THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Institut National Polytechnique de Toulouse (Toulouse INP)

Discipline ou spécialité :

Océan, atmosphère, climat

Présentée et soutenue par :

M. CAMILLE BESOMBES

le lundi 13 décembre 2021

Titre :

Parameterization using Generative Adversarial Networks for Control Space Reduction in Data Assimilation.

Ecole doctorale :

Sciences de l'Univers de l'Environnement et de l'Espace (SDU2E)

Unité de recherche :

Centre Européen de Recherche et Formation Avancées en Calcul Scientifique (CERFACS)

Directeur(s) de Thèse :

M. OLIVIER THUAL

M. CORENTIN LAPEYRE

Rapporteurs :

M. ERIC BLAYO, UNIVERSITE GRENOBLE ALPES

M. RONAN FABLET, IMT ATLANTIQUE

Membre(s) du jury :

MME HÉLÈNE ROUX, TOULOUSE INP, Président

M. CORENTIN LAPEYRE, CERFACS, Membre

M. MANUEL MARCOUX, TOULOUSE INP, Membre

MME LAURE RAYNAUD, CENTRE NATIONAL DE RECHERCHES METEO, Invité(e)

M. OLIVIER PANNEKOUCKE, CENTRE NATIONAL DE RECHERCHES METEO, Invité(e)

M. OLIVIER THUAL, TOULOUSE INP, Membre

Résumé

Ce travail porte sur l'utilisation des réseaux de neurones génératifs et plus particulièrement les GANs (generative adversarial networks) pour la paramétrisation dans le cadre des méthodes d'ensemble pour l'assimilation de données. L'assimilation de données permet d'estimer les paramètres initiaux ou l'état d'un modèle physique à l'aide d'observations en prenant en compte les incertitudes associées à ces dernières. Le filtre de Kalman donne une solution analytique lorsque le modèle physique est linéaire et les différentes sources d'erreurs suivent une distribution Gaussienne. Les méthodes d'ensemble permettent d'appliquer cette méthode à des systèmes physiques non-linéaires représentés par des modèles numériques. L'estimation de paramètres ne suivant pas une distribution Gaussienne reste un challenge dans beaucoup de domaines. Des méthodes de paramétrisation sont alors mises en place afin de transformer ces paramètres en de nouveaux, plus adaptés aux hypothèses des méthodes ensemblistes. Une autre limitation est la cohérence des paramètres estimés et l'utilisation d'information *a priori* comme les contraintes physiques que les paramètres doivent respecter. En effet, les paramètres estimés peuvent ne pas avoir de sens physique comme une température négative par exemple. Les méthodes de paramétrisation sont également utilisées afin de limiter ce phénomène. Enfin un dernier avantage de ces méthodes est qu'elles permettent de limiter le nombre de paramètres en réduisant leur dimension après transformation.

Dans cette étude une nouvelle méthode de paramétrisation utilisant les GANs, appliquée à la caractérisation de réservoirs souterrains est présentée. Lors de l'estimation à l'aide de méthodes d'ensemble de la disposition des différents types de roches au sein d'un réservoir, il est courant d'obtenir des formes d'hétérogénéités géologiques irréalistes. Ces hétérogénéités sont caractérisées par des formes et des motifs particuliers issus de phénomènes physiques connus. De plus, le type de roche n'est pas un paramètre continu respectant l'hypothèse de distribution Gaussienne et est de grande dimension pour des applications industrielles. L'utilisation d'une méthode de paramétrisation est alors requise, mais la conservation du réalisme géologique par ces dernières reste soit trop peu réaliste, soit trop coûteuse numériquement. Le GAN étant une technique issue des méthodes d'apprentissage automatique et ayant récemment gagné en notoriété pour sa capacité à pouvoir apprendre et générer des images complexes. Il constitue un choix prometteur pour son application dans le domaine de la caractérisation des réservoirs souterrains. Cette étude présente les résultats obtenus sur un cas de réservoir simplifié comportant des hétérogénéités en forme de chenaux, particulièrement difficile à paramétriser par les méthodes actuelles.

Une seconde application est abordée lors de cette étude portant sur la prédiction des champs atmosphériques à l'aide des méthodes d'assimilation de données. Lors de l'estimation de l'état de l'atmosphère, pour la prédiction météorologique par exemple, il est important de corriger l'état atmosphérique avec de nouvelles observations de manière que les nouveaux champs respectent les équilibres physiques qui régissent la circulation atmosphérique. Quand cela n'est pas le cas, des instabilités numériques peuvent apparaître lors de la simulation de l'état futur de l'atmosphère, détériorant l'information apportée par les observations. L'utilisation d'un GAN capable d'apprendre les contraintes physiques qui caractérise un champ atmosphérique à l'état d'équilibre peut s'avérer utile. C'est dans ce contexte que la seconde application de cette étude s'inscrit.

Ce travail vise à présenter les performances de la paramétrisation du GAN et son applicabilité multidisciplinaire aux lecteurs qui ne sont pas familiers avec le domaine de l'apprentissage profond. Les générations issues du GANs sont encodées dans un espace latent de faible dimension qui peut être

échantillonné à partir d'une distribution gaussienne adaptée à l'assimilation de données d'ensemble. La propriété non supervisée de ce type de paramétrisation le rend applicable à plusieurs domaines divers tels que l'apprentissage du modèle des hétérogénéités géologiques ou l'apprentissage des contraintes physiques qui rendent un état atmosphérique équilibré.

Cette étude montre comment entraîner des GANs pour deux applications différentes : les données de réservoir souterrain et les données de climat. L'utilisation de la paramétrisation dans un ensemble basé sur l'assimilation de données tel que le lisseur d'ensemble avec multiples assimilations de données (ES-MDA) est démontrée pour le réservoir souterrain. Enfin, le conditionnement a posteriori de la fonction du GAN est examiné en utilisant l'optimisation sans dérivation.

Abstract

This thesis examines the use of generative adversarial networks (GANs) as a parameterization tool for inverse problems solved with ensemble-based data assimilation methods. Ensemble methods often rely on the assumption of Gaussian distributed parameters in cases where this assumption is not valid, the parameter estimation can be invalid. Parameterization methods allow the transformation of these non-Gaussian parameters into a better suited distribution, and optimally reduce their dimension. Another limitation of ensemble methods is the injection of prior information of the physical relation as a constraint between parameters such as spatial coherence or physical balances. Optimal parameterization should encompass these different properties to facilitate the estimation. The novel approach presented in this work relies on GANs to achieve these objectives. Two application domains are tackled through the present work.

In a first application, subsurface reservoir characterization, the objective is to determine geological properties of a numerical reservoir model from the observation of the reservoir dynamical response by the way of data assimilation. Rock facies, that describe the type of rock present in each cell of the numerical model, have to be determined due to their strong influence on the dynamical response. The rock facies spatial distribution is ruled by geological phenomena such as sedimentation and forms well known patterns, like channels, called heterogeneities. The non-continuous property and their spatial coherence make their characterization by ensemble-based data assimilation algorithms difficult, and requires parameterization. Parameterization is a challenge for numerous heterogeneities, notably channels, due to the numerical cost or the statistical representation of their spatial distribution.

A Second application domain is the atmospheric balance in the context of numerical weather prediction. When new observations are available, correction of the atmospheric state is done using ensemble-based data assimilation methods. This correction step can introduce imbalance in the physical state and cause numerical instability during the integration in time of the atmosphere, deteriorating the information brought by the previous observations. The importance of generating or correcting balanced climate, also called initialized atmospheric state, during data assimilation is then a key step in numerical weather prediction.

This work aims at presenting the performance of GAN parameterization and its multi-disciplinary applicability to researchers who are not familiar with the domain of deep learning. GAN is an unsupervised deep learning method belonging to the deep generative network family, able to learn a dataset distribution and generate new samples from the learned distribution in an unsupervised way. These synthetic samples are encoded in a low-dimensional latent space that can be sampled from a Gaussian distribution that is suited to perform ensemble data assimilation. Their recent ability to generate complex images led us to consider them as a good candidate for parameterization method.

The unsupervised property of this type of parameterization makes it applicable to several diverse domains such as learning the pattern of geological heterogeneities or learning the physical constraints that make an atmospheric state balanced.

This study shows how to train GANs for two different applications : subsurface reservoir and climate data. The use of the parameterization in an ensemble based data assimilation such as ensemble smoother with multiple data assimilation (ES-MDA) is demonstrated for subsurface reservoirs. Finally, a posteriori conditioning of the GAN function is examined using derivative free optimization.

Remerciements

Ces trois années de thèse ont été menées grâce à l'aide d'une multitude de personnes que j'aimerai prendre le temps de remercier dans ce manuscrit.

Je voudrai commencer par remercier les membres du jury d'avoir accepté d'évaluer mes travaux de thèse et leur participation à la soutenance. Je suis conscient qu'il n'était pas facile de trouver des experts des différents domaines abordés par mes travaux de recherche, et je suis très satisfait de l'intérêt que vous avez porté à mes travaux et des échanges très intéressants qui ont pu en ressortir. Je remercie plus particulièrement Eric Blayo et Ronan Fablet qui ont accepté de rapporter cette thèse et dont les commentaires m'ont beaucoup apporté autant par des interrogations de personnes extérieurs à mes travaux que sur leur expertise des domaines d'applications abordés par mon travail. Enfin je remercie Hélène Roux d'avoir présidé ce jury et à Olivier Pannekoucke et Laure Raynaud d'en avoir fait partie.

Ma thèse a été réalisée en partenariat avec l'entreprise TotalEnergies et plus particulièrement l'équipe de Philippe Berthet. Je tiens à remercier toutes les personnes avec qui j'ai interagi de manière régulière : Philippe Berthet, Daniel Busby, Tatiana Chugunova, Anahita Abadpour qui m'ont dirigé et soutenu tout au long de ces trois années. Vous avez fait preuve de pédagogie pour me transmettre vos nombreuses connaissances et d'écoute sur les sujets qui ne vous étaient pas familiers. Votre accueil et les discussions que j'ai pu avoir lors de mes déplacements à Pau resteront pour moi des moments plaisants. Un remerciement spécial pour Rabeb Selmi qui est venue travailler au CERFACS pendant une bonne partie de la thèse qui m'a été d'une grande aide de part ses connaissances et avec qui il a été très intéressant d'échanger sur de nombreux sujets scientifiques.

Au fil de ces 3 années, une collaboration avec des experts du climat et plus particulièrement avec Olivier Pannekoucke et Benjamin Sanderson a abouti à la publication d'un article. Je voudrais les remercier pour leur contribution mais aussi pour leur curiosité, leur suivi et leurs explications claires et toujours intéressantes. Ces échanges ont toujours été pour moi des moments d'apprentissage passionnants.

Je voudrais évidemment remercier mes encadrants Olivier et Corentin. Olivier, ton optimisme, ta confiance et ta vision générale des sujets scientifiques que j'ai abordés lors de mon doctorat m'ont été d'une grande aide. Ca a été un vrai plaisir d'avoir été ton élève en école d'ingénieur où tu es indiscutablement apprécié pour ta pédagogie et ta bonne humeur sans faille. Cela a été encore plus le cas de pouvoir échanger avec toi lors des réunions, ton entrain a également beaucoup aidé dans les moments de doutes. Corentin, c'était un plaisir d'avoir été ton premier doctorant. J'ai beaucoup appris à tes côtés, je tiens à te remercier pour ta constance dans mon accompagnement, les nombreux conseils que tu m'as apportés et les différentes discussions qu'on a pu avoir sur les différents sujets scientifiques ou autres. Tu m'as transmis ta passion pour le domaine de l'intelligence artificielle pendant ces trois années qui est maintenant la voie dans laquelle je souhaite continuer. Merci à vous deux qui avez joué un rôle important dans l'aboutissement de ce travail.

Ensuite je veux remercier les personnes que j'ai cotoyées au CERFACS lors de ces trois années. Tout d'abord le département administratif qui a toujours fait preuve de soutien et de bienveillance : Brigitte, Marie, Chantal, Michèle, Nathalie et Lydia. L'équipe CSG que j'ai beaucoup sollicité et qui s'est toujours montrée très disponible lors de mes sollicitations : Isabelle, Fred, Gérard, Patrick, Fabrice et Nicolas. L'équipe GlobC/CECI avec qui j'ai pu avoir des discussions très intéressantes : Sophie, Bastien, Mélanie et Laurent. Enfin l'équipe Algo : Gabriel, Antoine et Selime. Merci à vous tous pour votre aide. Sur un plan plus personnel, également au CERFACS, je voudrais remercier mes

collègues de bureau Victor (courage pour la fin de ta thèse) et Nicolas avec qui j'ai partagé cette nouvelle passion et curiosité pour le domaine de l'IA, merci pour l'entraide, le soutien et la bonne ambiance. Un grand merci aux anciens qui m'ont accueilli et aux nouveau que nous avons ramené dans cette grande aventure qui est celle du club de foot de l'En Avant Cerfacs. Cela a été un très bon moyen de se changer les idées et de rigoler, un plaisir d'avoir été votre président et je vous souhaite un bonne continuation dans l'ascension de la ligue. Merci au grand groupe de collègues et qui sont maintenant des amis avec qui j'ai pu discuter autour d'une bière et faire la fête : David, Paul, Thomas L, Thibault G, Thibault D, Adèle, Adrien, Ekhi, Matthieu, Théo D, Thomas N, Etienne. Et tous ceux que je connaissais déjà de l'école d'ingénieur : Antoine, Nico B, Théo O, Nicolas U...

Je veux également remercier la team des Bills Alexis, Thomas, Arthur, Simon et Gabriel qui m'ont changé les idées lorsqu'il le fallait et avec qui il était très cool de partir en vacances ! Merci plus particulièrement à Alexis et Thomas pour le soutien entre thésards et tout mon courage pour l'année qu'il vous reste ! Il est important pour moi de remercier les amis de plus longue date que j'ai pu moins voir lors de ces trois années mais qui ont un rôle tout aussi important : la team prépa qui est un groupe qui m'est chère que je suis toujours impatient de retrouver : Thomas, Hippolyte, Nicolas, Paul, Henri, JT, Juliette et Ben au passage bon courage pour la fin de ta thèse ! Mes amis d'enfance de Bazas : Valentin, Marc, Elliot, Cedric, Léo, Matthieu G et Matthieu P qui lors de nos retrouvailles me ramène 10 ans en arrière instantanément. Enfin la team bordelaise merci à cette petite équipe inséparable qui m'a beaucoup apportée et qui est toujours un plaisir à retrouver merci Julien, Dov, Max et Nico. Nicolas tu sais à quel point c'est indispensable pour moi de te retrouver régulièrement en physique ou en virtuel avec vous tous à refaire le monde, rigoler ou simplement prendre des nouvelles.

Enfin les derniers remerciements vont à ma famille. Tout d'abord à la personne que j'ai rencontré au démarrage de cette thèse et qui partage ma vie depuis. Magaly, je tiens a figé dans ce document l'importance que tu as eu tout au long de cette thèse. Tu as sacrifié un grand nombre de choses importantes pour me rejoindre à Toulouse et tu as ensuite essuyé tous mes doutes et mes difficultés les uns après les autres pendant ces trois années. J'ai au moins autant appris personnellement à tes côtés que dans ce doctorat. Tu as été la source de mes plus grandes joies et la seule à avoir vu mes plus grandes peines. Ton courage, tes efforts et ton amour ont été inconditionnels, et te rendre tout cela dans nos projets futurs est mon objectif. Je sais que tu es autant soulagée que moi que tout cela soit terminé, merci pour ta confiance, tes relectures et tes surprises qui m'ont permis de relativiser dans les bons et mauvais moments. Je veux également remercier mes frères, Benjamin et Léo. La fierté dans vos yeux est ce qui me fait avancer. Sans vous deux, rien de tout cela n'aurait été possible. Merci de m'avoir encouragé et de m'avoir transmis votre confiance qui compte beaucoup. Je suis extrêmement heureux de partager ce lien fraternel si fort avec vous. Enfin merci à ma mère disparue peu avant le début de cette thèse. Mon seul regret est de ne pas t'avoir eu à cette soutenance. Je sais que tu aurais été très fière, sache que tu as réussi ton rôle de mère haut la main malgré les nombreuses difficultés. Ce diplôme est autant le tien que le mien, il est le fruit des valeurs que tu m'as transmises. Merci Mathieu pour ton aide dans tout cela, tu as eu un rôle indispensable qui m'a permis de garder le cap pendant ces trois années. Ce doctorat t'est également dédié.

Merci à tous.

Publications

Besombes, C., Pannekoucke, O., Lapeyre, C., Sanderson, B., & Thual, O. (2021). Producing realistic climate data with GANs. Nonlinear Processes in Geophysics Discussions, 1-39.

Contents

Introduction

French introduction

En assimilation de données, le problème dit inverse est l'étude de la manière d'estimer les paramètres du modèle à partir de l'observation. En physique, les modèles ne sont jamais une représentation parfaite de la réalité. En général, seuls les paramètres les plus importants sont pris en compte. Les équations du modèle, les conditions aux limites et les conditions initiales peuvent être simplifiées afin de rendre le modèle efficace numériquement. Sachant cela, le modèle n'est pas tout à fait juste et l'incertitude sur la justesse de la sortie du modèle pourrait être corrigée avec des données prises dans la réalité, les observations.

L'assimilation des données d'observation peut se faire en recherchant les paramètres d'un modèle qui produisent un champ ou une sortie qui correspond aux observations. De nombreuses méthodes d'assimilation existent aujourd'hui, mais elles reposent sur de lourdes hypothèses et peuvent varier selon les applications (*e.g.*, dimension de l'espace des paramètres, système chaotique, relation linéaire entre l'observation et les paramètres, difficultés à obtenir suffisamment d'observations, etc.) L'une des principales méthodes d'assimilation de données est le filtre de Kalman [58], dont l'une des premières applications a été le système de navigation de la mission Apollo. En raison de sa capacité à estimer les incertitudes et à les propager par intégration temporelle, la méthode du filtre de Kalman a ensuite été appliquée à de nombreux problèmes inverses.

Cependant, la méthode du filtre de Kalman est optimale pour les problèmes inverses sous des hypothèses fortes telles que la distribution gaussienne des erreurs et la linéarité de la fonction à inverser. Ces hypothèses ne sont pas toujours valables dans les nombreux problèmes où l'assimilation de données est utilisée. De plus, en probabilité bayésienne, principe sur lequel repose l'assimilation de données, il y a un avantage certain à exploiter le maximum d'informations *a priori* pour améliorer la certitude de l'estimation. Les contraintes physiques connues, comme la positivité de la température ou la cohérence entre les paramètres à estimer par exemple, doivent être intégrées dans le cadre de l'assimilation de données pour éviter une estimation qui ne respectent pas les lois de la physique. Ces informations préalables sont généralement intégrées à l'aide de méthodes de paramétrisation, également appelées reparamétrisations. Elles consistent à définir une transformation mathématique des paramètres à estimer en un nouveau jeu de paramètres. Le nouveau jeu de paramètres, s'il est choisi intelligemment, peut implicitement induire la contrainte comme information préalable. Par conséquent, le concept de paramétrisation peut être utile dans les problèmes où l'assimilation de données donne des estimations irréalistes ou non physiques. Mais il n'est pas facile de trouver une transformation de paramètres appropriée qui réponde à toutes les exigences énumérées précédemment.

Pour donner au lecteur une meilleure compréhension de ce que peut être une méthode de paramétrisation, nous donnons un exemple qui sera utile pour la suite de cette introduction. L'exemple de l'enseignement à un enfant de ce qu'est un carré : la première possibilité naïve est de lui présenter de multiples dessins de différents carrés et d'espérer qu'il généralisera assez bien pour le dessiner. La deuxième possibilité est de décomposer le carré en concepts mathématiques comme les angles con-

stants, les lignes parallèles et perpendiculaires, ce qui constitue en quelque sorte une paramétrisation de tous les carrés possibles. Les paramétrisations ne sont pas uniques, le cercle unitaire peut être caractérisé par une fonction mathématique $x^2 + y^2 = 1$, une fonction paramétrique $(\cos(t), \sin(t))$ ou l'ensemble des formes traçable par un compas... Les formes de base peuvent être relativement faciles à paramétrer, mais les concepts plus complexes peuvent être beaucoup plus difficiles.

Récemment, les méthodes d'apprentissage automatique aussi connues sous le nom d'apprentissage profond ont regagné beaucoup d'intérêt pour de multiples raisons, comme le développement de matériel spécifique connu sous le nom de GPU, le développement d'un algorithme de rétro-propagation efficace, etc. Ces méthodes sont capables d'apprendre automatiquement à partir de données, pour réaliser différentes tâches par le biais de l'optimisation ou de la minimisation d'une fonction d'erreur. Un exemple typique est la vision par ordinateur, où l'un des principaux défis, il y a quelques années, était de pouvoir classer des images dans différentes catégories. Avant l'apprentissage automatique, l'une des méthodes consistait à décomposer les images en différents concepts issus du traitement du signal, tels que les composantes de couleur, la détection des bords, etc. Cela s'appelait l'extraction de caractéristiques et constitue également une forme de paramétrisation. Désormais, les algorithmes d'apprentissage profond sont capables de déterminer automatiquement et plus efficacement les caractéristiques à extraire pour une tâche donnée. Une autre percée importante dans l'apprentissage profond a été l'avancée des méthodes génératives profondes, qui consistent à pouvoir générer de nouvelles images à partir de la même distribution que le jeu de données, comme les visages de personnes. Leur capacité à apprendre des images complexes intéresse la communauté de l'assimilation de données pour créer des méthodes de paramétrisation basées sur ces réseaux génératifs profonds.

C'est le sujet de la présente étude : les *generative adversarial networks*, une forme de réseau génératif profond, sont utilisés pour paramétrer deux applications du domaine : la caractérisation des réservoirs de subsurface et les champs atmosphériques équilibrés.

La paramétrisation de générateurs capables d'approximer la distribution d'états réalistes s'est développée dans le domaine depuis l'essor des algorithmes d'apprentissage statistique. Dans ce contexte, nous étudions le *generative adversarial network* (GAN) dont le but est d'échantillonner implicitement la distribution d'un ensemble de données donné.

Le chapitre 1 décrit le contexte des deux domaines d'application. Tout d'abord, il décrit le domaine de la caractérisation des réservoirs souterrains. Il met l'accent sur les méthodes de paramétrisation dans l'assimilation de données basée sur les ensembles pour la distribution spatiale des faciès rocheux. Ensuite, il décrit le domaine de la prévision numérique du climat et souligne la problématique du déséquilibre des champs atmosphériques dû à l'analyse et à la localisation.

Le chapitre 2 présente la théorie de l'assimilation de données depuis le filtre de Kalman jusqu'au lisseur d'ensemble avec assimilation de données multiples pour lequel la paramétrisation GAN est utilisée. Dans ce chapitre, un modèle simplifié unidimensionnel est étudié pour montrer le comportement de l'algorithme d'assimilation de données pour différentes propriétés analytiques de la fonction à inverser.

Le chapitre 3 introduit les principaux concepts de l'apprentissage profond et le développement théorique qui a conduit au modèle GAN et à sa version de Wasserstein. Il mentionne ensuite la bibliographie de l'utilisation des GANs appliqués aux problèmes inverses.

Le chapitre 4 explique comment former le GAN dans le contexte du réservoir souterrain et le développement de métriques pour évaluer la qualité des générations.

Le chapitre 5 applique la paramétrisation du GAN dans l'ES-MDA pour la caractérisation des réservoirs souterrains pour différents cas d'application. Il souligne le problème de déficience de rang et l'utilisation de l'inversion du sous-espace pour atténuer ce problème.

