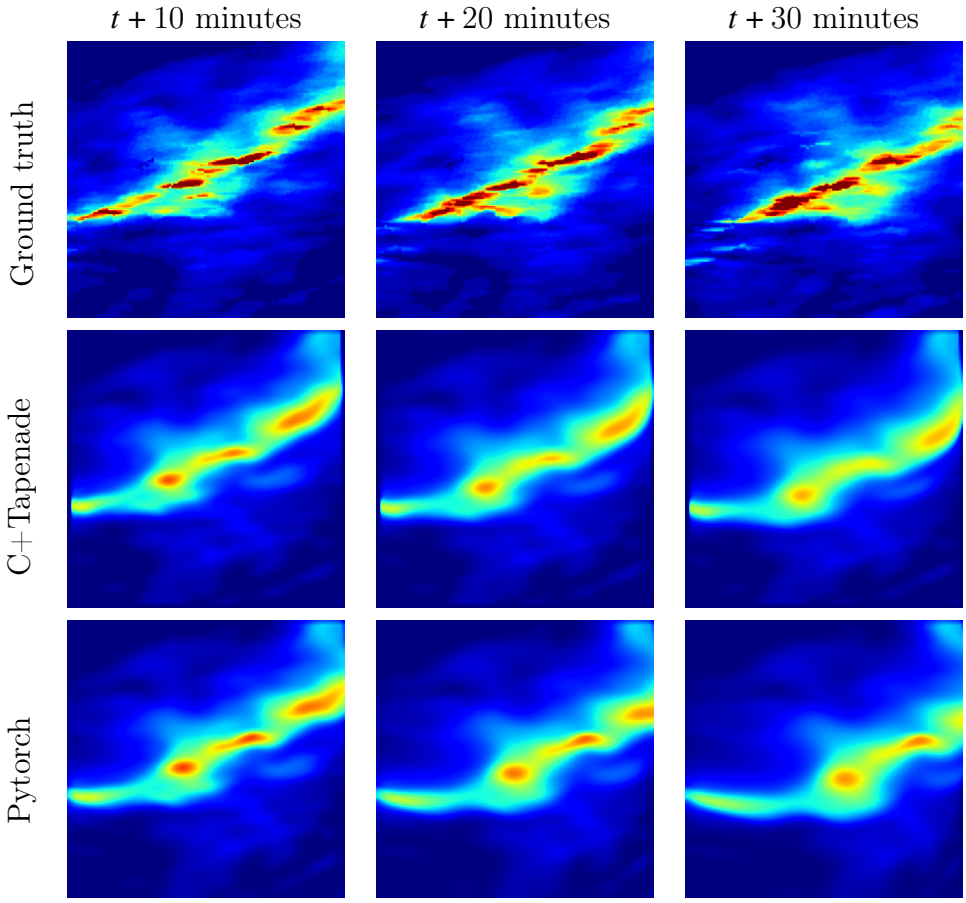


Figure 3.4: Example of rain map forecasts at three horizon times produced by C and Python codes

When repeating the assimilation algorithm over sliding windows, we can evaluate our forecast over a longer time period. In Figure 3.5 we consider a full day containing almost 800 acquisitions. After each assimilation, the forecasts are evaluated against the available ground truth using RMSE on the tracer I_t. Dealing with rain nowcasting, other metrics can be more informative than RMSE but in our case, it is enough to verify that both codes produce comparable performances.

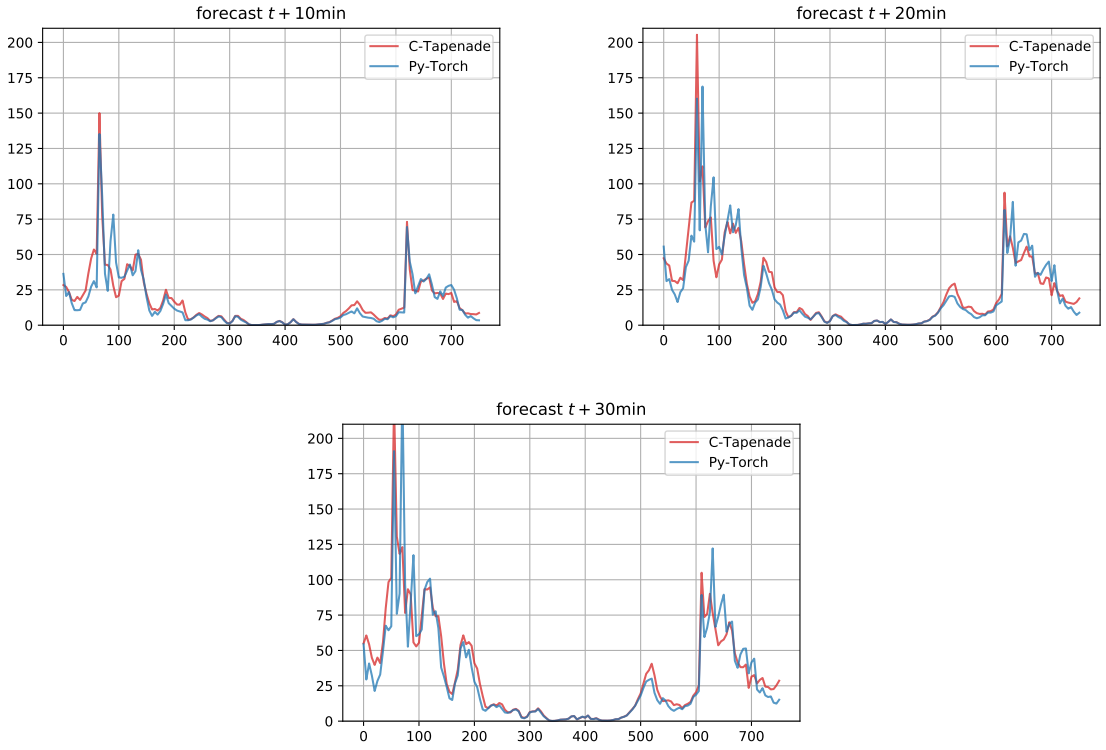


Figure 3.5: Rainfall forecasts at 10, 20 and 30 minutes: performances of C and Pytorch codes (RMSE)

Sea surface circulation from SST images

The exact same process is applied for SST forecasting. The 3-day windows are assimilated with the weak-constraint algorithm and forecasts at three horizon times as shown in Figure 3.6. SST circulation being a slow-evolving dynamics, it is hard to visualize differences.

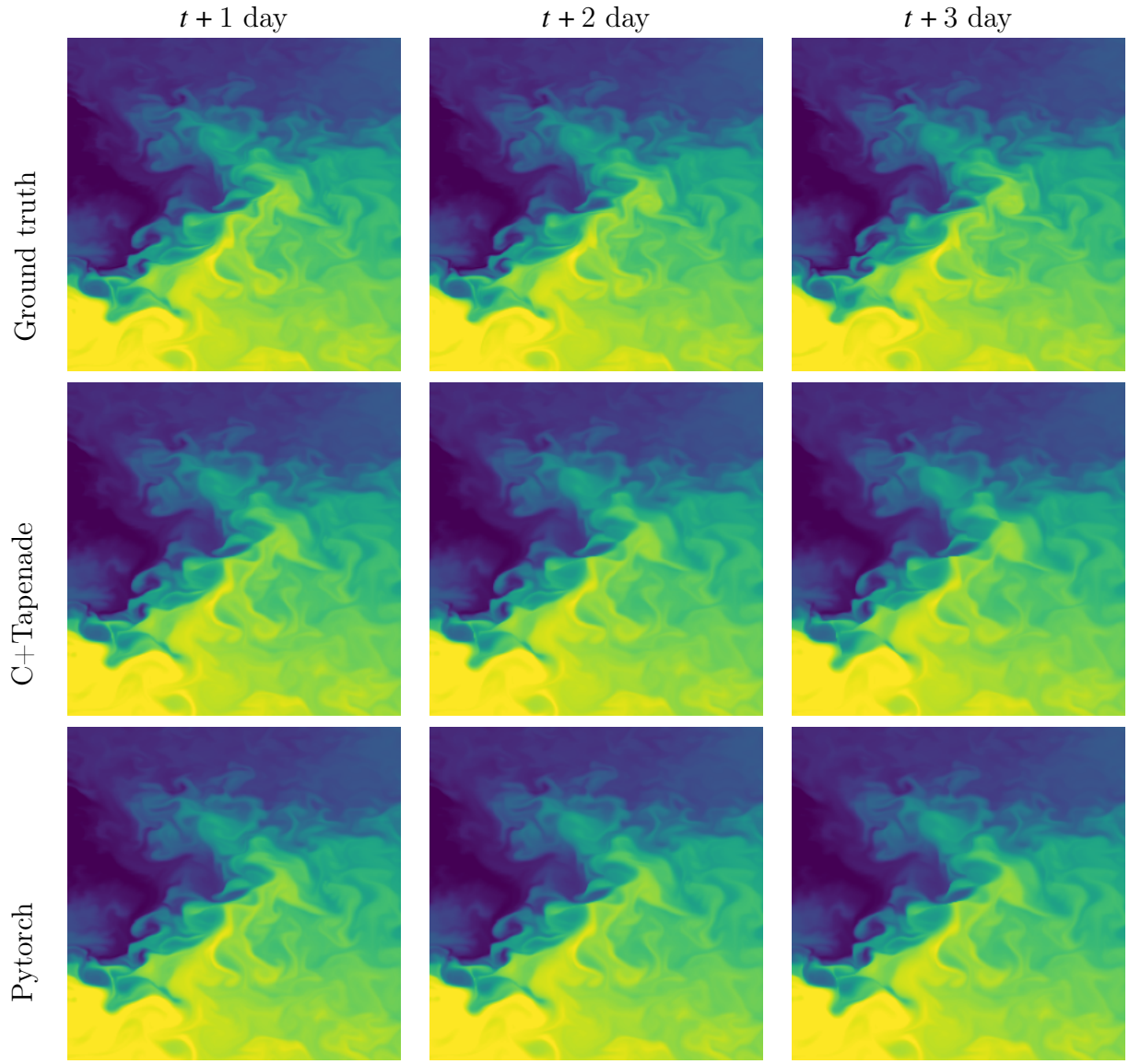


Figure 3.6: Example of SST forecasts at three horizon times produced by C and Python codes

However, we can similarly calculate the RMSE between the tracer $\boldsymbol{I}_t$ and the ground truth as plotted in Figure 3.7.

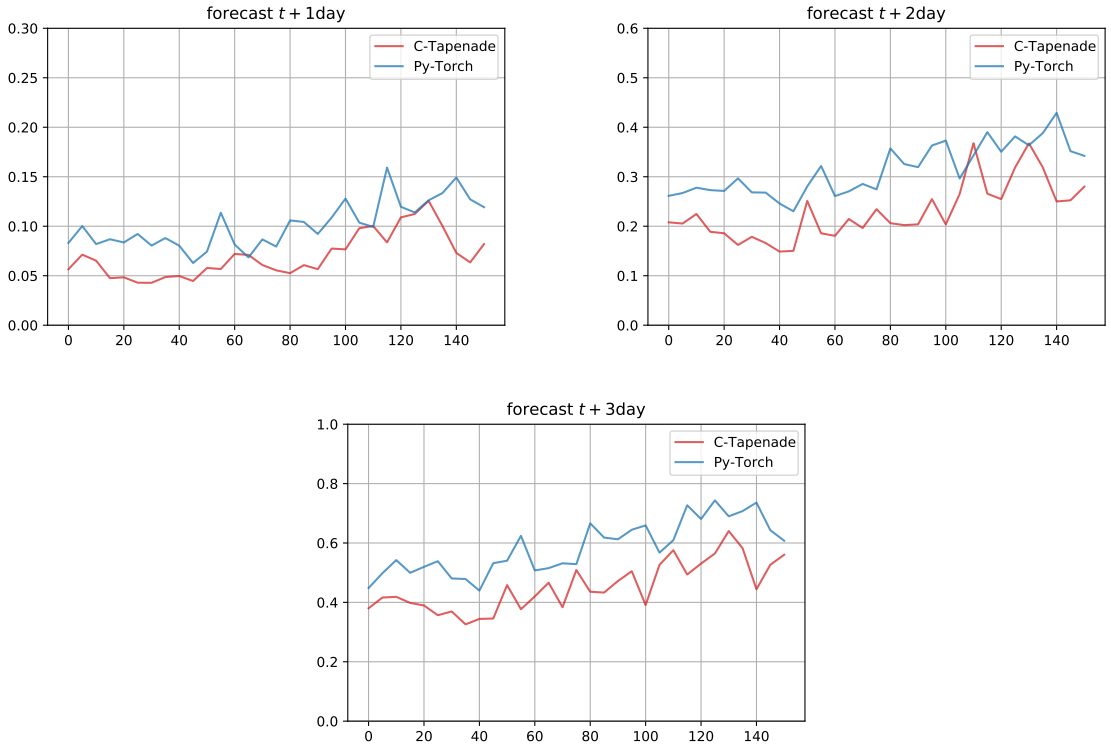


Figure 3.7: SST forecast: performances of C and Pytorch codes for three horizon times. Metric is RMSE

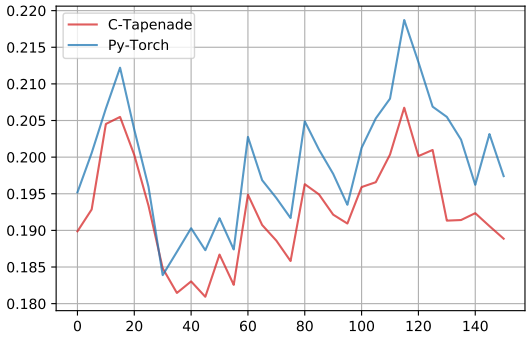
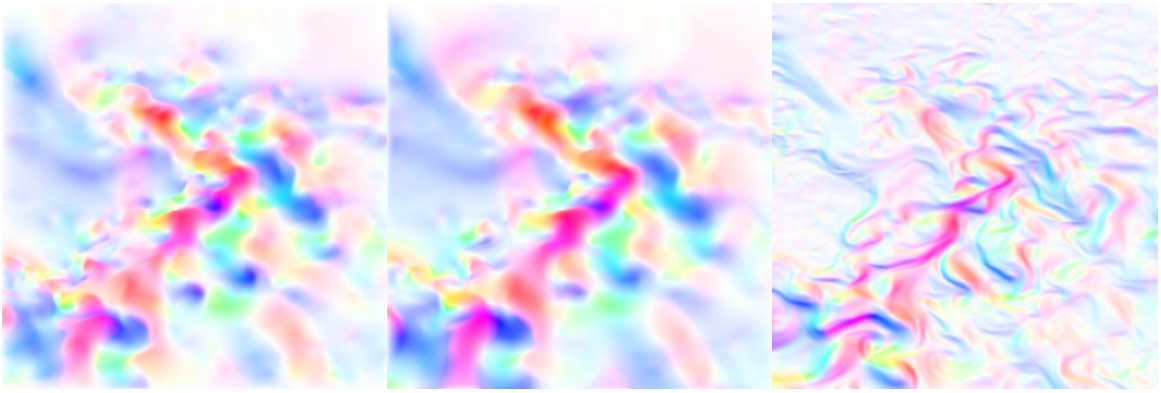


Figure 3.8: Performance of C and Pytorch codes, metric is EPE

In the particular case of SST data coming from another assimilation process, we have access to ground truth motion fields $\mathbf{w}$. In Figure 3.9, we can see assimilated motion fields against the ground truth, colors standing for motion vectors orientation [98]. We also can quantify these differences over time by calculating the end-point error (EPE) between assimilated fields and the ground truth as presented in Figure 3.8. Again, we can conclude that both codes are roughly equivalent.



(a) C+Tapenade (b) Pytorch (c) Ground-truth

Figure 3.9: Example of velocity maps obtained with C and Pytorch codes, compared to the ground-truth

Discussion

The Pytorch code aims at replicating the C reference as close as possible. However, results obtained are not exactly the same and we believe it can be explained by several reasons. First, the gradients coming from regularization terms are calculated manually in the C version but automatically in the other one. Second, even though both codes use an optimizer based on the same second-order quasi-Newton method, these optimizers came from different developments. C code uses the original implementation of L-BFGS [117] written in Fortran by the authors while Pytorch uses its own implementation. Third, the order of successive operations is not strictly respected which may lead to numerical rounding differences. Lastly, we compare in Table 3.2 the computation time needed in each case using a standard CPU architecture. Unsurprisingly, the C code version is much faster, the Pytorch simplicity could not be free. Also, we note that the gap between versions is more important in the strong case and this is because the C code has been optimized using high-performance computing techniques based on vectorization and parallel programming.

Code paradigm	Rain (strong)	SST (weak)
C-Tapenade	2.0 ± 0.3 s	25.8 ± 1.1 s
Pytorch	90.0 ± 43.6 s	71.3 ± 1.1 s

Table 3.2: Computation time comparison between 4DVAR in C and Pytorch on two different images assimilation tasks

3.1.4 Overfitting and Regularization

The code associated with the conducted experiments can be found at this GitHub address [3].

In this section, we provide baseline assimilation performances that only are applications of the existing 4DVAR version. We hope to exhibit the overfitting behavior of variational inversion when the noise level increases but also the regularizing effect of the Gaussian background prior hypothesis. To do so we will use the synthetic datasets detailed in Section 2.4.2 and study assimilation performances.

Overfitting with maximum likelihood estimation

When no background prior assumption is made, the 4DVAR algorithm only optimizes the observational cost, $\mathcal{J}(\mathbf{X}_0) = \frac{1}{2}\sum_{t=0}^{T} \|\mathbf{Y}_t - \mathbb{H}_t \circ \mathbb{M}_{0\to t}(\mathbf{X}_0)\|^2_{R_t}$, which corresponds to a maximum likelihood estimation. If the forward operator is invertible, this works well when observations are noiseless, but otherwise, the problem becomes ill-posed and the estimation is degraded by overfitting data. We conducted an assimilation experiment augmenting progressively the level of noise. In Figures 3.11 and 3.13, we display scores inside the assimilation window after optimization. We notice the expected behavior. The estimation fits practically perfectly the observation independently from the level of noise while the rest of the estimated trajectory quickly deteriorates. Looking at the estimated motion field displayed in Figures 3.10 and 3.12, we see that the noisier the observation, the more the motion field is unusable for a potential forecast.

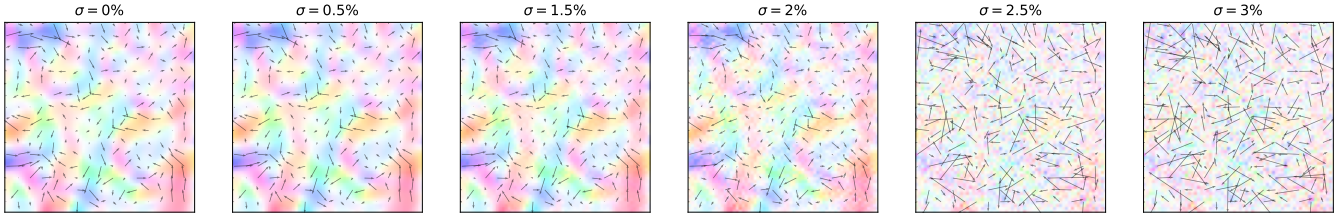


Figure 3.10: Shallow water system: Estimated motion field from 4DVAR without background, evolution regarding noise level

[3] `https://github.com/ArFiloche/Variational_Assimilation`

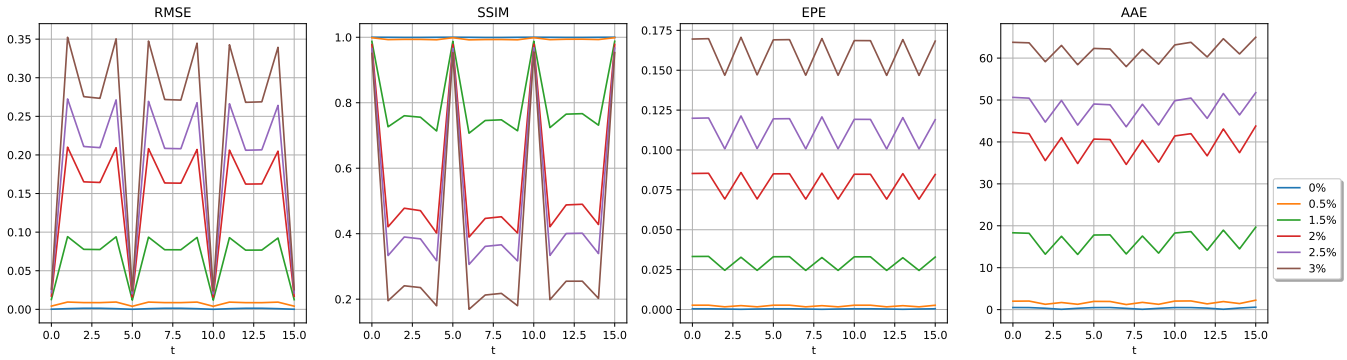


Figure 3.11: Shallow water system: Assimilation performances of 4DVar without background, evolution regarding noise level

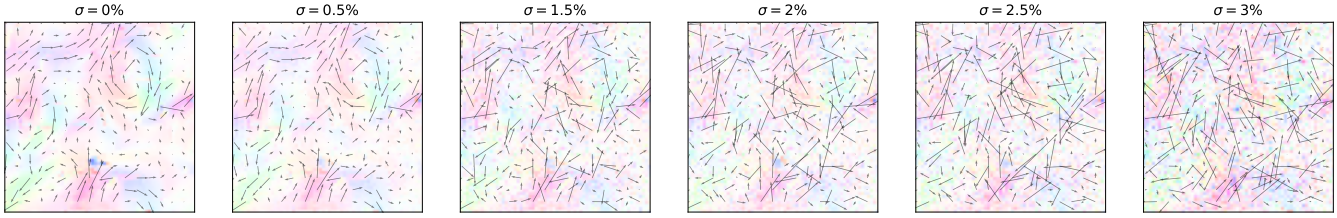


Figure 3.12: Advection system: Estimated motion field from 4DVAR without background, evolution regarding noise level

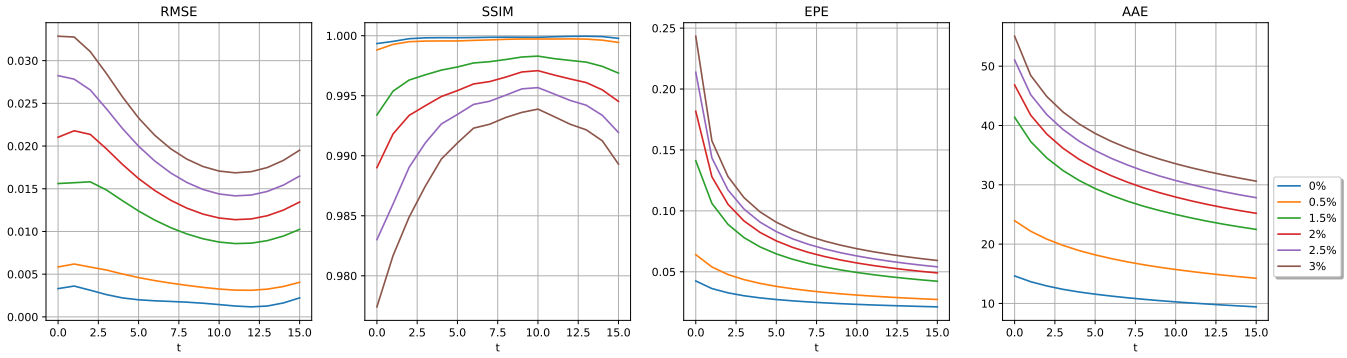


Figure 3.13: Advection system: Assimilation performances of 4DVAR without background, evolution regarding noise level

3.1.5 Regularization with Gaussian background prior

The Gaussian background prior hypothesis aims at regularizing the estimation. The 4DVAR algorithm then optimizes the observational and background cost: $\mathcal{J}(\mathbf{X}_0) = \frac{1}{2}\sum_{t=0}^{T} \|\mathbf{Y}_t - \mathbb{H}_t \circ \mathbb{M}_{0\to t}(\mathbf{X}_0)\|^2_{R_t} + \|\mathbf{X}_0\|^2_B$ which corresponds to a maximum a posteriori estimation. The employed background regularization is the one introduced in Section 2.4.2.

We conducted the same noise sensitivity experiment augmenting progressively the level of noise, displaying assimilation score in Figures 3.15 and 3.17 and estimated motion field in Figures 3.14 and 3.16. We clearly see the regularizing effect avoiding overfitting the noisy observation. On the motion estimation, the difference is blatant and estimated fields stay smooth for all the levels of noise.

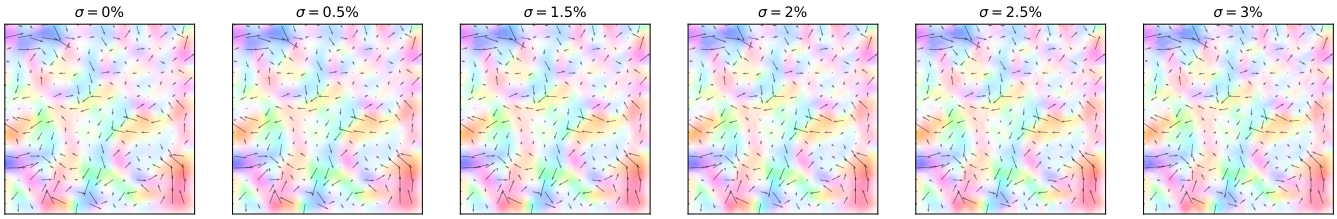


Figure 3.14: Shallow water system: Estimated motion field from 4DVAR with background regularization, evolution regarding noise level

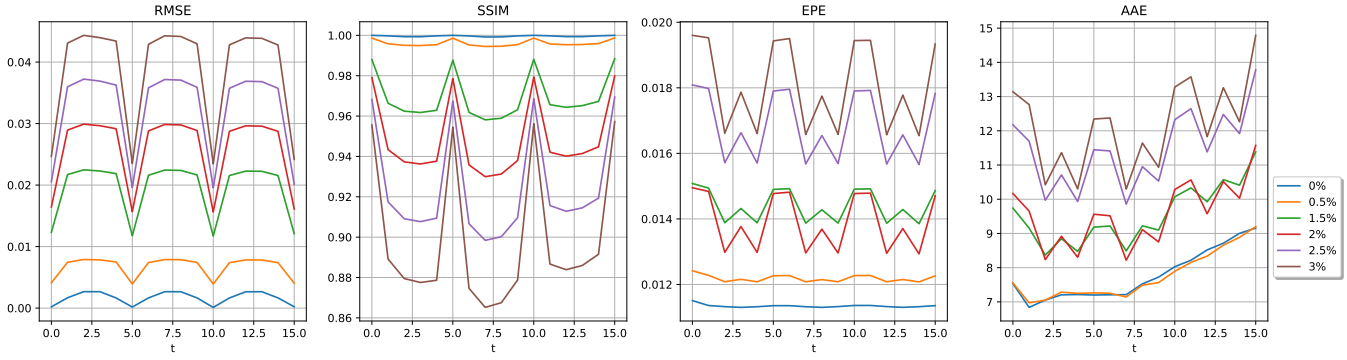


Figure 3.15: Shallow water system: Assimilation performances of 4DVAR with background regularization, evolution regarding noise level

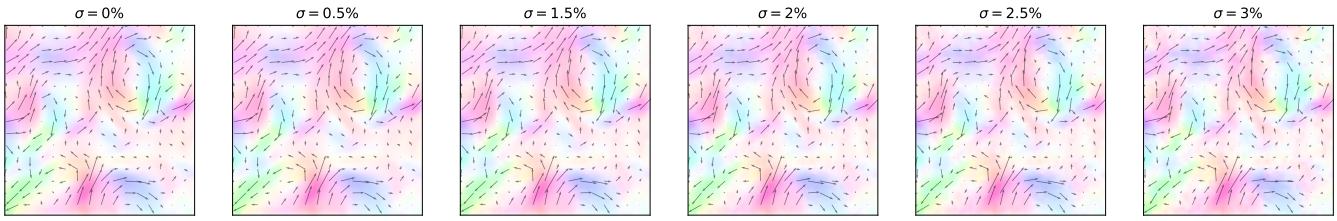


Figure 3.16: Advection system: Estimated motion field from 4DVAR with background regularization, evolution regarding noise level

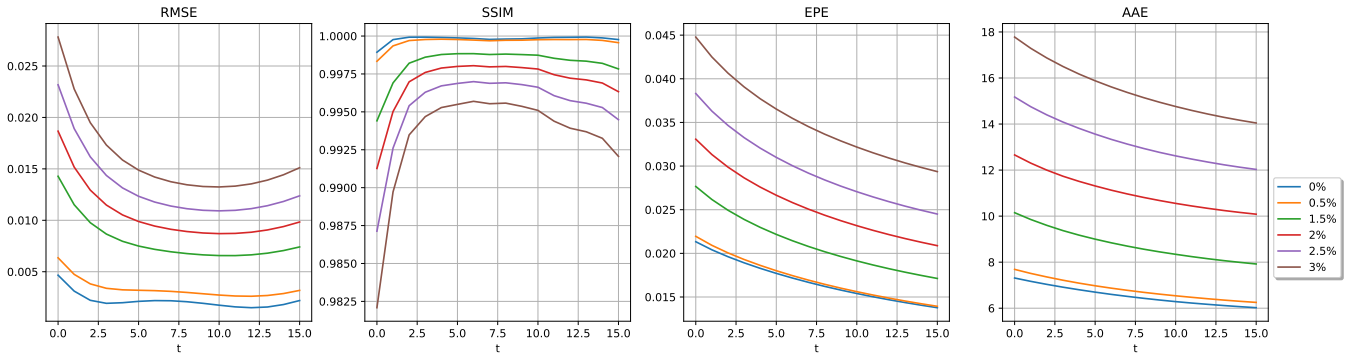


Figure 3.17: Advection system: Assimilation performances of 4DVAR with background regularization, evolution regarding noise level

3.2 Learning 4DVAR inversion directly from observation

3.2.1 Motivation

Again with the ambition to bridge DA-ML and after the encountered issue in the iterative experiment given in Section 2.6, we decided to look for an hybrid architecture. The latter should be trained on database in only one optimization step and still using only information about imperfect observation and a dynamical model. Starting modestly, we used the perfect model hypothesis so that in this section, $\mathbb{M}$ will be perfectly known.

3.2.2 Method

Inspired by physics-constrained deep architectures discussed in Section 2.3.4, we decided to use a neural network $\mathcal{F}_{\boldsymbol{\theta}}$ that should output initial conditions from observations and associated errors covariance (see Eq 3.1), which is exactly the task solved by 4DVAR. We make the convenient hypothesis that observational errors are also uncorrelated in space so that $\mathbf{R}^{-1}$ is diagonal and this diagonal can be reshaped in the observation format.

$$\mathcal{F}_{\boldsymbol{\theta}} : (\mathbf{Y}, \mathbf{R}^{-1}) \mapsto \widehat{\mathbf{X}}_0 \tag{3.1}$$

We consider having a dataset denoted $\mathcal{D} = \{\mathbf{Y}^{(i)}, \mathbf{R}^{-1(i)}\}_{i=1}^{N}$ not containing the ground truth so that the supervised setting is not an option. To train the neural network, we forward the estimated initial condition with the dynamical model and then calculate the observational, as in 4DVAR. Gradients are back-propagated through the dynamical model first and then through the neural network. A schematic view of the performed forward integration is drawn in Figure 3.18.

3.2.3 Bayesian view

We first consider a sample $\mathbf{Y}$ of $\mathcal{D}$ and the associated ground truth $\mathbf{X}$. Using the notation of the classical variational assimilation Bayesian framework introduced in Section 2.2.2, the perfect model hypothesis gives $p(\mathbf{X}) = p(\mathbf{X}_0)$ and the modeling choice gives $p(\mathbf{X}_0) = \delta(\mathbf{X}_0 - \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{Y}, \mathbf{R}^{-1}))$, where δ is the Dirac measure. Simply adapting the posterior leads to Eq. 3.2, where K is a constant.

$$\begin{aligned} -\log p(\mathbf{X} \mid \mathbf{Y}) &= \frac{1}{2} \sum_{t=0}^{T} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} - \log K \\ &\text{s.t.} \quad \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{Y}, \mathbf{R}^{-1}) = \mathbf{X}_0 \quad \text{and} \quad \mathbb{M}(\mathbf{X}_t) = \mathbf{X}_{t+1} \end{aligned} \tag{3.2}$$

Now, considering that all the trajectories in the dataset are independent and identically distributed, and denoting the desired ground truth set $\mathcal{T} = \{\mathbf{X}^{(i)}\}_{i=1}^{N}$, the posteriors for each trajectory are also independent as written in Eq. 3.3

$$-\log p(\mathcal{T} \mid \mathcal{D}) = \sum_{i=0}^{N} \log p(\mathbf{X}^{(i)} \mid \mathbf{Y}^{(i)}) \tag{3.3}$$

Cost function

Then the cost function associated with the MAP estimation can be developed as in Eq. 3.4. The adopted notation makes it complicated but the loss is just the sum of the observational loss on all the dataset. A simple way of thinking about it is to run multiple 4DVAR in parallel to optimize a common set of control parameters $\boldsymbol{\theta}$.

$$\mathcal{J}(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=0}^{N} \sum_{t=0}^{T} \|\varepsilon_{R_t^{(i)}}\|^2_{\mathbf{R}_t^{(i)}} = \frac{1}{2} \sum_{i=0}^{N} \sum_{t=0}^{T} \|\mathbf{Y}_t^{(i)} - \mathbb{H}_t \circ \mathbb{M}_{0\to t} \circ \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{Y}, \mathbf{R}^{(i)-1})\|^2_{R_t^{(i)}} \tag{3.4}$$

Gradient

We can deduce an analytical expression of the gradient, first using the linearity of the gradient (Eq. 3.5), then the chain rule (Eq. 3.6) and finally using the adjoint state method (Eq. 3.7) as introduced in Section 2.2. However, the gradient on the whole dataset will never be calculated as we will optimize the architecture in a deep learning fashion using mini-batch gradient descent. Interestingly, each sample in a batch will lead to only one backward integration of the dynamics.

$$\nabla_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}) = \sum_{i=0}^{N} \sum_{t=0}^{T} \nabla_{\boldsymbol{\theta}} \|\varepsilon_{R_t^{(i)}}\|^2_{\mathbf{R}_t^{(i)}} \tag{3.5}$$

$$\nabla_{\boldsymbol{\theta}} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} = \nabla_{\mathbf{X}_0} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} \nabla_{\boldsymbol{\theta}} \mathbf{X}_0 = \nabla_{\mathbf{X}_0} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} \nabla_{\boldsymbol{\theta}} \mathcal{F}_{\boldsymbol{\theta}}(\mathbf{Y}, \mathbf{R}^{-1}) \tag{3.6}$$

$$\nabla_{\mathbf{X}_0} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} = \left[\frac{\partial (\mathbb{H}_t \circ \mathbb{M}_{0\to t})}{\partial \mathbf{X}_0} \right]^{\top} {\mathbf{R}_t}^{-1} \varepsilon_{R_t} \tag{3.7}$$

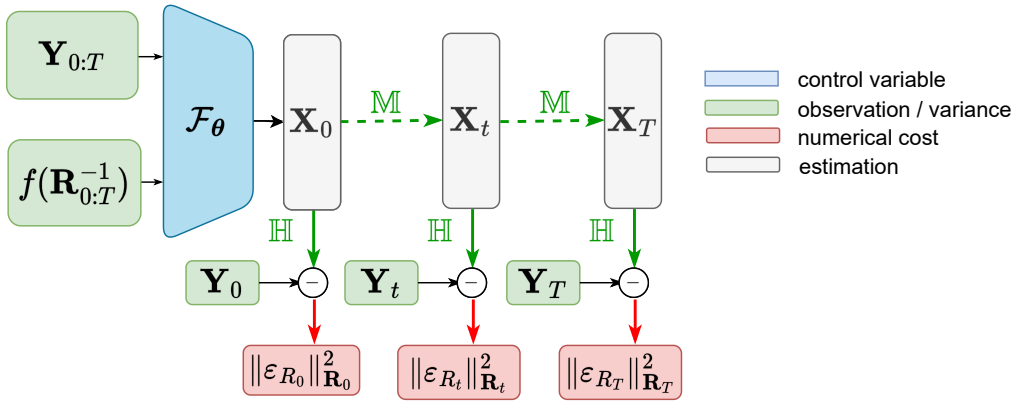


Figure 3.18: Forward computational graph of the hybrid architecture learning 4DVAR in an end-to-end manner. Functional f indicates an optional pre-processing for numerical purposes.

3.2.4 Experiment settings

Lorenz96 dynamical model

We use the Lorenz96 dynamics [118] as an evolution model (see Eq. 3.8) numerically integrated with Runge-Kutta 4 scheme. Here n indexes a one-dimensional space. On the right-hand side, the first term corresponds to an advection, the second term represents damping, and F is an external forcing. We use the parameters $dt = 0.1$ and $F = 8$ corresponding to a chaotic regime.

$$\frac{d\mathbf{X}_{t,n}}{dt} = (\mathbf{X}_{t,n+1} - \mathbf{X}_{t,n-2})\mathbf{X}_{t,n-1} - \mathbf{X}_{t,n} + F \tag{3.8}$$

Observation

Starting from white noise and after integrating during a spin-up period to reach a stationary state, we generate ground truth trajectories. To create associated observations, we use a randomized linear projector as an observation operator, making the observation sparse to finally add a white noise. Noises at each point in time and space can have different variances, $\varepsilon_{R_{n,t}} \sim \mathcal{N}(0, \sigma_{n,t})$, and we use the associated diagonal variance matrix defined by $\mathbf{R}^{-1}_{n,t} = \frac{1}{\sigma^2_{n,t}}$. Figure 3.19 displays an example of simulated observations. Variances are sampled uniformly such that $\sigma_{n,t} \sim \mathcal{U}(0.25, 1)$. When a point in the grid is not observed, we fix "$\mathbf{R}^{-1}_{n,t} = 0$", which corresponds to an infinite variance meaning a lack of information. From a numerical optimization view, no cost means no gradient back-propagated which is the desired behavior.

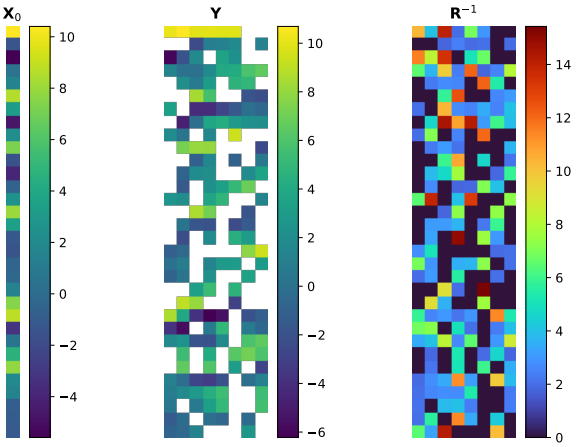


Figure 3.19: Observation generated with the Lorenz96 model, a randomized linear projector as observation operator and a white noise.

Algorithm benchmarks

We evaluate our method (NN-4DVAR-e2e) on the assimilation task which is estimating $\mathbf{X}_0$. We compare it with 4DVAR, a neural network trained on the output of the 4DVAR estimation (NN-4DVAR-iter), and a neural network trained with the ground truth (NN-perfect). The latter should represent the best-case scenario for the chosen architecture while NN-4DVAR-iter plays the role of the iterative method. The same neural architecture is used for all the methods involving learning. Its design is fairly simple, being composed of 5 convolutional layers using 3×3 kernels, ReLu activation, no down-scaling, and a last layer flattening the two-dimensional maps into the shape of $\mathbf{X}_0$. Table 3.3 sums up the differences between the algorithms.

Algorithm	$\mathbf{Y}$	$\mathbf{R}^{-1}$	$\mathbf{X}$	Optimization	# of $\mathbb{M}$ forward
4DVAR	✓	✓	✗	1 step N time	n_iter
NN-4DVAR-iter	✓	✓	✗	2 steps 1 time	$N \times n\_iter$
NN-4DVAR-e2e	✓	✓	✗	1 step 1 time	$N \times n\_epoch$
NN-perfect	✓	✓	✓	1 step 1 time	0

Table 3.3: Summary of the materials needed to optimize the algorithms: 4DVAR, NN-4DVAR-iter, NN-4DVAR-e2e and NN-perfect

3.2.5 Additional details

The Dataset is composed of 550 samples, 250 are used for training, 50 are used for testing if a learning procedure is involved and 250 are kept for validation. The displayed score

metrics will only concern the validation set. 4DVAR is optimized with an L-BFGS solver while for neural networks we use Adam. The learning procedure involves 50 epochs using a batch size of 16.

3.2.6 Results

We ran the experiment one time using a background matrix $B = \lambda \mathbf{I}_d$ to regularize 4DVAR estimation (see Fig 3.20) and one time without, optimizing 4DVAR only on the observational cost (see Fig 3.21). To evaluate the accuracy of an estimation, we use the $RMSE(\widehat{\mathbf{X}}_0, \mathbf{X}_0) = \sqrt{\sum_n (\widehat{\mathbf{X}}_{n,0} - \mathbf{X}_{n,0})^2}$ and the $Bias(\widehat{\mathbf{X}}_0, \mathbf{X}_0) = \sum_n (\widehat{\mathbf{X}}_{n,0} - \mathbf{X}_{n,0})$.

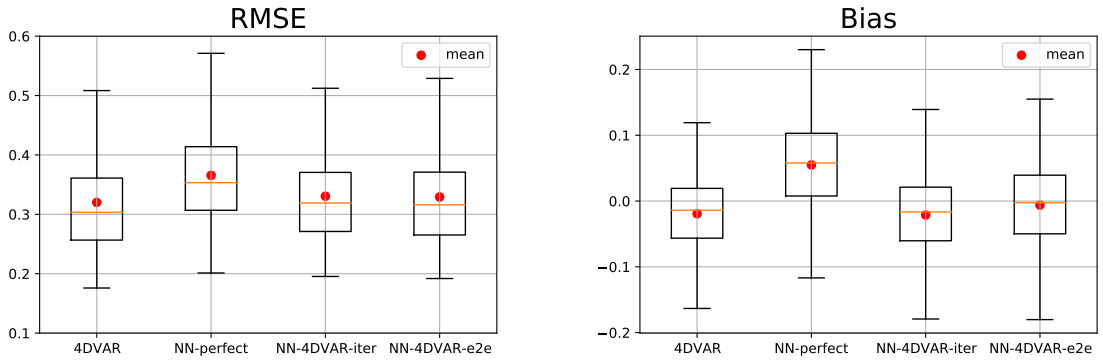


Figure 3.20: Lorenz96 assimilation experiment: Boxplot of RMSE and Bias, comparing 4DVAR-B, NN-4DVAR-iter, NN-4DVAR-e2e and NN-perfect, 250 samples

The first experiment shows that 4DVAR provides the best estimations. Astonishingly, the neural network trained on perfect data is the worst performing and we could not find an explanation. However, the 3 learning-based algorithms have not significant differences in terms of RMSE. Regarding the bias, 4DVAR-NN-e2e is the less biased algorithm.

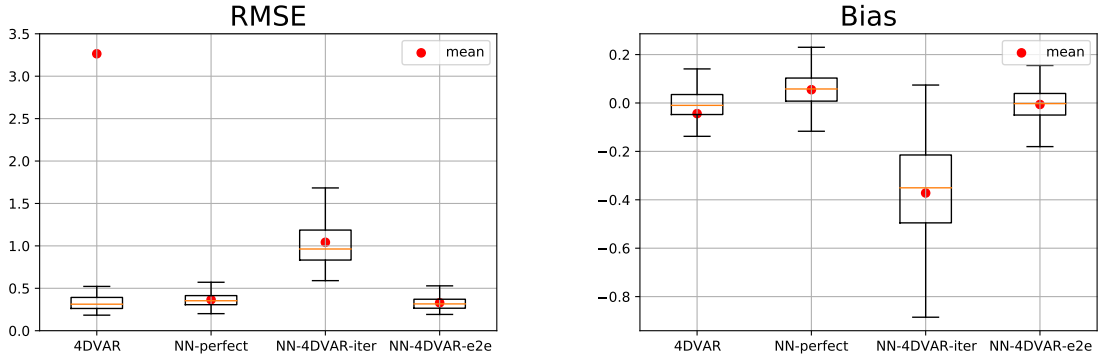


Figure 3.21: Lorenz96 assimilation experiment: Boxplot of RMSE and Bias, comparing 4DVAR, NN-4DVAR-iter, NN-4DVAR-e2e and NN-perfect, 250 samples

While the perturbation may be too artificial, the second experiment shows that if 4DVAR is not already tuned to provide good estimations, then building a dataset of estimation from it may be an issue. One cumbersome solution could be to filter manually the bad estimation but this would require designing a criteria.

3.2.7 Critical discussion

Some obvious critics can be made to the proposed method. First, if the observational noise is at least correlated in space, the associated covariance matrix cannot be used as a mask along the observation. One could argue that such a matrix being real symmetric is therefore diagonalizable, so that a change of basis could solve this issue. However, this transformation has to be the same for each trajectory for the proposed method to remain usable. Also, the method fixes the maximum assimilation window size. For smaller windows, it can still be used filling the masking variance with zeros accordingly but for larger ones, the only possibility is to use sliding windows, then raising to question of the coherence in time. Typically, the method in that form can not fit quasi-static strategies [119] employed in variational assimilation.

However, the method may have a computational interest. As depicted in Table 3.3, learning the inversion directly with our method may be less computationally costly, in terms of dynamics integration, depending on the ratio $\frac{n\_iter}{n\_epoch}$ (comparing the number of iteration in 4DVAR with the number of epoch in the learning). Also, once learned, the inversion will not require estimation, but this is also true for the iteratively-learned network.

Perspective

The lesson we took from this experiment is that it is probably hard to beat a well-tuned strongly-constrained 4DVAR and that optimizing on one specific window may be an advantage while learning over a database may have a smoothing effect. But we also exhibited that a neural architecture has desirable regularizing effects. Indeed the proposed architecture did not use additional regularization and provided good results optimizing only the observational cost. In the next chapter, we investigate more deeply this effect but assimilating on only one window.

CHAPTER 4

Variational Assimilation with deep prior

The classical variational assimilation cost function is derived from modeling errors prior with uncorrelated in times Gaussian distribution. The optimization then relies on error covariance matrices as hyperparameters. But such statistics can be hard to estimate particularly for background and model errors. We propose to replace the Gaussian prior with a deep convolutional prior circumventing the use of background error covariances.

To do so, we reshape the optimization so that the initial condition to be estimated is generated by a deep architecture. The neural network is optimized on a single observational window, no learning is involved as in a classical variational inversion. The bias induced by the chosen architecture regularizes the proposed solution with the convolution operators imposing locality.

We propose several experiments highlighting the regularizing effect of deep convolutional priors. First, we show that such prior can replace background regularization in a strong-constraint 4DVAR. We extend the idea in a 3DVAR set-up using spatio-temporal convolutional architecture to interpolate sea surface satellite tracks and obtain results on par with optimal interpolation with fine-tuned background matrix. Finally, we give perspective toward applying the same method accounting for model errors or uncertainty hoping to remove the need for model-errors covariances.

4.1 Strong-constraint 4DVAR with Deep Background Prior

The work presented in this section has partially been the subject of a communication at Climate Informatics 2022 [120]. The code associated with the conducted experiments can be found at this GitHub address [1].

4.1.1 Method

We use the exact same data assimilation framework described in Section 2.2 and consider the situation where the dynamical model $\mathbb{M}$ is time-independent and perfectly known so that $p(\mathbf{X}) = p(\mathbf{X}_0)$.

Deep background prior

A background usually provides prior information and acts like a regularization on the initial condition $\mathbf{X}_0$ to be estimated. As already well discussed in Section 2.3.3, the idea behind DIP is that using a well-suited untrained neural architecture to generate the solution of a variational inverse problem can act as a regularizer. We simply transposed the idea to data assimilation and ask generator network $g_{\boldsymbol{\theta}}$ to output our estimated initial condition $\widehat{\mathbf{X}}_0$ the solution, starting from a latent parameter z, see Eq. 4.1.

$$g_{\boldsymbol{\theta}}(z) = \widehat{\mathbf{X}}_0 \tag{4.1}$$

The prior then becomes $p(\mathbf{X}_0) = \delta(\mathbf{X}_0 - g_\theta(z))$, where δ is the Dirac measure, and adapting the posterior given in Section 2.2.2 we obtain the Eq. 4.2, where K is a constant.

$$\begin{aligned} -\log p(\mathbf{X} \mid \mathbf{Y}) = \frac{1}{2}\sum_{t=0}^{T} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} - \log K \\ \text{s.t.} \quad g_{\boldsymbol{\theta}}(z) = \mathbf{X}_0 \quad \text{and} \quad \mathbb{M}(\mathbf{X}_t) = \mathbf{X}_{t+1} \end{aligned} \tag{4.2}$$

Cost function

Then the cost function associated with the MAP estimation can be developed as in Eq. 4.3. A schematic view of the performed forward integration is drawn in Figure 4.1.

$$\mathcal{J}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{t=0}^{T} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} = \frac{1}{2}\sum_{t=0}^{T} \|\mathbf{Y}_t - \mathbb{H}_t \circ \mathbb{M}_{0\to t} \circ g_{\boldsymbol{\theta}}(z)\|^2_{R_t} \tag{4.3}$$

[1] `https://github.com/ArFiloche/Variational_Assimilation`

We notice that only the observational error is to be optimized over θ which would correspond to an MLE in the standard 4DVAR setting. From a purely computational perspective, control parameters have been shifted from the system state space to the neural network parameters space, constituting a kind of augmented state. In [29], they grid-search background matrix parameters using the observational error. They prove that this is equivalent to doing an MLE using an augmented state integrating background matrix parameters. This is very similar to what happens in our approach. Also, they made a locality assumption which we can also make implicitly using convolutional layers.

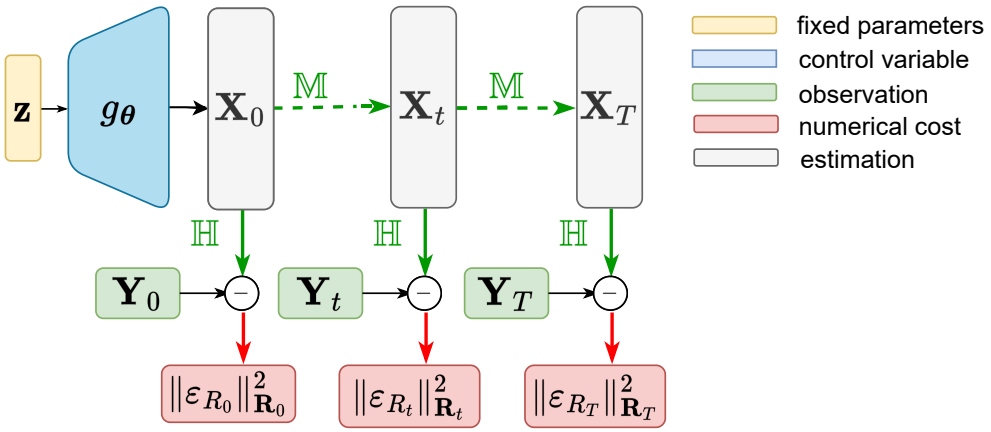


Figure 4.1: Schematic view of the forward integration used in strong-constraint deep background prior 4DVAR

Gradient

The gradient of $\mathcal{J}(\theta)$ can be determined analytically. First, the chain rule gives Eq. 4.4. Then using the adjoint state method we can develop $\nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0)$ as in Eq. 4.5.

$$\nabla_{\boldsymbol{\theta}}\mathcal{J}(\boldsymbol{\theta}) = \nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0)\nabla_{\boldsymbol{\theta}}\mathbf{X}_0 = \nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0)\nabla_{\boldsymbol{\theta}}g_{\boldsymbol{\theta}}(z) \tag{4.4}$$

$$\nabla_{\mathbf{X}_0}\mathcal{J}(\mathbf{X}_0) = \sum_{t=0}^{T}\left[\frac{\partial(\mathbb{H}_t \circ \mathbb{M}_{0\to t})}{\partial \mathbf{X}_0}\right]^\top {\mathbf{R}_t}^{-1}\varepsilon_{R_t} \tag{4.5}$$

Algorithm

Algorithm 4 is really similar to the vanilla 4DVAR already presented. In the next sections, we will not be detailing algorithms except in case of substantial variation.