Le chapitre 6 décrit l'entraînement du GAN pour produire des données climatiques réalistes et comprend l'article publié par l'auteur. Il est décrit comme une nouvelle méthode d'initialisation capable de générer un état atmosphérique d'équilibre et de nombreuses autres applications telles que l'augmentation d'ensemble ou le générateur de temps pour l'évaluation des risques.

Le chapitre 7 souligne les méthodes de conditionnement a posteriori de la génération du GAN en utilisant d'autres méthodes d'optimisation sans dérivation telles que la stratégie évolutionnaire d'adaptation de la matrice de covariance (CMA-ES) et le réseau d'inférence.

English introduction

In data assimilation, the so-called inverse problem is the study of how to estimate model parameters from observation. In physics, models are never a perfect representation of reality. Usually, only the most important parameters are taken into account. The model equations, the boundary, and initial conditions can be simplified in order to make the model computationally efficient. Knowing this, the model is not exactly right, and the uncertainty about the righteousness of the model output could be corrected with data taken from the reality, the observations.

Assimilating observation data into a model can be done by searching for the parameters that produce a field/output matching an optimal Bayesian inference between the model and the observations. Many assimilation methods exist today, but they rely on heavy assumptions and can vary for different applications (*e.g.*, Dimension of parameters space, chaotic system, linear relationship between observation and parameters, difficulty to get enough observations, etc.). One of the main data assimilation methods is the Kalman filter [58], one of the first applications of which was for the navigation system of the Apollo mission. Because of its capacity to estimate uncertainties and propagate them through time integration, the Kalman filter method was then applied for numerous inverse problems.

However, the Kalman filter method is optimal for inverse problems under quite strong assumptions such as the Gaussian distribution of errors and the linearity of the function being inverted. These assumptions are not always valid in the numerous problems where data assimilation is used. Moreover, in Bayesian probability, principle on which data assimilation relies there is a definite advantage to leverage the maximum of *a priori* information to improve the certainty of the estimation. Known physical constraints, such as positiveness of temperature or the coherence between the parameters being estimated for example, must be integrated in the data assimilation framework to avoid non-physical estimation. This prior information is usually integrated using parameterization methods, also called reparameterization, that consist in defining a mathematical transformation of the parameters being estimated in a new set of parameters. The new set of parameters, if chosen cleverly, can implicitly induce the different constraint as prior information. Therefore, the concept of parameterization can be useful in problems where data assimilation gives unrealistic or non-physical estimations. But finding a suitable parameter transformation that fits all requirements listed before is not an easy task.

To give the reader more understanding of what a parameterization method can be an example that will be useful for the following of this introduction is given. The example of teaching a child what is a square : the first naive possibility is to present multiple drawings of different squares and hope that he will generalize well enough to draw it. The second possibility is to decompose the square into

mathematical concepts like constant angles, parallel and perpendicular lines which in a way constitutes a parameterization of all the possible squares. Parameterizations are not unique, the unit circle can be characterized by a mathematical function $x^2 + y^2 = 1$, a parametric function $(\cos(t), \sin(t))$ or the drawable form by a compass... Basic shapes can be relatively easy to parameterize but more complex concepts might be much more difficult.

Recently, automatic learning methods known as machine learning or deep learning regained much interest for multiple reasons, such as specific hardware development known as GPUs, development of an efficient backpropagation algorithm, and so on. These methods are able to learn automatically from data, to realize different tasks by way of optimization or minimization of an error function. A typical example is computer vision, where one of the main challenges some years ago was to be able to classify images in different categories. Before automatic learning one of the methods was to decompose images into different concepts coming from signal processing such as color components, edges detection, etc. This was called feature extraction and is also a form of parameterization. Now deep learning algorithms are able to determine automatically and more efficiently which features to extract for a given task. Another important breakthrough in deep learning was the advance in deep generative methods, that consist in being able to generate new images from the same distribution as the dataset, such as people faces. Their ability to learn complex images are of interest in data assimilation community to create parameterization methods based on these deep generative networks.

This is the topic of the present study : generative adversarial networks a form of deep generative network are used to parameterize two domain applications : subsurface reservoir characterization and balanced atmospheric fields.

Chapter 1 describes the context of the two application domains. First, it describes the domain of subsurface characterization and emphasizes on the parameterization methods in ensemble based data assimilation for the spatial distribution of rock facies. Then it describes the numerical weather prediction domain and underlines the problematic of the unbalanced atmospheric field due to analysis and localization.

Chapter 2 presents the data assimilation theory from the Kalman filter to the ensemble smoother with multiple data assimilation for which the GAN parameterization is used. In this chapter a one-dimensional toy model is studied to show the behavior of the data assimilation algorithm for different analytical properties of the forward function.

Chapter 3 introduces the main concepts of deep learning and the theoretical development that lead the GAN model and its Wasserstein version. Then it mentions the bibliography of the use of GANs applied to inverse problems.

Chapter 4 explains how to train the GAN in the context of subsurface reservoir and the development of metrics to assess the quality of the generations.

Chapter 5 applies the GAN parameterization in the ES-MDA for subsurface reservoir characterization for different application cases. It underlines the rank deficiency problem and the use of subspace inversion to alleviate this problem.

Chapter 6 describes the GAN training for producing realistic climate data and includes the author published article. It is described as a new initialization method able to generate balance atmospheric state and numerous other application such as ensemble augmentation or weather generator for risk assessment.

Chapter 7 underlines methods for a posteriori conditioning of the GAN generation using other

derivative-free optimization methods such as covariance matrix adaptation evolutionary strategy (CMA-ES) and inference network.

Chapter 1

Physical context

1.1 History matching of hydrocarbon reservoirs

1.1.1 What is a reservoir ?

Hydrocarbon reservoirs are a limited energetic resource such as oil and/or gas that can be extracted or produced to be transformed into multiple useful products. These products are widely used in modern society such as fuel or plastic material. They are made of porous and permeable rocks deposited by a sedimentation process. Hydrocarbons created deeper in the ground from the transformation of organic beings into petroleum, are moving up to the surface due to pressure and temperature conditions in the subsurface. Then, they are trapped into reservoirs made of porous media and delimited by impermeable material, illustrated in Fig. 1.1.

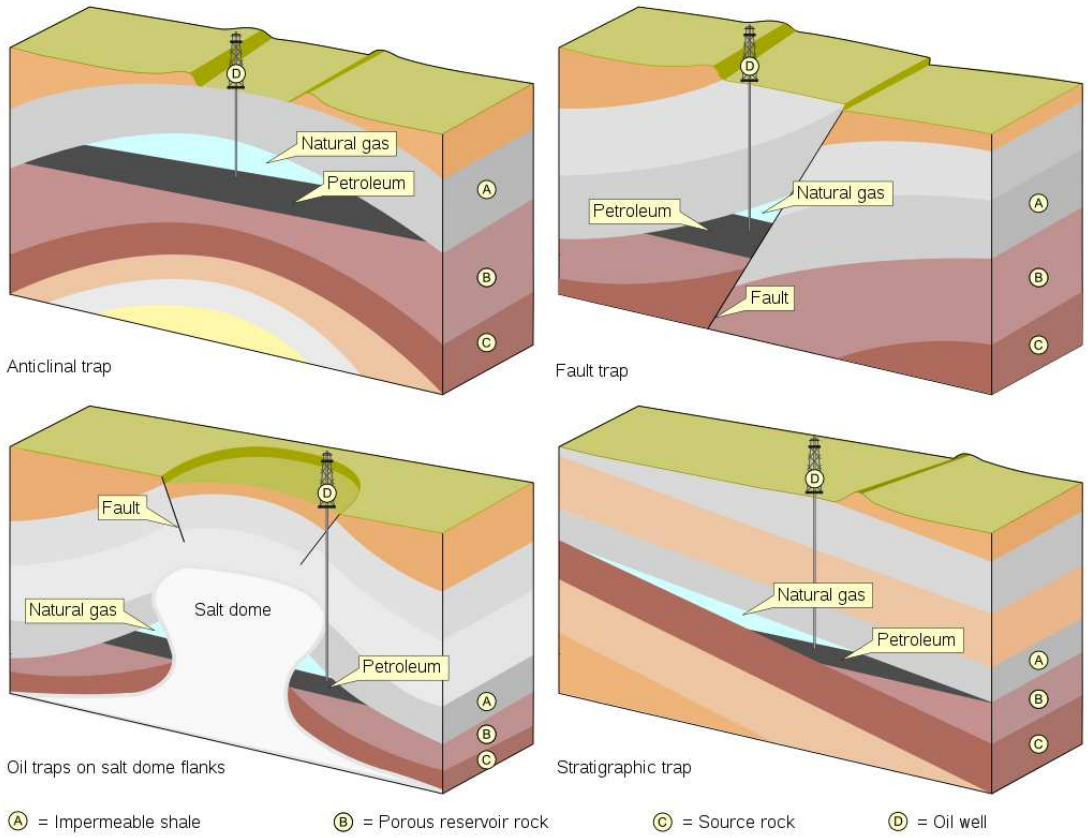


Figure 1.1 – Scheme of a sliced reservoir. Source : https://commons.wikimedia.org/wiki/Category:Petroleum_traps

The principle of oil recovery from reservoirs is to drill extractor and injector wells to porous reservoir rock. Injector wells inject a fluid, usually dilute solutions, to advect hydrocarbons in the direction of the extractor well, see Fig. 1.2.

Reservoirs are complex structures that can be found between 300 and 10000 meters under the surface with different shapes and sizes stretched over tens or hundreds of kilometers. They are made of multiple sedimentary basins with different shapes made by sedimentary deposits such as ancient river channels for example. These connections can also be split by geological phenomena such as faults, represented in the top right panel of Fig. 1.1. These geological phenomena and sedimentary structures are called geological heterogeneities and have an important influence on the fluid flow inside the reservoir and consequently on the amount of oil recovered by the injected fluid. That is why it is important to know the reservoir topology to plan an oil extraction strategy.

The workflow of oil companies is to locate hydrocarbon reservoirs, determine the reservoir topology, extract a maximum of oil and gas at a minimum cost, transform the crude oil, and send it to clients and retailers. This study focuses on the reservoir characterization which aims to determine the reservoir topology in order to assess available resources and to establish a producing strategy. To determine the topology and predict where to drill new wells to optimize the oil extraction, observations are gathered during the early phases of the reservoir exploitation. The goal is to replicate these observations using a reservoir numerical model that represents the reservoir topology and the rock and fluid properties. This model is used as input of a fluid flow numerical simulator that reproduces the response of the reservoir to the fluid injection *e.g.*, the amount of oil extracted. Once the reservoir numerical model replicates the behavior of the studied reservoir, the reservoir is considered characterized and predictions can be made using the numerical model.

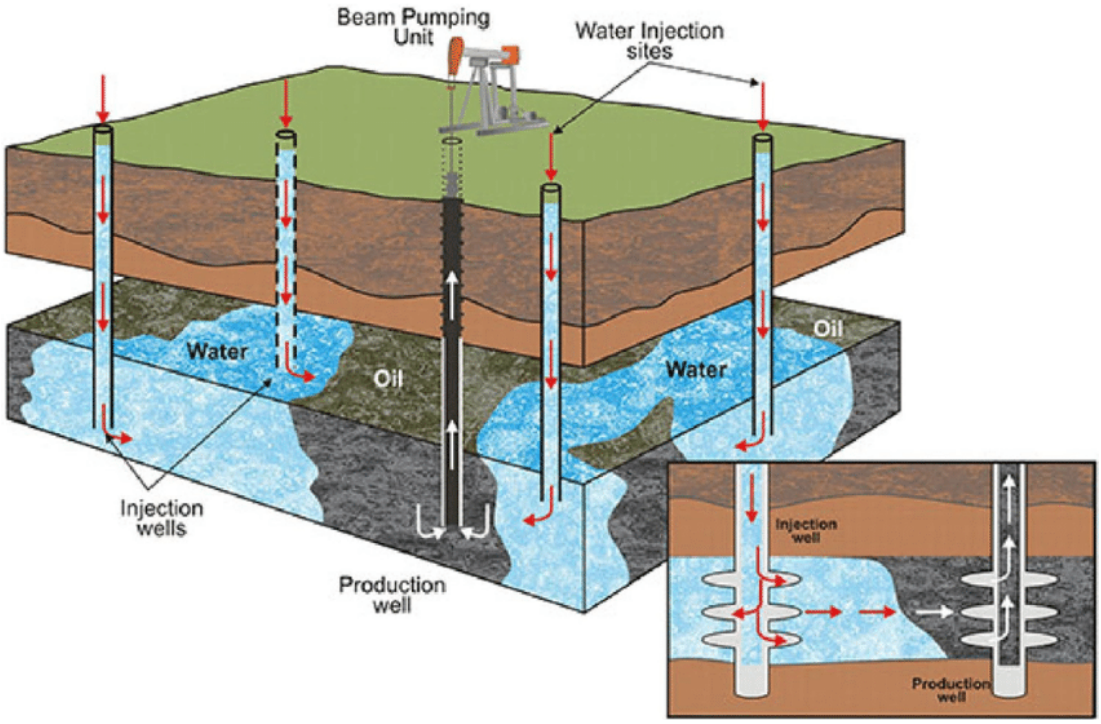


Figure 1.2 – Principle of oil recovery from subsurface reservoirs.

1.1.2 Reservoir numerical model

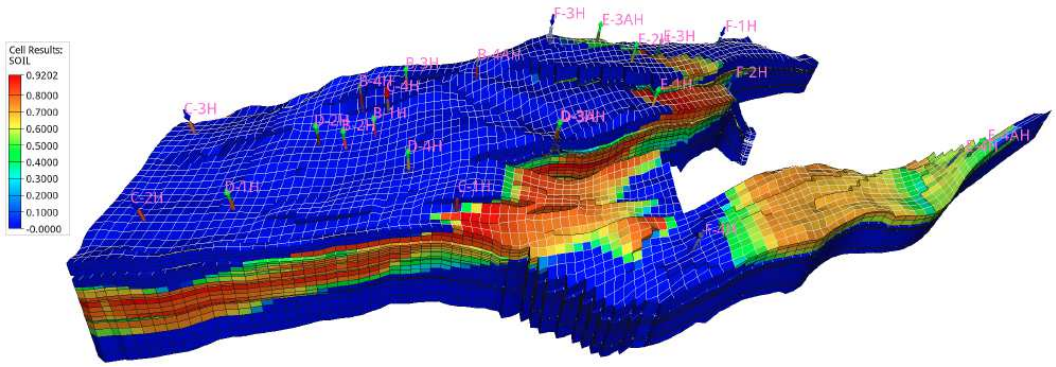


Figure 1.3 – Numerical reservoir model of the Norne field. Z-direction was exaggerated 5 times. [92]

Reservoirs' topology and properties are represented as numerical models, illustrated Fig. 1.3 and their behavior are assessed by running fluid flow simulations inside these numerical models. Numerical models are gridded arrays with a given resolution, each cell of the model contains local petro-physical properties, also called static properties of the reservoir such as porosity (fraction of the volume of voids over the total volume), permeability (ability of a media to allow fluids to pass through it), rock facies (type of rock with particular characteristics such as range of porosity, permeability...), physical properties of the different fluids *e.g.*, density, viscosity, ... When these static properties are defined the numerical model is given as input in a reservoir fluid flow simulator that will replicate the behavior of the reservoir when fluids are injected inside. The fluid flow simulator outputs several data such as the evolution of fluid saturation (fractional measure of the void, *i.e.*, "empty" spaces in a material occupied by a fluid) in each cell through time, but also data at the different wells connected to the reservoir for example the pressure at the bottom of wells or the flow rate of the different injected and extracted fluids.

In the last 40 years the complexity of reservoir numerical models have increased significantly. Nowadays, models can have thousands or millions grid cells that characterize the rock properties at each cell location. Knowing that a reservoir can cover hundreds of square kilometers, a numerical model will always contain errors from different origins. These errors can come from unresolved scales, the positions of heterogeneities because of the model resolution, one cell will represent tens of meters of the reservoir where rock properties will be averaged over the cell volume. This study aims to give a way to reduce the error on the estimation of rock properties and their spatial distributions using observation of the behavior of the exploited reservoir, it is called data assimilation.

1.1.3 Observations and measurements

One of the main difficulties in reservoir characterization is the rarity and the diversity of observations that are used to assess the petro-physical properties of the reservoir. The different types can be split in 2 categories : static data that do not evolve during the exploitation of the field in contrast to dynamic data. In static data, different types of observation can also be measured. Seismic data are obtained by sending seismic waves in the studied area and their echos, that originate from a change of rock property, are analyzed to get an image of the different geological layers in the subsurface. Seismic data can cover a wide area but have a limited resolution, tens of meters for the most precise.

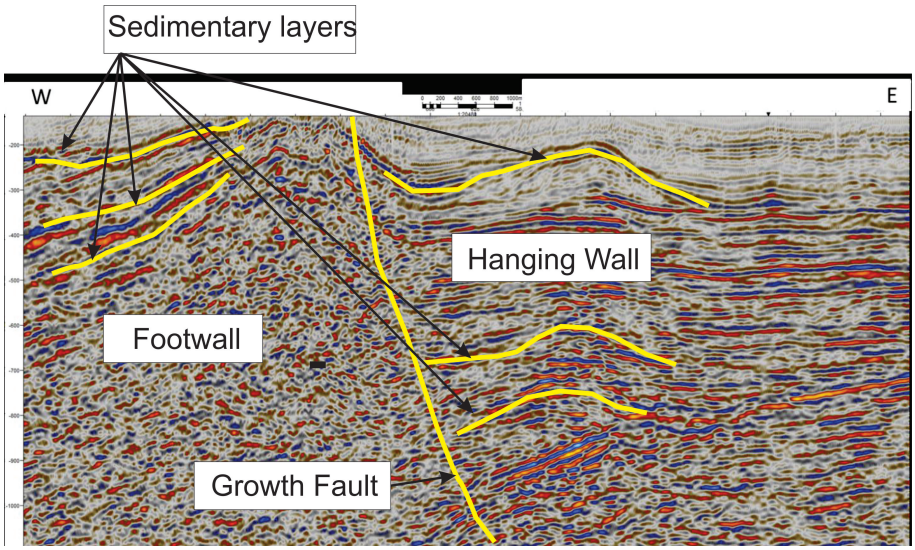


Figure 1.4 – Example of seismic data. Source : https://en.wikipedia.org/wiki/Growth_fault

Another type of static data is given by well logs, when exploratory wells are drilled rock samples are extracted and studied in the laboratory. The analysis can determine the different rock types, their petro-physical properties listed in Sec. 1.1.2. This type of data gives high resolution information but in a very local area (at well bore). Finally, the last type of static data available is the geological expertise from geoscientists that can assess the type of rock and detect some heterogeneities from geographic position of the field for example. These static data are propagated through the field using geostatistics methods detailed in Sec. 1.1.5. Geostatistical methods can measure covariance of two points as a function of their distance which is useful when some observations at different locations are available. However, these methods cannot detect geological heterogeneities such as channels or faults.

Finally, dynamic data are obtained thanks to measurement tools in the wells. Wells are able to measure the physical properties as flow rate, pressure . . . which are highly driven by the petro-physical properties of the reservoir. For example, by injecting water in a reservoir only hydrocarbons will be extracted from the reservoir until a given time when the extracted fluid will be a mix of hydrocarbon and injected water. This is called the water-cut and is information related to the connectivity of the geological heterogeneities and the size of the reservoir. Dynamic data are also the output of a reservoir simulator. By comparing the dynamic data obtained from the real field exploitation and those from simulations using a numerical reservoir model as input, error between observations and predictions can be measured. The measurement of the error allows us to assess the reliability of the numerical model and correct it if necessary. This is the principle of history matching.

1.1.4 History matching

In the history matching problem applied to reservoir characterization, the objective is to estimate reservoir model parameters given observations from a real reservoir being produced, this is the reservoir characterization inverse problem. The parameters and observations being respectively the static and the dynamic properties of the reservoir. After creating a first guess of the parameters, the reservoir simulation gives predictions that are compared with observations. Then, different methods exist to correct the parameters in order to reduce the error between predictions and observations *i.e.*, to solve the inverse problem. Inverse problems are usually solved using data assimilation methods which are used in different domains *e.g.*, trajectory estimation, weather prediction... The algorithmic methods

used depend on the particularity of the forward problem such as the stability of the assimilated phenomenon (chaotic behavior of climate for example), the linearity between parameters and observations, the cost of the execution of the forward problem... In reservoir characterization, the particularities are the following :

1. Non-linear relation between observation data and reservoir model parameters : The relation between the spatial distribution of rock properties and production data such as the oil flow extracted at a well is non-linear.
2. Sparse observations : It is complicated to gather observations on the complete reservoir field. Instead, it is common to have observations at already drilled wells (rock properties but also dynamic data like oil flow rate, pressure...). It is also common to use seismic data that are highly dimensional and necessitate a particular processing.
3. Ill-posed history matching problem : Usually there are more model parameters to determine than independent observation data. Reservoir models can have millions of cells, and for each cell, all the static properties have to be determined. The inverse problem does not have a unique solution and regularization methods are necessary.
4. Non-Gaussian distribution of the parameters : Certain data assimilation methods are theoretically proved and efficient when parameters follow Gaussian distributions. Due to properties of geological heterogeneities this assumption is not generally verified and can result in physically unrealistic estimation. Parameterization methods are necessary, see Sec. 1.1.5.

As said previously, the complexity of numerical reservoir models is constantly increasing because of an increase in computational power and the representation quality that such high dimensional models offer. During a history match, it is always useful to reduce the number of parameters to estimate, it is common to retain the most important ones and discard others for example. Indeed, the rock properties at a given location is dependent on the properties of the surrounding rocks due to the particular shapes of geological objects. This implies that the dimension of the realistic parameters to estimate live in a smaller dimensional space. That is why reservoir engineers use parameterization of reservoir models to reduce the dimension of the inverse problem. It is an important part in the reservoir characterization process that has a great influence on the quality of the results of history matching processes. The Sec. 1.1.5 aims to give an overview of the different methods of parameterization used.

1.1.5 Parameterization methods in reservoir characterization

Geoscientists have local sparse static data, a highly dimensional parameter space and a data assimilation method able to estimate parameters by computing the error between numerical prediction and observations. However, these data assimilation methods rely on different assumptions such as specific parameters distribution and do not take into account the realism of the estimation. It means for example that an estimated reservoir topology produces predictions that match observations, but this topology is not realistic *i.e.*, no natural sedimentation process can produce such topology. One of the solutions is to define a parameterization scheme. Parameterization transforms the parameter space into another one. It is also the occasion to induce some properties into the new parameter space to reduce the complexity of the inverse problem. For example, reducing the dimension of the parameter space, induce probability distributions adapted to data assimilation method assumptions, induce prior knowledge by producing only realistic parameters using the new parameter space.

The principle of determining the value of a variable at a location with respect to other known values at other locations belongs to geostatistics. It is a task closely related to interpolation methods but more

general. The idea behind geostatistics is to consider unknown values at specific reservoir locations as correlated random functions. Random functions can represent uncertainties for the associated variable. In this way unknown variables can be constrained to close known values by defining a cumulative density function (CDF). For example, assuming that an elevation field in a landscape has to be assessed and measurements at different locations are available. Unknown elevation values at locations close to a measurement will be correlated by spatially close measurements. On the contrary, locations where no close measurements are available will have a high uncertainty, consequently its variance will be more important. In this example, the landscape has to be smooth enough for the method to be efficient. On the contrary, if the landscape is heterogeneous because of the presence of canyons for example, more complex methods would be required.