Algorithm 4 – Deep prior 4DVAR

Initialize: fixed parameter z, control variables $\boldsymbol{\theta}$
while stop criterion **do**
  forward: integrate $\mathbb{M}_{0\to T}(g_{\boldsymbol{\theta}}(z))$ and compute $\mathcal{J}$
  backward: automatic differentiation returns $\nabla_{\boldsymbol{\theta}}\mathcal{J}$
  update: $\boldsymbol{\theta}$ = optimizer$(\boldsymbol{\theta}, \mathcal{J}, \nabla_{\boldsymbol{\theta}}\mathcal{J})$
end while
return $\boldsymbol{\theta}, \mathbf{X}_0$

4.1.2 Experiments

We test our strong-constraint 4DVAR with deep background prior (4DVAR-DIP) in twin experiments involving the dynamical systems introduced in Section 2.4.2 with the exact same framework used in the experiments presented in Section 3.1.4 so that we have comparison points with standard 4DVAR with or without background regularization (4DVAR-B or 4DVAR, respectively). We nevertheless remind that if not precised, the white noise variance is fixed to 1.5% of half the peak-to-peak signal amplitude.

Architecture

The choice of architecture is the handcrafted prior. In this experiment, we use the generative convolutional architecture introduced in [58], which is known as a good baseline to generate images. We modified it a bit to avoid checkerboard artifacts, replacing deconvolution operations by an upsampling operator followed by a convolutional layer, as described in [121]. The relevance of the prior lies in the fact that it has produced well on several image processing experiments. The architecture, procedure, and numerical details are available on GitHub. To optimize the neural network we use the Adam optimizer [113].

Shallow water case study

Results We ran the assimilation experiment over 100 series of observations. In Figure 4.2 we look at the RMSE and the AAE inside the assimilation window. We can observe the regularizing effect of the deep prior on the RMSE, 4DVAR-DIP fit the data and the estimation is not degraded between observational points as for the 4DVAR-B. But looking at the AAE, the estimated motion field appears to be very imprecise, displaying poor performances in comparison with 4DVAR.

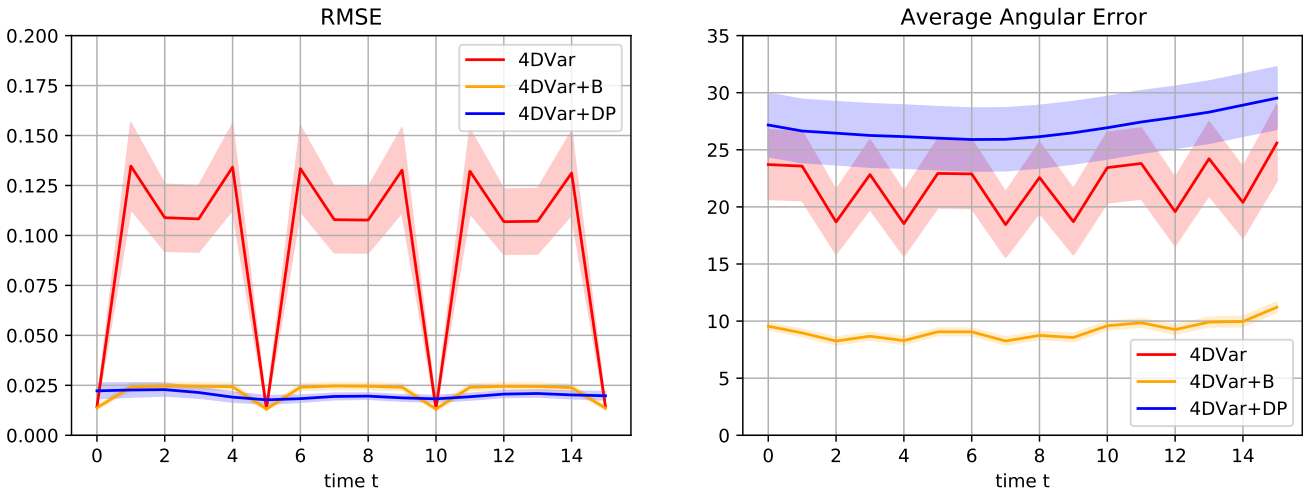


Figure 4.2: Shallow water system: Assimilation score inside the window, averaged on 100 estimations, 1.5% Gaussian white noise, transparent area is bounded by the average score ±0.2 standard deviation

However, if we dig into smoothness statistics of the estimated fields, $\|\nabla \mathbf{w}_0\|_2$, $\|\nabla . \mathbf{w}_0\|_2$ and $\|\Delta \mathbf{w}_0\|_2$, it seems that 4DVAR-DIP is able to capture complex statistics of the true motion field. Similar behavior has been noticed in [63]. The histograms in Figure 4.3 show that 4DVAR-B and 4DVAR-DIP estimation have smoothness statistics very close to that of the ground truth motion field while the 4DVAR estimation is distorted (see Figure 3.10), exhibiting the regularizing effect of the deep prior.

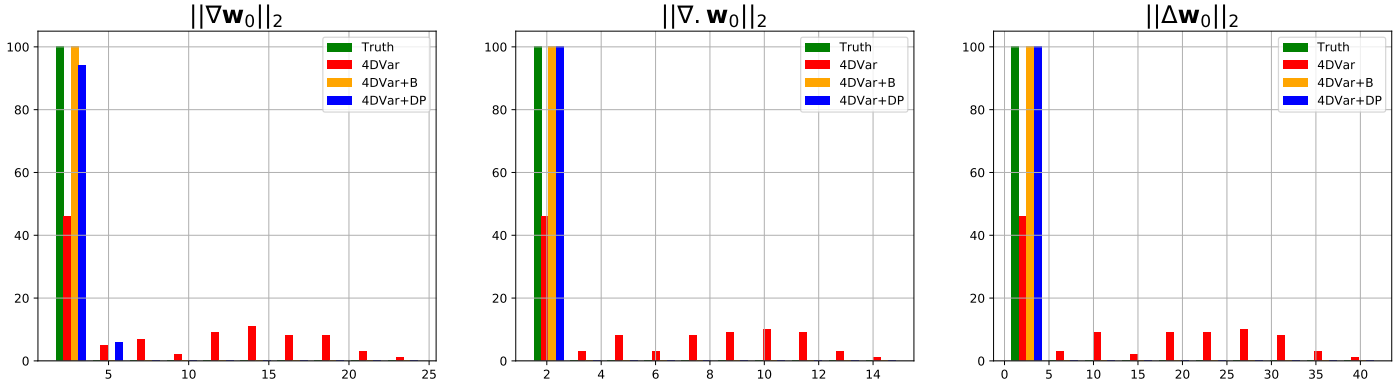


Figure 4.3: Shallow water system: Natural statistics of estimated motion fields from various 4DVAR versions, 1.5% Gaussian white noise

Noise sensitivity We performed the exact same noise sensitivity experiment as in Section 3.1.4. First looking at estimated motion fields (Figure 4.4), we confirm that estimations are smooth but more than that, they seem to conserve these characteristics

even when the level of noise in the observation increases, as in 4DVAR-B (Figure 3.14) but not 4DVAR (Figure 3.10). Then, comparing assimilation results in Figure 4.5, 4DVAR-DIP appears less sensitive to increasing noise and less prone to over-fitting observations than 4DVAR-B (see Figure 3.15) and 4DVAR (see Figure 3.11). This is in line with the original "Deep Image Prior" paper exhibiting that convolutional architecture can not overfit noise easily. Again, we notice the regularizing effect of the deep prior.

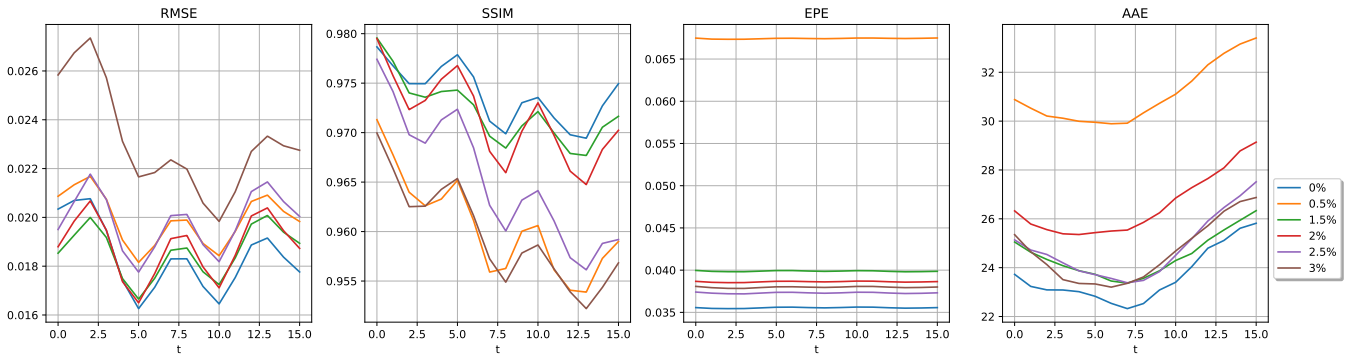


Figure 4.4: Shallow water system: Assimilation performances of 4DVAR with deep background prior, evolution regarding noise level

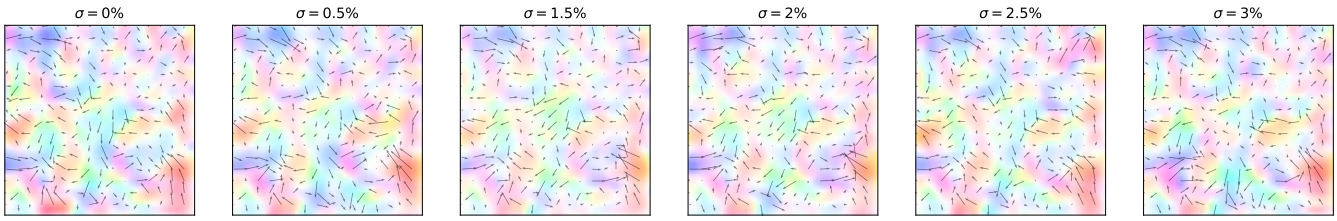


Figure 4.5: Shallow water system: Estimated motion field from 4DVAR with deep background prior, evolution regarding noise level

Advection case study

Results Conclusions drawn for the shallow water system stand for the advection one regarding natural statistics of the estimated motion field (see Figure 4.6). In contrast, we see that 4DVAR-DIP estimation of the motion field is more precise (AAE in Figure 4.7). In this configuration, it is interesting to look at forecast performances (see Figure 4.8) to gauge the balance between fitting the observation and having a good estimation of the motion field. And it seems that the 4DVAR-B estimation has the best balance considering longer-term forecasts. Even if the design background regularization appears to be more accurate, the deep prior still shows interesting properties.

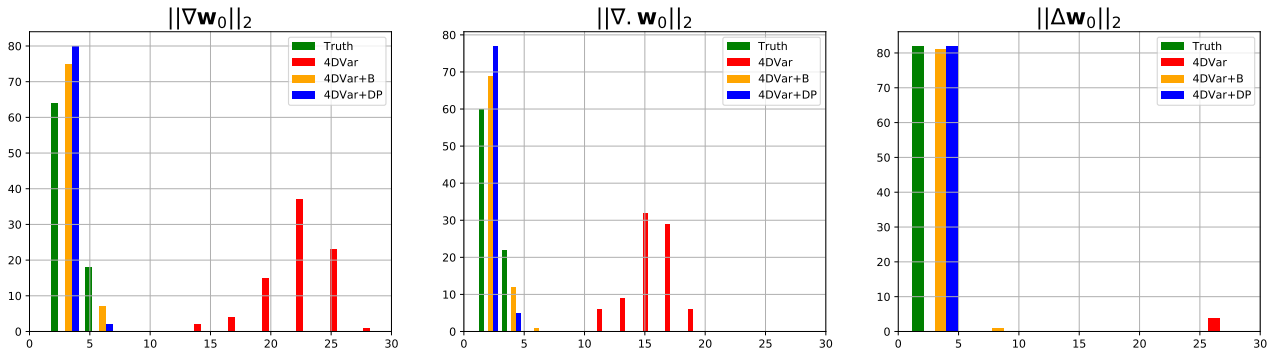


Figure 4.6: Advection system: Natural statistics of estimated motion fields from various 4DVAR versions, 1.5% Gaussian white noise

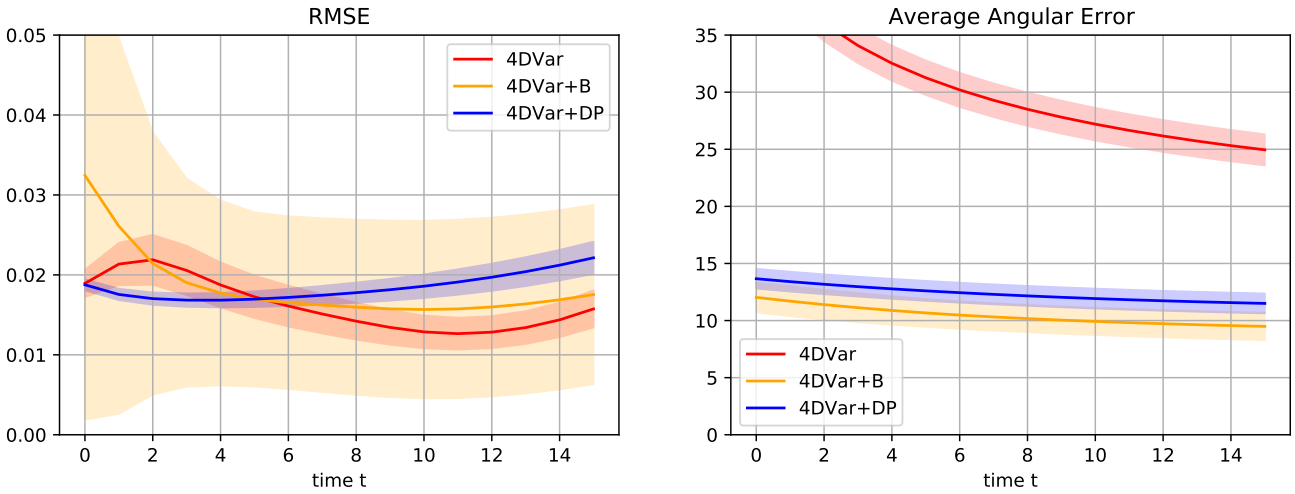


Figure 4.7: Advection system: Assimilation score inside the window, averaged on 100 estimations, 1.5% Gaussian white noise, transparent area is bounded by the average score ±0.2 standard deviation

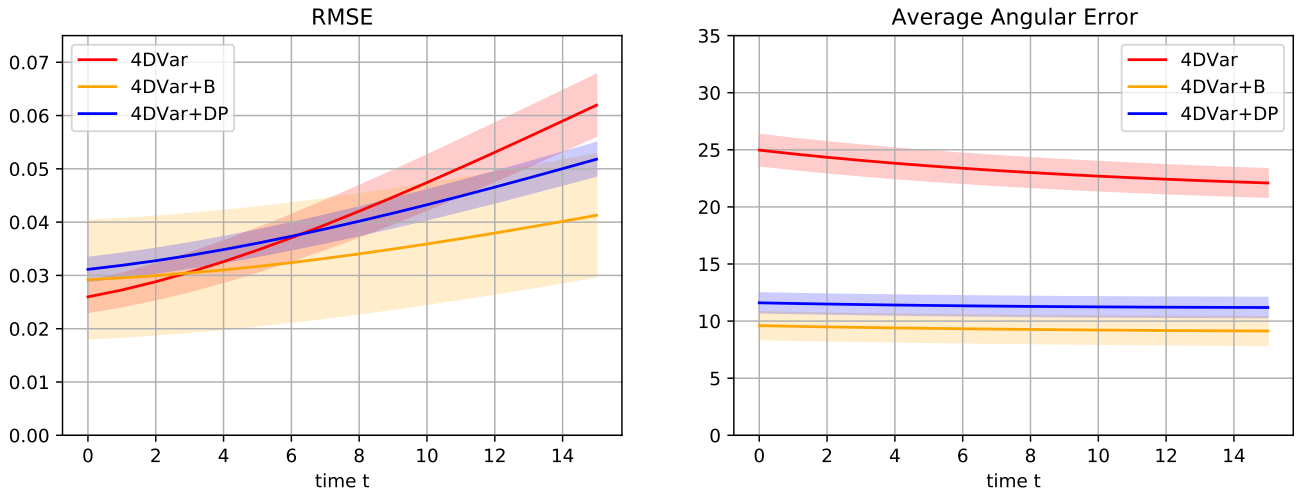


Figure 4.8: Advection system: Forecast scores, averaged on 100 estimations, 1.5% Gaussian white noise, transparent area is bounded by the average score ±0.2 standard deviation

Noise sensitivity The 4DVAR-DIP estimated motion stay smooth when the level of noise in the data augments (see Figure 4.9). Looking at assimilation results (Figure 4.10), 4DVAR-DIP again shows a strong denoising effect. The variability of the assimilation score is even more consistent between different noise levels.

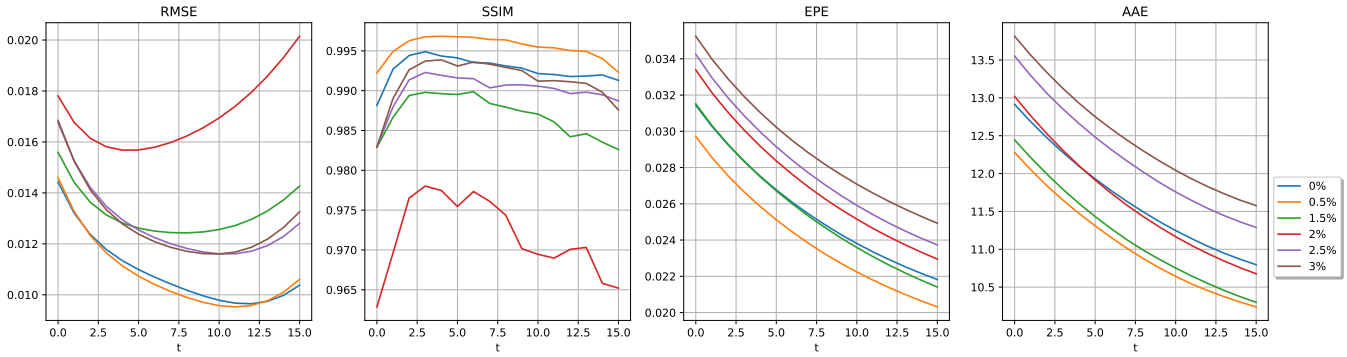


Figure 4.9: Advection system: Assimilation performances of 4DVAR with deep background prior, evolution regarding noise level

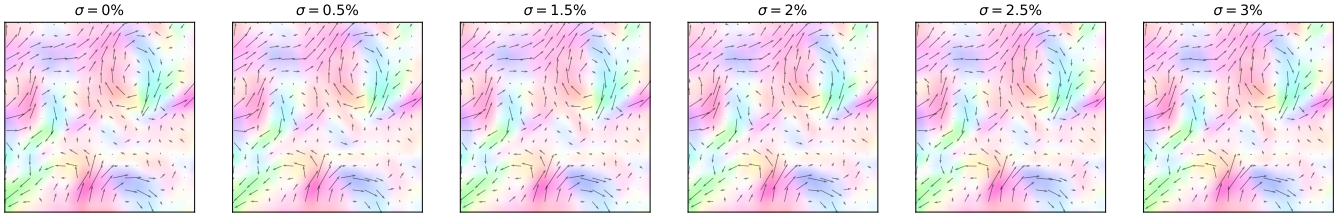


Figure 4.10: Advection system: Estimated motion field from 4DVAR with deep background prior, evolution regarding noise level

Ensemble of deep prior

By training multiple times the same network with different weights initialization we noticed significant differences in performances. It is now well documented that an ensemble of neural networks can produce significantly better results [122]. Even though our approach is untrained, we observed similar behaviors. Also, even using the exact same initialization, a small perturbation in the data can greatly influence the optimization path. This is what happened in the noise sensitivity experiment in Figures 4.4 and 4.9, where the 4DVAR-DIP performances do not exactly decrease when the level of noise augments. With the objective of having a more accurate estimation of such sensitivity, we believe the experiment should be run with multiple noises and multiple weights initialization, which is computationally costly.

Averaging to enhance performances Optimizing multiple deep priors with different weights initialization and averaging their outputs, we obtain better estimation than the best one performed by an individual network as depicted in Figures 4.11 and 4.12. Doing so, an ensemble of 4DVAR-DIP is competitive with 4DVAR-B but to be fair, we should compare it with an ensemble of 4DVAR, for instance, EnsVAr [123]. Moreover, the added computational complexity is not worth it. Details about computational considerations are discussed at the end of the section.

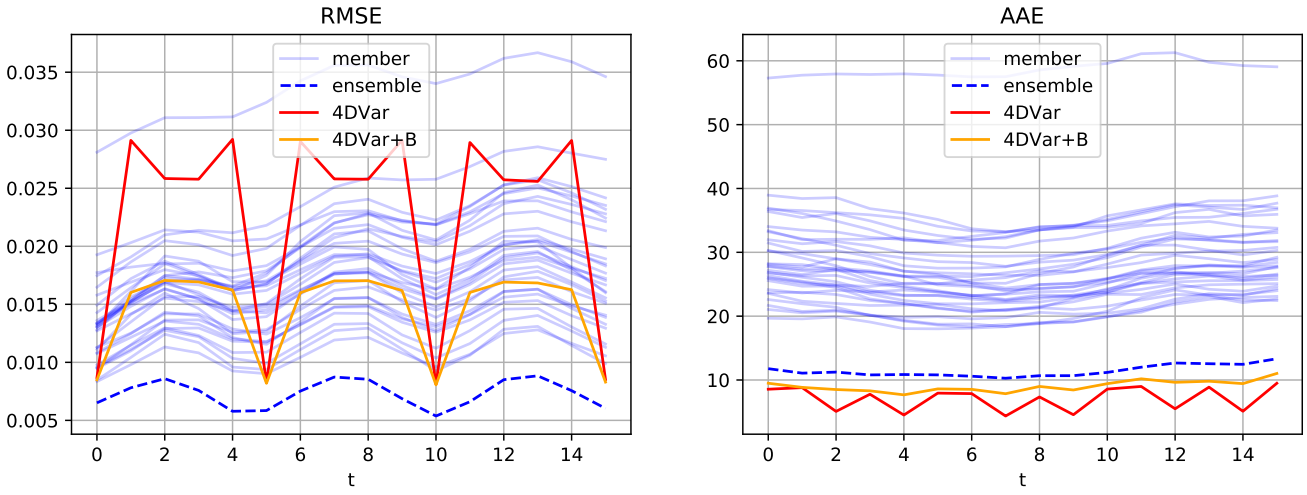


Figure 4.11: Shallow water system: 4DVAR-DIP assimilation score for different weights initialization, 1.5% Gaussian white noise

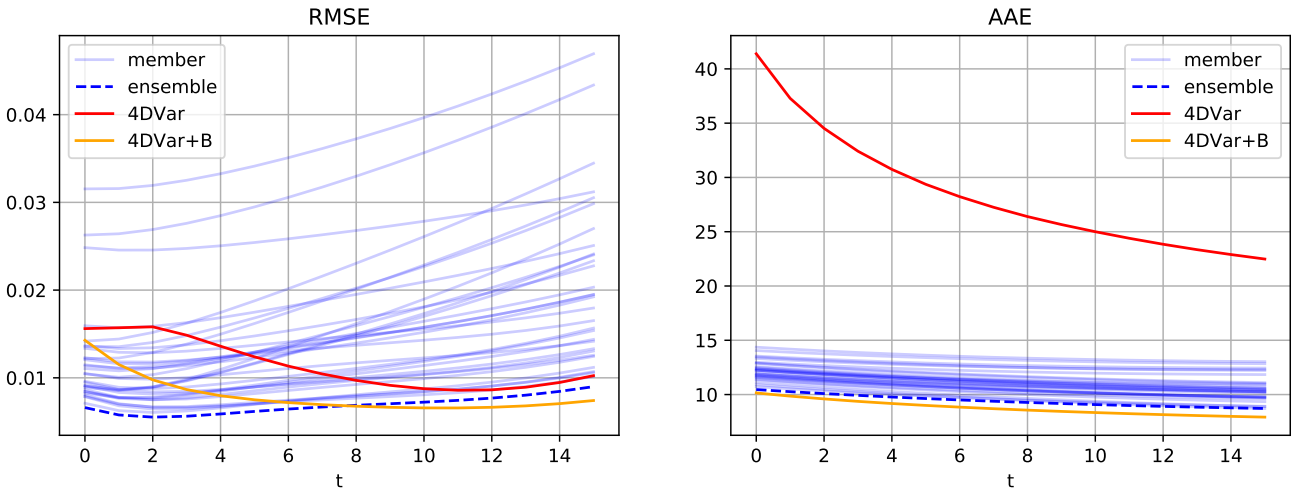


Figure 4.12: Advection system: 4DVAR-DIP assimilation score for different weights initialization, 1.5% Gaussian white noise

Uncertainty quantification ? Having an ensemble at disposal, it is natural to want to use it to characterize uncertainty. Using the result of the same ensemble experiment, we displayed in Figures 4.13 and 4.14 the error maps on I of the estimation and the standard deviation of the ensemble along the assimilation window, for the shallow water and the advection system, respectively. It is blatant for the shallow water case and it is also the case for the advection one, there is no correlation between these two maps so that we don't think the ensemble provides a useful uncertainty quantification. Our interpretation is that the stochasticity in the 4DVAR-DIP initialization introduces some bias in the estimation which can be eliminated by averaging several samples.