The usage of geostatistics is the most common for parameterization of reservoir models. Due to the progress in the last decades of geostatistics, different parameterization methods have emerged. It usually depends on the method used for adjusting the model parameters and the utilization of the fitted reservoir a posteriori *i.e.*, when the history matching step is terminated. The parameterization is usually a random function depending on spatial coordinates or defined on a grid array representing the rock properties of the reservoir. Another important aspect of parameterization is that estimating the parameters independently can lead to a match of observed data of all complexity without being physically realistic. For example parameters might not respect the physical constraints such as connectivity or physically unrealistic values.

The formalism used in Oliver and Chen [85] will be used where it defines the parameterization by using a basis vector called $\mathbf{q}$ of size N_b to parameterize the set of parameters $\delta\mathbf{m}$ of size N_m. The parameterization can be written as

$$\delta\mathbf{m} = \mathbf{A}\mathbf{q} \tag{1.1}$$

where $\mathbf{A}$ of size $N_m \times N_b$ is the matrix whose columns are the basis vectors and $\mathbf{q} = [q_1, \ldots, q_{N_b}]$ is a column vector that contains the coefficient for each basis vectors. We have $N_m >> N_b$ for an efficient parameterization for the task described above. In the next paragraphs, the progress of the parameterization methods will be tackled from the simpler 1D problem to complex geology.

1.1.5.1 Zonation methods

The most basic method is zonation that consists in defining zones of multiple 1D cells a priori in the reservoir where the basis functions are constant. The function of rock properties is then a piecewise constant function along the reservoir domain. It is a method that has been investigated by Jacquard et al. [56] and Shah et al. [99]. It is known to be efficient to reduce quickly the data misfit between observation data and prediction of the reservoir model estimated. But it fails to give a reduced enough data misfit due to the non-optimal position of the different zones defined a priori. Moreover, the resulting reservoir properties are highly discontinuous which is undesirable. These techniques are used for simplified reservoir models with single phase on one-dimensional problems.

1.1.5.2 Point control

Zonation is a constant function over a domain zone but an interpolation function can also be used. Using an interpolation technique from some well-chosen points is called the point control method, also widely known as kriging. Kriging method [39] defines the Best Linear Unbiased Estimator (BLUE) of parameters values from a linear combination of the measured points (also called control points)

coupled with a covariance model, illustrated Fig. 1.5. For example, let suppose a spatial estimation problem, $(z(\mathbf{x_1}), \ldots, z(\mathbf{x_n}))$ with $\mathbf{x_i} \in \mathbb{R}^2$ where $z(\mathbf{x_i})$ is a measurement of the studied variable such as porosity value at coordinate $\mathbf{x_i}$. Let $z(\mathbf{x_0})$ be the unknown value at location $\mathbf{x_0}$. The estimator of the unknown value can be written :

$$z^*(\mathbf{x_0}) = \sum_{i=1}^{n} \lambda_i z(\mathbf{x_i}) + \left(1 - \sum_{i=1}^{n} \lambda_i\right) \mu_z \tag{1.2}$$

where λ_i are the data weights, μ_z is the mean over the domain of the z variable. In order to have an unbiased estimator it is important that $\Sigma_{i=1}^{n} \lambda_i = 0$ that is how the weight in front of μ_z is defined. This can be understood by imagining a case where data measurements are not giving enough information for the estimation, in this case it is coherent to impose the value of the mean due to the lack of information. Kriging is the optimal way to choose the λ_i in the sense of the minimum variance estimate. Now it is possible to reframe the problem as residual estimation *i.e.*, different from the mean. Simple kriging will be tackled with stationarity assumption which means that the mean is known and does not depend on the location in the domain. It follows that :

$$\begin{aligned} z^*(\mathbf{x_0}) - \mu_z &= \sum_{i=1}^{n} \lambda_i (z(\mathbf{x_i}) - \mu_z) \\ y^*(\mathbf{x_0}) &= \sum_{i=1}^{n} \lambda_i y(\mathbf{x_i}) \qquad \text{given } y = z - \mu \end{aligned} \tag{1.3}$$

where y is the difference with the mean. Then, the last step is to find all the λ_i, as said before kriging method gives the optimal solution *i.e.*, minimize the estimation variance. The estimation variance can be written as :

$$\begin{aligned} \mathbb{E}\left[(y^*(\mathbf{x}) - y(\mathbf{x}))^2\right] &= \mathbb{E}\left[(y^*(\mathbf{x}))^2\right] - 2\mathbb{E}\left[y^*(\mathbf{x}) y(\mathbf{x})\right] + \mathbb{E}\left[y(\mathbf{x})^2\right] \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j \mathbb{E}\left[y(\mathbf{x_i}) y(\mathbf{x_j})\right] - 2\sum_{i=1}^{n} \lambda_i \mathbb{E}\left[y(\mathbf{x}) y(\mathbf{x_i})\right] + C(\mathbf{x_0}, \mathbf{x_0}) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} \lambda_i \lambda_j C(\mathbf{x_i}, \mathbf{x_j}) - 2\sum_{i=1}^{n} \lambda_i C(\mathbf{x_i}, \mathbf{x}) + C(\mathbf{x_0}, \mathbf{x_0}) \end{aligned} \tag{1.4}$$

where $C(\mathbf{x_i}, \mathbf{x_j})$ is the covariance between all data measurement, $C(\mathbf{x_i}, \mathbf{x})$ is the covariance between data measurement and unknown value and $C(\mathbf{x_0}, \mathbf{x_0})$ is the variance of the unknown value. The first term controls the influence of the redundancy of the data measurements, the second one controls the influence of the distance of data measurements from the unknown value and the third term shows the influence of the variance of the unknown value.

By deriving partially with respect to λ_i and setting the derivative equal to 0 (to find the minimum of estimation variance), the following linear system can be written :

$$\sum_{j=1}^{n} \lambda_j C(\mathbf{x_i}, \mathbf{x_j}) = 2C(\mathbf{x}, \mathbf{x_i}) \tag{1.5}$$

By defining an arbitrary covariance model or derived from the data, the linear system can be solved under the assumption of the covariance matrix being positive semi-definite which gives the BLUE,

with the following variance for simple kriging (SK) given by injecting Eq. 1.5 in Eq. 1.4 :

$$\sigma^2_{SK} = C(\mathbf{x_0}, \mathbf{x_0}) - \sum_{i=1}^{n} \lambda_i C(\mathbf{x}, \mathbf{x_i}) \tag{1.6}$$

Covariance model shows that unknown points close to control points will be highly correlated with the value at those control points. However, the covariance decreases rapidly with respect to the distance from these control points. Kriging remains a widely used method, but it is not sufficient to model geological facies. It is a deterministic method, by definition it cannot generate different realizations in order to create an ensemble of possible scenarios and represent uncertainties.

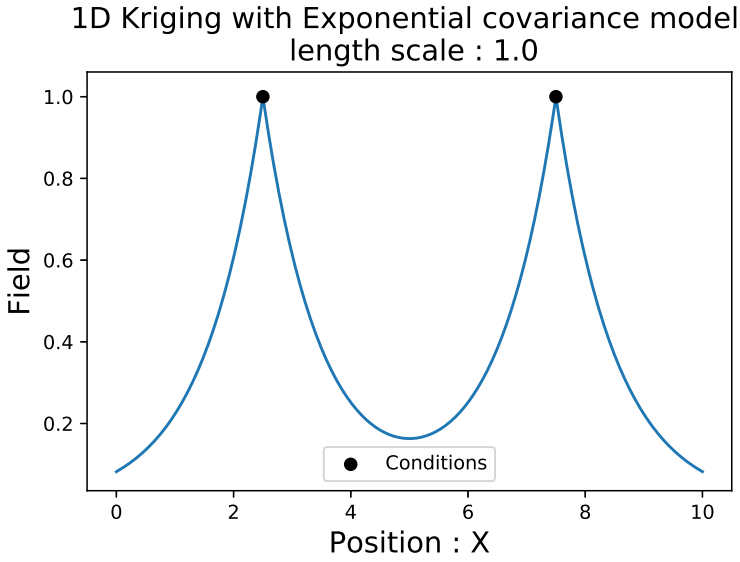


Figure 1.5 – Exponential covariance model of 2 points at position 2.5 and 7.5.

In order to have stochastic simulations, it is necessary to reconsider the problem as a random function problem and more precisely as Gaussian process. This means that the random function over the studied domain has a multivariate Gaussian distribution. Thus, observations are considered as samples from a random function, and the objective is now to determine the 2 first statistical moments of the random functions at unknown locations. The simple kriging estimator $y^\star$ coincides with the conditional expectation $\mathbb{E}\left[y(\mathbf{x_0})|y(\mathbf{x_1}), \ldots, y(\mathbf{x_n})\right]$ due to linear combination of Gaussian distributions. Because of the variance of the mean square error, σ_{SK} does not depend on the estimate $y^\star$, both distributions are uncorrelated, thus independent. The variance of the conditional distribution given $y^\star$ is $\mathbb{E}\left[(y(\mathbf{x_0}) - y^\star(\mathbf{x_0}))^2|y^\star(\mathbf{x_0})\right] = \mathbb{E}\left[(y(\mathbf{x_0}) - y^\star(\mathbf{x_0}))^2\right] = \sigma^2_{SK}$. Finally, the expression of the random function at location $\mathbf{x_0}$ is the distribution : $\mathcal{N}(y^\star(\mathbf{x_0})|\sigma_{SK}(\mathbf{x_0}))$. An inference example of 100 samples using gaussian processes with the covariance model shown in Fig. 1.5 is illustrated in Fig. 1.6.

The position of the pilot points are defined by the user and their positions are not related to data sensitivity. Control points position can be determined by the data sensitivity to the field values but the computational work to determine these control points are significant (see Rodrigues [96]). Unfortunately kriging method and Gaussian processes are regression methods, which imply that the stochastic simulations will be smooth realizations which cannot be appropriate to represent geologic heterogeneities as explained in the landscape analogy at the beginning of Sec. 1.1.5.

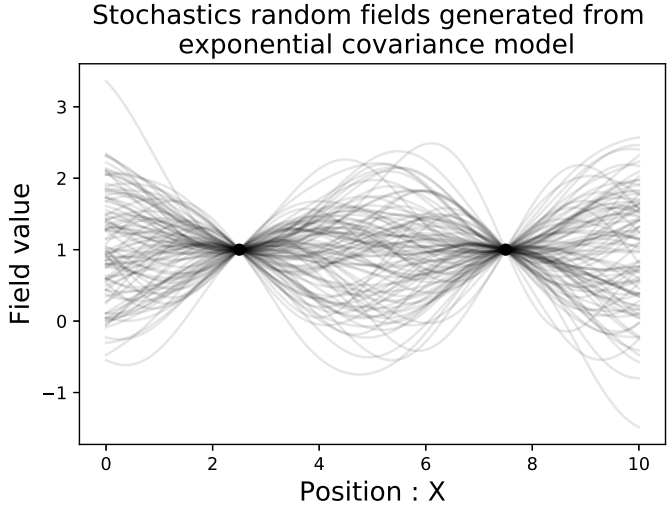


Figure 1.6 – 100 stochastic random fields generated from the covariance model of Fig. 1.5

1.1.5.3 Truncated Gaussian simulation

In data assimilation, it is common to use methods based on a square data misfit and a regularization term that lead to the form of the least squares difference. One of the main assumptions in these methods is that model variables have to be approximately Gaussian or can be transformed to be approximately Gaussian. In order to represent the multiple point statistics that can be found in stochastic earth models, a method was introduced by Matheron et al. [79] to the domain of reservoir simulation, called Truncated Gaussian Simulations (TGS). TGS is still a widely used method to represent stochastic earth models, the next section will describe TGS in order to give an understanding and its pros and cons.

The TGS is a very well-known and still broadly used method to represent spatial distribution of the facies or lithofacies Galli et al. [36]. Reservoir characterization is usually a hierarchical process where it is important to separate the different types of rocks before setting the permeability inside them. Facies determination helps at the characterization of static properties especially far from data measurements.

The principle of the method for a 2D example is to draw two Gaussian random functions Y_1 and Y_2 in the reservoir domain with defined covariance functions illustrated in top panels in Fig. 1.7. The covariance function has to represent the different trends on the different directions of the domain. Then, a truncation map, panel (c) of Fig. 1.7 has to be determined from available data or expert knowledge. For each pixel, the truncation map is the indicator function Eq. 1.7 applied to the random functions values and gives the facies index at the pixel location, illustrated in panel (d) in Fig. 1.7. The truncation map is chosen from facies proportion and represents the transition between facies in the domain *e.g.*, the black facies in the example can never be in contact with the light color facies.

Let F be a random set that represents the facies we want to model in our domain study. An indicator function is associated : $1_F(x)$ where x is the position on the grid that represents the discrete subdivision of the reservoir. Assuming we want to study n facies, we first define a stationary Gaussian random function Y then we define as lithofacies such that :

$$F_i = \{x \in \mathbb{R}^2; s_{i-1} \leq Y_j(x) \leq s_i \text{ with} j \in \{1, 2\}\} \tag{1.7}$$

It follows :

$$d_i = E(1_{F_i}) = P(s_{i-1} \leq Y(x) \leq s_i) = D(s_i) - D(s_{i-1}) \tag{1.8}$$

with d_i the proportion of facies F_i and D the cumulative normal distribution. s_i are the threshold of the truncation map and are defined by :

$$s_i = D^{-1}\left(\sum_{j=1}^{i} d_j\right) \Leftrightarrow d_i = D(s_i) - D(s_{i-1}) \tag{1.9}$$

with $s_0 = -\infty$ and $s_n = +\infty$. The Eq. 1.9 tells us that the thresholds are in bijection with the proportion of facies so a user knowing the facies proportion can have the corresponding thresholds easily.

Such parameterization can be useful in a history matching problem where it is possible to estimate the boundaries of the truncation map [91] instead of the cell values of the reservoir model. However, it remains difficult for particular heterogeneities such as channels characterized by particular continuous patterns. Particular patterns are generally too complex to be determined by two-points statistical methods such as TGS. Higher order geostatistics methods known as multiple point statistics are used to parameterize geological heterogeneities.

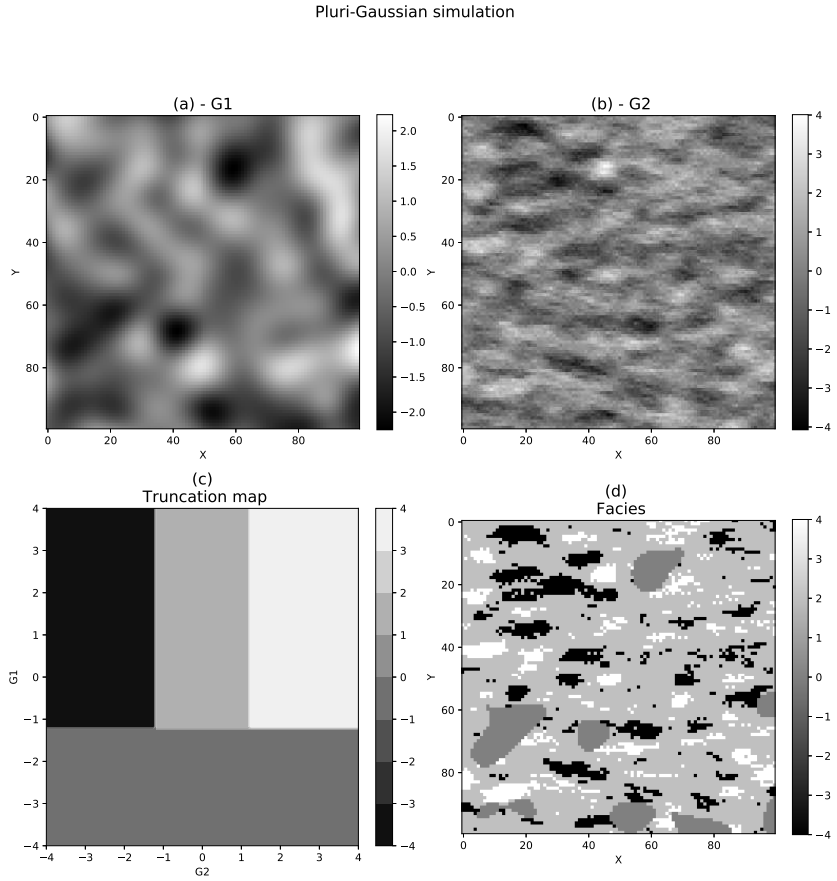


Figure 1.7 – Example of a Pluri-Gaussian Simulation (PGS). Panels (a) and (b) are Gaussian random fields, panel (c) is the truncation map and panel (d) is the result of the TGS

1.1.5.4 Multiple Point Statistics (MPS)

Methods such as kriging and TGS are categorized as two-points statistics methods and are limited by not considering structural properties such as connected patterns over long ranges such as meandering channels, clusters of similar properties etc. These limitations conducted geoscientists to use more complex statistical tools called Multiple Point Statistics (MPS). Using MPS methods requires a certain amount of data that are usually not available in the domain of geosciences, especially in subsurface characterization with only well logs. To solve this problem geoscientists developed object-based methods that use a training image (TI), that is a synthetic image created by experts and containing all the known geostatistical properties that need to be reproduced in the realizations. The MPS algorithm can be divided into 3 categories : pixel-based, object-based and hybrid methods.

Due to the important number of MPS algorithms, this section will give an overview of the literature with references for details. For pixel-based methods, one of the most known algorithm is the Extended Normal Equation Simulation (ENESIM) [44], the idea is based on an extended indicator kriging where at each visited unpredicted point, a pattern is defined from data in the neighborhood of the unpredicted point. Then, it scans the TI in order to find occurrences of such pattern and build a conditional distribution from occurrences. Finally, a sample from the built distribution is placed at the unpredicted point and become a data point, then the algorithm loop to another unknown point. Some limitations of the method have to be underlined. Because it has to scan the TI at each iteration the method is very CPU and memory demanding. It was later modified by using a search tree as a preprocessing to access the scan of the TI without computing it again and called SNESIM [102]. Simulated annealing used by Deutsch [23], Manwart et al. [78], Yeong and Torquato [111] for porous media reconstruction is also widely used, it consists in initializing the domain with random values and hard data. Then add a small perturbation to the initialized domained and compute an objective function. If the objective function was lowered, then the new state is accepted with probability equal to 1, if not the probability acceptance is computed from the difference between objective function value of the original state and perturbed state. One of the main weakness of pixel-based methods is the lack of continuity for geological heterogeneities, Tahmasebi [103] underlined this phenomenon. All these methods require a lot amount of CPU time and are not suited for real test cases containing channel heterogeneities.

The alternative solution is an object-based or pattern-based method. The idea behind such methods is to directly add a patch (group of pixels) from the TI in the realizations. Simulation of pattern (SIMPAT) introduced by Arpat and Caers [4] where a database is built of all the patterns in the TI of a given size. Then, the algorithm visits different locations in the realization through a random path and chooses the most similar pattern of the local environment in the realization computed thanks to a distance between patterns. If no data is present it chooses randomly. This method is very CPU demanding, the result is also very close to the TI and the algorithm tends to underestimate the spatial uncertainties. This principle can be useful for comparing different ensembles of images generated by different algorithms. Other methods were developed such as filter-based simulations (FILTERSIM) or cross-correlation simulations (CCSIM) see Tahmasebi [104].

These MPS methods are difficult to compare with our method due to the lack of open source implementations which is the consequence of their potential importance in commercial context. Additionally, their dependence on a particular TI makes it difficult for the comparison with dataset-based methods. Specific parameterization methods using deep learning will be tackled in Sec. 3.6.2. The present work elaborates a new dataset-based method relying on generative neural networks. Its main asset is to parameterize with few parameters and interesting properties the distribution of images, here reservoir topologies, represented by a dataset in an unsupervised manner.

1.2 Data assimilation for weather forecast

The weather forecast has an important place in the development of data assimilation in the last decades. The necessity for weather forecasts drove scientists to put lots of effort into weather prediction in order to anticipate future extreme weather events such as storms, drought etc. The example of the Presidents' Day snowstorm that hit the United-States of America in February 1979 is a true example of an extreme event wrongly predicted with serious consequences. Even today, the best models struggle to accurately predict these extreme events. Forecast quality was lowered by the sensitivity to small errors in the initial atmosphere state in the northwestern Pacific four days before [52]. This section aims to describe the background of the weather forecast and its remaining challenges.

The Earth climate system is very complex with a lot of interactions between 5 subsystems such as the atmosphere (air), hydrosphere (water), lithosphere (planet surface), biosphere (vegetation and being organisms) and cryosphere (sea ice at the poles). Atmosphere is the most variable component of the earth's climate system in both space and time. Bjerknes [9] referred to the weather forecast as the ultimate problem in meteorology and described an approach to solve it. Its approach is based on two conditions :

1. The present state of the atmosphere must be characterized as accurately as possible.
2. The intrinsic laws, according to which the subsequent states develop out of the preceding ones, must be known.

It describes weather forecast as an initial-value problem with a determinist approach because assertion is made about the complete determination of a future state of the atmosphere from a previous one. Bjerknes [9] subdivided into 3 main tasks to tackle the initial-value problem :

(i) The observation component.

(ii) The diagnostic or analysis component.

(iii) The prognostic component.

The 2 first tasks are related to the characterization of the current state of the atmosphere, condition 1. The third is related to condition 2. The first component mentions the necessity of an observation network of the atmosphere. At each observation location, measurement of variables such as temperature, wind component etc. have to be measured. The distribution of the observation points needs to cover well enough the different space and time scales of characteristic phenomena of the atmospheric circulation. Figure 1.8 shows the wide variety of the space and time scales of the atmospheric circulation, and underlines that such an observation network is theoretical and even now is far from a satisfying sampling of these scales. Daley [22] tells how, along the history, the weather prediction started with sparse local measurements and subjective analysis to a worldwide data collection using complex methods for data selection in order to realize objective estimation of the weather. Even if the number of observations gathered is constantly increasing using satellites, civil aviation, meteorological probes and even animals equipped with probe [97] some places remain under sampled like deep ocean or high atmospheric layers.

The second task is about processing these observations. Analysis is a data assimilation term that refers to the state estimated from the measurements usually on a gridded domain representing the atmosphere in a numerical model. All the dependent variables (mass, temperature, wind humidity...) have to be defined at each grid point. If measurements are not available at certain locations, spatial interpolation or other methods have to be used in order to have a completely defined atmospheric state.

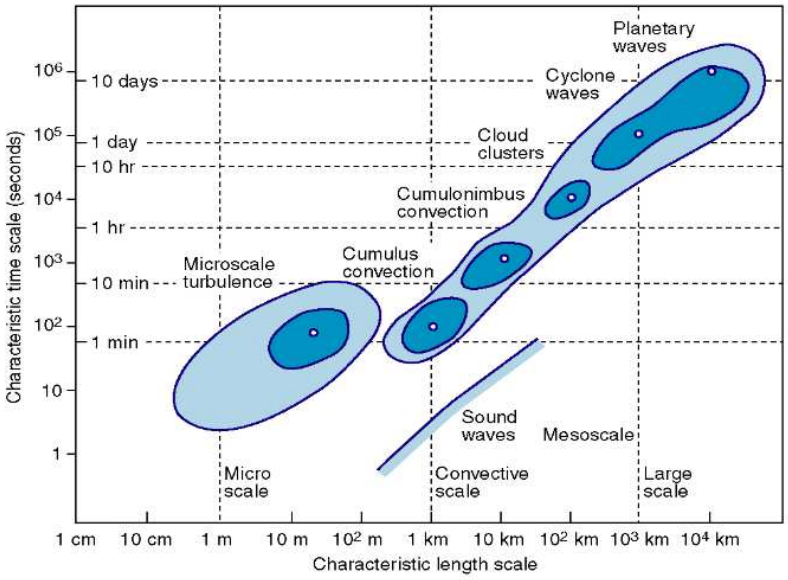


Figure 1.8 – Representation of the different space and time scales of the atmospheric circulation. Source : Owens and Hewson [86]

Finally, prognostic components are obtained by integrating the atmospheric state in time using an atmospheric global circulation model (AGCM) which will be described in the Sec. 1.2.1. One should note that this is a simplification of the weather forecast complete system, this study focuses on atmosphere circulation but lots of climate components have to be coupled with atmosphere circulation such as ocean, sea ice, atmospheric chemistry etc. For sake of simplicity our study will consider all these components as parameterized. The reader should note that in the context of climate, parameterization has another sense than in reservoir characterization. Parameterization of a climate phenomenon means that its influence is represented by a mathematical approximated model instead of using physical equations.

1.2.1 Atmospheric global circulation models

The Earth dynamical system is described as a "pure" fluid *i.e.*, dry air that interacts and is modified by a minor constituent : water in its three phases. It is also subject to several forces from external systems such as its interaction with oceans and heat fluxes from the solar annual cycle. The dynamic of the atmosphere is represented by a numerical model. Atmospheric global circulation models (AGCM) uses a discretized domain over the entire planet with constant angular spacing over the latitudes and longitudes, illustrated Fig. 1.9 in order to simulate the dynamic of the atmosphere. For the height resolution, the grid is made of tens of layers. Such models aim at solving the primitive equations *i.e.*, the Navier-Stokes equations on a rotating sphere with thermodynamics and appropriate approximations. These equations characterize different balances present in the atmosphere :

- Continuity equation representing the conservation of mass.
- Conservation of momentum representing the Navier-Stokes equation on a sphere under the assumptions of low vertical motion compared to horizontal motion and that the size of the fluid layer is small compared to the size of the sphere.
- Thermal energy equation representing the different temperature sources and sinks
- Hydrostatic approximation.

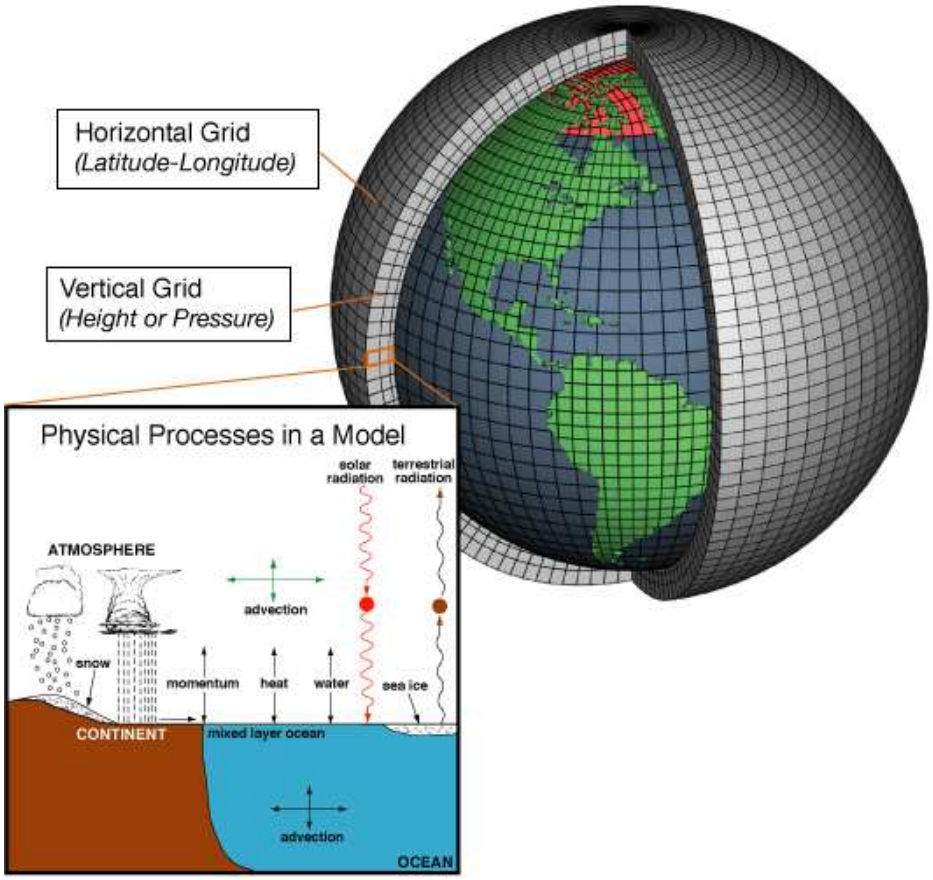


Figure 1.9 – Scheme of a numerical model of the atmosphere.

AGCMs can use finite difference methods or spectral methods to solve the primitive equations over the discretized domain. Some finite difference methods use the grid with constant (for simple numerical models but resolution can change for more complex models) angular spacing that converges to the poles. This type of grid causes computational instabilities due to the shrinking of the grid towards the poles, filtering over the latitude close to the pole can be a solution to leverage this limitation. Spectral method generally uses Gaussian grids that have the advantage to conserve a constant spacing between points over each latitude and an absence of point at the pole. These different resolutions have an important impact on the quality of the forecast by numerical models. The error of the model can be reduced but is unavoidable with the current computational power, the computation of all the necessary variables such as temperature, pressure, velocity components, etc., over such a large domain are computationally demanding. Thus, the resolution of numerical models are limited by computational resources and are not fine enough to take into account phenomena that occur on a small scale that is below the spatial resolution *e.g.*, moist convection or cloud coverage. These phenomena must be handled via mathematical parameterization where their impact on the atmosphere is represented from mathematical models instead of solving the physical equations.

A large variety of numerical atmospheric models are currently in production ; Météo-France for example uses two different models. One AGCM is named ARPEGE and represents the atmosphere of the entire planet. One regional circulation model (RCM) is named AROME and covers metropolitan France with a refined grid and exchanges inputs and outputs at its boundaries with the AGCM model. Moreover, all these models can cooperate by sharing their prediction in order to increase the quality of the numerical weather predictions (NWP) that consist of using AGCM coupled with a data assimilation algorithm to forecast the state of the atmosphere.

1.2.2 Numerical weather prediction

Numerical weather prediction (NWP) is the scientific area where the objective is to predict the weather given the observations of its current state, by taking into account :

- Errors coming from measurements of observations.
- Errors coming from initial state estimation.
- Errors coming from the time integration by the AGCM.

Similarly to the reservoir application of data assimilation, the first step for NWP is to initialize the state space. One should note that the main difference with the reservoir application is that, in NWP, the estimation is on the state of the atmosphere which is also the output of the AGCM. It means that the parameter space is a set of variables that vary through time while history matching deals with constant geological parameters. These two kinds of problems are respectively referenced as state and parameter estimation. NWP requires a sequential data assimilation consisting in realizing the 3 tasks mentioned in Sec.1.2 at regular intervals (see Fig. 1.10) which differs from reservoir parameter estimation. This is due to the chaotic property of the climate system. Chaotic nature of the atmosphere can be described as : close initial conditions can produce very different outputs after a certain time, that is the reason in NWP it is important to gather information and inject it in the model regularly. A direct fundamental consequence given by [74–77] is the limit of about 2 weeks of predictability of the atmosphere given a perfect model and perfect observations. Analysis can cause difficulties due

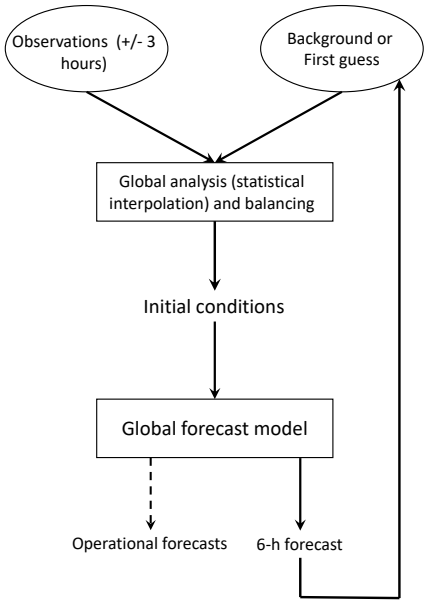


Figure 1.10 – Scheme of data assimilation cycle for 6h analysis. Reproduced from Kalnay [59].

to sparse observations and the complexity of generating a realistic atmospheric state, in the sense of a balanced state from sparse data. Indeed, a non-balanced state can generate non-physical gravity waves. Gravity waves is a natural phenomenon of the atmosphere that occurs when a fluid is displaced from its equilibrium state. This usually allows the atmosphere to maintain or retrieve the geostrophic balance by generating high frequency low amplitude gravity waves that have only a local influence on the atmosphere and low life expectancy. However, an unbalanced analysis of the atmosphere will

also generate gravity waves during time integration but with high amplitudes which will decrease the quality of the prediction by adding a noisy component (see Phillips [89] for further details). Different methods were used in order to remove the noise component of the analysis, a description of the different families of methods used are listed below :

Filtered equations The first method introduced by Charney et al. [19] in order to remove the spurious oscillations was the modification of the set of equations into the set of quasi-geostrophic equations. It is named the filtering method, and involves approximations that are not always verified. Consequently, the filter can also eliminate meteorological information brought by atmosphere observations [22]. This method was widely used in production but due to its limitations to integrate useful information, research focused on another type of solution. Instead of adapting the model to remove these oscillations, the source of the phenomenon was tackled, the unbalanced fields. The new objective was to generate balanced field after correction by the observations this process is called initialization.

Static initialization Another idea proposed by Hinkelmann [51] was to work on the initial state itself. Because oscillations are generated by unbalanced grid points, Hinkelmann [51] proposed to use a balanced wind field deduced from the pressure field as initial state. The price of this approximation was to not take into account wind observations and the remaining noise level in the predictions was still not acceptable. Static initialization refers to all methods that determine an initial physical field without using predictive equations of the model.

Dynamic initialization The dynamic initialization, introduced by Miyakoda and Moyer [81], consists of an initialization of a velocity field and running an AGCM forward and backward in time in order to reach the equilibrium of the velocity field. The method does not require a modification of the observation of the wind fields and output a noise-free wind field and consequently avoid the gravity/inertia waves dampened during the initialization process. But, it is extremely computationally demanding to run the model back and forth and other physically significant motions are damped as well.

1.3 Discussion

Both applications share a common limitation, the lack of a method able to understand the physical constraints in their respective domains. In reservoir characterization, the particular patterns of geological heterogeneities have to be characterized by a set of parameters, optimally living in a space with a limited dimension to perform parameter estimation. In numerical weather prediction, correction of an atmospheric state can produce imbalance. The ability to characterize balance states by a set of parameters could also be useful. Therefore, the development of a generator able to create geological patterns or balance atmospheric states from a set of parameters would be a significant improvement for the limitations encountered in the data assimilation techniques used in both domains.

Determining the physical equations that rule both physical phenomenon, sedimentation process or balancing atmosphere, would be too difficult or too computationally demanding. However, the family of data driven generator methods could be a way to learn the physical constraints without the obligation to determine and solve the physical equations. Another advantage of these techniques is their transferability between different application domains. This work investigates the usefulness of the generative methods able to statistically learn the implicit rules that produce very complex phenomena such as balanced climate states or complex subsurface properties' distribution. It should

be emphasized that this work tackles two application domains to demonstrate its applicability in numerous other domains. The present manuscript was designed to be didactic and underlines the different specificities to take into account when transferred to a new application.

Chapter 2

Gradient-free methods for Inverse Problem.

In the inverse problem the objective is to find the causes of a phenomenon from observations. To solve such a problem, one has to first model a direct problem, noted $\mathcal{M}$, numerically (or experimentally if possible). The model has to simulate the behavior of the phenomenon, which means producing the same effect y outputs for a given set of causes x, it can be written as :

$$\mathcal{M}(x) = y \tag{2.1}$$

The objective is to determine the right set of causes, parameters, state, condition that produces the same values measured on the real phenomenon. The inverse problem is a general formulation that can be used in different contexts such as tomography (applied in medicine, astronomy, geosciences), hydrology, geology and more. As an example one can think of a trajectory estimation algorithm that has to determine the thrust and the direction for a vehicle in order to reach a known position. The model will contain the set of governing equations characterizing how the vehicle moves into space (position, external forces, friction...).

The choice of the method depends on the properties of the studied phenomenon. Usually the most important property is the linearity of the mapping function $\mathcal{M}$. The phenomenon can also be chaotic, and can have a high number of parameters, such as the climate for example. It can also be hard to gather a satisfying amount of observations, the inverse problem can be ill-posed meaning that different sets of parameters can produce the same prediction that fit observations such as in subsurface flow or reservoir characterization. The availability of the inverse of the model in the sense of an inverse model or adjoint that can characterize theoretically the behavior of the inverse phenomenon, is also an important aspect. It can help to choose the appropriate and most efficient method to solve the inverse problem. One of the mainly used methods for inverse problems containing a dynamical aspect is known as data assimilation. It can be separated into two categories : adjoint methods (with an adjoint model that can compute the derivative of $\mathcal{M}$) such optimization methods using gradient descent for example or gradient-free methods (without adjoint model) usually these methods rely on covariance estimation instead of the unavailable gradient ; hybrid methods of these two techniques are also getting more attention, particularly in weather prediction domain.

2.1 Data assimilation

Data assimilation is used in the context of an inverse problem, to estimate the state of a complex system or its initial conditions for example using predictions given by a numerical model and observations. It also estimates the associated uncertainties due to multiple sources. For complex systems such as atmospheric circulation for example, the model contains errors due to the numerical discretization of the problem for example. Observations are also usually sparse and contain errors. The objective of

data assimilation is to find the best estimate taking into account errors and observations. It is indeed, difficult to model accurately such systems and models for different limitations :

1. The physical constraints that rule the model behavior and the state manifold.
2. Ill-posed history matching problem : Usually there are a lot more model parameters to determine than independent observation data. That is why it is common to regularize the problem using prior knowledge.
3. Non-linear relation between data and model parameters.
4. High dimensional parameter space.
5. Sparse observations.

Usually, data assimilation methods can be separated into two steps. First, when an observation is available, the correction of the system state using the information brought by the observation is called the analysis. The second step is the forecast of the corrected state through time until another observation is available. The analysis step can be the source of unrealistic corrections (such as an unbalanced mass-wind field in atmospheric circulation model as mentioned in Sec. 1.2.2) and can lead to deterioration of the forecasted property.

2.1.1 Problem definition

The current section will introduce the notations in Carrassi et al. [17] and theory as basis for the rest of the manuscript. For sake of generality the data assimilation method will be described for both application domains. The atmospheric state will be noted $\mathbf{x} \in \mathbb{R}^{N_x}$. In the oil reservoir domain, the inverse problem is defined as an augmented state estimation, which means that the method estimates the static parameters $\{\mathbf{m_i}\}_{i\in\{0,\dots,N_{stat}\}}$ and the state of the dynamical system $\{\mathbf{p_j}\}_{j\in\{0,\dots,N_{dyn}\}}$. We define one parameter state that combines static and dynamic parameters $\mathbf{x} \in \mathbb{R}^{N_x}$, where $N_x = N_{stat} + N_{dyn}$ is the dimension of the parameter space.

$$\mathbf{x} = \begin{pmatrix} \mathbf{m} \\ \mathbf{p} \end{pmatrix} \tag{2.2}$$

a dynamical model can be defined such as :

$$\mathbf{x}_k = \mathcal{M}_{k-1:k}(\mathbf{x}_{k-1}) + \mathbf{q}_k \tag{2.3}$$

where $\mathbf{x}_k = \mathbf{x}(t_k)$ and t_k is the time index, and $\mathbf{q}_k$ is the model error. $\mathcal{M}$ here represents the dynamical model that integrates the parameters where index $k-1:k$ indicates from time t_{k-1} to t_k. It is a representation of the natural, usually nonlinear process that is simulated. In the reservoir domain, it is a simulator that solves the fluid flow equations inside a porous media and in NWP it is an AGCM. Because simulators are not perfect due to the approximations made or unresolved scales for example, a noise vector $\mathbf{q}$ is added. An observational operator is also defined :

$$\mathbf{y}_{pred;k} = \mathcal{H}_k(\mathbf{x}_k) + \mathbf{e}^o_k \tag{2.4}$$

$$\mathbf{y}_{obs;k} = \mathcal{H}_k(\mathbf{x}^t_k) + \mathbf{e}^o_k \tag{2.5}$$

where $\mathcal{H}$ is the observation operator also often non-linear, $\mathcal{H} : \mathbb{R}^{N_x} \to \mathbb{R}^{N_y}$ and $\mathbf{y}_k = \mathbf{y}(t_k) \in \mathbb{R}^{N_y}$ is the observable variable produced by the model or natural process, of dimension N_y, which is the consequence of given set of parameters $\mathbf{x}_k$. The observations are not always in the same space as the parameters (*e.g.*, radiance measure from satellite to determine temperature in weather forecast). It

can also be a source of error thus another noise vector $\mathbf{e}_k^o$ is added to represent the uncertainties of the measurements. The vector $\mathbf{x}^t$ is the unknown true state model where the observations come from.

Our objective is to estimate the model state given observations and taking into account uncertainties on model parameters and measurements. Usually observations are sparse in space and time, the data assimilation method aims to interpolate these observations thanks to a dynamical model and the observation operator. The initial model state, before assimilation of the observations, can be named the background ($\mathbf{x}^b$) in the case of initial condition estimation, it is also called the forecast ($\mathbf{x}^f$) for sequential state estimation. In the following the forecast notation will be used for generality. This following derivation of the filtering problem is more adapted for the application to NWP because of the sequential aspect of the state estimation. For the parameter estimation application associated with reservoir characterization see the particular case of ensemble smoother Sec. 2.3.3. The new estimate given observations is named the analysis ($\mathbf{x}^a$). Once the model state is updated, it is integrated in time by the dynamical model $\mathcal{M}$ until the next observation. The forecasted analysis is then taken as the new forecast for the next assimilation. This problem formulation is known as a *filtering problem*. The goal is to estimate the model state given the observations. The data assimilation procedure consists in maximizing the likelihood of the state $\mathbf{x}_k$ given observations $\mathbf{y}_{obs;1:k} = \mathbf{y}_{1:k} = (y_1, \ldots, y_k)$. From Bayes' theorem we get:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_{1:k}|\mathbf{x}_k)p(\mathbf{x}_k)}{p(\mathbf{y}_{1:k})} \tag{2.6}$$

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) \propto p(\mathbf{y}_{1:k}|\mathbf{x}_k)p(\mathbf{x}_k) \tag{2.7}$$

which leads to :

$$p(\mathbf{x}|\mathbf{y}_{1:k}) \propto e^{-\mathcal{J}(\mathbf{x})} \quad \text{where} \tag{2.8}$$

$$J(\mathbf{x}_k) = \frac{1}{2}(\mathbf{x}_k - \mathbf{x}_k^f)^T \mathbf{P^f}^{-1}(\mathbf{x}_k - \mathbf{x}_k^f) + \frac{1}{2}[\mathbf{y}_{1:k} - \mathcal{H}(\mathbf{x}_k)]^T \mathbf{R}^{-1}[\mathbf{y}_{1:k} - \mathcal{H}(\mathbf{x}_k)] \tag{2.9}$$

where $\mathbf{P}^f$ is the model state error covariance matrix and $\mathbf{R}$ the measurement error covariance matrix.

2.2 Kalman filter

The principle of the Kalman filter introduced by Kalman [58] relies on two steps that can be seen as a feedback control procedure : the filter estimates the process state at a certain time called prediction step, and then get feedback from noisy measurements for correction called the analysis step. Under certain assumptions, the Kalman filter is the best possible estimator. This section will first derive the Kalman filter under these assumptions and then tackle its extension to more complex cases.

2.2.1 Linear case

The Kalman filter addresses the general problem of state estimation for cases where the dynamical system is linear and errors associated with the model and measurement are independent and follow Gaussian distributions. Let's describe the set of equations of the Kalman filter for a multi-dimensional

Gaussian linear case. The linearity assumption implies :

$$\mathbf{y}_{pred;k} = \mathbf{H}_k \mathbf{x}_k + \mathbf{e}_k^o \tag{2.10}$$

and the Gaussian assumption $\mathbf{e}_k^f \sim \mathcal{N}(0, \mathbf{P}_k^f)$ and $\mathbf{e}_k^o \sim \mathcal{N}(0, \mathbf{R}_k)$. The model state error covariance matrix is then defined by :

$$\mathbf{P}^f = E\left[(\mathbf{x}_k^t - \mathbf{x}_k^f)(\mathbf{x}^t - \mathbf{x}_k^f)^T\right] = E\left[\mathbf{e}_k^o (\mathbf{e}_k^o)^T\right] \tag{2.11}$$

the superscript f refers to the forecast and E describes the expected value. The assimilation error is defined by $\mathbf{x}_k^t - \mathbf{x}_k^a = \mathbf{e}_k^a$. If the assimilation error follows a centered Gaussian distribution, it follows:

$$\mathbf{P}_k^a = E\left[\mathbf{e}_k^a (\mathbf{e}_k^a)^T\right] \tag{2.12}$$

the measurement error covariance matrix is defined by :

$$\mathbf{R}_k = E\left[\mathbf{e}_k^o \mathbf{e}_k^{oT}\right] \tag{2.13}$$

and finally the model error covariance :

$$\mathbf{Q}_k = E\left[\mathbf{q}_k {\mathbf{q}_k}^T\right] \tag{2.14}$$

These assumptions can be summarized by the following equations :

$$\begin{aligned} \overline{\mathbf{e}_k^f} = 0, \ \overline{\mathbf{e}^f \mathbf{e}^{fT}} = \mathbf{P}^f, \\ \overline{\mathbf{e}_k^a} = 0, \ \overline{\mathbf{e}^a \mathbf{e}^{aT}} = \mathbf{P}^a, \\ \overline{\mathbf{e}_k^o} = 0, \ \overline{\mathbf{e}_k^o \mathbf{e}_k^{oT}} = \mathbf{R}_k, \\ \overline{\mathbf{e}_k^f \mathbf{e}_k^{oT}} = 0, \ \overline{\mathbf{q}_k {\mathbf{q}_k}^T} = \mathbf{Q}_k, \end{aligned} \tag{2.15}$$

2.2.1.1 Analysis step

Under the given assumption, the *a posteriori* estimate *i.e.*, the analysis is a linear combination between the *a priori i.e.*, the forecast and a weighted difference between the observation and the prediction of the model such as :

$$\mathbf{x}^a = \mathbf{x}^f + \mathbf{K}(\mathbf{y}_{obs} - \mathbf{H}\mathbf{x}^f) \tag{2.16}$$

The objective is to determine the weight of this combination $\mathbf{K}$, called the Kalman gain. For the analysis step the time index k is dropped for readability.

$$J(\mathbf{x}) = \frac{1}{2} ||\mathbf{x} - \mathbf{x}^a||^2_{(\mathbf{P}^a)^{-1}} + C \tag{2.17}$$

where C is a constant. This leads to :

$$\nabla_{\mathbf{x}} J(\mathbf{x}^a) = (\mathbf{P}^a)^{-1} \tag{2.18}$$

By injecting Eq. 2.16 into Eq. 2.12 for $\mathbf{P}^a$ which yields :

$$\mathbf{P}^a = E\left[\left[(\mathbf{I}-\mathbf{KH})\left(x^t - x^f\right) - K\mathbf{e}^o\right]\left[(\mathbf{I}-\mathbf{KH})\left(x^t - x^f\right) - K\mathbf{e}^o\right]^T\right] \tag{2.19}$$

here $\mathbf{x}^t - \mathbf{x}^f$ is the error of the prior estimate. Because it is not correlated with the measurement noise $\mathbf{e}^o$, it follows :

$$\mathbf{P}^a = (\mathbf{I}-\mathbf{KH})\,E\left[\left(\mathbf{x}^t - \mathbf{x}^f\right)\left(\mathbf{x}^t - \mathbf{x}^f\right)^T\right](\mathbf{I}-\mathbf{KH}) + \mathbf{K}E\left[\mathbf{e}^o\mathbf{e}^{oT}\right]\mathbf{K}^T \tag{2.20}$$