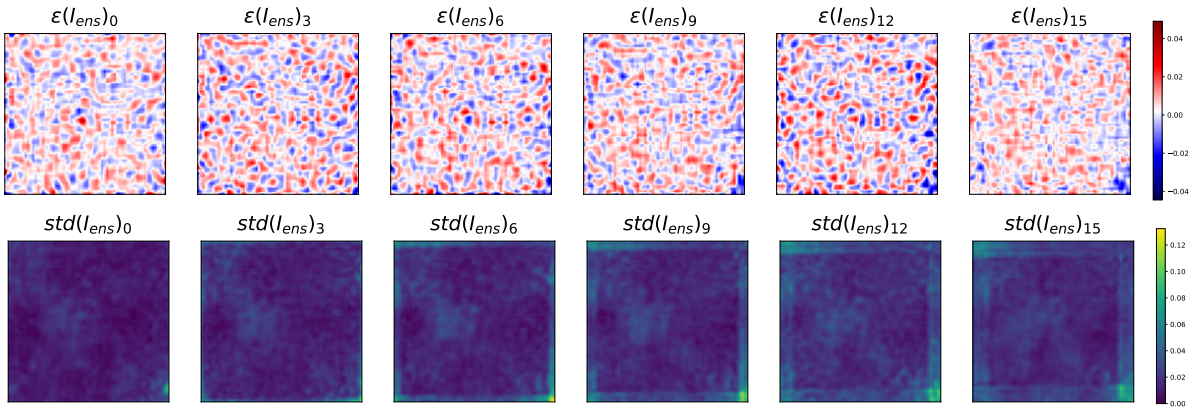


Figure 4.13: Ensemble shallow water DIPs: Error and standard deviation maps of the 4DVAR-DIP ensemble assimilation

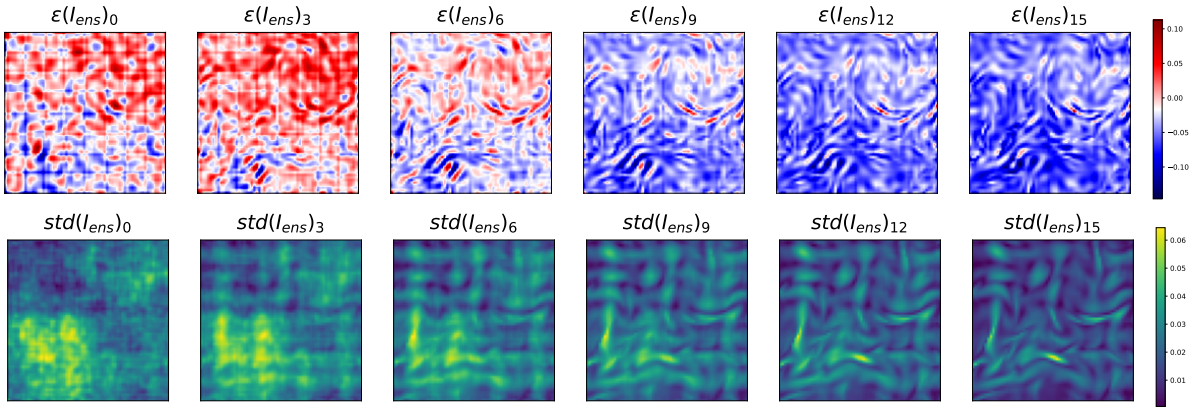


Figure 4.14: Ensemble of advection DIPs: Error and standard deviation maps of the 4DVAR-DIP ensemble assimilation

Adding regularization

Adding regularization to deep prior methods is already investigated, for instance, in [66] they add a total variation for more regularization. Regarding motion estimation using variational assimilation, the background cost enforces smoothness but also promotes small motion values. From a numerical optimization view, such regularization is particularly useful to avoid undesirable behavior of the control parameters. Indeed, the initial motion field is the control here and it is not constrained by observations, only by the background. In our 4DVAR-DIP, only the observational cost is optimized so that nothing promotes small motion values. This is why we decided to try a L_2-regularization on the initial motion field, corresponding to a Gaussian prior and leading to a MAP. We repeated the ensemble experiment with such regularization, results are plotted in Figures 4.15 and 4.16 and show that it improves drastically the accuracy of the assimilation for the shallow water while it degrades it for the advection system.

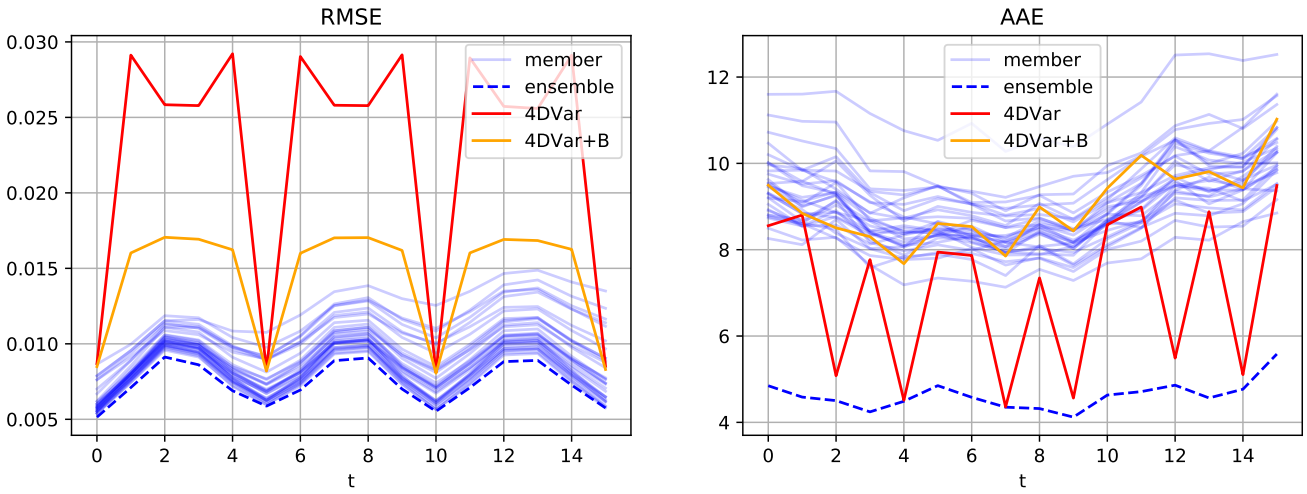


Figure 4.15: Shallow water system: 4DVAR-DIP using L_2-regularization, assimilation score for different weights initialization, 1.5% Gaussian white noise

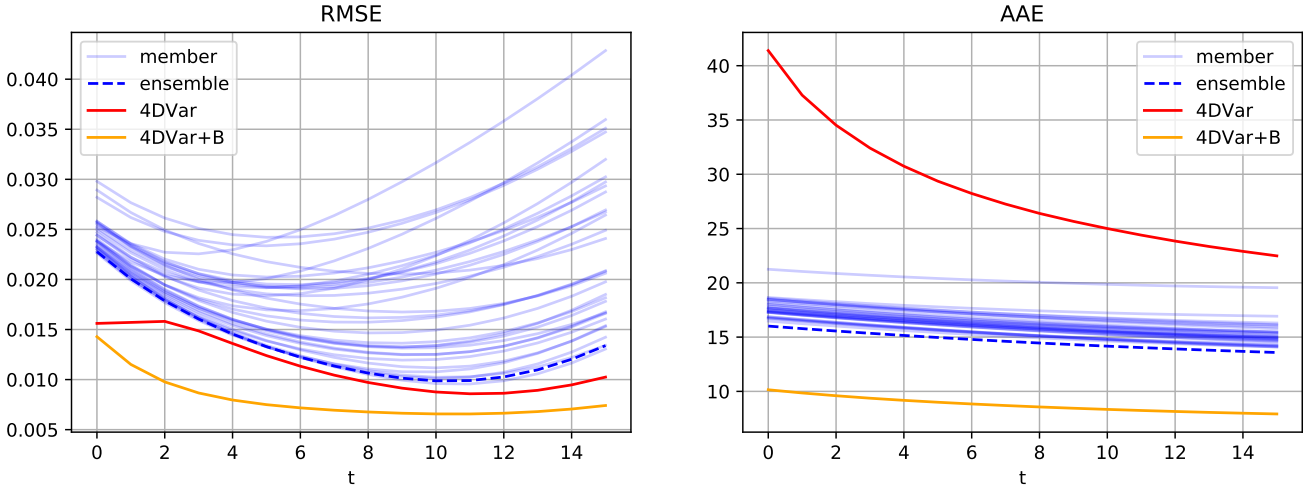


Figure 4.16: Advection system: 4DVAR-DIP using L_2-regularization, assimilation score for different weights initialization, 1.5% Gaussian white noise

The need for an additional regularization opens the door for criticism: why should one use a deep prior if one needs to design an additional prior? An answer we find acceptable is that even if L_2 and background regularization both correspond to Gaussian priors, the former is much simpler in the sense that there is only one parameter to tune. Our take on this is that the deep convolutional prior is able to handle the needed spatial correlation for a regular solution but not letting some control parameters totally free can be helpful.

4.1.3 Critical discussion

Computational complexity

Using a deep prior adds a layer of complexity on top of 4DVAR which is already non-trivial. There is still the issue of having the numerical optimized on GPU for better compatibility that we will not discuss here. During our experiments, we noticed that the numerical scheme integration was the computationally costly part of the forward model. The number of integration made in 4DVAR and 4DVAR-B was limited to 150 while for 4DVAR-DIP 15000 and 2500 iterations were made for the shallow water and advection system, respectively. It corresponds to between one and two orders, of magnitude in terms of model integration, which is totally prohibitive for operational application. And it is not even considering the ensemble of deep priors.

Convergence criterion

The main criticism addressed to the "Deep Image Prior" inspired method is that it needs an early-stopping criterion even though some already exist [65].

Over-fitting Without such criterion, the deep prior may over-fit the observation version leading to a distorted estimation. In Figures 4.18 and 4.17 we monitored the assimilation scores during the optimization, deliberately fixing a large number of iterations for 4DVAR-DIP to highlight this over-fitting effect. Regarding the shallow water system, we clearly see that at some point in the optimization, the deep prior starts to over-fit regarding the estimated motion field. But this criticism can also be addressed to 4DVAR: even the regularized version exhibits an over-fitting behavior. In the advection experiment, this is less blatant but still, 4DVAR-DIP seems to over-fit. It is to be noted that the number of iterations and so integration of the dynamics is very large, so the early-stopping is also computationally beneficial.

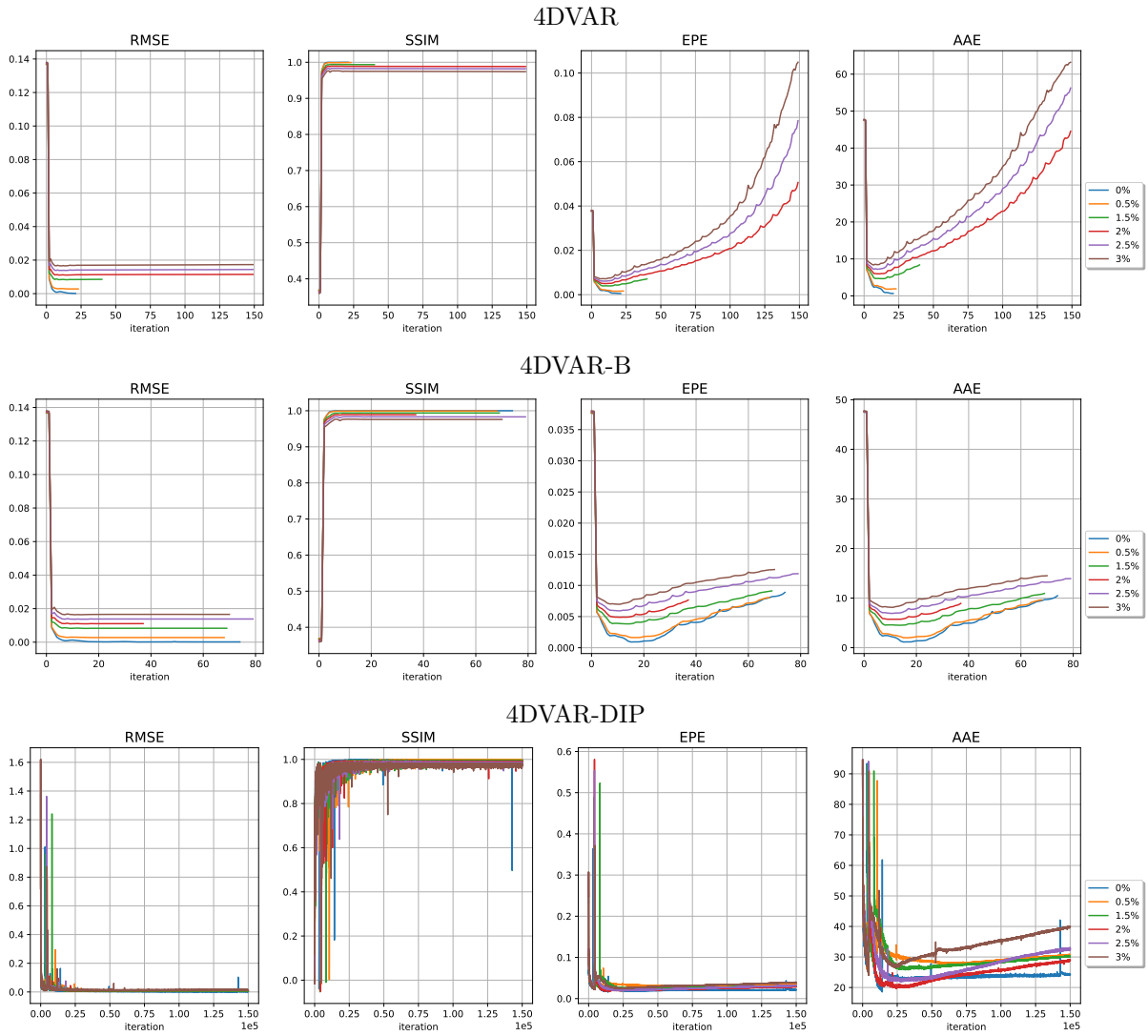


Figure 4.17: Shallow water system: Monitoring of assimilation score during the optimization for 4DVAR, 4DVAR-B, and 4DVAR-DIP, using different levels of noise.

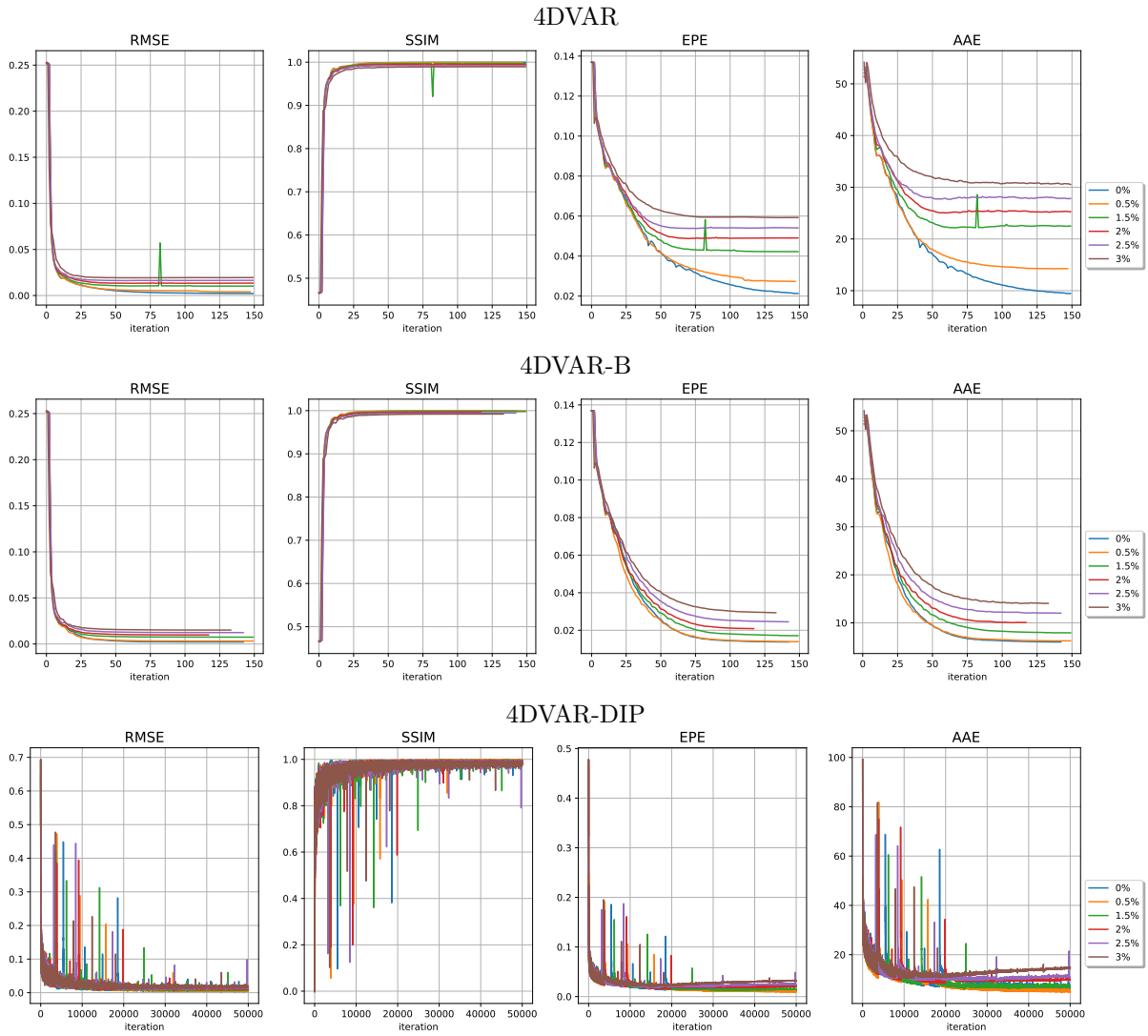


Figure 4.18: Advection system: Monitoring of assimilation score during the optimization for 4DVAR, 4DVAR-B, and 4DVAR-DIP, using different levels of noise.

Hyper-parameters tuning

The most important hyper-parameters to tune in our 4DVAR-DIP are then the learning rate and the number of optimization steps, which are intertwined parameters. As 4DVAR and 4DVAR-DIP are both "unlearned methods", we can use the same procedure. When the ground truth is not available, a good objective metric is the forecast performances on the observed variable, $\boldsymbol{I}$ in our cases. The solution we ended up adopting was to fix the number of optimization steps and finding a complementary learning rate. An interesting work could have been to look for hyper-parameters maintaining a certain accuracy while minimizing the number of integration of the dynamics.

Architectures design

Obviously, the choice of architecture is influential as it constitutes the prior. In the original DIP paper, they show that diverse convolutional architectures reach similar accuracy. However, they also show that the depth of the used neural network plays an important role. For instance, the deeper the network the better the in-painting results. It is to be noted that natural images are present so we obviously cannot transpose these conclusions. In our experiments, we used a deep prior with 5 layers. An interesting sensitivity experiment we could have made would have been to remove these layers one by one and see the influence on the estimation. A similar experiment could be made on the number of filters, looking forward to the simplest architecture possible maintaining accuracy. In [67] authors design the "Deep Decoder" architecture to reach DIP accuracy while being severely under-parameterized, having less weights than the number of pixels in the fitted image.

4.2 Simultaneous Downscaling and Assimilation

The work presented in this section has been the subject of a communication at RFIAP (Congrès Reconnaissance des Formes, Image, Apprentissage et Perception) [124]. The code associated with the conducted experiments can be found at this GitHub address [2].

4.2.1 Downscaling ocean simulation

Increasing the resolution of numerical simulations allows to represent finer physical dynamics, instead of representing it with parametrization [125], and then better explains spatio-temporal variability of eddy fields. But when coupled with various satellite observations in a data assimilation scheme, the interpolation tends to smooth small-scale processes [126] and results in diminishing forecast skills [127]. By simultaneously assimilating and increasing the resolution of observations, we hope to participate in developing methods to soften this issue.

4.2.2 Super-resolution

The single image super-resolution task consists in recovering a particular high resolution image denoted $\mathbf{X}^h$ from a low-resolution observation $\mathbf{X}^l$, modeled as the output of a decimation operator denoted d, such that $\mathbf{X}^l = d(\mathbf{X}^h)$. Estimating super-resolution from an observation can be seen as an optimization problem minimizing an energy function of the general form $\|d(\mathbf{X}^h) - \mathbf{X}^l)\|$. By rewriting the observation operator $\mathbb{H} = d \circ p$ where p depends on the application, we highlight the fact that the variational data assimilation framework is suited for simultaneous system state estimation and super-resolution. However, as d is non-injective, it increases the ill-posedness of the inverse problem. For the sake of simplicity, we won't use the notation $\mathbf{X}^h$ so that when $\mathbf{X}$, it is implicitly at high-resolution.

4.2.3 Shallow water twin experiment

The proposed methodology is tested within a twin experiment where data are generated from a numerical dynamical model. Observations are then created by sub-sampling, decimating (down-sampling), and adding noise. The aim of this experiment is to highlight the regularizing effect of DIP in an excessively ill-posed, ocean-like data assimilation problem. To do so, we compare several super-resolution 4DVAR algorithms, one using a deep image prior. We repeat the same experiments with different downscaling ratios and different levels of noise. In all the assimilation experiments, the dynamical model $\mathbb{M}$ and the decimation operator d are assumed perfectly known. The objective is to estimate

[2] `https://github.com/ArFiloche/RFIAP_simulataneous_assimilation_downscaling`

the high-resolution true system state $\mathbf{X}$ from low-resolution observation of ocean surface height deviation η.

Experiments details

Observations At regularly-spaced observational dates, a low resolution of η is observed, up to an additive white noise. The velocity field $\mathbf{w}$ is never observed. The chosen decimation operator d is a 2-dimensional average pooling convolution of kernel size $r \times r$, r being the upscaling factor. This means that at observational date t, the observation operator $\mathbb{H}$ is the composition of a linear projector and d so that $\mathbb{H}\mathbf{X}_t = d(\eta_t)$. White noise is then added. Examples of generated observations are displayed in Figure 4.19. The considered temporal window has a fixed length of $10 \times dt$ so that $T = 9$. Observations are sampled at date $t = 0, 3, 6, 9$. Several upscaling factors, $r = 1, 2, 4, 8$, are investigated, $r = 4, 8$ roughly corresponding to the factor between high-resolution numerical simulations and sea surface height satellite resolution [128]. Several levels of white noise are investigated. The standard deviation of the noise is expressed as a percentage of the low-resolution observation dynamic range. We investigate 0%, 1%, 2%, and 3% which is the level of noise to be expected in the most recent altimeter [90]. For each upscaling factor and each level of noise, 100 series of observations are generated.

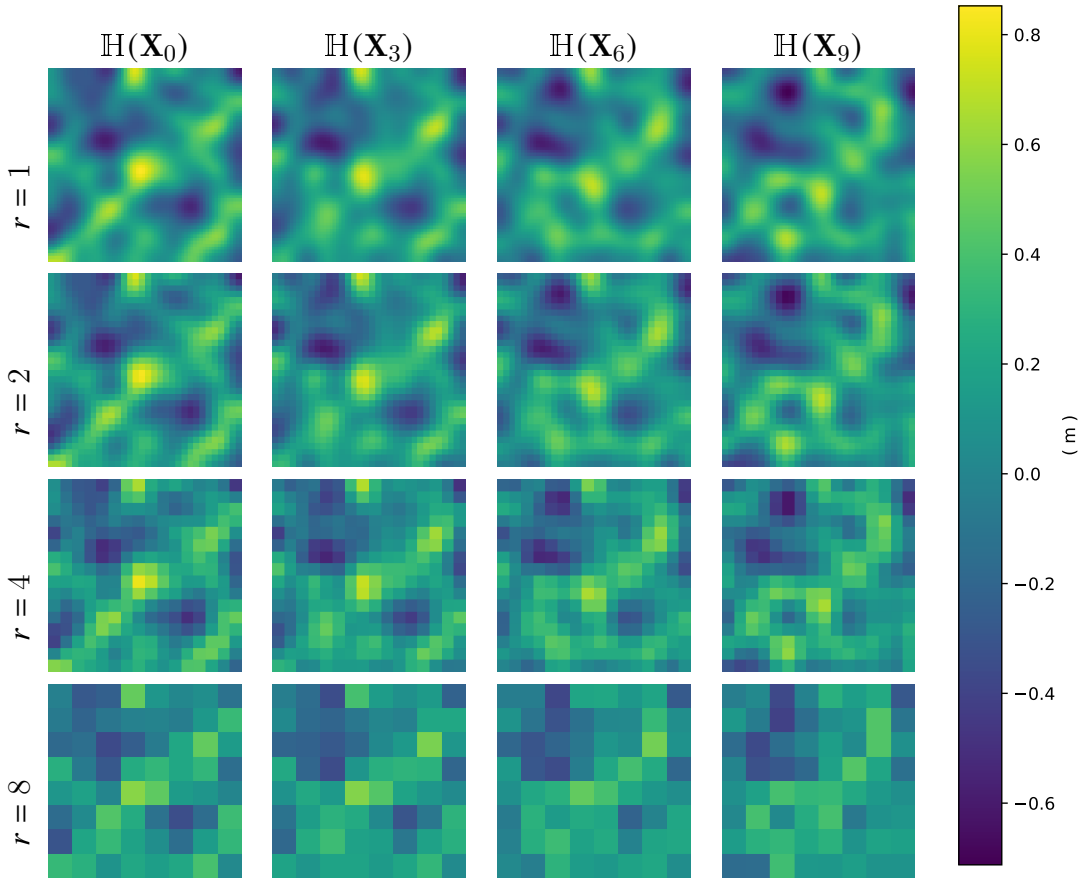


Figure 4.19: Examples of one noiseless generated observation series, using different downsampling ratio r

Compared algorithms

We compared the already presented 4DVAR, 4DVAR-B and 4DVAR-DIP algorithms.