Using Eq. 2.11 and Eq. 2.13 :

$$\mathbf{P}^a = (\mathbf{I}-\mathbf{KH})\,\mathbf{P}^f\,(\mathbf{I}-\mathbf{KH})^T + \mathbf{KRK}^T \tag{2.21}$$

where $\mathbf{P}^f$ is the prior estimate of $\mathbf{P}^a$ Eq. 2.21 is the error covariance update equation. The sum of the diagonal elements of a matrix is the trace of a matrix. In the case of the error covariance matrix the trace is the sum of the mean squared errors. Therefore, the mean squared error may be minimized by minimizing the trace of $\mathbf{P}^a$. The trace of $\mathbf{P}^a$ is first differentiated with respect to $\mathbf{K}$ and the result set to zero in order to find the conditions of this minimum.

$$\mathbf{P}^a = \mathbf{P}^f - \mathbf{KHP}^f - \mathbf{P}^f\mathbf{H}^T\mathbf{K}^T + \mathbf{K}\left(\mathbf{HP}^f\mathbf{H}^T + \mathbf{R}\right)\mathbf{K}^T \tag{2.22}$$

$$\mathrm{Tr}\left[\mathbf{P}^a\right] = \mathrm{Tr}\left[\mathbf{P}^f\right] - 2\,\mathrm{Tr}\left[\mathbf{KHP}^f\right] + \mathrm{Tr}\left[\mathbf{K}\left(\mathbf{HP}^f\mathbf{H}^T + \mathbf{R}\right)\mathbf{K}^T\right] \tag{2.23}$$

where, $\mathrm{Tr}\left[\mathbf{P}\right]$ is the trace of the matrix $\mathbf{P}$ Differentiating with respect to $\mathbf{K}$ gives;

$$\frac{d\,\mathrm{Tr}\left[\mathbf{P}\right]}{d\mathbf{K}} = -2\left(\mathbf{HP}^f\right)^T + 2\mathbf{K}\left(\mathbf{HP}^f\mathbf{H}^T + \mathbf{R}\right) \tag{2.24}$$

setting to zero and re-arranging gives :

$$\left(\mathbf{HP}^f\right)^T = \mathbf{K}\left(\mathbf{HP}^f\mathbf{H}^T + \mathbf{R}\right) \tag{2.25}$$

Now solving for $\mathbf{K}$ gives :

$$\mathbf{K} = \mathbf{P}^f\mathbf{H}^T\left(\mathbf{HP}^f\mathbf{H}^T + \mathbf{R}\right)^{-1} \tag{2.26}$$

which yields to the usual form of the state analysis equation that can be written in multiple ways :

$$\boxed{\mathbf{x}^a = \mathbf{x}^f + \mathbf{P}^f\mathbf{H}^T\left(\mathbf{HP}^f\mathbf{H}^T + \mathbf{R}\right)^{-1}\left(\mathbf{y}_{obs} - \mathbf{Hx}^f\right)} \tag{2.27}$$

and injecting Eq. 2.26 into Eq. 2.22 :

$$\mathbf{P}^a = (\mathbf{I}-\mathbf{KH})\mathbf{P}^f \tag{2.28}$$

Under the hypotheses of linearity for the observation operator and the dynamical model, the analysis step of the Kalman filter is the optimal solution and coincides with the Best Linear Unbiased Estimator (BLUE). The mean and covariance estimation characterizes the probability density thanks to Gaussian linear assumption. Gaussianity assures that the forecast is totally determined by its two first moments and linearity assures that Gaussian distributions integrated in time remain Gaussian. Now that observations have been assimilated, it is necessary to integrate in time the model state until new

observations are available.

2.2.1.2 Forecast step

After the forecast (prior) was corrected by the observations, the next step is to forecast the system state from time t_k to t_{k+1} using the dynamical model $\mathcal{M}_{k:k+1}$ under linearity assumption will be written $\mathbf{M}_{k+1}$. The linearity assumption assures that the estimator is unbiased. The forecast error is :

$$\begin{aligned} \mathbf{e}^f_{k+1} &= \mathbf{x}^f_{k+1} - \mathbf{x}_{k+1} \\ &= \mathbf{M}_{k+1}(\mathbf{x}^a_k - \mathbf{x}_k) - (\mathbf{x}_{k+1} - \mathbf{M}_{k+1}\mathbf{x}_k) \\ &= \mathbf{M}_{k+1}\mathbf{e}^a_k - \mathbf{q_k} \end{aligned} \tag{2.29}$$

from which it is possible to calculate the forecast error covariance matrix :

$$\begin{aligned} \mathbf{P}^f_{k+1} &= E\left[\mathbf{e}^f_k(\mathbf{e}^f_k)^T\right] \\ &= \mathbf{M}_{k+1}\mathbf{P}^a_k\mathbf{M}^T_{k+1} + \mathbf{Q} \end{aligned} \tag{2.30}$$

2.2.2 Kalman filter equations synthesis

As a synthesis, the equations of the Kalman filter for the mean and the covariance are :

Analysis step

$$\mathbf{x}^a_k = \mathbf{x}^f_k + \mathbf{K}_k\left(\mathbf{y}_{obs;k} - \mathbf{H}\mathbf{x}^f_k\right) \tag{2.31}$$

$$\mathbf{K}_k = \mathbf{P}^f_k\mathbf{H}^T\left(\mathbf{H}\mathbf{P}^f_k\mathbf{H}^T + \mathbf{R}_k\right)^{-1} \tag{2.32}$$

$$\mathbf{P}^a_{k+1} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}^f_k \tag{2.33}$$

Forecast step

$$\mathbf{x}^f_{k+1} = \mathcal{M}_{k:k+1}(\mathbf{x}^a_k) + \mathbf{q}_{k+1} \tag{2.34}$$

$$\mathbf{P}^f_{k+1} = \mathbf{M}_{k+1}\mathbf{P}^a_k\mathbf{M}^T_{k+1} + \mathbf{Q} \tag{2.35}$$

2.2.3 Non-linear case

In the case of non-linearity for the evolution model $\mathcal{M}$ and/or the observation operator $\mathcal{H}$ which is usually the case in numerous applications, a new development of equations is necessary. The extended Kalman filter is a method for the non-linear case where :

$$\begin{aligned} \mathbf{y}_{pred;k} &= \mathcal{H}_k\left(\mathbf{x}_k\right) + \mathbf{e}^o_k \\ \mathbf{x}_{k+1} &= \mathcal{M}_k\left(\mathbf{x}_k\right) + \mathbf{q}_k \end{aligned} \tag{2.36}$$

In the non-linear case it is necessary to define the tangent linear models of the observation operator and the evolution model. Tangent linear model (TLM) is a first order approximation of a model using Taylor extension. The TLM of the observation operator and evolution model will be noted respectively $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{M}}$. The linearization is done discarding all the terms with statistical moments greater than the second order considered to have a negligible influence :

Analysis step:

$$\begin{aligned}\mathbf{K}_k &= \left(\tilde{\mathbf{H}}_k \mathbf{P}_k^f\right)^T \left[\tilde{\mathbf{H}}_k \left(\tilde{\mathbf{H}}_k \mathbf{P}_k^f\right)^T + \mathbf{R}_k\right]^{-1} \\ \mathbf{x}_k^a &= \mathbf{x}_k^f + \mathbf{K}_k \left(\mathbf{y}_{obs;k} - \mathcal{H}_k\left(\mathbf{x}_k^f\right)\right) \\ \mathbf{P}_k^a &= \left(\mathbf{I} - \mathbf{K}_k \tilde{\mathbf{H}}_k\right) \mathbf{P}_k^f\end{aligned} \tag{2.37}$$

Forecast step:

$$\begin{aligned}\mathbf{x}_{k+1}^f &= \mathcal{M}_{k,k+1}\left(\mathbf{x}_k^a\right) \\ \mathbf{P}_{k+1}^f &= \tilde{\mathbf{M}}_{k,k+1} \mathbf{P}_k^a \tilde{\mathbf{M}}_{k,k+1}^T + \mathbf{Q}_k\end{aligned} \tag{2.38}$$

With linearization, the system of equations is now closed but by solving it the results will remain an approximation. It was shown that the assumption on negligible influence of higher order statistical moments is not always verified and can lead to an unbounded error growth [29]. Another limitation concerning the Kalman filter and its extended version is the computational cost concerning the storage of error covariance matrix (n^2 unknowns for an n-dimensional model state) and its $2n$ cost for the time integration by the dynamical model. In conclusion, such implementation is useful for a relatively low-dimensional problem.

2.3 Ensemble methods

In Evensen [30], it was shown that instead of using the approximate error covariance equation Eq. 2.37 that is invalid when the dynamical model is highly non-linear. A Monte Carlo method can be used to solve an equation for the time evolution of the probability density of the model state. For a non-linear model where we assume that it is not perfect and contains model errors the following stochastic differential equation can be written :

$$d\mathbf{x}_t = \mathcal{M}(\mathbf{x}_t)dt + \boldsymbol{\sigma}(\mathbf{x}_t)d\mathbf{q}_t \tag{2.39}$$

It states that an increment in time yields to an increment in $\mathbf{x}$ with an influence from the stochastic forcing term $\boldsymbol{\sigma}(\mathbf{x}_t)d\mathbf{q}_t$ representing the model error. The $d\mathbf{q}_t$ term represent a vector of Brownian motion process with covariance $\mathbf{C}_{qq}dt$. $\mathcal{M}$ is a non-linear operator that does not depend on $d\mathbf{q}_t$ so the Itô interpretation of the stochastic differential equation is used (instead of Stratonovich interpretation, see Evensen [30] for further details). Under the assumption that additive Gaussian model errors are forming a Markov process, the Fokker-Planck equation (also known as Kolmogorov's equation) can be

derived :

$$\frac{\partial f}{\partial t} + \sum_i \frac{\partial (m_i f)}{\partial x_i} = \frac{1}{2} \sum_{i,j} \frac{\partial^2 f(\boldsymbol{\sigma} \mathbf{C_{qq}} \boldsymbol{\sigma}^T)_{i,j}}{\partial x_i \partial x_j} \tag{2.40}$$

This equation characterizes the time evolution of the probability density $f(\mathbf{x})$ of the model state. It does not enforce important approximations and can be considered as the fundamental equation for the time evolution of model error statistics. m_i is the component number i of the model operator $\mathcal{M}$ and $\boldsymbol{\sigma} \mathbf{C_{qq}} \boldsymbol{\sigma}^T$ is the covariance matrix for the model error. In the case of a linear Gauss-Markov model where initial distributions are taken from a normal distribution *i.e.*, the model is entirely described by its mean and covariance, the equation 2.40 can be solved for the 2 first statistical moments. It is equivalent to solving the Kalman Filter problem. For a non-linear model the mean and covariance of Kolmogorov's equation are not sufficient to represent the time evolution of $f(\mathbf{x})$ but can represent the mean path and the dispersion about that path. Thus, it is possible to solve approximate equations for the 2 first statistical moments such as in the Extended Kalman filter.

2.3.1 Markov Chain Monte Carlo

In the Ensemble Kalman Filter (EnKF) method [30] a Markov Chain Monte Carlo (MCMC) method is used to solve Eq. 2.40. An ensemble of model states are used to represent the probability density function. Integrating in time these model states according to the model dynamic following Eq. 2.39 yields an ensemble prediction that is equivalent to solving Eq. 2.40. The main advantage of the method is to avoid the use of closing approximations such as in the extended Kalman filter because the non-linear terms are represented by the dynamical model. Since with MCMC the error covariance matrix will now be empirically estimated from the different ensemble members, the estimation error will decrease proportionally to $\frac{1}{\sqrt{N_{ens}}}$ for N_{ens} model states of dimension N_x. These ensemble members can be represented as a particle cloud in an m-dimensional space. See the $2D$ example in Fig. 2.1. These particles can be represented by a probability density function when N_{ens} goes to infinity such as :

$$f(\mathbf{x}) = \frac{dN_{ens}}{N_{ens}} \tag{2.41}$$

Where dN_{ens} is the number of particles in a small unit of volume and N_{ens} the total number of particles. In this way an estimation of the statistical moments of the probability density characterized by the ensemble is accessible at any time.

2.3.2 Ensemble Kalman Filter

Using the MCMC to solve Kolmogorov's equation Eq. 2.40, an analysis scheme can be derived. An ensemble observation vector is defined as a random variable :

$$\mathbf{y}_{obs;j} = \mathbf{y}_{obs} + \mathbf{e}^o_j \tag{2.42}$$

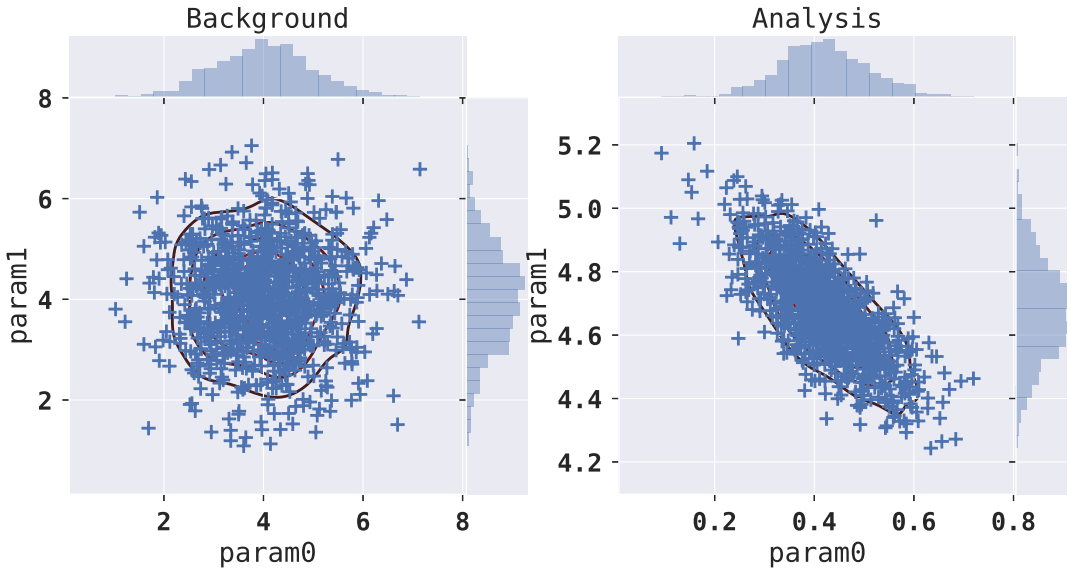


Figure 2.1 – Evolution in parameter space of the model states for 2D case describe in 2.5

here j is the index of ensemble members and $\mathbf{e}^o$ is a noise vector with 0 mean. The definition of the ensemble error covariance matrix of measurements follows :

$$\mathbf{R}_e = \overline{\mathbf{e}^o \mathbf{e}^{oT}} \tag{2.43}$$

$\mathbf{R}$ converges to the error covariance matrix of the Kalman filter when N_{ens} goes to infinity. With a large enough ensemble, the sampling error introduced in the ensemble error matrix of measurement can be less than the initial uncertainty in the exact $\mathbf{R}$. Finally, the forecast step of the error covariance matrix is done using the dynamical model. The analysis step of the ensemble Kalman filter is :

$$\mathbf{x}_j^a = \mathbf{x}_j^f + \mathbf{P}_e^f \mathbf{H}^T \left(\mathbf{H} \mathbf{P}_e^f \mathbf{H}^T + \mathbf{R}_e \right)^{-1} \left(\mathbf{y}_{obs;j} - \mathbf{y}_{pred;j} \right) \tag{2.44}$$

$\mathbf{y}_{pred;j} = \mathbf{H}(x_j^f)$ is the prediction by the dynamical model applied to the forecasted model state $\mathbf{x}_j^f$. Equation 2.44 is an approximation of the update of each model state that constitutes the ensemble. It should be underlined that in the case of a number of measurements greater than the number of ensemble members, the matrix $\mathbf{H}\mathbf{P}_e^f\mathbf{H}^T + \mathbf{R}_e$ can be singular and it will be necessary to use a subspace inversion method described in Sec. 2.3.5. Note that Eq. 2.44 can be rewritten for the mean of the ensemble

$$\overline{\mathbf{x}_j^a} = \overline{\mathbf{x}_j^f} + \mathbf{P}_e^f \mathbf{H}^T \left(\mathbf{H} \mathbf{P}_e^f \mathbf{H}^T + \mathbf{R}_e \right)^{-1} \left(\mathbf{y}_{obs} - \mathbf{H}(\overline{\mathbf{x}_j^f}) \right) \tag{2.45}$$

The mean of the analysis $\overline{\mathbf{x}_j^a}$ represents the best estimate of the ensemble, but usually the form of the Eq. 2.44 is preferred because the analyzed ensemble mean is still accessible at low computational cost. The covariance of the ensemble indicates the uncertainty of the parameter estimation. The analysis scheme is equivalent to the Kalman filter, except the approximation of error covariance matrix from

the ensemble. It should be noted that there is no approximation of the linearity of the model because the non-linear terms are considered when the model states are integrated in time by the non-linear dynamical model. This characterizes the benefit of the ensemble Kalman method. However, it is still solved for the 2 first moments of the model state probability density function (pdf), which means that if distributions are too far from a Gaussian distribution this approximation is not verified. Miller et al. [80] explains that non-linear dynamics can transform Gaussian parameter distribution into non-Gaussian distributions that will consequently not be entirely defined by their 2 first statistical moments. As a consequence, the results could be impacted.

2.3.3 Ensemble Smoother

The Ensemble Smoother [107] is different in the way that the method uses all the observations, in the sense of past and present observations, to fit its target. This method corresponds to parameter estimation such as the history matching performed in reservoir characterization. A new dynamical model can be defined :

$$\mathbf{x}_k = \mathcal{M}_{1:k}(\mathbf{x}_1) + \mathbf{q}_k \tag{2.46}$$

and a new observational operator :

$$\mathbf{y}_{pred;1:k} = \mathcal{H}_{1:k}(\mathbf{x}_1) + \mathbf{e^o}_k \tag{2.47}$$

$$\mathbf{y}_{obs} = \mathcal{H}_{1:k}(\mathbf{x}_1^t) + \mathbf{e^o}_k \tag{2.48}$$

It means that instead of taking into account the observation at time t_k, all the observations from the beginning until time t_l where $l \geq k$ in order to estimate the parameters at time t_k of the model. Finally, we reframe the estimation problem such as :

$$p(\mathbf{x}_k|\mathbf{y}_{1:l}) = \frac{p(\mathbf{y}_{1:l}|\mathbf{x}_k)p(\mathbf{x}_k)}{p(\mathbf{y}_{1:l})} \tag{2.49}$$

$$p(\mathbf{x}_k|\mathbf{y}_{1:l}) \propto p(\mathbf{y}_{1:l}|\mathbf{x}_k)p(\mathbf{x}_k) \tag{2.50}$$

Using the ensemble smoother problem formulation yields to :

$$\mathbf{x}_k^a = \mathbf{x}_k^b + \mathbf{B}_e\mathbf{H}^T\left(\mathbf{H}\mathbf{B}_e\mathbf{H}^T + \mathbf{R}_e\right)^{-1}\left(\mathbf{y}_{obs;1:l} - \mathbf{y}_{pred;1:l}\right) \tag{2.51}$$

the ensemble index was dropped for readability. $\mathbf{B}$ is the parameter error covariance matrix when a static problem is considered. Here we do not consider the forecast of the state error covariance matrix. Thus, in the present work the notation background of the initial state is preferred. The parameter error covariance is then noted $\mathbf{B}$ and the parameters before analysis are noted $\mathbf{x}_k^b$.

Discussion

Here the Ensemble Smoother (EnKS) removes the sequential aspect of the Kalman filter to update the model state taking into account all the observations available simultaneously. It allows initial parameter estimation also called history matching by ensemble methods that will be the application of the present work in the context of reservoir characterization. This change of formalism is analog to the differences between 3D and 4D variational methods widely used in numerical weather prediction.

3D-VAR method tries to determine the best estimate for observations of a given time whereas 4D-VAR tries to find the best estimate given observation on a given time window and propagates back this information to the initial state of the time window [73]. This smoother can result in an unacceptable data mismatch. Because of non-linear phenomenon, the parameters update can be underestimated or overestimated. This phenomenon was reproduced and explained in a simple case in Sec. 2.5.2. To remove this drawback it is possible to do multiple data assimilation [26] using the ensemble smoother with an inflated error covariance matrix of measurements and a disturbed observation vector. The objective in this method is to be equivalent to a single data assimilation with an ensemble Kalman smoother for the linear Gaussian case by proceeding multiple small updates instead of a unique one.

It should be noted that the Ensemble Smoother is different from the Ensemble Kalman Smoother Evensen and Van Leeuwen [32]. It is a version of a sequential Ensemble Smoother where instead of updating the parameter states from all the observations it recursively considers more and more observations and realizes several updates during this recursion. This method will not be discussed in this study because it is not applicable in the context of reservoir history matching.

2.3.4 Ensemble smoother with Multiple Data assimilation

Emerick and Reynolds [27] shows that Ensemble Smoother with Multiple Data Assimilation (ES-MDA) equations can be written as :

$$\mathbf{x}^a_{k;i} = \mathbf{x}^b_{k;i} + \mathbf{B}_{e;i}\mathbf{H}^T \left(\mathbf{H}\mathbf{B}_{e;i}\mathbf{H}^T + \alpha_i \mathbf{R}_e\right)^{-1} \left(\tilde{\mathbf{y}}_{obs;1:l} - \mathbf{y}_{pred;1:l}\right) \tag{2.52}$$

$\tilde{\mathbf{y}}_{obs;1:l}$ is the perturbed observation vector such as $\tilde{\mathbf{y}}_{obs;1:l} = \mathbf{y}_{obs;1:l} + \sqrt{\alpha_i}\mathbf{R}^{1/2}\boldsymbol{\eta}$ where $\boldsymbol{\eta} \sim \mathcal{N}(0, \mathbf{I})$. The index i stands for the multiple data assimilation (MDA) iteration index. Equation. 2.52 is the equation for one iteration of the MDA procedure. In order to do the complete method it is necessary to replace the background model parameter $\mathbf{x}^b$ by the analysis $\mathbf{x}^a$, add a resampled noise vector to observations and update α and apply again the Eq. 2.52. A condition is imposed on α_i in order to respect the equivalence with the single data assimilation with an ensemble smoother for a linear Gaussian case.

$$\sum_{i=0}^{N_a} \frac{1}{\alpha_i} = 1 \tag{2.53}$$

In Emerick and Reynolds [26] it was shown that choosing a decreasing α over the MDA iterations improves only slightly the results. In our study, α is chosen constant for the different ESMDA iterations for simplicity.

2.3.5 Subspace inversion

The subspace inversion method [31] was created in the case where the dimension of the observation vector is greater than the dimension of the ensemble. Kepert [61] shows that in the case where $N_{ens} \leq N_{obs}$, the analysis will be rank deficient resulting in an ensemble collapsing to a single ensemble member. In reservoir characterization, such a case happens in most of the cases, when seismic data are assimilated for example. Subspace inversion is usually coupled with a rescaling step of the measurements [26]. The goal of this method is to avoid ensemble collapse phenomenon and reduce the cost of the matrix inversion Eq. 2.52 when it is necessary using a singular value decomposition (SVD). The step of the method is described below. The matrix that holds the predictions of the ensemble

perturbation is defined Eq. 2.54 :

$$\Delta\mathbf{D} = \left(\mathbf{D}_k - \overline{\mathbf{D}}\right) \tag{2.54}$$

Where $\overline{\mathbf{D}}$ is the mean over the ensemble of predictions. This matrix can be rescaled by dividing the deviation of the predictions by the uncertainty of each observation point.

$$\Delta\mathbf{Z} = \left(\mathbf{Z}_k - \overline{\mathbf{Z}}\right) \tag{2.55}$$

Where $\overline{\mathbf{Z}}$ is the mean over the ensemble of parameters. We define the matrix to invert in the Kalman gain expression as :

$$\mathbf{C} = (N_{ens} - 1)\left(\mathbf{H}\mathbf{B}_{e;i}\mathbf{H}^T + \alpha_i\mathbf{R}_e\right) \tag{2.56}$$

By injecting Eq. 2.54, 2.55 in Eq. 2.56 :

$$\mathbf{C} = \Delta\mathbf{D}(\Delta\mathbf{D})^T + (N_{ens} - 1)\mathbf{R} \tag{2.57}$$

Truncated singular value decomposition of the prediction deviation can be written as :

$$\Delta\mathbf{D} \approx \mathbf{U}\mathbf{S}\mathbf{V} \tag{2.58}$$

Where $\mathbf{S}$ is a diagonal matrix of eigenvalues of $\mathbf{D}$ in decreasing order. The SVD procedure can be truncated in order to keep only the most important eigenvalues, if no truncation is done equality instead of approximation should be written in Eq. 2.58. If the truncation is done to keep N_r the largest singular values : $\mathbf{U}$ is an $N_{obs} \times N_r$ matrix, $\mathbf{S}$ is an $N_r \times N_r$ matrix and $\mathbf{V}$ is an $N_r \times N_{ens}$.

$$\begin{aligned}\mathbf{C} &\approx \left[\mathbf{U}\mathbf{S}\mathbf{S}^T\mathbf{U}^T + (N_{ens} - 1)\,\mathbf{R}\right] \\ &\approx \mathbf{U}\mathbf{S}\left[\mathbf{I}_{N_r} + (N_{ens} - 1)\mathbf{S}^{-1}\mathbf{U}^T\mathbf{R}\mathbf{U}\mathbf{S}^{-1}\right]\mathbf{S}\mathbf{U}^T\end{aligned} \tag{2.59}$$

Where $\mathbf{I}_{N_r}$ is the identity matrix of size $N_r \times N_r$ resulting from the approximation $\mathbf{V}\mathbf{V}^T = \mathbf{I}_{N_r}$. We define :

$$\mathbf{X} = (N_{ens} - 1)\mathbf{S}^{-1}\mathbf{U}^T\mathbf{R}\mathbf{U}\mathbf{S}^{-1} \tag{2.60}$$

And because $\mathbf{X}$ is positive semi-definite, it can be decomposed as $\mathbf{X} = \mathbf{Z}\mathbf{\Gamma}\mathbf{Z}^T$ Finally injecting Eq. 2.60 in Eq. 2.59 :

$$\mathbf{C} \approx (\mathbf{U}\mathbf{S}\mathbf{Z})\left[\mathbf{I}_{N_r} + \mathbf{\Gamma}\right](\mathbf{U}\mathbf{S}\mathbf{Z})^T \tag{2.61}$$

Which can be easily inverted :

$$\mathbf{C}^{-1} \approx \left(\mathbf{U}\mathbf{S}^{-1}\mathbf{Z}\right)\left[\mathbf{I}_{N_r} + \mathbf{\Gamma}\right]^{-1}\left(\mathbf{U}\mathbf{S}^{-1}\mathbf{Z}\right)^T \tag{2.62}$$

Because $\mathbf{I}_{\mathbf{N_r}}$ and $\mathbf{\Gamma}$ are diagonal matrices, their inversion is straightforward. Moreover, the truncated singular value decomposition of $\Delta\mathbf{D}$ controls the reduction of uncertainty in the parameter space by EnKS. It will be demonstrated that this property can be useful to avoid ensemble collapse.

2.4 Limitations in data assimilation

Data assimilation methods are so widely used across diverse scientific domains that it is important to separate the general limitations from applied limitations. First example of a recurrent limitation is the usage of prior information. Prior information also called *a priori* helps to constrain the problem in order to make it well-posed as a regularization method. The example for history matching is the parameterization method. Performing EnKS for estimating every cell of the numerical reservoir model will give non-physical spatial distribution. A priori information such as known structure and pattern of geological patterns has to be injected in the data assimilation workflow.

A good estimate of uncertainty associated with the first guess (*i.e.*, first ensemble) is also necessary to solve data assimilation problems. Because it is difficult to generate physically realistic states in some domains such as NWP, where an AGCM has to be run for a given time to reach a balanced atmospheric state, data assimilation methods are penalized by the lack of useful information.

In this research we have focused on the usage of data-driven methods and more specifically deep generative methods that are able to learn statistical features at a relatively low computational cost. This work will focus on two applications of this method, Numerical Weather Prediction (NWP) and Reservoir Characterization (RC) that share similar data assimilation limitations. The research questions addressed in this study are :

- Can generative neural networks be used as prior knowledge in data assimilation methods ?
- Can they be used as a mapping to respect the Gaussian assumption ?

The reader should keep in mind the fact that these application cases are just examples and this method could be used for every application where prior information are available as data of multiple forms.

2.5 Toy model

To give an overview of the ES-MDA algorithm, it is interesting to see how it behaves in simple cases. In this section, some results on toy models will be investigated to see advantages and limitations of this method. These will stay relevant when applied on complex cases such as history matching in hydrocarbon reservoirs.

In the example, multiple experiments using one-dimensional mathematical functions with different analytical properties representing the dynamical function will be used. These functions take as input a parameter and output the dynamical response of the model *i.e.*, a prediction value. The reader can think of the following analogy where : parameters of the dynamical function represent the spatial distribution of the rock properties. The dynamical function is the reservoir simulator that computes the model predictions given the parameters that are the dynamical response of the reservoir *e.g.*, the pressure or the fluid flow of oil at the different wells. Ensemble Kalman smoother will first be applied to a linear problem, then on a non-linear monotonic problem and finally on a non-linear, non-monotonic problem. The performance of classical ensemble smoother and ESMDA will be compared.

2.5.1 Linear dynamical function

The first experiment is an Ensemble Kalman smoother applied on a one-dimensional linear function $\mathcal{H}$ (Eq. 2.63). This case could be the application of the pressure estimation from the height of a water column linked by the following linear equation : $z = \frac{1}{\rho g}P - \frac{1}{\rho g}P_0$. Where P is the pressure applied at the height z, P_0 is the atmospheric pressure, ρ the fluid density and g the gravitational acceleration. An observation of the water height is available and the objective is to find the pressure value corresponding to the water height. The dynamical function represents our simulator that is considered perfect.

$$\begin{aligned} \mathcal{H}\colon Parameter\ Space &\mapsto Observation\ Space \\ x \quad &\mapsto \quad y = \mathcal{H}(x) = 2x + 4 \end{aligned} \tag{2.63}$$

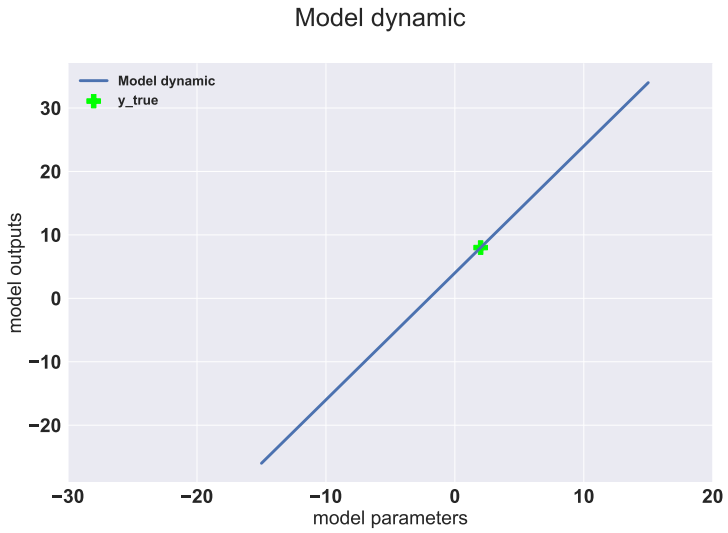


Figure 2.2 – Linear model dynamical function. The green cross represents the measured value $y = 8$ corresponding to the parameters that has to be retrieved $x = 2$.

Figure 2.2 shows the dynamical function chosen for the toy model, the green cross shows the observation value on the vertical axis $y = 8$ and the parameter value on the horizontal axis $x = 2$ that needs to be retrieved by the assimilation algorithm. Gaussian noise was added to the observation to represent the measurement error, the noise was drawn in the normal distribution $\mathcal{N}(0, 0.05)$. The blue line represents the set of images of $\mathcal{H}$ with respect to the parameter space. The dynamical function is linear and monotonic which is an optimal use case for the Ensemble Kalman Smoother (EnKS). The smoothing term here is used because of the absence of time dynamic in the model, it is considered that all the observations are assimilated at once (here only one observation is available for one parameter value.)

An EnKS analysis was done which is equivalent to a single iteration of the ESMDA algorithm, case where $\alpha = 1$ in Eq. 2.64 *i.e.*, $\tilde{\mathbf{y}}_{obs}$ is the perturbed observation vector defined by $\tilde{\mathbf{y}}_{obs;1:l} = \mathbf{y}_{obs;1:l} + \mathbf{R}^{1/2}\boldsymbol{\eta}$ that simulates the error of the measurements.

$$\mathbf{x}_k^a = \mathbf{x}_k^f + \mathbf{B}_e\mathbf{H}^T\left(\mathbf{H}\mathbf{B}_e\mathbf{H}^T + \mathbf{R}_e\right)^{-1}\left(\tilde{\mathbf{y}}_{obs} - \mathbf{y}_{pred}\right) \tag{2.64}$$

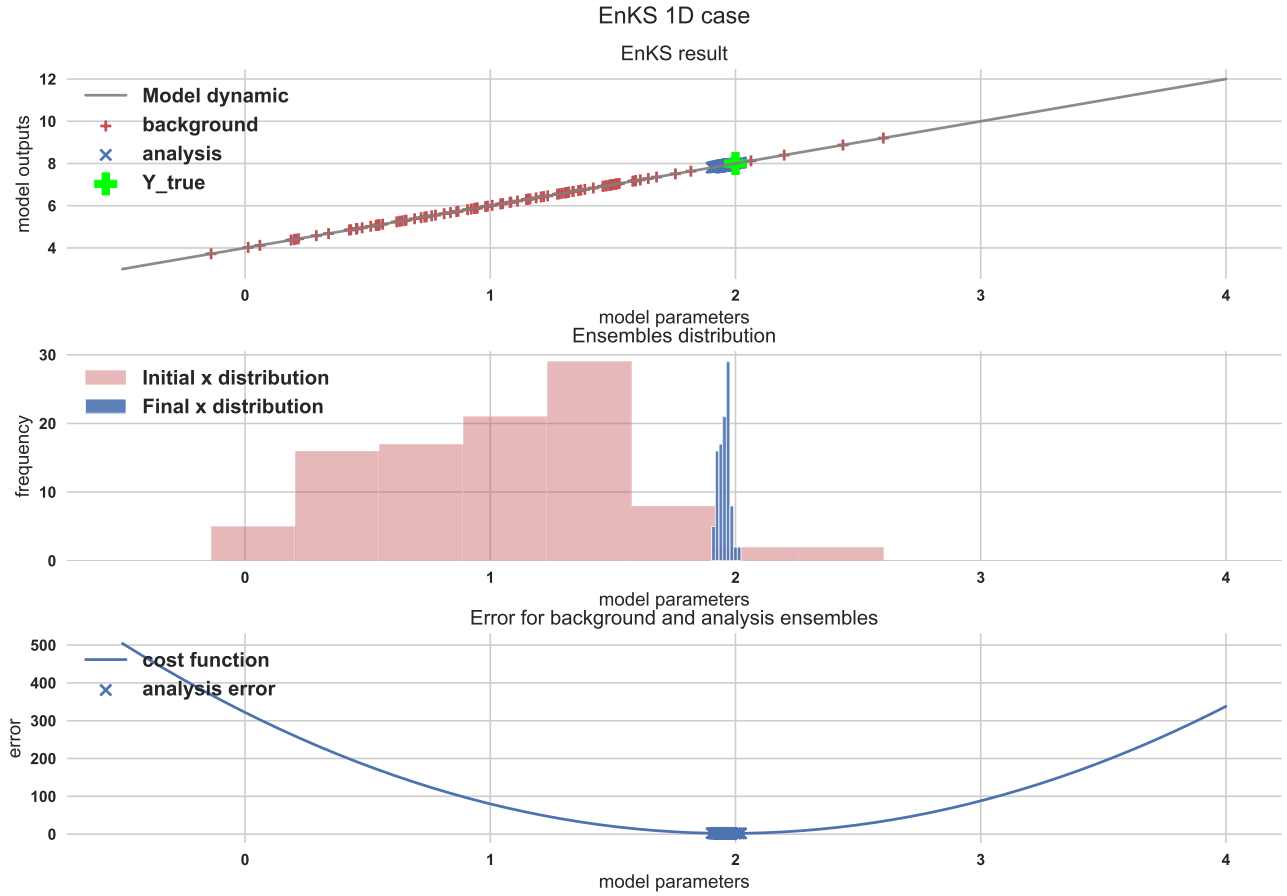


Figure 2.3 – Result of the EnKS with 100 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(1, 0.5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at forecast and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

Figure 2.3 shows the EnKS results in the linear case with 100 ensemble members measurement error variance of 0.05. The background ensemble is drawn from the normal distribution $\mathcal{N}(1, 0.5)$ and ensemble members are represented by red crosses for the background and blue crosses for the analysis. For each ensemble member the parameter can be read on the horizontal axis and its prediction value on the vertical axis. It shows a good estimation of the true parameter value $x = 2$ which can also be observed on the position of the analysis ensemble distribution on the cost function. It can also be noted that the shape of the cost function is quadratic for a linear case. The analysis ensemble distribution is close to a Gaussian distribution due to the linear aspect of the dynamical function. The quality of the results are mainly because of good estimation of the covariance on such a simple case in the sense of respecting the assumption of the Kalman filter theory.

One should observe in the second panel of Fig. 2.3 on the blue histogram, which shows the distribution of the analysis that the mean analysis is not exactly centered on the target value, due to the noise added to the observation. Decreasing the observation error improves the parameter estimation see Fig. 2.4 where the standard deviation associated to the observation is now 0.01 instead of 0.05. Reduction of observation error is possible because of the artificial case but in real condition the error

associated with the observation is not a parameter a user can modify.

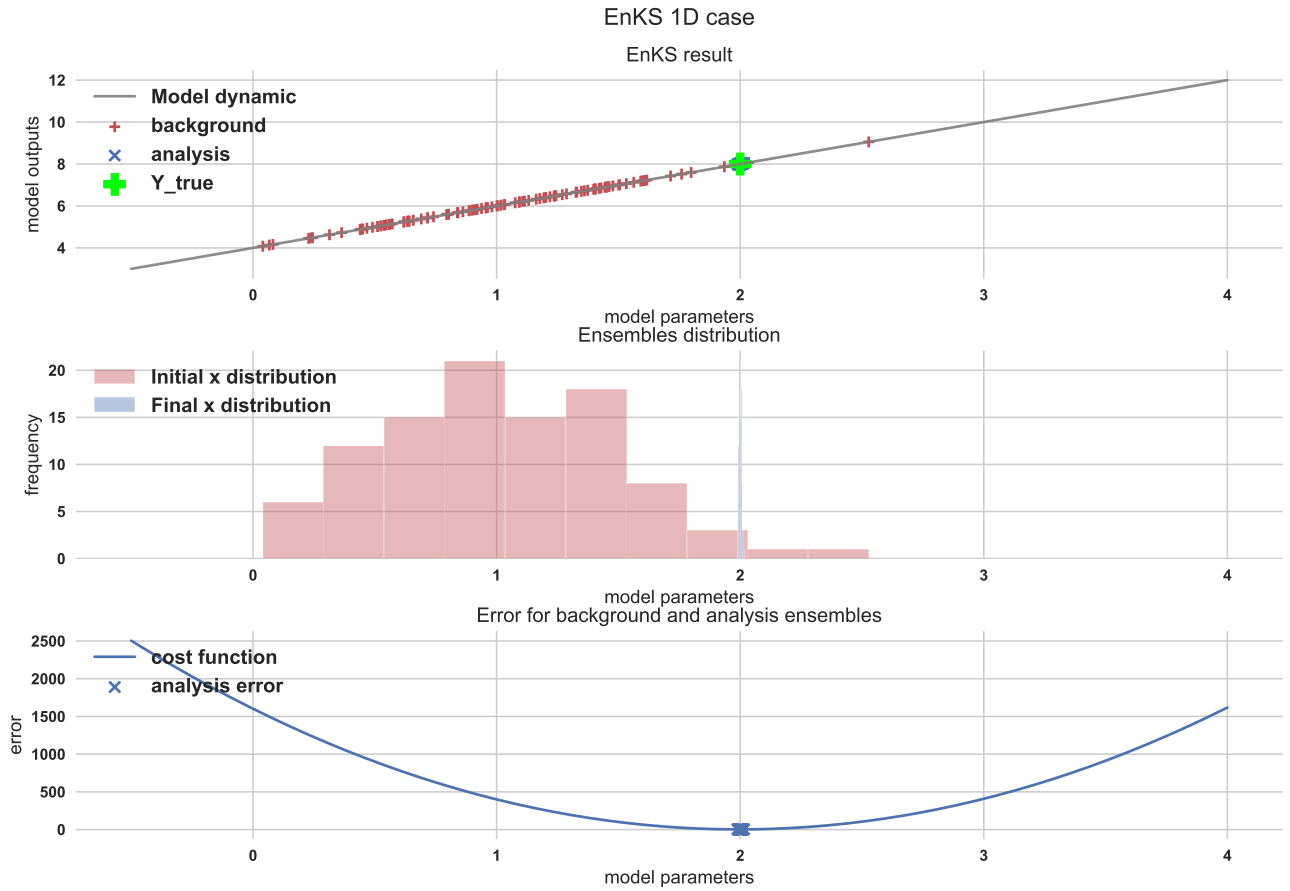


Figure 2.4 – Result of the EnKS with 100 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(1, 0.5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at background and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

In real condition the user can increase the number of ensemble members in order to improve the parameter estimation. This is shown Fig. 2.5 where the experiment is done with 1000 ensemble members and a standard deviation of the observation error equal to 0.05. Comparison is illustrated in Fig. 2.6 which shows distribution analysis for runs with low observation standard deviation error or high number of ensemble members.

The reader should notice the similarity with a gradient descent in this particular linear case. Covariance estimation is equivalent to estimating the gradient of the dynamical function on the area sampled by the ensemble. In the next examples, it will be visible that this global gradient estimation leads to over or under estimation in the non-linear case.

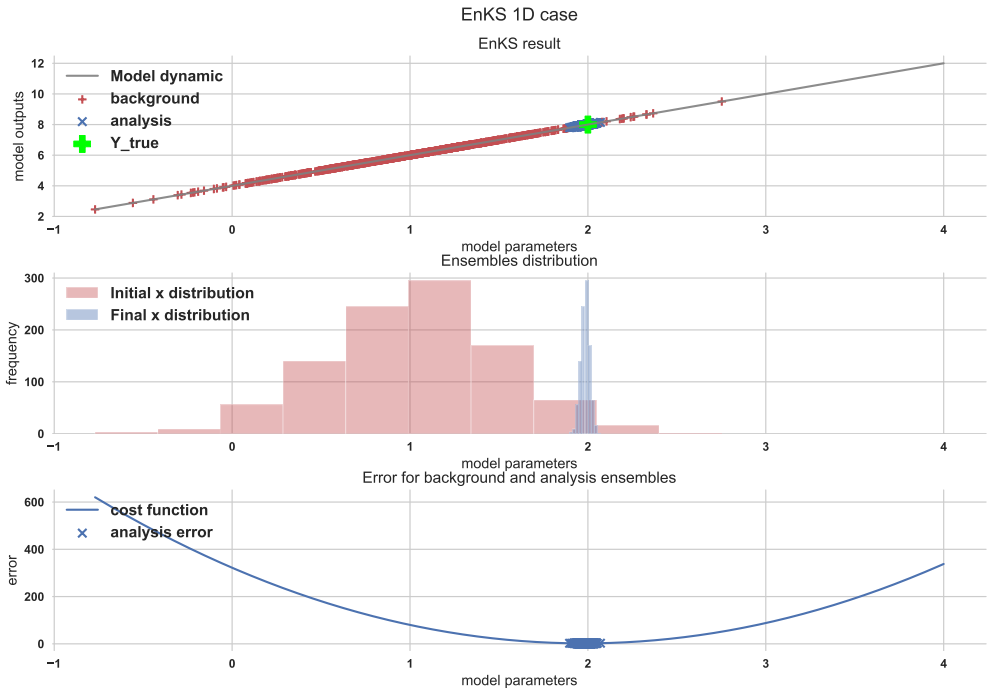


Figure 2.5 – Result of the EnKS with 1000 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(1, 0.5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at background and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

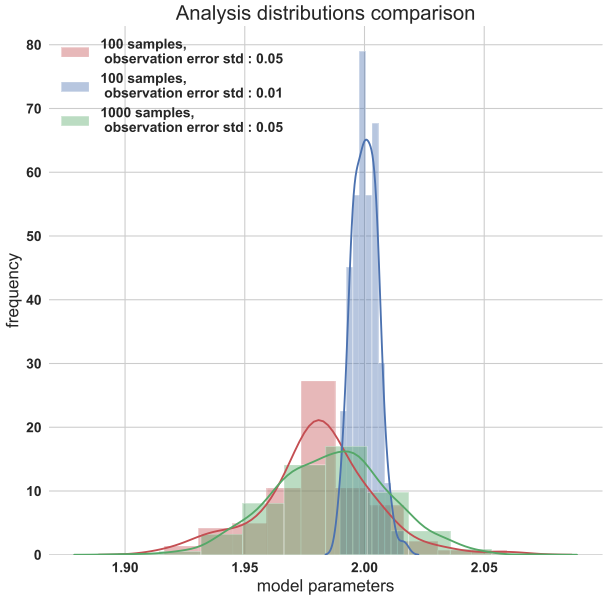


Figure 2.6 – Comparison of final ensembles for assimilation with different observation error std and different number of ensemble members. Red histogram is the analysis distribution for an observation error equal to 0.05 and 100 ensemble members, blue histogram is for an observation error std equal to 0.01 and 100 ensemble members and green histogram is for an observation error std of 0.05 with 1000 ensemble members.