Hyper-parameters tuning

For each downscaling factor, hyper-parameters are tuned using Bayesian optimization on low-resolution observations forecasts so that ground truth is never used. The noise level is fixed at 1%. Regarding DIP, we made the choice to fix the number of epochs to 1000 and only having the learning rate as a hyper-parameter and experimentally found similar optimal learning rates for each downscaling factor.

4.2.4 Results

For each downscaling factor and each level of noise, we optimized the three different versions of 4DVAR on 100 series of observations. As the dynamical model is considered perfect, the initial condition entirely characterizes the estimated state so that $\widehat{\eta}_0$ and $\widehat{\mathbf{w}}_0$ are the quantities to look at to assess the quality of the estimation. Once such a

condition is estimated, it is possible to produce a forecast integrating the dynamics, we will then use $\widehat{\eta}_{T+1}$ and $\widehat{\eta}_{T+5}$ to evaluate forecasts performances. All these estimations are in high-resolution and are then compared quantitatively and visually to the ground truth.

Quantitative results

Quantitative results of the main experiment are displayed in Table 4.1. Such a table is available for each level of noise. We decided to display the one for 1% level of noise as it is the closest to the satellite noise range and because we tuned hyper-parameters at this level. The first interesting result to note is that when no downscaling task is performed ($r = 1$), classical 4DVAR algorithms perform better, which makes sense as the optimized cost function has been designed assuming Gaussian errors. However, augmenting r, the 4DVAR-DIP algorithm seems to be performing better, at least in forecast performances. Regarding metrics quantifying the quality of the motion field estimation, 4DVAR-DIP systematically outperforms 4DVAR and more particularly 4DVAR-B.

Table 4.1: Metrics quantifying the quality of the assimilation and the following forecast for various downscaling factors and a fixed level of noise of 1%

	Assimilation				Forecast			
Quantity	$\widehat{\eta}_0$		$\widehat{w}_0$		$\widehat{\eta}_{T+1}$		$\widehat{\eta}_{T+5}$	
Metric	RMSE ($\times 10^2$)	SSIM	EPE ($\times 10^2$)	AAE	RMSE ($\times 10^2$)	SSIM	RMSE ($\times 10^2$)	SSIM
				$r = 1$				
4DVAR	1.4 ± 0.2	$\mathbf{0.98 \pm 0.01}$	$\mathbf{1.1 \pm 0.2}$	7 ± 1	3.9 ± 0.8	0.87 ± 0.05	3.8 ± 0.7	0.89 ± 0.04
4DVAR-B	$\mathbf{1.3 \pm 0.2}$	$\mathbf{0.98 \pm 0.01}$	1.1 ± 0.3	$\mathbf{6 \pm 1}$	$\mathbf{1.8 \pm 0.3}$	$\mathbf{0.96 \pm 0.01}$	$\mathbf{1.8 \pm 0.3}$	$\mathbf{0.97 \pm 0.01}$
4DVAR-DIP	2.8 ± 0.9	0.94 ± 0.04	5.6 ± 1.3	31 ± 5	4.2 ± 1.3	0.92 ± 0.04	4.8 ± 1.5	0.91 ± 0.04
				$r = 2$				
4DVAR	7.2 ± 2.7	0.74 ± 0.12	5.4 ± 1.8	29 ± 6	16.7 ± 5.7	0.40 ± 0.07	17.7 ± 5.6	0.39 ± 0.12
4DVAR-B	7.0 ± 5.8	0.77 ± 0.19	$\mathbf{1.9 \pm 0.6}$	$\mathbf{11 \pm 2}$	6.6 ± 4.7	0.76 ± 0.18	6.9 ± 4.5	0.76 ± 0.16
4DVAR-DIP	$\mathbf{2.9 \pm 0.9}$	$\mathbf{0.93 \pm 0.05}$	5.5 ± 1.0	30 ± 5	$\mathbf{4.4 \pm 1.4}$	$\mathbf{0.92 \pm 0.05}$	$\mathbf{5.0 \pm 1.6}$	$\mathbf{0.90 \pm 0.05}$
				$r = 4$				
4DVAR	$\mathbf{2.4 \pm 0.3}$	$\mathbf{0.95 \pm 0.01}$	2.1 ± 0.4	13 ± 2	6.0 ± 1.1	0.76 ± 0.07	6.1 ± 1.1	0.77 ± 0.05
4DVAR-B	4.7 ± 1.3	0.85 ± 0.06	$\mathbf{1.6 \pm 0.3}$	$\mathbf{9 \pm 1}$	$\mathbf{4.5 \pm 0.9}$	0.85 ± 0.05	$\mathbf{4.6 \pm 1.0}$	0.85 ± 0.04
4DVAR-DIP	3.3 ± 0.8	0.92 ± 0.04	6.1 ± 1.3	33 ± 6	5.3 ± 1.3	$\mathbf{0.89 \pm 0.04}$	5.7 ± 1.6	$\mathbf{0.88 \pm 0.04}$
				$r = 8$				
4DVAR	8.1 ± 1.6	0.71 ± 0.11	$\mathbf{3.9 \pm 0.3}$	$\mathbf{25 \pm 5}$	11.8 ± 0.9	0.51 ± 0.06	11.9 ± 1.0	0.51 ± 0.06
4DVAR-B	8.1 ± 1.6	0.71 ± 0.11	$\mathbf{3.9 \pm 0.3}$	$\mathbf{25 \pm 5}$	11.7 ± 1.0	0.51 ± 0.06	11.8 ± 1.0	0.51 ± 0.07
4DVAR-DIP	$\mathbf{7.9 \pm 1.7}$	$\mathbf{0.74 \pm 0.12}$	9.7 ± 3.0	47 ± 9	$\mathbf{11.2 \pm 1.3}$	$\mathbf{0.66 \pm 0.08}$	$\mathbf{10.9 \pm 1.4}$	$\mathbf{0.67 \pm 0.09}$

Qualitative results

To better understand error sources, we displayed in Figures 4.20, 4.21 and 4.22 error maps for $\widehat{\eta}_0$, $\widehat{\mathbf{w}}_0$ and $\widehat{\eta}_{T+5}$. First looking at assimilation results, we see that square patterns tend to appear in estimated states with 4DVAR and 4DVAR-R while 4DVAR-DIP estimation stays smooth. The harder the downscaling task the bigger the squares. When combined, square degradation propagates through the dynamical model which explains the better forecasts performances of 4DVAR-DIP. However, 4DVAR-DIP seems less precise and more intense and this is confirmed in Figure 4.23. In an experiment with no noise, we see that the used deep prior can't perfectly fit observations, contrary to the classical prior.

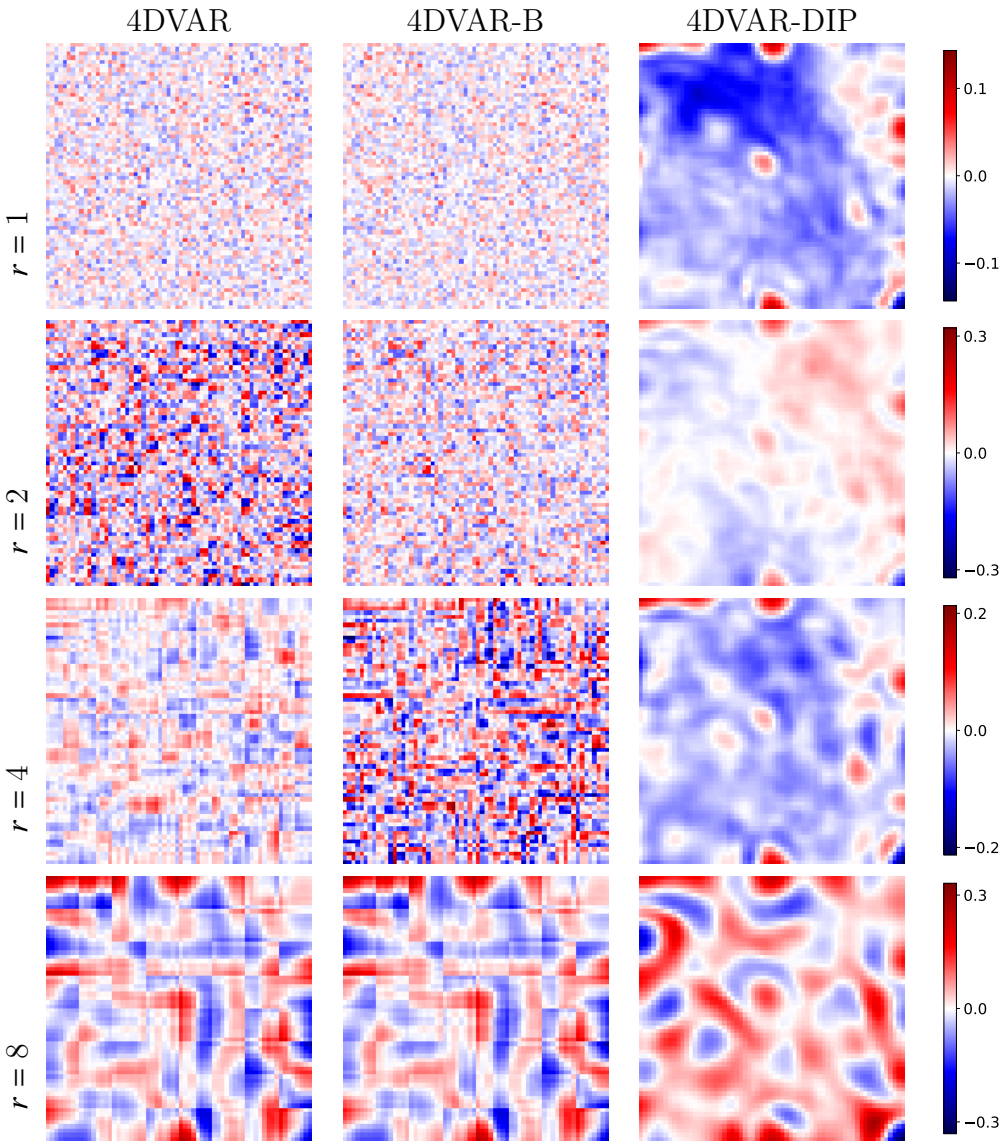


Figure 4.20: Assimilated height $\widehat{\eta}_0$ error maps for each downscaling factor and 4DVar algorithms, noise level of 1%.

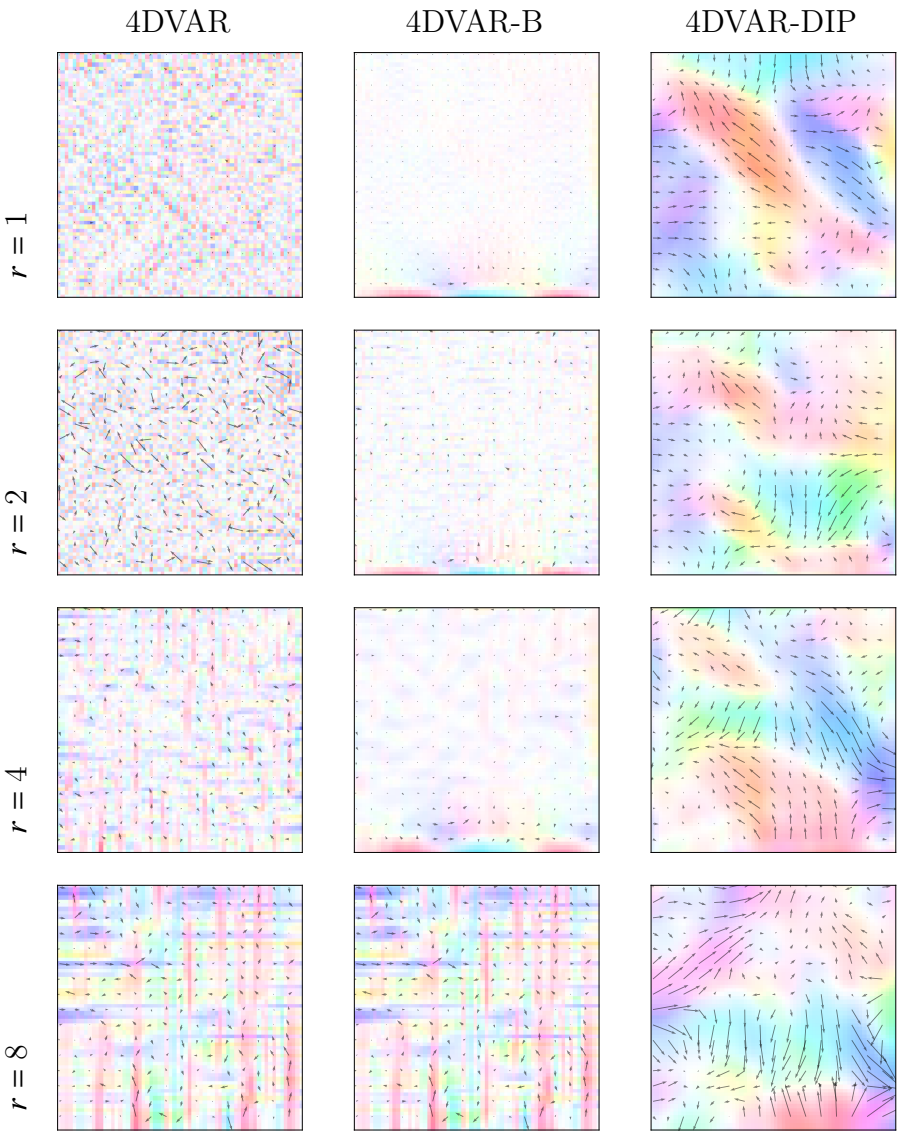


Figure 4.21: Assimilated $\widehat{\mathbf{w}}_0$ error maps for each downscaling factors and 4DVAR algorithms, noise level of 1%. Color corresponds to the motion field orientation but the intensity is normalized, quiver arrows quantify the intensity of the motion field.

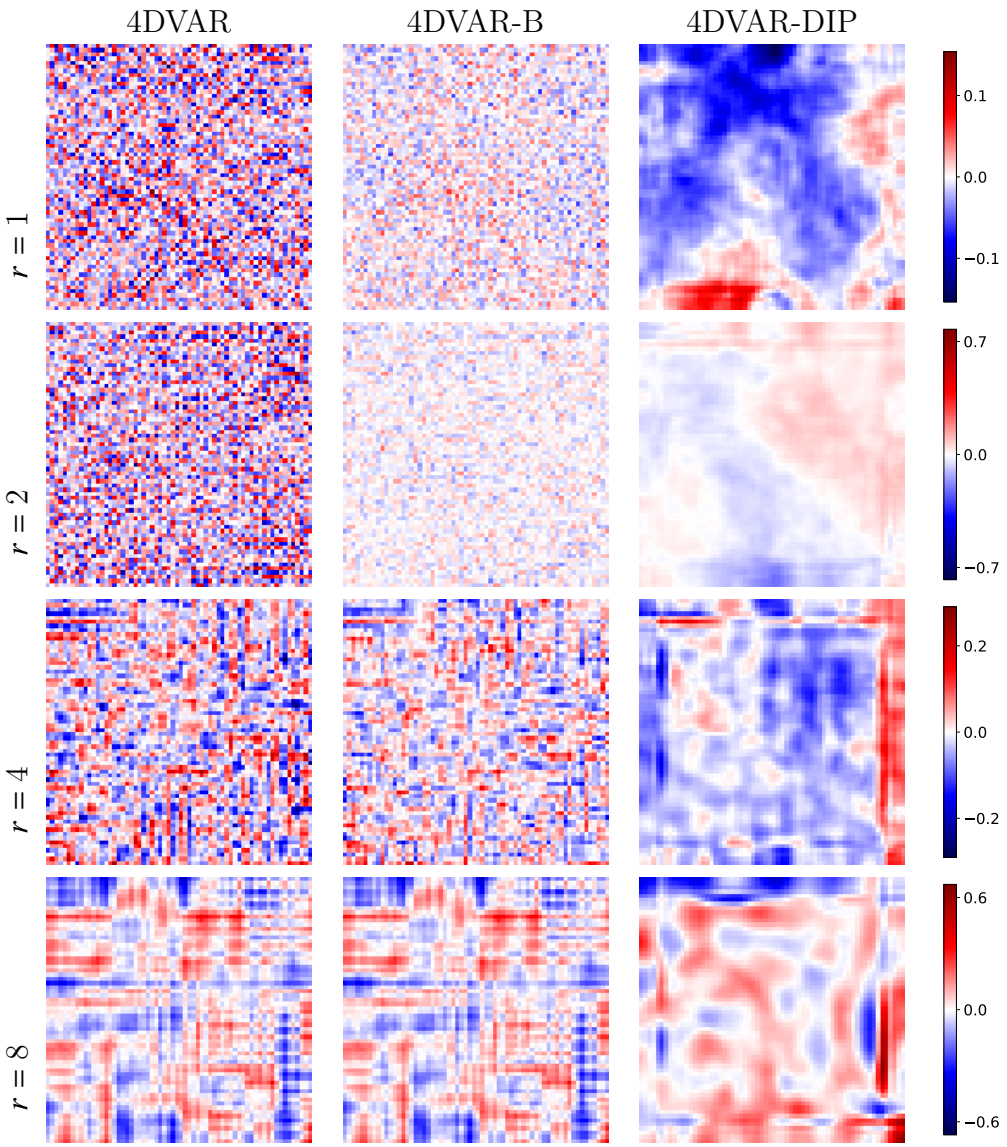


Figure 4.22: Forecasted $\widehat{\eta}_{T+5}$ error maps for each downscaling factors and 4DVAR algorithms, noise level of 1%.

Sensitivity to noise

We repeated similar experiments with different levels of noise and noticed in Figure 4.23 that the used deep prior is robust against noise increases. It is to be reminded that hyper-parameters have been tuned at a single level of noise so they may be less relevant in other cases. We believe that the inductive bias from the architecture choice brings a desirable regularity to our problem. For instance, the qualitative results show that the estimated motion fields have interesting characteristics that are not translated in the chosen optical flow metrics. So we looked at $\|\nabla \mathbf{w}_0\|_2$, $\|\nabla . \mathbf{w}_0\|_2$ and $\|\Delta \mathbf{w}_0\|_2$, statistics characterizing smoothness and observed in Figure 4.24 that are naturally close to the desired ones and not very sensitive to noise increase.

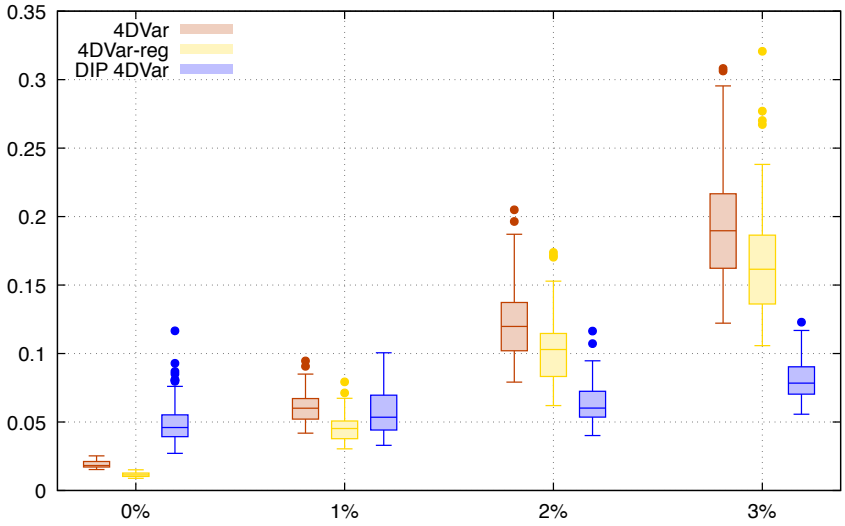


Figure 4.23: Evolution of RMSE of $\widehat{\eta}_{T+5}$ forecasts regarding the level of noise in the observation, with $r = 4$, for each 4DVAR algorithm.

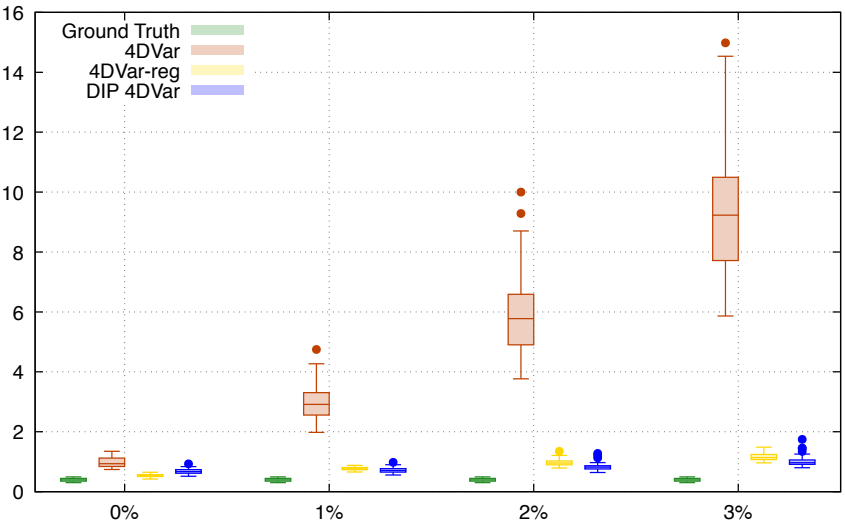


Figure 4.24: Evolution of $\|\Delta\widehat{\mathbf{w}}_0\|$ regarding the level of noise in the observation, with $r = 4$, for each 4DVAR algorithm and for the ground truth.

Ensemble of DIP

As discussed in Section 4.1.2, an ensemble of DIP removes bias in the estimation. In Figure 4.25 we focus on the ×4 super-resolution assimilation of one series of observations images. We then optimize 50 Deep Image Prior, and note that scores of averaged estimation are superior to the average score. We arbitrarily displayed RMSE and AAE

assimilation scores but the same behaviors are obtained with all the metrics. A similar plot with various downscaling factor is presented in Figure 4.26 and show that the enhanced performances appear every time for every measured metric. Looking at errors for different members in Figure 4.26, we see that they tend to compensate. We again conclude that an ensemble of deep image priors enhances performances.

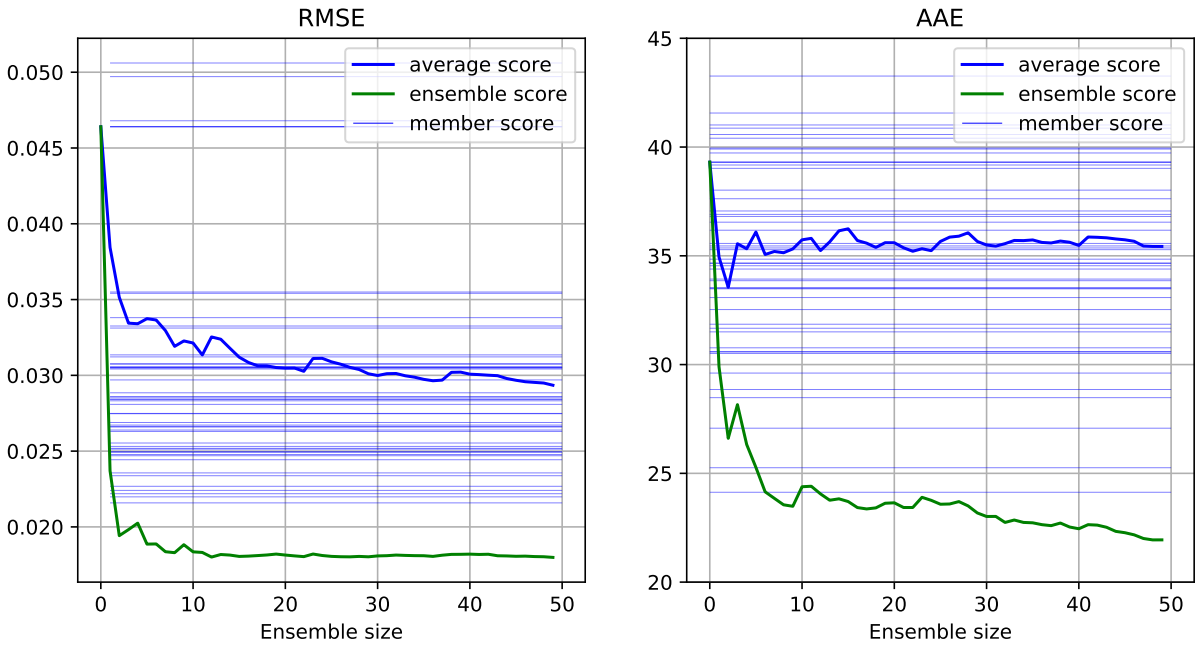


Figure 4.25: Evolution of deep prior ensembles RMSE and AAE scores, realized on one series of observations; downscaling factor $r = 4$, noise level 1%.

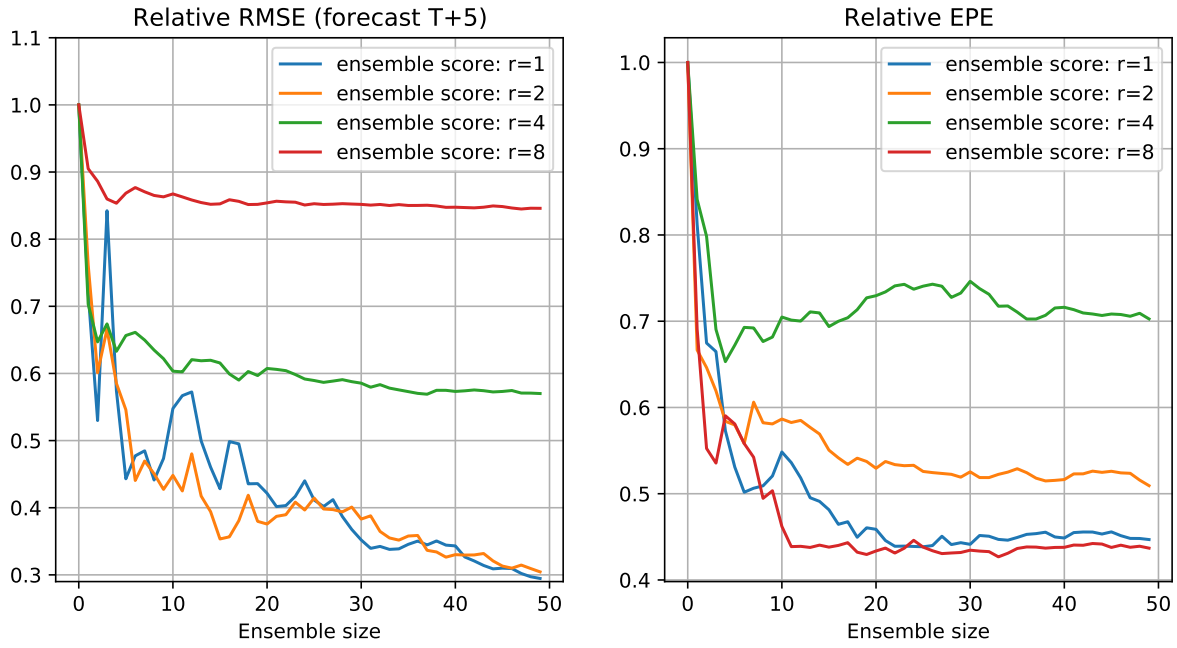


Figure 4.26: Evolution of deep prior ensembles RMSE (forecast) and EPE scores, realized on one series of observations, downscaling factor $r = 1, 2, 4, 8$, noise level 1%, scores presented here are relative so that the whole curve is normalized by the performance of the first member.

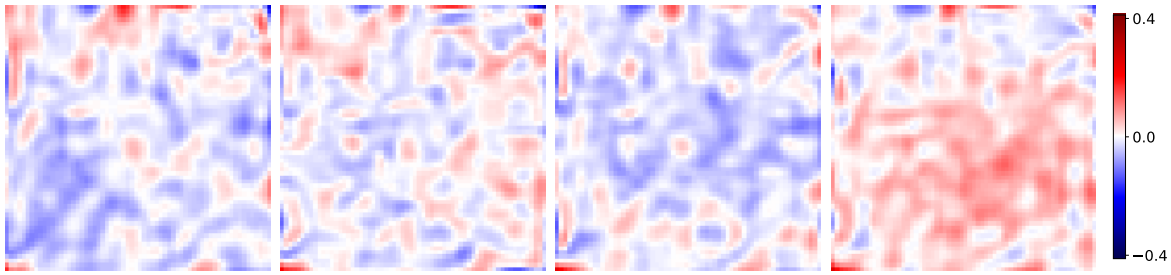


Figure 4.27: Evolution of deep prior ensembles RMSE (forecast) and AAE scores, realized on one series of observations; downscaling factor $r = 1, 2, 4, 8$, noise level 1%.

4.3 3DVAR with deep spatio-temporal prior

The work presented in this section has been the subject of a communication at ECML/PKDD Workshop on Machine Learning for Earth Observation and Prediction (MACLEAN) [129]. The code associated with the conducted experiments can be found at this GitHub address [3].

4.3.1 Geoscientific motivations

Monitoring and modeling the ocean is a constant scientific concern whether for global climate understanding or numerical weather prediction. To do so, information from various sensors is processed in order to estimate the state of the ocean. Surface circulation is usually a variable of great interest as it explains the transport of numerous quantities. It can partially be derived from sea surface heights which are observed thanks to altimeter satellites [130, 131]. However, such data are very sparse in space and time so that interpolating them leads to challenging inverse problems. Even though classical least square methods relying on second-order statistics data [48, 47] have a strong operational record and are still getting better thanks to the growing number of available observations [128], deep learning techniques have revolutionized inverse problems solving [59]. But in the Earth science context, ground truth is not available, so that a supervised learning setup is not realistic. In this work, we investigated the deep prior method [33], optimizing a neural architecture on only one spatio-temporal observation of sea surface heights. We show that the designed deep prior provides efficient regularization.

4.3.2 Optimal interpolation of sea surface height

Observing System Simulation Experiment

The used dataset and the simulation experiment framework have been introduced in [132] and we use the pre-processing of [108]. The interest here is to estimate the full space-time trajectory of the sea surface height (SSH) variable. The considered ground truth is the result of NATL60 high-resolution ocean simulation [133] re-scaled at $(1/20)°$. We denote the 3D-volume of dimensions (T, n_x, n_y) representing a ground truth space-time trajectory $\mathbf{X}$, an example is displayed in Figure 4.28.

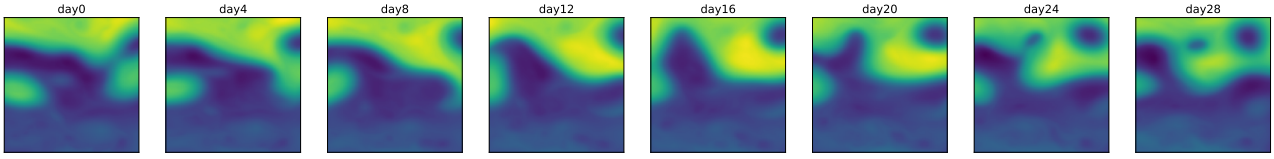


Figure 4.28: Example of reference sea surface height trajectory

[3] `https://github.com/ArFiloche/MACLEAN_deep_spatiotemporal_prior`

The observation operator used to create the dataset aims at simulating two satellite sources. The first is a constellation of 4 nadir altimeters [130] with small spatio-temporal coverage. The second is from a wide-swath altimeter, replicating the Surface Water and Ocean Topography (SWOT) upcoming mission, and made possible thanks to the observation simulator introduced in [131]. Observation $\mathbf{Y}$ denoted available at regular time-steps, per daily interval and obeys the following observation equation $\mathbf{Y} = \mathbb{H}\mathbf{X} + \boldsymbol{\varepsilon}$, where $\mathbb{H}$ is a linear projector associated with satellite tracks and $\boldsymbol{\varepsilon}$ a measurement noise. An example is displayed in Figure 4.29, pointing to a significant sparsity in space.

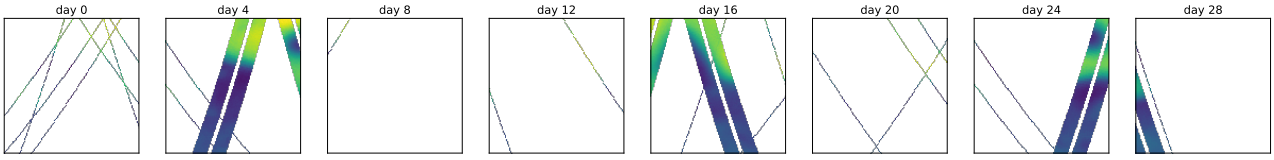


Figure 4.29: Example of sea surface height satellite observation along a trajectory

DUACS analysis

The Data Unification and Altimeter Combination System (DUACS) [128] analysis is a result of a best linear unbiased estimation (BLUE) introduced in Section 2.2.3. This estimation relies on the knowledge of second-order statistics, covariance matrices of state, and noise that we denote $\mathbf{B}$ and $\mathbf{R}$, respectively. To estimate $\mathbf{B}$, DUACS leverages 25 years of reprocessed sea level altimetry so that this estimation is a strong baseline. The produced estimation $\widehat{\mathbf{X}}_{blue} = \mathbf{B}\mathbb{H}^T(\mathbb{H}\mathbf{B}\mathbb{H}^T + \mathbf{R})^{-1}$ can be achieved equivalently in a variational manner [49], minimizing the energy function detailed in Eq. 4.6 and condensed in Eq. 4.7.

$$\mathcal{J}(\mathbf{X}) = (\mathbf{Y} - \mathbb{H}\mathbf{X})^T\mathbf{R}^{-1}(\mathbf{Y} - \mathbb{H}\mathbf{X}) + \mathbf{X}^T\mathbf{B}^{-1}\mathbf{X} \tag{4.6}$$

$$= \|\mathbf{Y} - \mathbb{H}\mathbf{X}\|^2_{\mathbf{R}} + \|\mathbf{X}\|^2_{\mathbf{B}} \tag{4.7}$$

Deep spatio-temporal prior

Similarly, we use a well-suited neural network to generate the estimation, acting as a handcrafted regularization, leveraging bias induced by the convolutional architecture. This means that the control parameters are shifted from the system state space to the neural network parameters space. From a practical standpoint, a generator network $g_{\boldsymbol{\theta}}$ outputs the solution from a latent state z such that $g_{\boldsymbol{\theta}}(z) = \widehat{\mathbf{X}}$. In our case, we ask the network to output the spatio-temporal system state trajectory on a specified window, then we optimize it using the observational cost $\mathcal{J}(\theta)$ given in Eq. 4.8.

$$\mathcal{J}(\boldsymbol{\theta}) = \|\mathbf{Y} - \mathbb{H} \circ g_{\boldsymbol{\theta}}(z)\|^2_{\mathbf{R}} \tag{4.8}$$

Architecture The global design of the used deep prior is largely inspired by generative convolutional architectures introduced in [134]. To avoid checkerboard artifacts, we replaced deconvolution operations as described in [121]. Finally, to ensure spatio-temporal coherence of the generated solution, we used (2+1)D convolution [135], which is an alternative to 3D convolutions being less expensive computationally. A schematic view of the architecture is provided in Figure 4.30.

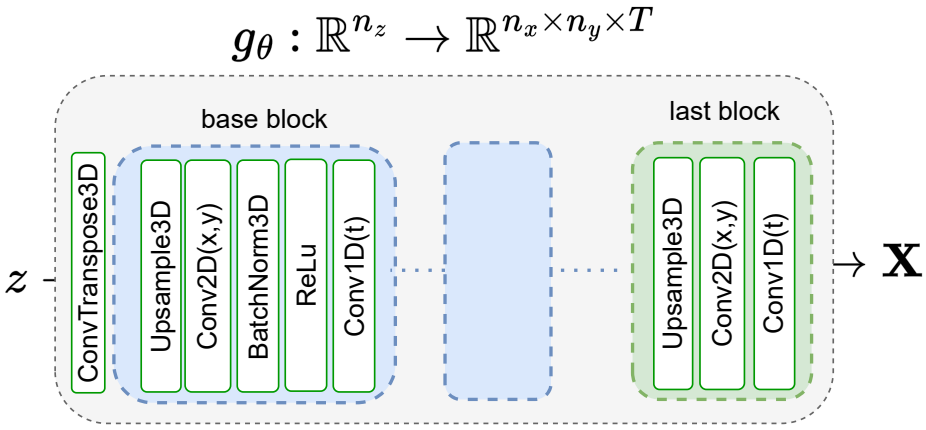


Figure 4.30: Schematic view of the deep spatio-temporal prior architecture.

4.3.3 Experimental results

Observational window

We tested the method on 32-day windows with 128×128 sized observation. Referring to Section 4.1.2 about ensembles of deep priors, we observed similar behavior training multiple networks with different initialization and obtained better results averaging the ensemble. In Figure 4.31, estimation from DUACS and deep priors are compared using the root mean square errors (RMSE) metric. We observe that the ensemble is indeed beneficial and performs slightly better than DUACS interpolation. We also notice border effects, such that deep prior estimation deteriorates at the beginning and at the end of the temporal window. Logically, the DUACS optimal interpolation does not suffer from border effects as considered estimation were window-centered.

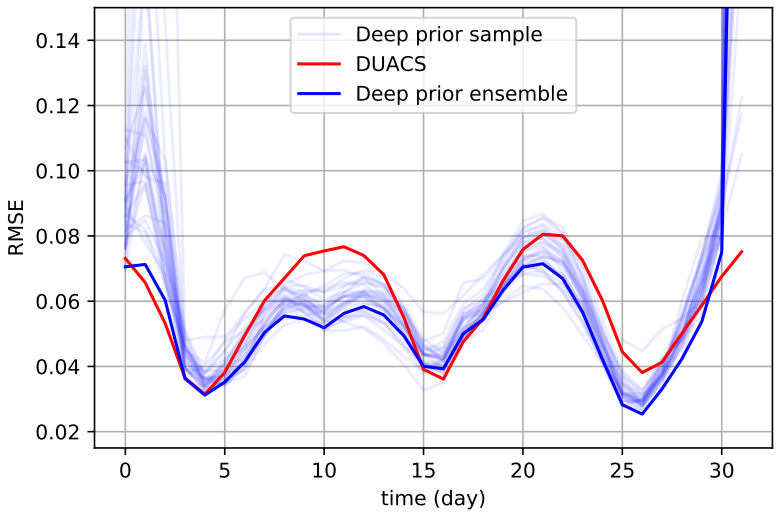


Figure 4.31: RMSE comparison of optimal interpolation from DUACS and deep spatio-temporal prior on a single 32-day observational window example

Looking at the error maps displayed in Figure 4.32, we conclude that both methods have very similar spatial structures. We also notice that error maps for the DUACS optimal interpolation present checkered numerical artifacts while the deep prior ones are smoother. Our interpretation is that various biases induced by the chosen deep architecture emphasize low-frequency patterns avoiding high-frequency artifacts introduced by numerical optimization directly at the pixel level.

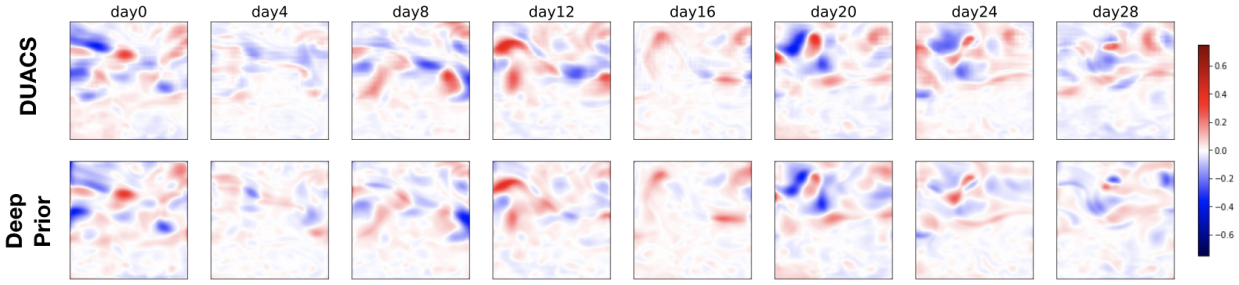


Figure 4.32: Error maps of DUACS and deep prior estimation at various times in the same observational window

Year-long analysis

We also compared both methods on a year-long analysis. But training an ensemble of deep priors at each window can be computationally cumbersome. To overcome this issue but still benefit from ensemble performances, we adopted a sliding window along the year and averaged estimation from different windows excluding border estimation. Results are

displayed in Figure 4.33. As for the single window experiment, RMSE scores are in the same range and slightly better with an ensemble of deep prior.

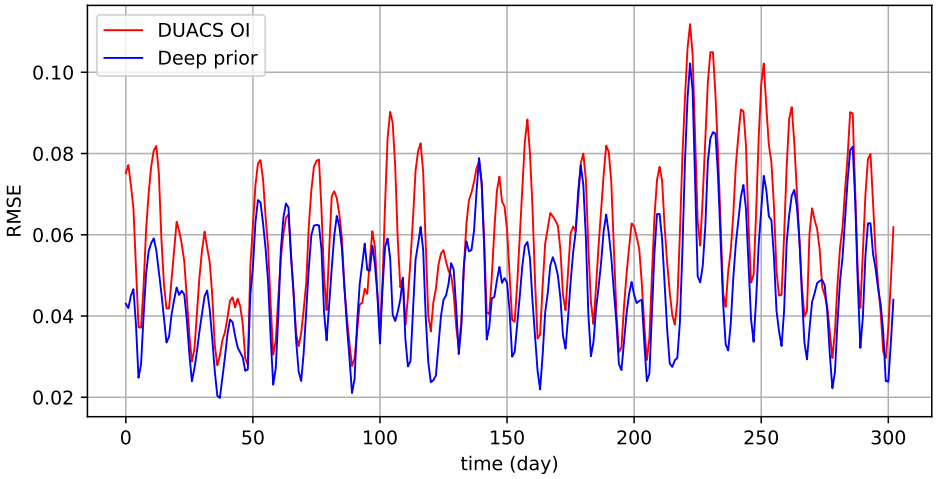


Figure 4.33: RMSE comparison of optimal interpolations from DUACS and sliding-averaged deep spatio-temporal prior on a year-long period

Conv(2+1)D ablation

To justify the use of (2+1)D convolutions, we performed a similar experiment using only 2D convolutions and considering the time as channels. Results displayed in Figure 4.34 show that such prior lacks temporal coherence and degrades performances, particularly at times when observations are very sparse.

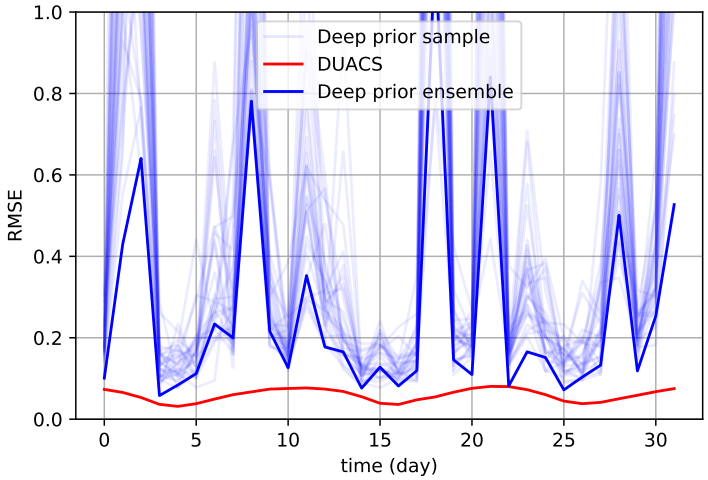


Figure 4.34: RMSE comparison of optimal interpolation from DUACS and deep prior with vanilla convolutional architecture, on a single 32-day observational window example

4.4 Perspective on variational assimilation with deep prior

4.4.1 Accounting for model error with deep spatio-temporal prior

Motivation

The first results obtained so far show that a convolutional neural network may play the role of a suitable background matrix. Also, in the previous section, we saw that using convolution in space and time, such architecture can account for spatio-temporal error modeling. In the conducted experiments involving a dynamical model, the perfect model hypothesis was made and even though it can be useful, it is rarely verified. A natural yet ambitious idea would be then to transpose ideas developed previously to account for model errors in the estimation without needing to specify the covariances matrices $\mathbf{Q}_t$. Also, using a spatio-temporal convolutional architecture, we hope to get rid of the *uncorrelated in time errors* hypothesis.

In Figure 4.37 we look into various error sources comparing similar dynamical models and see how differences in the dynamics propagate and accumulate over time. Error clearly shows temporal correlation. Even though it is not exactly comparable with a weak 4DVAR setup where controls at each time step may compensate such error propagation, we can have a glimpse on why it could be interesting to loosen the *uncorrelated in time hypothesis*. So that in this section, we will use a spatio-temporal deep prior $g_{\boldsymbol{\theta}}$ with the architecture given in the previous Section 4.30.

The question that remains is how to integrate the weak-constraint 4DVAR scheme with the architecture? Our first attempt was to replicate the classical additive error scheme, asking the deep prior to output the initial condition and the model errors over time, as drawn in Figure 4.35. The ambition here was to optimize model errors still only using the observational loss function. But the architecture was hard to optimize, probably due to the complex retro-propagation path similar to the one found in recurrent neural networks. To make it converge, we add to employ additional L_2-regularization and even with this, the accuracy of the estimation could not justify the additional methodological and computational complexity.

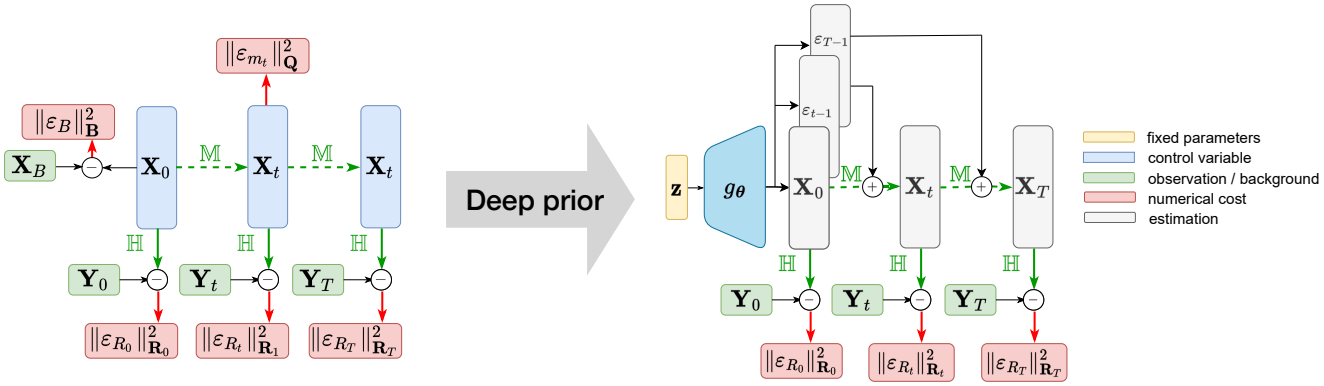


Figure 4.35: First attempt to account for model error with deep spatio-temporal prior

Method

Moving forward, we decided to use the spatio-temporal deep prior to generate the whole trajectory as in the previous section, such that $g_{\boldsymbol{\theta}}(z) = \widehat{\mathbf{X}}$. The log-posterior to be maximized is given in Eq. 4.9, where K is a constant. We could not find a proper analytical expression for the associated prior $p(\mathbf{X})$. The used cost function is similar to the weak 4DVAR one, except that the model error covariance matrix becomes a dilatation, $\mathbf{Q} = \lambda \mathbf{I}_d$. As discussed in Section 4.1.2, this also corresponds to a Gaussian prior hypothesis, the advantage would be to avoid designing a particular matrix $\mathbf{Q}$. A simple schematic view of the method is drawn in Figure 4.36 and we name the associated algorithm weak 4DVAR-STDP, STDP standing for Spatio-Temporal Deep Prior.

$$\begin{aligned} \log p(\mathbf{X} \mid \mathbf{Y}) = -\frac{1}{2} \sum_{t=0}^{T} \|\varepsilon_{R_t}\|^2_{\mathbf{R}_t} - \frac{\lambda}{2} \sum_{t=0}^{T-1} \|\varepsilon_{m_t}\|_2^2 + \log K \\ \text{s.t.} \quad g_{\boldsymbol{\theta}}(z) = \mathbf{X}_0 \quad \text{and} \quad \mathbb{M}(\mathbf{X}_t) + \varepsilon_{m_t} = \mathbf{X}_{t+1} \end{aligned} \tag{4.9}$$

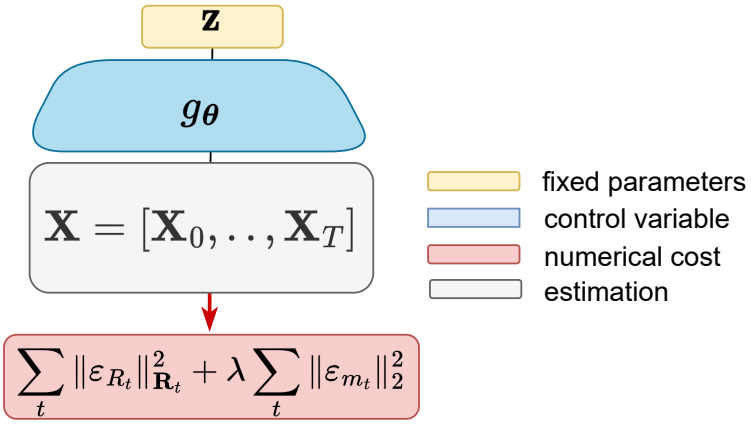


Figure 4.36: Schematic view of weak 4DVAR-STDP

Preliminary Experiment

We briefly tested the method in a twin experiment using the advection system, where the dynamical model is partially known. We simulated various model error sources by either using another scheme, removing the non-linear advection part, diminishing the time step size by 75%, or all at once. In Figure 4.37, errors between the reference model used to generate the data and the various degraded dynamics are displayed. We compared 3 algorithms: 4DVAR-B, 4DVAR-DIP and 4DVAR-STDP.