2.5.2 Non-linear, monotonic dynamical function

It was demonstrated in Sec. 2.2 that the Kalman method from which the ensemble version is an approximation is the best (in the sense of uncertainty minimization) unbiased way to estimate parameters in a linear case and under Gaussian assumption. When ensemble methods are used on real application cases, it is usually not that easy. It is possible to define a one-dimensional non-linear function to visualize how the EnKS method behaves. The non-linear dynamical function is now represented by the cubic function :

$$\begin{aligned} \mathcal{H}\colon \textit{Parameter Space} &\mapsto \textit{Observation Space} \\ x \quad &\mapsto \quad y = \mathcal{H}(x) = x^3 \end{aligned} \tag{2.65}$$

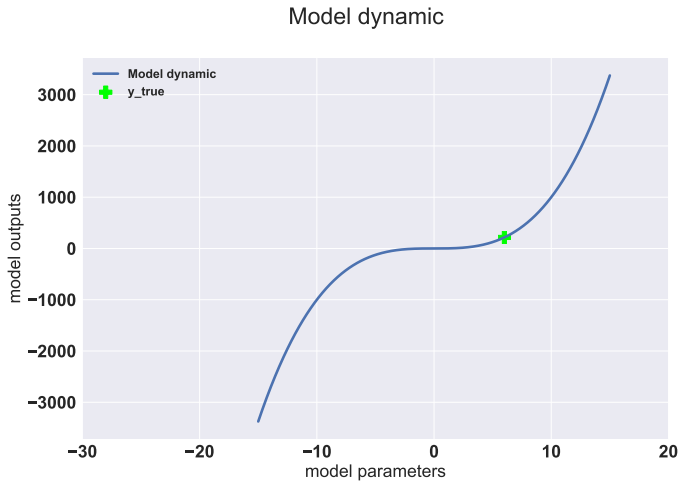


Figure 2.7 – Non-linear model dynamical function. The green cross represents the measured value $y = 8$ corresponding to the parameter value that has to be retrieved $x = 6$.

Figure 2.8 shows a clear overshoot of the parameters update. This effect can be explained by the erroneous covariance estimation, equivalent to a gradient estimation of the dynamical function, due to the non-linearity of the function. This overestimation is the result of the initial ensemble placed on a very local part of the parameter space where the slope of the function is weak. Because the algorithm relies on a linearity assumption, an important update of the parameter is necessary to reach the ordinate equal to $5^3 = 125$. This is equivalent to taking the linear tangent of the dynamical function at the abscissa where the initial ensemble is located. Figure 2.9 shows that the initial ensemble has an influence on the result. Because the initial ensemble is placed where the function has a more important gradient, an underestimation of the update is observed.

This result emphasizes the importance of a correct initial ensemble to estimate covariance not on a local area of the dynamical function but on average on a large set of parameter values such as illustrated in Fig. 2.10. The ESMDA method was developed in order to alleviate the problem of over and under estimation due to non-linear dynamical function. Instead of estimating the covariance only once which is strongly dependent on the background ensemble, the ESMDA method does smaller iterative updates with covariance estimation after each update. An example on the same case as in Fig. 2.7 illustrates ESMDA results on a non-linear case. Figure 2.11 shows the result of ESMDA for 5 iterations, one should observe the improved quality of the prediction at the last iteration compared to the EnKS. Underestimation is still visible at the first iteration but the analysis of the first iteration

will become the new background from which covariance is re estimated for the next iteration etc.

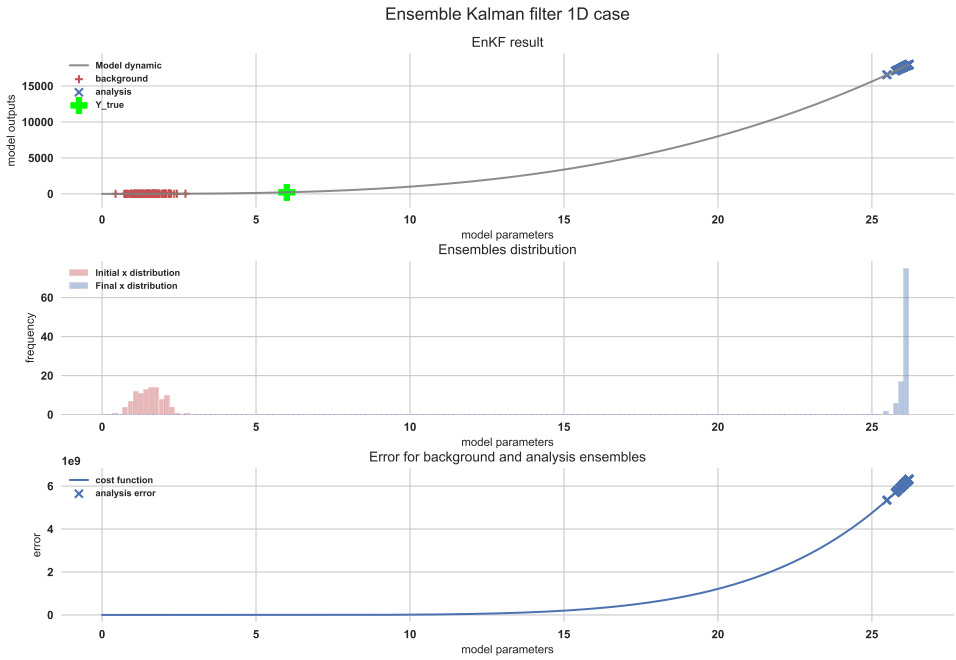


Figure 2.8 – Result of the EnKS with 100 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(1.5, 0.5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at background and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

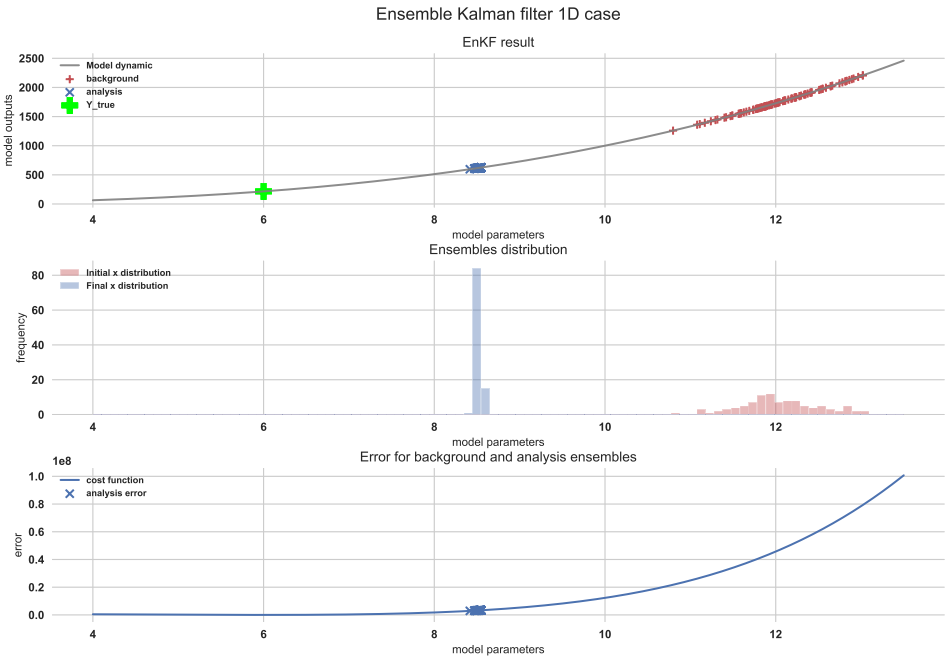


Figure 2.9 – Result of the EnKS with 100 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(12, 0.5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at background and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

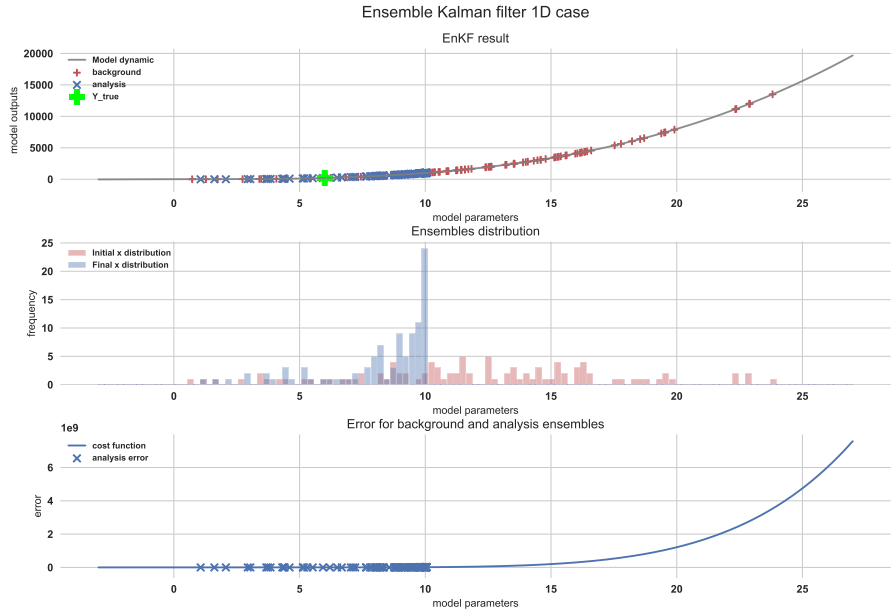


Figure 2.10 – Result of the EnKS with 100 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(12, 5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at background and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

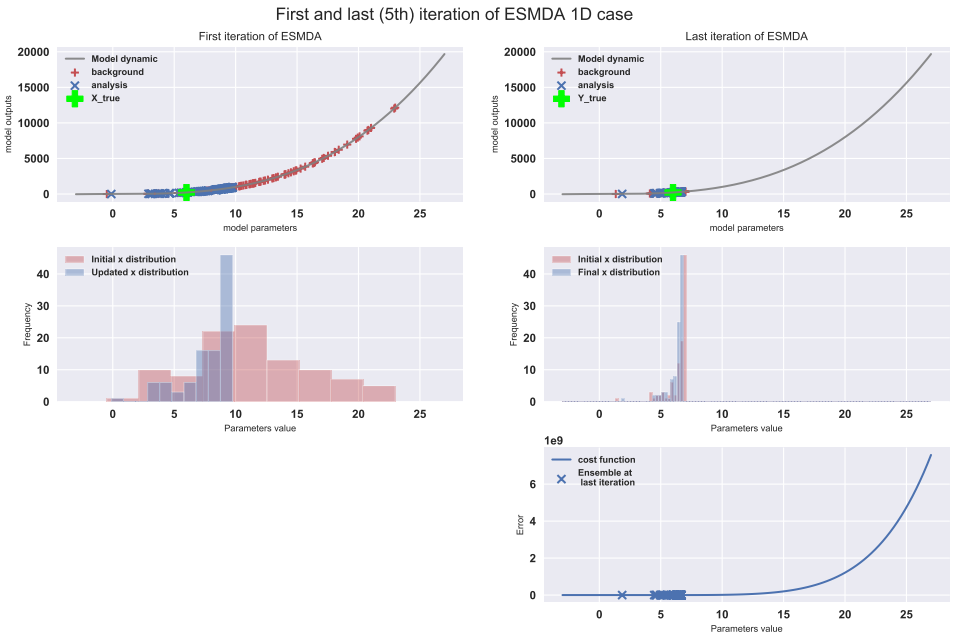


Figure 2.11 – Result of ESMDA with 5 iteration with 100 ensemble members. On the left-hand side the first ESMDA iteration is represented, in the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(12, 0.5)$ and blue crosses the analysis. Distribution of ensemble members at background and analysis are shown on middle panels. On the right-hand side, the last ESMDA iteration is represented. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

2.5.3 Non-linear, non-monotonic dynamical function

For the last example it is important to show how an ensemble method behaves when the dynamical function is non-monotonic. Non-monotonic function implies that multiple parameters are solutions for the same observation value. The following equation was chosen as non-linear non-monotonic dynamical function :

$$\begin{aligned} \mathcal{H}\colon \textit{Parameter Space} &\mapsto \textit{Observation Space} \\ x \quad &\mapsto \quad y = \mathcal{H}(x) = 2.5x + 4 + 9\sin(1.7x) \end{aligned} \tag{2.66}$$

One of the principal changes due to non-monotonic property is the equifinality of the problem *i.e.*, different sets of parameters can give the true observation value, the problem is called under-constrained or ill-posed. The function introduced in Eq. 2.66 aims to illustrate how EnKS behaves when it is applied to an ill-posed problem. The function is a linear function with a sinusoidal component, an observation is chosen such that multiple parameter values can output a prediction corresponding to this observation.

First an example is shown Fig. 2.13 where the EnKS is used with an ensemble initialization sampling multiple periods of the dynamical function. The ensemble analysis was updated towards the solution of $x = 2$. This can be explained by the covariance estimation that detects the linear component of the dynamical function thanks to a large sampling of the parameter space. If the initial ensemble

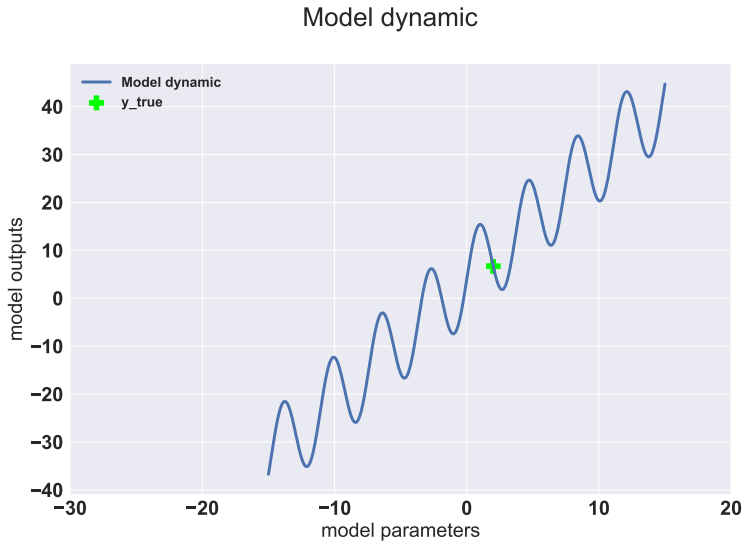


Figure 2.12 – Non-linear model dynamical function. The green cross represents the measured value $y = 8$ corresponding to the parameter value that has to be retrieved $x = 2$.

was located on a single period of $\mathcal{H}$ the estimated slope would be different such as illustrated in the non-linear monotonic case, Fig. 2.9. The recovery of the linear component of the observation is also allowed due to the high number of ensemble members compared to the dimension of the problem. Usually in data assimilation parameter space is highly dimensional and it is more difficult to realize such sampling of the parameter space. This explains why an important ensemble size results in a better estimation. The sampling of the parameter space smooths out the undesirable correlation that the algorithm could detect on a positive slope due to the sinusoidal signal instead of the linear trend.

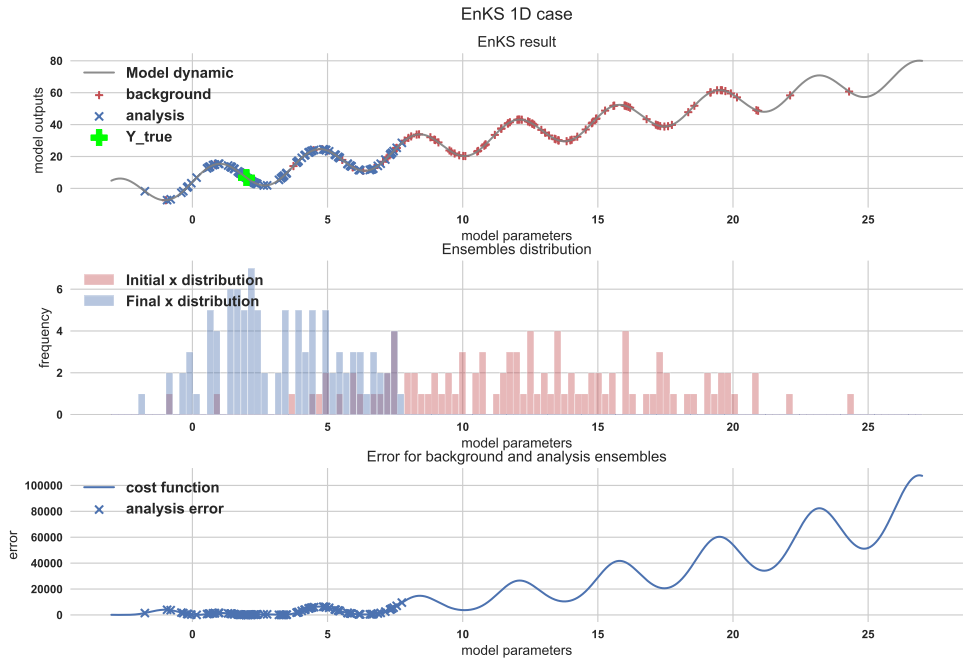


Figure 2.13 – Result of EnKS with 100 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(12, 0.5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at background and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

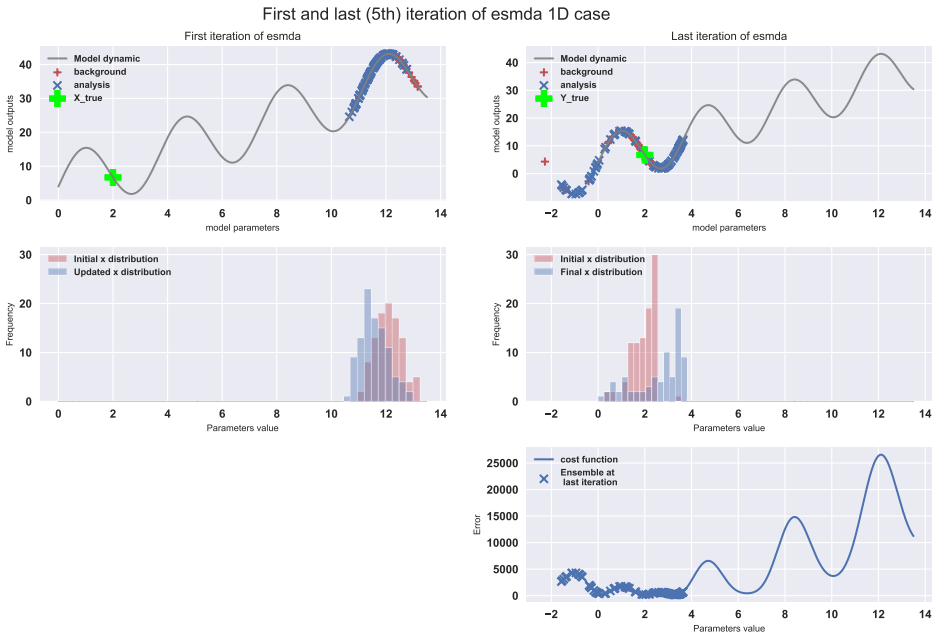


Figure 2.14 – Result of 5 iterations of ESMDA with 100 ensemble members. In the top panel, the red crosses represent the ensemble members at initialization drawn from a normal distribution $\mathcal{N}(12, 0.5)$. The blue crosses are the ensemble members at analysis. Distribution of ensemble members at background and analysis can be shown on the middle panel. In the bottom panel, the error associated with analysis ensemble members on the cost function is represented.

To avoid the necessity of large ensembles to correctly sample the dynamical function on real highly dimensional cases and keep the computational cost reasonable the ESMDA method was constructed. Figure. 2.14 shows the same case using ESMDA with a realistic initial condition that does not sample the dynamical function in order to detect linear components. Five ESMDA iterations were done, and the final analysis converged on a close solution of $x = 2$. Middle right panel shows that the final ensemble distribution is located around $x = 3.3$ which is one of the solutions. But not the one that corresponds to the observation.

From these results it can be concluded that even if the assumptions of linearity and Gaussianity are not respected, the EnKS method shows satisfying results because the dynamical function remains close to a linear function. But this requires a number of ensemble members too important when it is applied to highly dimensional problems. ESMDA is one of the solutions that is able to alleviate the non-linearity limitation by realizing multiple updates and covariance estimations. The different toy model experiments also showed that this kind of solution is not adapted to find multiple satisfying solutions in case of equifinal problems. Only added constraints as regularization can help to retrieve the true parameter. These different conclusions have to be kept in mind by the reader for a good understanding of ESMDA results applied to reservoir characterization in Chap. 5.

Chapter 3

Deep learning background

This chapter aims to give the main principles of deep learning for readers not familiar with these methods. It will introduce the main concepts of neural networks (NN) and their training process, convolutional networks and generative networks. The objective is to give a general understanding of the deep learning methods to data assimilation researchers by underlining the important similarities between both domains. Finally, it will focus on the domain of generative adversarial networks that has gained an important interest in the last years and is usually not known from those who are not directly involved in the deep learning research field.

3.1 Neural networks

A neural network can be defined as a non-linear application parameterized by weights w that associates to an entry x an output $y = f(x, w)$. One can consider x, y and w as real scalar for simplicity, but the expression holds for multi-dimensional cases. The objective of a neural network is to be trained using statistical learning on samples from a dataset to find w such as f approximates a target continuous function *e.g.*, regression or classification function. The function to minimize, referred to as loss function, to achieve the approximation of the target function is non-convex, which requires an optimizer to find local (or global) minimum. A first analogy can be done with a data assimilation method which is very similar. Usually, data assimilation methods hold w constant and try to estimate the parameter state x. Both methods solve an inverse problem (also known as bayersian inference), that is why strong analogy can be found between them Geer [37].

Figure. 3.1 shows the usual representation of a neural network called neural layer perceptron (NLP) or fully connected neural network (FCNN). It is one of the most basic and simple neural networks on which some theoretical results were demonstrated. Neurons, illustrated Fig. 3.2, are simple mathematical functions, called activation functions and written $\sigma(\Sigma_{j=1}^{n} w_j x_j + b)$ for n neuron inputs, with parameters called weights w_j and bias b, neuron are represented by circles in Fig. 3.1. An example of an activation function is the sigmoid : $\sigma(z) = \dfrac{1}{1 + \exp(-z)}$.

3.1.1 Universal approximation theorem

Cybenko [21] demonstrated through the universal approximation theorem (UAT) that "a finite linear combination of compositions of a fixed, univariate function and a set of affine functionals can uniformly approximate any continuous function of n real variables with support in the unit hypercube;" *i.e.*, composition of activation function in a single hidden layer NLP can approximate any continuous function to a given precision. Cybenko [21] demonstrated the UAT for sigmoid functions but it was later extended to other functions in Hornik [54], Leshno et al. [71], Pinkus [90]. It must be underlined

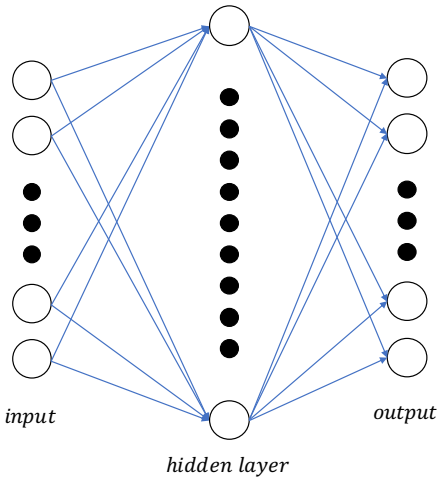


Figure 3.1 – Neural network scheme. Circles are referred to as neurons, blue lines are referred to as connection and black dots represent other neurons not drawn for readability. One column of neurons is called a layer. Each neuron of one layer is connected to all neurons of the next layer.