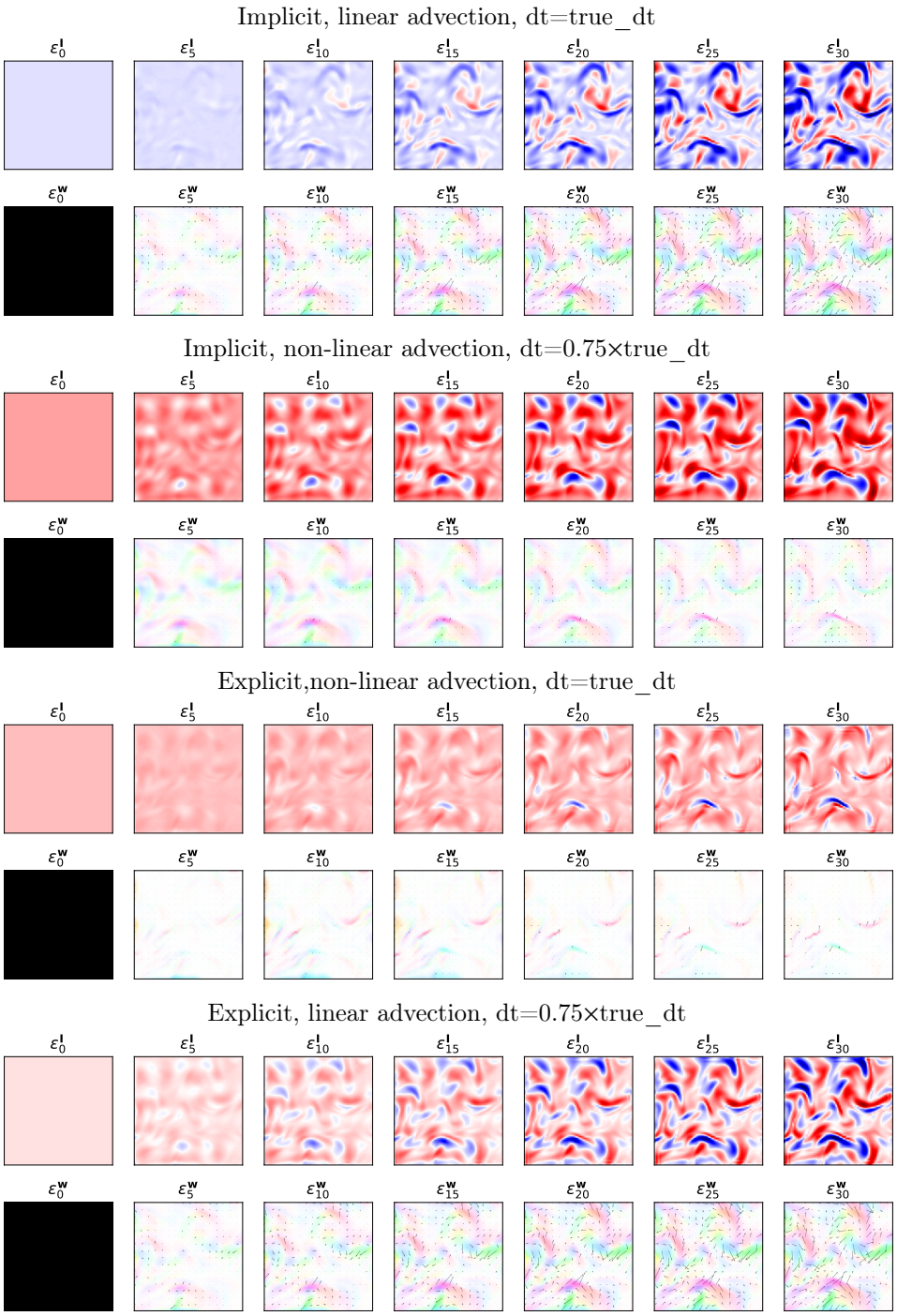


Figure 4.37: Advection model: Difference between integrated trajectory by a reference model versus a degraded one, starting from the exact same initial condition, using various degradation. Reference model: Implicit Semi-Lagrangian scheme for non linear advection.

The assimilation and forecast results (Figure 4.39 4.39) show that the best performing algorithm in this context is 4DVAR-B which is a bit disappointing. However, 4DVAR-STDP performs slightly better than 4DVAR-DIP which still constitutes an improvement.

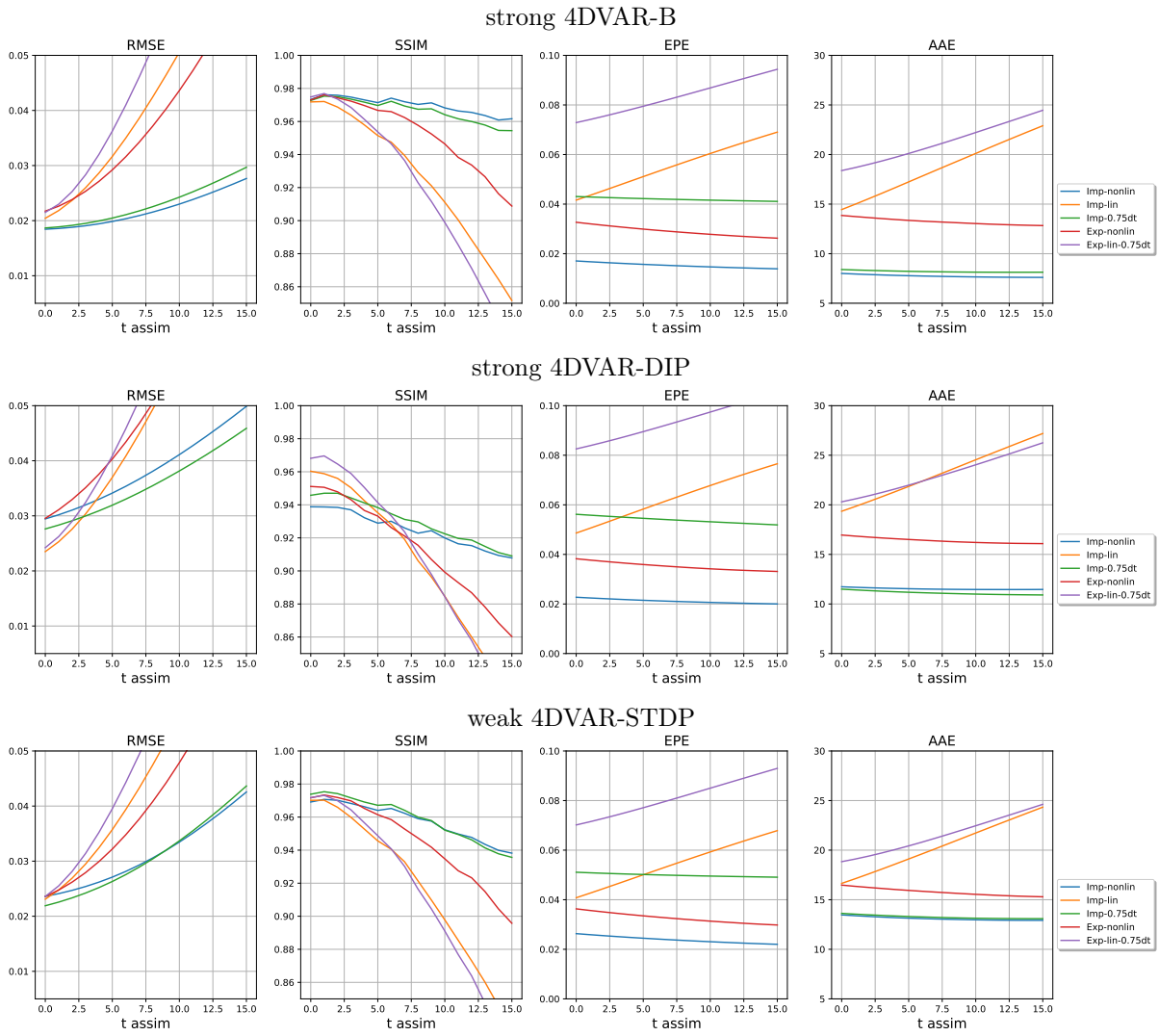


Figure 4.38: Advection system: Assimilation scores of 4DVAR-B, 4DVAR-DIP, 4DVAR-STDP using various degraded dynamics, average on 5 examples, 1.5% white noise

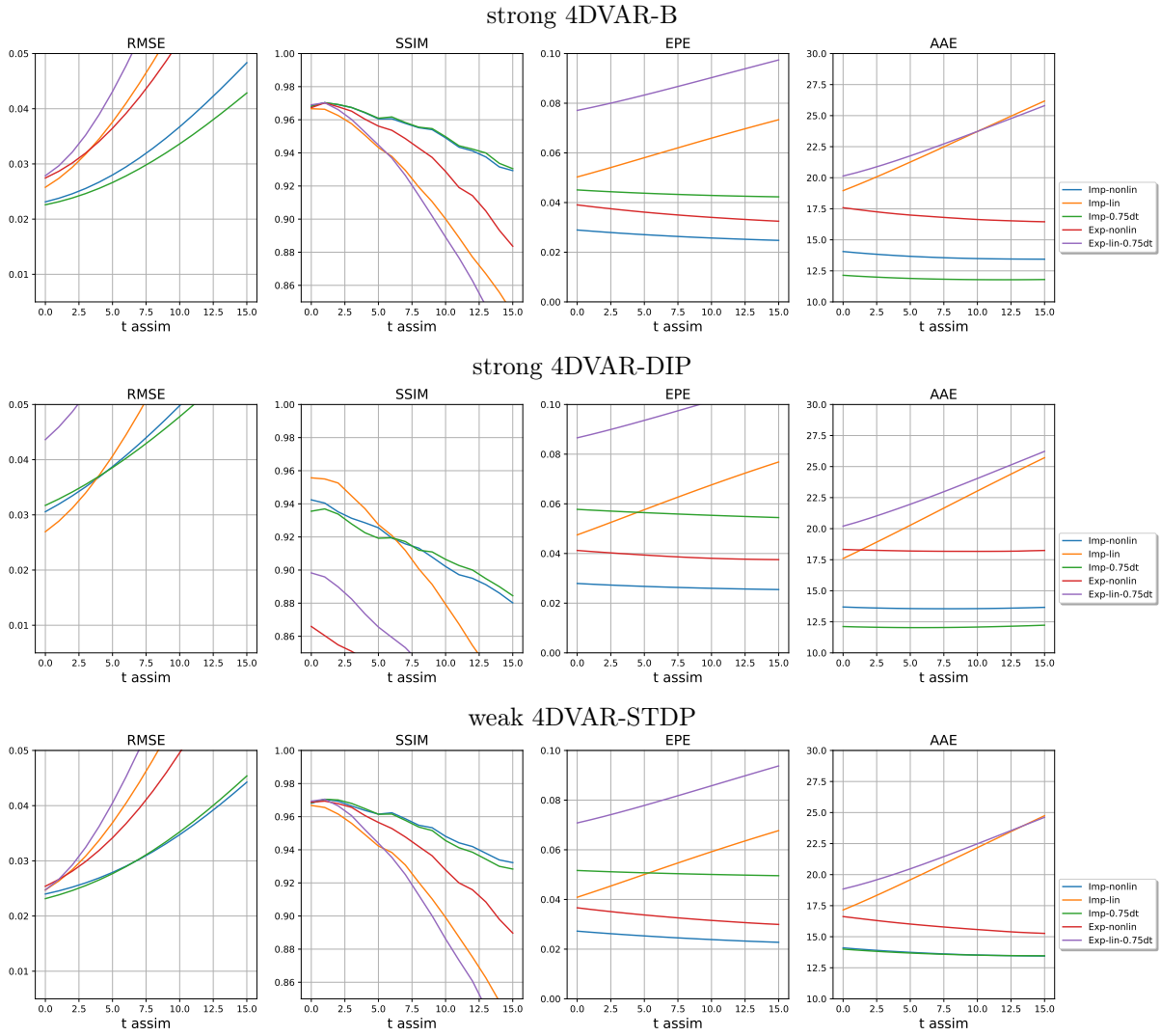


Figure 4.39: Advection system: Forecast scores of 4DVAR-B, 4DVAR-DIP, 4DVAR-STDP using various degraded dynamics, average on 5 examples, 1.5% white noise

4.4.2 Uncertainty quantification with Bayesian deep prior

Motivation

Bayesian deep learning [136, 137] is a tool allowing us to produce uncertainty quantification with deep neural networks. In [68], authors performed such Bayesian inference using deep image prior with Gaussian weights. To do so they used stochastic gradient Langevin dynamics [138] (SGLV) allowing to sample from the posterior. They show that their Bayesian deep prior avoids overfitting and allows relevant uncertainty quantification.

Method

We basically replicated the procedure used in [68] and adapted it to our physics-constrained inverse problem enabling uncertainty quantification. We are not confident enough to ensure that it really corresponds to SGLV. However, we did copy their optimization procedure carefully, using Algorithm 5 derived from their code[4].

Algorithm 5 – 4DVAR with deep bayesian prior

Initialize: fixed parameter z, control variables $\boldsymbol{\theta}$
while stop criterion **do**
  forward: integrate $\mathbb{M}_{0\to T}(g_{\boldsymbol{\theta}}(z))$ and compute $\mathcal{J}$
  backward: automatic differentiation returns $\nabla_{\boldsymbol{\theta}}\mathcal{J}$
  add noise: $\nabla_{\boldsymbol{\theta}}\mathcal{J} = \nabla_{\boldsymbol{\theta}}\mathcal{J} + \epsilon \quad \text{s.t.} \quad \epsilon \sim \mathcal{N}(0, lr)$
  update: $\boldsymbol{\theta} = \text{optimizer}(\boldsymbol{\theta}, \mathcal{J}, \nabla_{\boldsymbol{\theta}}\mathcal{J})$
  if n_iter $>$ n_burninphase **then** keep $\mathbf{X}_0$ as a sample
  end if
end while
return $\boldsymbol{\theta}, \mathbf{X}_0$ samples

Preliminary Experiment

We used the same shallow water experimental setting as in Section 4.1.2, and simply changed the algorithm. The assimilation scores are displayed in Section 4.40 and show that while still competing with 4DVAR-B, the method allows us to sample from the posterior, naturally giving an ensemble.

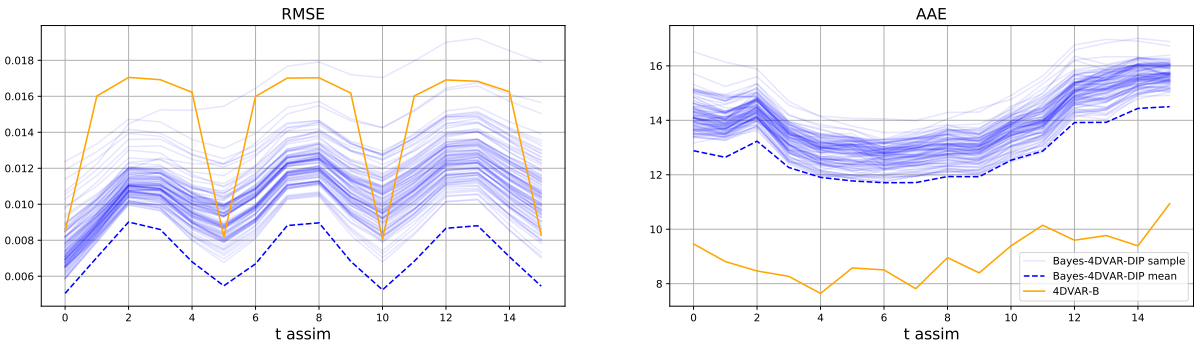


Figure 4.40: Shallow water: Assimilation scores of the Bayes 4DVAR-DIP assimilation

Looking at the error maps on $\boldsymbol{I}$ of the estimation and the standard deviation of the sampled trajectories in Figure 4.41, we doubt that the uncertainty introduced around

[4] `https://github.com/ZezhouCheng/GP-DIP/`

the weights provides information to quantify the uncertainty of the estimation. Our interpretation is the same as in Section 4.1.2, the introduced stochasticity is a numerical trick allowing to reduce the bias of the algorithm. However, one interest of such an approach could be to have the performance of an ensemble of 4DVAR-DIP in only one optimization.

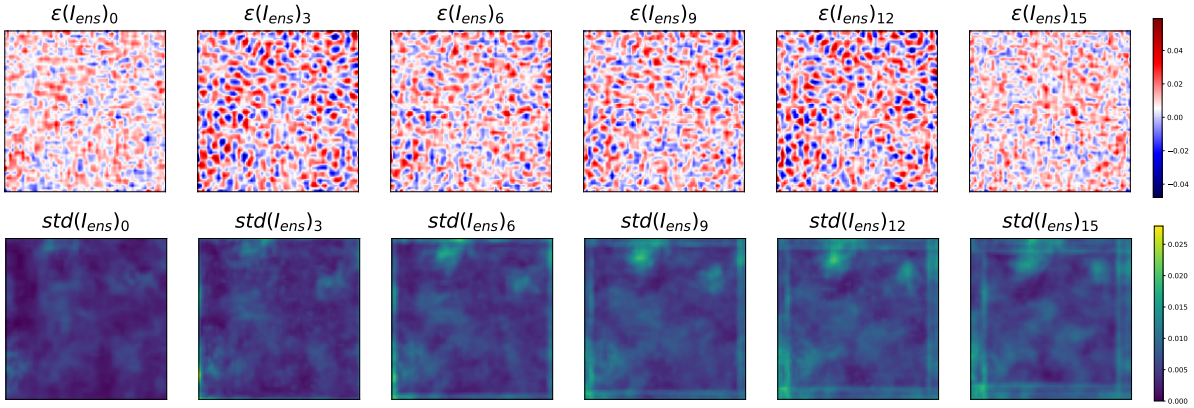


Figure 4.41: Shallow water: Error and standard deviation maps of the Bayes 4DVAR-DIP assimilation

CHAPTER 5

Conclusion

5.1 Overview

The idea of the thesis began with the statement that variational assimilation and deep learning are closely related and with the willingness to leverage recent deep learning advances to robustify assimilation. The first step was to clearly understand the mechanisms at stake in the 4DVAR algorithm but also re-coding it using deep learning tools. The flexible embedded automatic differentiation in the latter made the 4DVAR algorithm much more accessible, opening the door for hybrid methods. Having this in hand, we followed the recent literature on algorithms iterating data assimilation and machine learning steps, trying to learn mechanistic model errors as a dynamics, parameterized with a deep neural network. Setting up an experiment, we partially succeeded in retrieving missing dynamics but most importantly, the technical issues encountered had us questioning the method. In the hope of circumventing the cumbersome iterations and inspired by existing physics-constrained neural networks, we designed a hybrid architecture allowing us to learn the 4DVAR inversion directly from observation, unifying deep learning and variational assimilation in a single optimization phase. The results of the associated experiment led us to believe that the learning approach may have a smoothing effect, diminishing the overall accuracy of the assimilation, but also that the inductive bias of the used convolutional architecture had desirable regularizing properties. From this perspective and after discovering the paper "Deep Image Prior", we decided to investigate similar hybrid architectures but optimizing them on only one observational window. We exhibited in a strong-constraint 4DVAR and 3DVAR experiments, that a well-suited deep architecture can replace the usual background prior modeling, circumventing the

Gaussian background error hypothesis and the associated covariance matrix. Finally, we went back to our initial ambition to correct the model error and adapted the method in a weakly-constrained framework, and also tried to account for uncertainty quantification. To this day, this work is still in progress.

5.2 Software

An important part of the thesis concerned algorithmic development and a contribution we wanted to make is an easy-to-adapt code for potential users interested in developing their own 4DVAR algorithm, given at this GitHub address. As emphasized in Section 3.1.1, 4DVAR has long been hard to access due to tangent linear adjoint modeling but it is much easier now using tools like Pytorch with native automatic differentiation. Particularly, different versions of 4DVAR presented in the manuscript and coded in Pytorch, including vanilla's and deep prior's, are at disposal in the folder 'assimilation'. The code has been simplified over time leading to a minimalist version of 4DVAR. Adapting our code, the only thing you need is to format your data in the right shape and translate your forward model (dynamics and observation operator) in Pytorch, which is easy if you already have it in Python/Numpy. Also, classes are available to experiment in a few lines of code as shown in Listing 1. Such experiments using Lorenz96, Advection-based, and the Shallow water model can be found in the `notebook_demo` folder. Finally, the vast majority of the experiments presented in the manuscript are reproducible following the `Experiment` folder.

```
############################

# define a dynamical model
dynamics = SW()

# define a ground truth simulator
simulator = Simulator_SW()

# simulate ground truth
X = simulator.sample()

# define observation operator
H = Observation_operator_sw()

# define observation simulator with the desired noise
simulator_obs = Simulator_obs(H,sigma_perc=1)

# sample observation from ground truth trajectory
Y, Rm1, sigma = simulator_obs.sample(X)

# choose assimilation algorithm - strong 4DVar
assimilation = strong_4DVar(dynamics, H)

# fit / optimization
assimilation.fit(Y, Rm1)

# result / trajectory estimation
X_hat = assimilation.X_hat

# make a forecast ?
X_forecast = assimilation.forecast(n_step)

############################
```

Listing 1: Pytorch 4DVAR experiment exemple

5.3 Perspectives

5.3.1 Scaling

All the designed algorithms presented in the thesis have been tested on small-scale simulated systems. This setup is quite far from operational consideration concerning data quality, knowledge of the underlying physics, and complexity of the dynamical model. The question we can discuss but cannot answer now is the following: could the proposed algorithm be used in a tougher operational environment? Machine Learning and Data Assimilation are fields intimately tied to computational engineering. While the latter has been operating Numerical Weather Prediction for decades, handling "Big Data" before it was even a term, the former has been gaining traction at an ever-increasing speed. Tech giants already proved that deep learning-based methods can scale. In [139] they coined the term "Neural Earth System Modelling" and speculate on the convergence of both paradigms with certitudes on the hybridization. My personal opinion on the question is that the Data Assimilation community has the expertise advantage regarding data processing and physical modeling while the Machine Learning community has the methodological advantage, having addressed similar inverse problems toward tracking intractable integrals as discussed in Section 2.3.4.

5.3.2 End-to-end learning

High-dimensional applications inspired by deep image prior already exist [64] even though they avoid gridded space using geometric deep learning [140]. As for Data assimilation, these methods need to be optimized for each desired inference while deep learning-based inference is computationally cheap, being optimized only once. We would like to point out that the variational assimilation with deep prior can be transposed in a learning set-up using the framework given in Section 3.2, losing the advantage of working on a specific sample but gaining statistical knowledge of a database.

5.3.3 Reliable uncertainty quantification

Quantifying uncertainty is a constant preoccupation in numerical weather forecasting as considering different scenarios can be critical depending on the application. Regrettably, this has only been superficially addressed in the thesis. Uncertainty quantification is naturally done by ensemble data assimilation [6, 7] and similar ensembling methods can be used in a variational setting [123]. Machine learning also provides tools for generative modeling for instance using approximate variational inference [141, 138] and those are already applied for NWP [142]. From a far enough point of view, all these methods introduce arbitrary stochastic components that may be artificial in the sense that estimated uncertainty and actual errors are not correlated as it has been pointed out in [143]

studying ensemble Kalman filtering and observed in some of our variational experiments, as described in Section 4.1.2 and 4.4.2. To the best of our knowledge, a challenging perspective would be to work toward reliable uncertainty quantification.

Bibliography

[1] R. E. Kálmán and R. S. Bucy, "New results in linear filtering and prediction theory," *Journal of Basic Engineering*, vol. 83, pp. 95–108, 1961. 1, 8

[2] M. Bonavita, Y. Trémolet, E. Holm, T. Lang, M. Chrust, M. Janiskova, P. Lopez, P. Laloyaux, P. de Rosnay, H. M., M. Hamrud, and E. Stephen, "A strategy for data assimilation," Tech. Rep. 800, ECMWF, Apr. 2017. 1, 8, 32

[3] F.-X. Le Dimet and O. Talagrand, "Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects," *Tellus*, vol. 38A, no. 10, p. 97, 1986. 1, 8, 10, 17

[4] O. Talagrand, "Assimilation of observations, an introduction," *Journal of Meteorological Society of Japan*, vol. 75, no. 1B, pp. 191–209, 1997. 1, 8, 10

[5] Y. Trémolet, "Accounting for an imperfect model in 4D-Var," *Quarterly Journal of the Royal Meteorological Society*, vol. 132, pp. 2483–2504, 2006. 1, 3, 8, 10, 15

[6] G. Evensen, "The ensemble Kalman filter: Theoretical formulation and practical implementation," *Ocean Dynamics*, vol. 53, pp. 343–367, 2003. 1, 8, 94

[7] P. Houtekamer and H. L. Mitchell, "Data assimilation using an ensemble kalman filter technique," *Monthly Weather Review*, vol. 126, pp. 796–811, 1998. 1, 8, 94

[8] T. J. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*. Springer Series in Statistics (SSS), 2001. 2, 8, 9

[9] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais, and Prabhat, "Deep learning and process understanding for data-driven earth system science," *Nature*, vol. 566, pp. 195–204, 2019. 2, 18, 25

[10] A. J. Geer, "Learning earth system models from observations: machine learning or data assimilation?," *Philosophical Transactions of the Royal Society A*, vol. 379, 2021. 2, 18, 31

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016. 2, 16

[12] Y. B. Yann LeCun and G. Hinton, *Deep Learning*. Nature, 2015. 2

[13] S. Rasp, P. D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, and N. Thuerey, "Weatherbench: A benchmark dataset for data-driven weather forecasting," 2020. 2

[14] A. H., P. Rozdeba, and S. Shirman, "Machine learning: Deepest learning as statistical data assimilation problems," *Neural Computation*, vol. 30, no. 8, pp. 2025–2055, 2018. 2, 3, 32

[15] M. Bocquet, J. Brajard, A. Carrassi, and L. Bertino, "Bayesian inference of chaotic dynamics by merging data assimilation, machine learning and expectation-maximization," *Foundations of Data Science*, vol. 2, no. 1, p. 55–80, 2020. 2, 24

[16] J. Brajard, A. Carrassi, M. Bocquet, and L. Bertino, "Combining data assimilation and machine learning to infer unresolved scale parametrization," *Philosophical Transactions of the Royal Society A*, vol. 379, 2021. 2, 24

[17] M. Bocquet, J. Brajard, A. Carrassi, and L. Bertino, "Data assimilation as a learning tool to infer ordinary differential equation representations of dynamical models," *Nonlinear Processes in Geophysics*, vol. 26, no. 3, pp. 143–162, 2019. 2, 24

[18] D. Nguyen, S. Ouala, L. Drumetz, and R. Fablet, "Assimilation-based Learning of Chaotic Dynamical Systems from Noisy and Partial Data," in *ICASSP 2020 : International Conference on Acoustics, Speech, and Signal Processing*, 2020. 2, 24

[19] E. de Bézenac, A. Pajot, and P. Gallinari, "Deep learning for physical processes: Incorporating prior scientific knowledge," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2019, no. 12, p. 124009, 2019. 2, 17, 25

[20] L. Mosser, O. Dubrule, and M. Blunt, "Stochastic seismic waveform inversion using generative adversarial networks as a geological prior," *Mathematical Geoscience*, pp. 53–79, 2018. 2, 17, 25

[21] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. F. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, Çaglar Gülçehre, H. F. Song, A. J. Ballard, J. Gilmer, G. E. Dahl, A. Vaswani, K. R. Allen, C. Nash, V. Langston, C. Dyer, N. M. O. Heess, D. Wierstra, P. Kohli, M. M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," *ArXiv*, vol. abs/1806.01261, 2018. 2, 3, 16, 25, 32

[22] R. Fablet, B. Chapron, L. Drumetz, É. Mémin, O. Pannekoucke, and F. Rousseau, "Learning variational data assimilation models and solvers," *Journal of Advances in Modeling Earth Systems*, vol. 13, 2021. 2, 25

[23] R. Fablet, Q. Febvre, and B. Chapron, "Multimodal 4DVarNets for the reconstruction of sea surface dynamics from SST-SSH synergies." arXiv, 2022. 2, 25

[24] T. Janjić, N. Bormann, M. Bocquet, J. A. Carton, S. E. Cohn, S. L. Dance, S. N. Losa, N. K. Nichols, R. Potthast, J. A. Waller, and P. Weston, "On the representation error in data assimilation," *Quarterly Journal of the Royal Meteorological Society*, vol. 144, no. 713, pp. 1257–1278, 2018. 3, 8

[25] A. Carrassi and S. Vannitsem, "Accounting for model error in variational data assimilation: A deterministic formulation," *Monthly Weather Review*, vol. 138, pp. 3369–3386, 2010. 3

[26] M. Bocquet, C. A. L. Pires, and L. Wu, "Beyond gaussian statistical modeling in geophysical data assimilation," *Monthly Weather Review*, vol. 138, pp. 2997–3023, 2010. 3, 15

[27] Y. Trémolet and M. Fisher, "Weak constraint 4D-Var," *ECMWF Newsletter*, vol. 125, pp. 12–16, 2010. 3, 15

[28] M. Bonavita, L. Raynaud, and L. Isaksen, "Estimating background-error variances with the ecmwf ensemble of data assimilations system: some effects of ensemble size and day-to-day variability," *Quarterly Journal of the Royal Meteorological Society*, vol. 137, 2011. 3, 15

[29] B. Chapnik, G. Desroziers, F. Rabier, and O. Talagrand, "Diagnosis and tuning of observational error in a quasi-operational data assimilation setting," *Quarterly Journal of the Royal Meteorological Society*, vol. 132, 2006. 3, 15, 53

[30] M. Bonavita and P. Laloyaux, "Machine learning for model error inference and correction," *Journal of Advances in Modeling Earth Systems*, vol. 12, 2020. 3, 16

[31] A. Farchi, P. Laloyaux, M. Bonavita, and M. Bocquet, "Using machine learning to correct model error in data assimilation and forecast applications," *Quarterly Journal of the Royal Meteorological Society*, vol. 147, pp. 3067 – 3084, 2021. 3, 16, 24

[32] T. M. Mitchell, "The need for biases in learning generalizations," tech. rep., Rutgers University, New Brunswick, NJ, 1980. 3, 10, 16

[33] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Computer Vision on Pattern Recognition (CVPR)*, 2017. 3, 16, 78

[34] N. Cohen and A. Shashua, "Inductive bias of deep convolutional networks through pooling geometry," in *International Conference on Learning Representation (ICLR)*, 2017. 3, 16

[35] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. Hamprecht, Y. Bengio, and A. Courville, "On the spectral bias of neural networks," in *Proceedings of Machine Learning Research (PMLR)*, vol. 97, pp. 5301–5310, 2019. 3, 16

[36] A. Tarantola, *Inverse Problem Theory and Methods for Model Parameter Estimation*. Society for Industrial and Applied Mathematics, 2005. 6, 32

[37] J. S. Hadamard, "Sur les problèmes aux derivées partielles et leur signification physique," *Princeton University Bulletin*, vol. 13, pp. 49–52, 1902. 6

[38] A. N. Tikhonov and V. Y. Arsenin, "Solutions of ill-posed problems," *SIAM*, vol. 21, no. 2, 1977. 7

[39] M. Asch, M. Bocquet, and M. Nodet, *Data assimilation: methods, algorithms, and applications*. Fundamentals of Algorithms, SIAM, 2016. 7, 8, 11, 12, 32

[40] A. Carrassi, M. Bocquet, L. Bertino, and G. Evensen, "Data assimilation in the geosciences: An overview of methods, issues, and perspectives," *Wiley Interdisciplinary Reviews: Climate Change*, vol. 9, no. 5, p. e535, 2018. 7

[41] D. P. Dee, "Bias and data assimilation," *Quarterly Journal of the Royal Meteorological Society*, vol. 131, no. 613, pp. 3323–3343, 2005. 8

[42] C. W. Fox and S. J. Roberts, "A tutorial on variational bayesian inference," *Artificial Intelligence Review*, vol. 38, pp. 85–95, 2011. 9

[43] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, pp. 503–528, Aug 1989. 12, 32

[44] A. M. Bradley, "PDE-constrained optimization and the adjoint method." `https://cs.stanford.edu/~ambrad/adjoint_tutorial.pdf`. 12, 32

[45] L. Hascoët and V. Pascual, "The Tapenade Automatic Differentiation tool: Principles, Model, and Specification," *ACM Transactions On Mathematical Software*, vol. 39, no. 3, 2013. 13, 32

[46] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS Autodiff Workshop*, 2017. 13, 19, 32

[47] C. Henderson, "Best linear unbiased estimation and prediction under a selection model," *Biometrics*, vol. 31, no. 2, pp. 423–447, 1975. 14, 78

[48] F. Bretherton, E. Russ Davis, and C. Fandry, "A technique for objective analysis and design of oceanographic experiments applied to mode-73," *Deep Sea Research and Oceanographic Abstracts*, vol. 23, no. 7, pp. 559–582, 1976. 14, 78

[49] S. Puntanen and G. Styan, "The equality of the ordinary least squares estimator and the best linear unbiased estimator," *The American Statistician*, vol. 43, no. 3, pp. 153–161, 1989. 14, 79

[50] C. Johnson, B. Hoskins, and N. Nichols, "A singular vector perspective of 4D-Var: Filtering and interpolation," *Quarterly Journal of the Royal Meteorological Society*, vol. 131, pp. 1–19, 2005. 15

[51] J. Harlim, *Nonlinear and Stochastic Dynamics*, ch. Model error in data assimilation, pp. 276–317. Cambridge University Press, Jan. 2017. 15

[52] P. Laloyaux, M. Bonavita, M. Chrust, and S. Gürol, "Exploring the potential and limitations of weak-constraint 4D-Var," *Quarterly Journal of the Royal Meteorological Society*, vol. 146, pp. 4067 – 4082, 2020. 15

[53] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, 05 2015. 16

[54] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997. 16

[55] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, pp. 84 – 90, 2012. 16

[56] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *ArXiv*, vol. abs/1505.04597, 2015. 16

[57] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Computer Vision on Pattern Recognition (CVPR)*, pp. 770–778, 2016. 16

[58] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representation (ICLR)*, 2016. 16, 54

[59] G. Ongie, A. Jalal, C. Metzler, R. Baraniuk, A. Dimakis, and R. Willett, "Deep learning techniques for inverse problems in imaging," *Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 39–56, 2020. 16, 17, 31, 78

[60] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "FlowNet: Learning optical flow with convolutional networks," in *International Conference on Computer Vision (ICCV)*, pp. 2758–2766, 2015. 16

[61] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Computer Vision on Pattern Recognition (CVPR)*, pp. 1647–1655, 2017. 16

[62] Z. Teed and J. Deng, "RAFT: Recurrent all-pairs field transforms for optical flow," in *European Conference on Computer Vision (ECCV)*, 2020. 16

[63] J. Yoo, K. Jin, H. Gupta, J. Yerly, M. Stuber, and M. Unser, "Time-dependent deep image prior for dynamic MRI," *IEEE Transactions on Medical Imaging*, vol. PP, pp. 1–1, 05 2021. 17, 55

[64] F. Williams, T. Schneider, C. T. Silva, D. Zorin, J. Bruna, and D. Panozzo, "Deep geometric prior for surface reconstruction," in *Computer Vision on Pattern Recognition (CVPR)*, pp. 10122–10131, 2019. 17, 94

[65] H. Wang, T. Li, Z. Zhuang, T. Chen, H. Liang, and J. Sun, "Early stopping for deep image prior," *CoRR*, vol. abs/2112.06074, 2021. 17, 63

[66] G. Mataev, P. Milanfar, and M. Elad, "Deepred: Deep image prior powered by red," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 17, 61

[67] R. Heckel and P. Hand, "Deep decoder: Concise image representations from untrained non-convolutional networks," in *International Conference on Learning Representation (ICLR)*, 2019. 17, 66

[68] Z. Cheng, M. Gadelha, S. Maji, and D. Sheldon, "A bayesian perspective on the deep image prior," in *Conference on Computer Vision and Pattern Recognition*, 2019. 17, 88, 89

[69] T. Archambault, A. Filoche, A. A. Charantonis, and D. Béréziat, "Unlearned Downscaling of sea surface height with Deep Image Prior," in *IA for Earth Sciences Workshop The International Conference on Learning Representations (ICLR)*, (Virtual conference, United States), Apr. 2022. 17

[70] T. Beucler, M. Pritchard, S. Rasp, J. Ott, P. Baldi, and P. Gentine, "Enforcing analytic constraints in neural networks emulating physical systems," *Physical Review Letters*, vol. 126, pp. 1079–7114, Mar 2021. 17

[71] T. Beucler, S. Rasp, M. S. Pritchard, and P. Gentine, "Achieving conservation of energy in neural network emulators for climate modeling," in *International Conference on Machine Learning (ICML) Workshop, Climate Change: How Can AI Help?*, vol. abs/1906.06622, 2019. 17

[72] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part I): Data-driven solutions of nonlinear partial differential equations," *ArXiv*, vol. abs/1711.10561, 2017. 17, 25

[73] S. Cuomo, V. S. D. Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli, "Scientific machine learning through physics-informed neural networks: Where we are and what's next," *J. Sci. Comput.*, vol. 92, p. 88, 2022. 17

[74] J. Pathak, S. Subramanian, P. Z. Harrington, S. Raja, A. Chattopadhyay, M. Mardani, T. Kurth, D. Hall, Z.-Y. Li, K. Azizzadenesheli, P. Hassanzadeh, K. Kashinath, and A. Anandkumar, "Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators," *ArXiv*, vol. abs/2202.11214, 2022. 17

[75] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin, "Accelerating eulerian fluid simulation with convolutional networks," in *International Conference on Machine Learning, ICML*, pp. 5258–5267, 2017. 17, 25

[76] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 113, pp. 3932 – 3937, 2016. 18

[77] L. Zanna and T. Bolton, "Data-driven equation discovery of ocean mesoscale closures," *Geophysical Research Letters*, vol. 47, no. 17, 2020. 18

[78] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part II): Data-driven discovery of nonlinear partial differential equations," *ArXiv*, vol. abs/1711.10566, 2017. 18

[79] K. Cranmer, J. Brehmer, and G. Louppe, "The frontier of simulation-based inference," *Proceedings of the National Academy of Sciences*, vol. 117, pp. 30055 – 30062, 2020. 18

[80] G. Papamakarios, E. T. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *J. Mach. Learn. Res.*, vol. 22, pp. 57:1–57:64, 2021. 18

[81] J. Hermans, A. Delaunoy, F. Rozet, A. Wehenkel, and G. Louppe, "Averting a crisis in simulation-based inference," *ArXiv*, vol. abs/2110.06581, 2021. 18

[82] A. Delaunoy, J. Hermans, F. Rozet, A. Wehenkel, and G. Louppe, "Towards reliable simulation-based inference with balanced neural ratio estimation," *ArXiv*, vol. abs/2208.13624, 2022. 18

[83] S. Mishra-Sharma, "Inferring dark matter substructure with astrometric lensing beyond the power spectrum," *Machine Learning: Science and Technology*, vol. 3, 2022. 18

[84] J. Brehmer and K. Cranmer, "Simulation-based inference methods for particle physics," *Artificial Intelligence for High Energy Physics*, 2022. 18

[85] P. O. Gislason, J. A. Benediktsson, and J. R. Sveinsson, "Random forests for land cover classification," *Pattern Recognit. Lett.*, vol. 27, pp. 294–300, 2006. 18

[86] R. Grimm, T. Behrens, M. Märker, and H. Elsenbeer, "Soil organic carbon concentrations and stocks on barro colorado island — digital soil mapping using random forests analysis," *Geoderma*, vol. 146, pp. 102–113, 2008. 18

[87] S. Rasp, P. D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, and N. Thuerey, "Weatherbench: A benchmark data set for data-driven weather forecasting," *Journal of Advances in Modeling Earth Systems*, vol. 12, 2020. 18

[88] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W. chun Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *NIPS*, 2015. 18

[89] V. Bouget, D. Béréziat, J. Brajard, A. A. Charantonis, and A. Filoche, "Fusion of rain radar images and wind forecasts in a deep learning model applied to rain nowcasting," *Remote. Sens.*, vol. 13, p. 246, 2021. 18

[90] M. Jiang, K. Xu, X. Xu, L. Shi, X. Yu, and P. Liu, "Range noise level estimation of HY-2b radar altimeter and its comparison with Jason-2 and Jason-3 altimeters," in *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, pp. 8312–8315, 2019. 19, 68

[91] G. Madec, R. Bourdallé-Badie, P.-A. Bouttier, C. Bricaud, D. Bruciaferri, D. Calvert, J. Chanut, E. Clementi, A. Coward, D. Delrosso, *et al.*, "Nemo ocean engine," 2017. 20

[92] S. Pulkkinen, D. Nerini, A. A. Pérez Hortal, C. Velasco-Forero, A. Seed, U. Germann, and L. Foresti, "Pysteps: an open-source python library for probabilistic precipitation nowcasting (v1.0)," *Geoscientific Model Development*, vol. 12, no. 10, pp. 4185–4219, 2019. 21

[93] A. Zebiri, D. Béréziat, E. Huot, and I. Herlin, "Rain nowcasting from multiscale radar images," in *International Conference on Computer Vision Theory and Applications*, 2019. `https://hal.sorbonne-universite.fr/hal-02048500v1`. 21, 34, 35

[94] D. Béréziat and I. Herlin, "Image-based modelling of ocean surface circulation from satel-lite acquisitions," in *International Conference on Computer Vision Theory and Applications (VISAPP)*, 83, pp. 288–295, 2014. 21, 23

[95] D. Béréziat and I. Herlin, "Motion and Acceleration from Image Assimilation with evolution models," *Digital Signal Processing*, vol. 83, pp. 45–58, 2018. 21, 27, 34, 35

[96] A. Staniforth and J. Côté, "Semi-Lagrangian Integration Schemes for Atmospheric Models—A Review," *Monthly Weather Review*, vol. 119, pp. 2206–2223, 09 1990. 21, 27

[97] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981. 23, 27

[98] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal on Computer Vision*, vol. 92, pp. 1–31, March 2011. 23, 39

[99] D. Suter, "Motion estimation and vector splines," in *Conference on Vision and Pattern Recognition (CVPR)*, 1994. 23

[100] D. Béréziat, I. Herlin, and L. Younes, "A generalized optical flow constraint and its physical interpretation," in *Computer Vision on Pattern Recognition (CVPR)*, vol. 2, pp. 487–492, 2000. 23

[101] Y. Lepoittevin and I. Herlin, "Regularization terms for motion estimation. Links with spatial correlations," in *International Conference on Computer Vision Theory and Applications VISAPP*, vol. 3, pp. 458–466, 2016. 23

[102] P. Düben, U. Modigliani, A. Geer, S. Siemen, F. Pappenberger, P. Bauer, A. Brown, M. Palkovic, B. Raoult, N. Wedi, *et al.*, "Machine learning at ecmwf: A roadmap for the next 10 years," *ECMWF Technical Memoranda*, vol. 878, 2021. 24

[103] G. Shutts, M. Leutbecher, A. Weisheimer, T. Stockdale, L. Isaksen, and M. Bonavita, "Representing model uncertainty: stochastic parametrizations at ecmwf," *ECMWF*, pp. 19–24, 2011. 24

[104] A. Farchi, M. Bocquet, P. Laloyaux, M. Bonavita, and Q. Malartic, "A comparison of combined data assimilation and machine learning methods for offline and online model error correction." arXiv 2107.11114, 2021. 24

[105] P. Tandeo, P. Ailliot, M. Bocquet, A. Carrassi, A. Carrassi, T. Miyoshi, M. Pulido, M. Pulido, and Y. Zhen, "A review of innovation-based methods to jointly estimate model and observation error covariance matrices in ensemble data assimilation," *Monthly Weather Review*, 2020. 24

[106] M. Pulido, P. Tandeo, M. Bocquet, A. Carrassi, and M. Lucini, "Stochastic parameterization identification using ensemble kalman filtering combined with maximum likelihood methods," *Tellus A: Dynamic Meteorology and Oceanography*, vol. 70, pp. 1 – 17, 2018. 24

[107] Z. Ghahramani and S. Rowis, "Learning nonlinear dynamical systems using an em algorithm," *Adv. Neural Inf. Processing Syst.*, vol. 11, 03 1999. 24

[108] R. Fablet, S. Ouala, and C. Herzet, "Bilinear residual neural network for the identification and forecasting of dynamical systems," in *European Signal Processing Conference (EUSIPCO)*, 2017. 25, 78

[109] R. Fablet, M. 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," in *Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XXIV ISPSR Congress*, 2021. 25

[110] A. Filoche, J. Brajard, A. A. Charantonis, and D. Béréziat, "Completing physics-based models by learning hidden dynamics through data assimilation," in *NeurIPS 2020, AI for Earth Sciences Workshop*, pp. 1–4, 2020. 25

[111] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Learning and Intelligent Optimization*, pp. 507–523, 2011. 28

[112] Z. Zhu, J. Wu, B. Yu, L. Wu, and J. Ma, "The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects," 2018. 32

[113] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015. 32, 54

[114] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, 1986. 32

[115] A. Filoche, D. Béréziat, J. Brajard, and A. A. Charantonis, "Variational assimilation of Geophysical images leveraging deep learning tools," in *ORASIS 2021*, (Saint Ferréol, France), Centre National de la Recherche Scientifique [CNRS], Sept. 2021. 34

[116] G. Larvor, L. Berthomier, V. Chabot, B. Le Pape, B. Pradel, and L. Perez, "MeteoNet, an open reference weather dataset by Meteo-France," 2020. `https://meteonet.umr-cnrm.fr/`. 34

[117] J. L. Morales and J. Nocedal, "Remark on algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization," *ACM Transaction on Mathematical Software*, vol. 38, pp. 7:1–7:4, December 2011. 40

[118] E. N. Lorenz, *Predictability of Weather and Climate*, ch. Predictability – a problem partly solved, pp. 40–58. Cambridge University Press, Dec. 2006. 47

[119] C. A. L. Pires, R. Vautard, and O. Talagrand, "On extending the limits of variational assimilation in nonlinear chaotic systems," *Tellus A*, vol. 48, pp. 96–121, 1996. 50

[120] A. Filoche, D. Béréziat, and A. A. Charantonis, "Deep prior in variational assimilation to estimate ocean circulation without explicit regularization," in *Climate Informatics*, (Asheville, NC, United States), May 2022. 52

[121] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, 2016. 54, 80

[122] Z. Allen-Zhu, Y. Li, and Z. Allen-Zhu, "Towards understanding ensemble, knowledge distillation and self-distillation in deep learning." arXiv preprint, December 2020. 59

[123] G. Desroziers, J.-T. Camino, and L. Berre, "4DEnVar: link with 4D state formulation of variational assimilation and different possible implementations," *Quarterly Journal of the Royal Meteorological Society*, vol. 140, no. 684, pp. 2097–2110, 2014. 59, 94

[124] A. Filoche and D. Béréziat, "Simultaneous Assimilation and Downscaling of Simulated Sea Surface Heights with Deep Image Prior," in *Congrès Reconnaissance des Formes, Image, Apprentissage et Perception (RFIAP)*, July 2022. 67

[125] T. Bolton and L. Zanna, "Applications of deep learning to ocean data inference and subgrid parameterization," *Journal of Advances in Modeling Earth Systems*, vol. 11, no. 1, pp. 376–399, 2019. 67

[126] H. Hewitt, M. Bell, E. Chassignet, A. Czaja, D. Ferreira, S. Griffies, P. Hyder, J. McClean, A. New, and M. Roberts, "Will high-resolution global ocean models benefit coupled predictions on short-range to climate timescales?," *Ocean Modelling*, vol. 120, pp. 120–136, Dec. 2017. 67

[127] P. Sandery and P. Sakov, "Ocean forecasting of mesoscale features can deteriorate by increasing model resolution towards the submesoscale," *Nature Communications*, vol. 8, p. 1566, Nov 2017. 67

[128] G. Taburet, A. Sanchez-Roman, M. Ballarotta, M.-I. Pujol, J.-F. Legeais, F. Fournier, Y. Faugere, and G. Dibarboure, "DUACS DT2018: 25 years of reprocessed sea level altimetry products," *Ocean Science*, vol. 15, no. 5, pp. 1207–1224, 2019. 68, 78, 79

[129] A. Filoche, T. Archambault, A. A. Charantonis, and D. Béréziat, "Statistics-free interpolation of ocean observations with deep spatio-temporal prior," in *ECML/PKDD Workshop on Machine Learning for Earth Observation and Prediction (MACLEAN)*, Sept. 2022. 78

[130] V. Enjolras, P. Vincent, J.-C. Souyris, E. Rodriguez, L. Phalippou, and A. Cazenave, "Performances study of interferometric radar altimeters: from the instrument to the global mission definition," *Sensors*, vol. 6, no. 3, pp. 164–192, 2006. 78, 79

[131] L. Gaultier, C. Ubelmann, and L.-L. Fu, "The challenge of using future swot data for oceanic field reconstruction," *Journal of Atmospheric and Oceanic Technology*, vol. 33, no. 1, 2016. 78, 79

[132] M. Ballarotta, E. Cosme, and A. Albert, "ocean-data-challenges/2020a_SSH_mapping_NATL60: Material for SSH mapping data challenge," Sept. 2020. https://doi.org/10.5281/zenodo.4045400. 78

[133] A. Ajayi, J. Le Sommer, E. Chassignet, J.-M. Molines, X. Xu, A. Albert, and E. Cosme, "Spatial and temporal variability of north atlantic eddy field at scale less than 100 km.," *Earth and Space Science Open Archive*, p. 28, 2019. 78

[134] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *International Conference on Learning Representation (ICLR)*, 2016. 80

[135] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Computer Vision on Pattern Recognition (CVPR)*, 2018. 80

[136] L. V. Jospin, W. L. Buntine, F. Boussaid, H. Laga, and Bennamoun, "Hands-on bayesian neural networks—a tutorial for deep learning users," *IEEE Computational Intelligence Magazine*, vol. 17, pp. 29–48, 2022. 88

[137] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," *ArXiv*, vol. abs/1505.05424, 2015. 88

[138] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *International Conference on Machine Learning (ICML)*, 2011. 88, 94

[139] C. Irrgang, N. Boers, M. Sonnewald, E. A. Barnes, C. Kadow, J. Staneva, and J. Saynisch-Wagner, "Towards neural earth system modelling by integrating artificial intelligence in earth system science," *Nat. Mach. Intell.*, vol. 3, pp. 667–674, 2021. 94

[140] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velivckovi'c, "Geometric deep learning: Grids, groups, graphs, geodesics, and gauges," *ArXiv*, vol. abs/2104.13478, 2021. 94

[141] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2014. 94

[142] S. V. Ravuri, K. Lenc, M. Willson, D. Kangin, R. Lam, P. W. Mirowski, M. Fitzsimons, M. Athanassiadou, S. Kashem, S. Madge, R. Prudden, A. Mandhane,

A. Clark, A. Brock, K. Simonyan, R. Hadsell, N. H. Robinson, E. Clancy, A. Arribas, and S. Mohamed, "Skilful precipitation nowcasting using deep generative models of radar," *Nature*, vol. 597, pp. 672 – 677, 2021. 94

[143] K. J. H. Law and A. M. Stuart, "Evaluating data assimilation algorithms," *American Meteorological Society*, vol. 140, pp. 3757–3782, Nov. 2012. 94