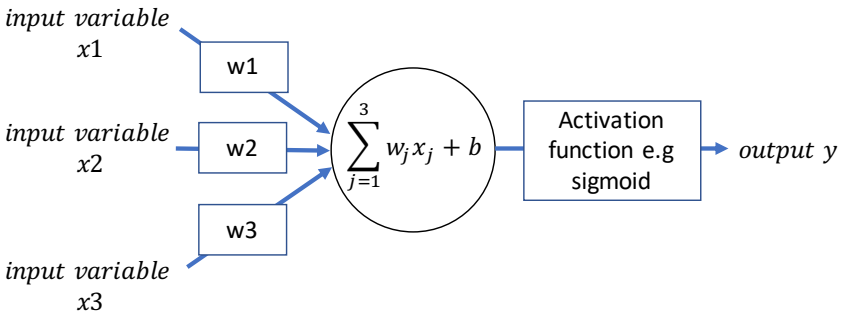


Figure 3.2 – Neuron function scheme.

that although UAT affirms the possibility to realize such approximation, in reality the hidden layer might be unfeasible. UAT does not tell how to construct such NN, how many neurons are necessary in the hidden layer for example or how to determine the value of the NN parameters. However, NN should be seen as a clever tool to model such a universal approximator of function, helped by powerful optimization methods to determine the parameters value. Although most of the theoretical knowledge concerns NLP, empirically lots of rules of thumb exist. For example, stacking hidden layers in an FCNN to form a multi-layer perceptron (MLP) helps to improve the approximation of a target function by inducing more non-linearity to the NN function.

This is one of the main results why NNs are today a mainly used method to solve complex problems. Many problems can be reduced to a function approximation such as classification, translating a text, defining a title from the content of a movie etc. The second main advance that was required to take advantage of the full capacity of NNs, is a way to estimate NNs' parameters from learning samples by gradient descent.

3.1.2 Parameter estimation

Training a NN by statistical learning can be seen as finding the best statistical estimate of the set of parameters such that the probability distribution of the model (here the neural network) p_{model} fits the unavailable true probability distribution where the learning samples come from p_{data}. Considering a set of learning samples from a dataset $\mathbf{X} = \left\{x^{(1)}, \ldots, x^{(m)}\right\}$ and their associated label $\mathbf{Y} = \left\{y^{(1)}, \ldots, y^{(m)}\right\}$ for example their class or category for a classification task. The objective for supervised learning is to estimate the conditional maximum likelihood :

$$w_{ML} = \underset{w}{\operatorname{argmax}} \prod_{i=1}^{m} p(y_i|x_i; w) \tag{3.1}$$

Where w_{ML} is the set of parameters corresponding to the maximum likelihood. By using the log-likelihood to pass from a product of probability that is numerically inconvenient to a sum of probability :

$$w_{ML} = \underset{w}{\operatorname{argmax}} \sum_{i=1}^{m} \log\left(p(y_i|x_i; w)\right) \tag{3.2}$$

and dividing by the number of learning samples m to get an expression using the expectation with respect to the empirical data distribution without changing the argmax function :

$$w_{ML} = \underset{w}{\operatorname{argmax}} \ \mathbb{E}_{x\sim p_{data}} \log\left(p(y_i|x_i; w)\right) \tag{3.3}$$

this expression can be linked to the Kullback-Leibler divergence (KL divergence) that measures the dissimilarity between two probability distributions :

$$D_{KL}\left(p_{data}||p_{model}\right) = \mathbb{E}_{x\sim p_{data}}\left[\log p_{data}(y|x) - \log p_{model}(y|x; w)\right] \tag{3.4}$$

where the term depending on p_{data} is a function of the data generation process which is not a parameter the user can modify. Thus, minimizing the KL divergence is equivalent to maximizing Eq. 3.3 *i.e.*, minimizing the cross-entropy between data and model distributions.

In software implementation, the training process consists in minimizing a loss function usually derived from the log-likelihood. As an example, minimizing a mean square error is equivalent to minimizing the KL divergence between an empirical distribution and a Gaussian model. The reader can refer to Goodfellow et al. [42] for further details.

Finally, the last important piece of the neural network framework is the optimization procedure, that aims at minimizing the loss function with respect to the NN's parameters. Gradient descent algorithm uses the local gradient of the loss function with respect to the parameters to determine the update of the parameters. The computation of the local gradient is done using backpropagation algorithm or backprop [98]. Backpropagation algorithm is a way to propagate the information backward in the neural network by computing the derivative of functions formed by composing other functions whose derivatives are known. It means that backpropagation computes the gradient (which is the derivative of a tensor operation) of the loss function. It uses computational graph theory that is above the topic of this study but for more detail the reader can refer to Goodfellow et al. [42], LeCun et al. [70], Rumelhart et al. [98]. Moreover, readers familiar with data assimilation should notice the analogy with adjoint method which is sensibly the same method that allows to propagate information backward and get the local gradient of the forward model.

The gradient of the loss function with respect to parameters being available a stochastic gradient descent (SGD) algorithm also called an optimizer is used to update the parameters. SGD differs from the original gradient descent method by using only a set of samples from the dataset instead of the entire dataset to compute the parameters' update. In deep learning this subset of samples is called a batch. Its size is defined by the user and corresponds to the number of data samples processed before the update of the parameters. Different versions of optimizers exist to alleviate some limitations such as the early stopping in local minima using inertia or managing highly-dimensional parameter space. The main gradient descent algorithms are for example classical SGD [62, 95], RMSProp [105] and Adam [64] algorithms.

This section gives the basis of how to train a basic neural network in theory. Lots of topics are not tackled such as regularization, over-fitting, data normalization, generalization... These are important topics for those who want to build and train neural networks but not necessary for a global understanding of how they work. Deep learning literature for real application and fundamental theory is available among them [8, 20, 42] are a very strong basis.

3.2 Convolutional network

Neural layer perceptron is adapted for vector shaped data, but when images are processed by FCNN, most of the spatial information is discarded by flattening the image to match the input shape of the NN. This is the main reason why Le Cun et al. [69] developed convolutional neural networks (CNNs). Convolution principle, illustrated Fig. 3.3, allows extracting spatial features by processing groups of pixels spatially closed by passing a kernel (2D window) on the processed image. Convolutional layers in a neural network have multiple kernels with trainable parameters *e.g.*, w, x, y and z in Fig. 3.3. After training, each kernel can detect different patterns *e.g.*, a circle, an edge, textures.... The training of a CNN is the same as described in Sec. 3.1 only the way it processes the pixels of an image is different.

Convolution are translation invariant which means that if a convolutional layer has a kernel that can detect an edge in an image, due to the convolution property it can detect edges over the entire image as opposed to NLPs. This property allows us to reduce the number of parameters of the NN to perform the same task.

Another important property of CNN is to be able to learn spatial hierarchies of patterns by stacking multiple convolutional layers. The first layer will for example detect local patterns such as edges and circles. The next layer will use these extracted features as inputs to deduct larger patterns such as the face of a human because it can be decomposed as multiple symmetrical circles like eyes and nostrils for example. This results in a particularly adapted framework to process general images such as multiple channels images (RGB) where the same convolutional kernels are applied to all channels of an image. The convolution is one of the main advances during the past decade in deep learning. This success was catalyzed by the development of Graphical Processor Units GPUs that are adapted to perform matrix computation faster than a classical CPU. It is not limited to three channel images but can also be applied to large 3D data such as demonstrated in Besombes et al. [7] where it is applied on atmospheric fields corresponding to an image of 64 by 128 pixels on 82 channels. 3D convolutions could be used but remains very memory demanding which is the limiting resource in GPUs.

One of the limitations in CNN is the computation of long range pixels that is difficult due to the local property induced by the use of convolutional kernels. An example is the number of legs when using CNN for computer vision on animal images. A classifier of cats and dogs images might not be surprised by a six-legged cat because of the distance between the legs being too important and

cannot be processed by the same convolutional kernel. Solutions are being proposed concerning this limitation on the receptive fields of CNN such as the concept of attention layers coming from neural language processing [108].

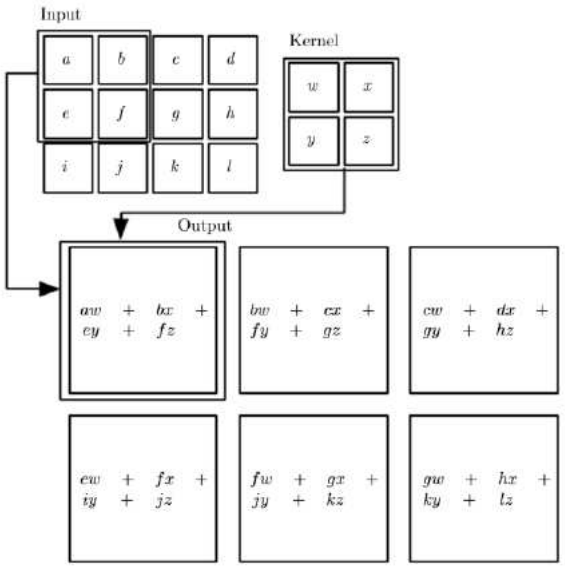


Figure 3.3 – Example of a convolution with a 2 kernel.

3.3 Generative networks

Generative networks aim to learn the data distribution in order to generate new samples from the same distribution. Although, the progress in supervised learning such as classification or regression tasks were partly due to the growing dataset in size and quality. Supervised learning requires labeled datasets that are expensive to build, unsupervised learning however is another category of deep learning models that does not need labeled data. They can be applied in a lot of domains that produce important quantities of data without the need to use human supervision to annotate data samples. Generative networks belong to the unsupervised category and are very popular because of their potential application to domains without high quality labeled datasets. Among generative networks, generative adversarial networks (GAN) are one of the generative models that gained lots of attention during the last years. In order to understand the recent success of GANs, it is important to compare it with other generative models. Goodfellow [40] gives a comparison illustrated Fig. 3.4 between generative models that maximize likelihood directly or that can be derived to do so. A clear distinction can be made between generative models, those who compute an explicit density function and others that use an implicit density function. Models that use an explicit density function that is computationally tractable have to be carefully designed to represent complex density functions while maintaining low computational costs such as Fully Visible Belief Networks [33, 34]. The main drawback with such methods are the computational cost that is proportional to the data dimension. GANs were designed to be able to generate samples in parallel to avoid this problem, GANs are also flexible regarding their design and architecture which is also an advantage compared to explicit models.

Another way to build explicit generative models is to use intractable density functions coupled with approximations to minimize the log likelihood. Approximations can be divided in two categories : Variational and Markov Chain approximations. Variational methods define a loss function that is computationally tractable and bounded by the log-likelihood. The most used NN in the family of

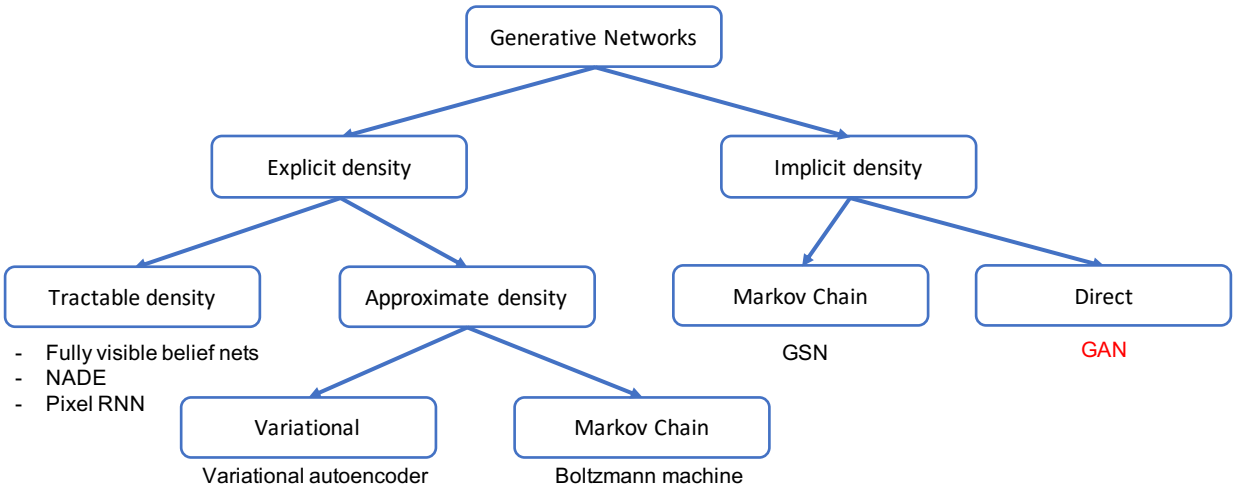


Figure 3.4 – Taxonomy of generative networks, reproduced from Goodfellow [40].

variational learning is Variational AutoEncoders (VAEs) [63]. As of this writing it was not proven yet that VAEs are asymptotically consistent meaning that there is no proof that with a weak approximation on prior or posterior distribution VAEs can learn something different from p_{data}. Contrary to the GAN where it was proven that under certain assumptions the GAN is a universal approximator. Markov chain approximation consist in generating data x' using a transition operator such as $x' \sim q(x'|x)$ repeatedly. It is difficult to check the convergence of the chain and this method becomes computationally too costly when the dimension of the data increases. GAN were also designed to avoid the use of Markov chain approximation.

Finally, implicit generative models can be again separated in two domains between models that can generate a sample in a single step, of which GANs belong, and others that need more computations. Before GANs there were no generative models able to generate samples in a parallelized manner, but they have been joined recently by models based on kernelized moment matching methods [72]. Another advantage is because the generator is trained using the gradient from the loss function computed on the discriminator output, the generator cannot simply copy samples from the dataset.

3.4 The GAN framework

Generative Adversarial Networks (GANs) were introduced in Goodfellow et al. [41], it is a generative model made of two neural networks trained peculiarly. The first model is called the generator that from a drawn noisy vector generates a sample from the data distribution, p_{data}. The second is called the discriminator, its objective is to measure the similarity between the dataset distribution represented by the samples in the dataset (p_{data}) and the distribution generated by the model (p_{model}). Both networks, in their classical version applied to images, have convolutional neural network (CNN) architectures Fig. 4.2, 4.3 using convolution layer in the discriminator and transposed convolution layer in the generator.

Figure 3.5 represents the GAN framework. The training phase is a game between the generator and the discriminator. The task of the generator is to create samples coming from the same distribution as the dataset. The discriminator has to determine whether the samples it processes are real (from the dataset) or fake (created by the generator also called synthetic). Discriminator outputs the probability of a sample to come from the dataset and uses labels as feedback, as in a supervised training. The label

being real or fake, it does not require a labeled dataset. The generator is given the same feedback *i.e.*the classification of fake samples by the discriminator, but its task is to fool the discriminator, meaning that its samples have to be predicted as true by the discriminator. The knowledge of the prediction made by the discriminator is used as feedback for minimizing $\mathbb{E}_{z\sim p_{\text{model}}}[1 - \log D(G(z))]$.

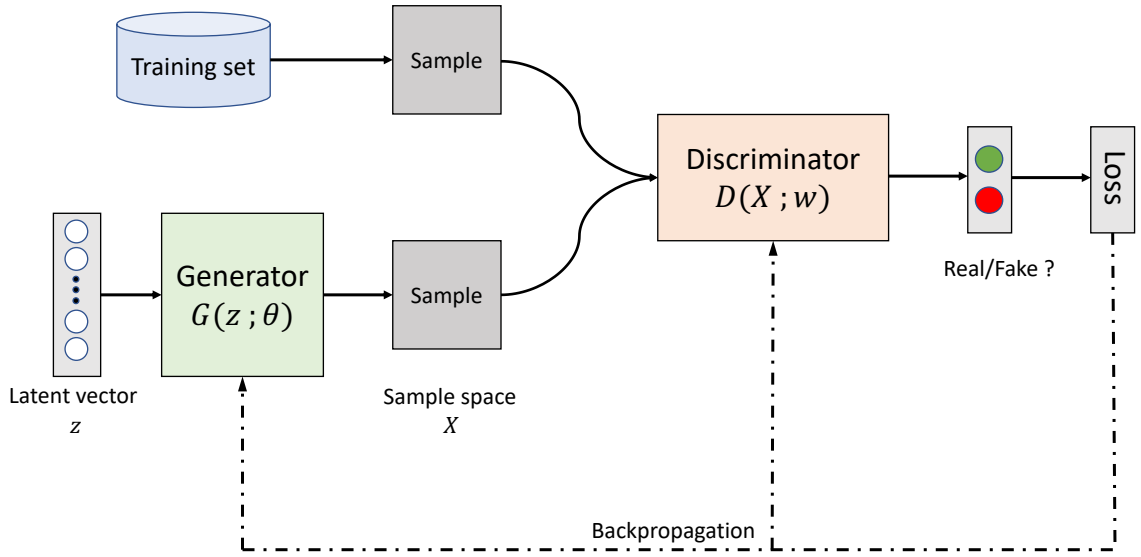


Figure 3.5 – GAN framework scheme.

Both networks are considered as functions that are differentiable with respect to their inputs, and their weights θ and w respectively for the generator and the discriminator. The cost function is defined with respect to the weights of each network $J = J(\theta, w)$. Alternatively a mini batch of samples from either the generator or the dataset is given to the discriminator, then an update by stochastic gradient descent (SGD) of the weights of one network is done by *freezing* the weights of the other network. The following step, another mini batch of samples is considered and the other network's weights are updated while the other one has its weights frozen. The fact that both networks' cost function depends on the weights of the other network changes the optimization problem into a game problem. The goal is no longer to find a (local) minima but to find a Nash equilibrium [40]. It yields the following loss function :

$$\min_G \max_D J(D;G) = \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_{\text{model}}}[1 - \log D(G(z))] \tag{3.5}$$

Equation 3.5 is the cross-entropy loss that is used when training a binary classifier using a sigmoid output, where labels for real samples are 1 and 0 for fake ones. The principal difference is that batches of samples come from different places, the dataset and the generator. We can write the cost function to minimize for the discriminator only :

$$J^D(D;G) = -\frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] - \frac{1}{2}\mathbb{E}_{z\sim p_{\text{model}}}[1 - \log D(G(z))] \tag{3.6}$$

and then for the generator :

$$J^G(D;G) = -J^D(D;G) = \frac{1}{2}\mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \frac{1}{2}\mathbb{E}_{z\sim p_{\text{model}}}[1 - \log D(G(z))] \tag{3.7}$$

It was proven in Goodfellow et al. [41] that minimizing Eq. 3.5 is similar to minimizing the Jensen-Shannon divergence between p_{data} and p_{model}. The convergence to a global minimum $p_{\text{data}} = p_{\text{model}}$ of the training algorithm was also proven.

The training needs to be carefully designed in order to avoid problems such as mode collapse also called the Helvetica scenario where the generator exploits a local minima of D by generating the same images ignoring noise input. The mode collapse was addressed by the Wasserstein GAN [3] which is detailed in Section 3.5.

3.5 Wasserstein Generative Adversarial Network to characterize a physical system

This section introduces the notations and theoretical development for the parameterization of a physical system by the Wasserstein GAN (WGAN). This section is taken from Besombes et al. [7] included in the present manuscript Chap. 6 that is applied for climate field generation. It was slightly modified in order to make it more general and applicable to reservoir application.

3.5.1 Parametrizing a physical system

The physical system is considered as being the solution of an evolution equation

$$\partial_t \chi = \mathcal{M}(\chi), \tag{3.8}$$

where χ denotes the state of the system at a given time and $\mathcal{M}$ characterizes the dynamics including the forcing terms. While χ should stand for continuous multivariate fields, we consider its discretization in a finite grid so that $\chi \in X$ with $X = \mathbb{R}^n$, where n denotes the dimension. The physical system is the set of states of the system along its time evolution. It is characterized by a distribution or a probability measure, denoted p_{sys}.

Obtaining a complete description of p_{sys} is intractable usually due to the complexity of the system, and because p_{data}, is limited by numerical resources and is only a proxy for this distribution. Thus, p_{data} lives in the n-dimensional space X, but it is non-zero only on an m-manifold $\mathbb{M}$ (where $m \ll n$) that can be fractal. The objective is to learn a mapping

$$g : Z \mapsto X, \tag{3.9}$$

from $Z = \mathbb{R}^m$, the so-called latent space, to X.

Moreover, g must transform a Gaussian $\mathcal{N}(0, \mathbf{I}_m)$ to $p_{\text{data}} \subset \mathbb{M}$. Our objective is to approximate the physical distribution that is included in the high dimensional space X by a simpler distribution that is included in a relatively low dimensional space Z that we will describe in further details in the next section. The main advantage of such a formulation is to have a function g that maps a low dimensional continuous space Z to $\mathbb{M}$. This property could be useful in the domain of geoscience notably in climate sciences or subsurface characterization where a high dimensional space is ruled by important physical constraints and parameters.

Here the generator is a good candidate to learn the physical constraints that make a system state realistic without the need to run a complete simulation of the physical equations.

3.5.2 Background on Wasserstein generative adversarial networks

To characterize the system, we first introduce a simple Gaussian distribution $p_z = \mathcal{N}(0, \mathbf{I}_m)$ of zero mean and covariance the identity matrix $\mathbf{I}_m$, defined on the space $Z = \mathbb{R}^m$, called the latent space. The objective of an adversarial network is to find a non-linear transformation of this space Z to X as written in Eq. 3.9 so that the Gaussian distribution maps to the system distribution *i.e.*$g_{\#}(p_z) = p_{\text{sys}}$ where $g_{\#}$ denotes the pushforward of a measure by the map g, defined here as follows: for any measurable set E of X, $g_{\#}(p_z)(E) = p_z(g^{-1}(E))$ where $g^{-1}(E)$ denotes the measurable set of Z that is the pre-image of E by g. The latent space, Z, can be seen as an encoded state space where each point drawn from p_z corresponds to a realistic system state and where the generator is the decoder. Looking for such a transformation is non-trivial, and is directly linked to the parameterization problematic in data assimilation. This work describes a way to find such a transformation in an automated manner and easily transferable from one domain to another.

The search is limited to a family of transformations $\{g_\theta\}$, characterized by a set of parameters θ. Thus, for each θ, the non-linear transform of the Gaussian p_z by g_θ is a distribution p_θ. The goal is then to find the best set of parameters, θ^* such that $\theta^* = \text{argmin}_\theta \ \ di(p_\theta, p_{\text{sys}})$ where di is a measure of the discrepancy between the two distributions, so that p_{θ^*} approximates p_{sys}. This method is known as the generative learning, where g_θ is implemented as a neural network of trainable parameters θ. Note that, being a neural network, the resulting g_θ is then a differentiable function.

Even with this simplified framework, the search for an optimal θ is not easy. One of the difficulties is that the differentiability of g_θ requires the comparison of continuous distribution p_θ with p_{sys}, which is not necessarily a density on a continuous set. To alleviate this issue, Arjovsky et al. [3] introduced an optimization process based on the Wasserstein distance defined for the two distributions p_{sys} and p_θ by

$$W(p_\theta, p_{\text{sys}}) = \inf_{\gamma \in \Pi(p_\theta, p_{\text{sys}})} \mathbb{E}_{(x,y)} \left[\|x - y\| \right], \tag{3.10}$$

where $\Pi(p_\theta, p_{\text{sys}})$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively $\int_y \gamma(\cdot, dy) = p_\theta$ and $\int_x \gamma(dx, \cdot) = p_{\text{sys}}$. The Wasserstein distance, also called Earth mover distance (EMD), comes from optimal transport theory and can be seen as the minimum work required (in the sense of *mass* $\times$ *transport*) to transform the distribution p_θ into the distribution p_{sys}. Thus, the set $\Pi(p_\theta, p_{\text{sys}})$ can be seen as all the possible mappings, also called couplings, to transport the mass from p_θ to p_{sys}. The Wasserstein distance is a *weak distance* : it is based on the expectation, which can be estimated whatever the kind of distributions. Hence, the optimization problem states as

$$\theta^* = \text{argmin}_\theta W(p_\theta, p_{\text{sys}}), \tag{3.11}$$

which leads to the Wasserstein GAN (WGAN) approach.

One of the major advantages of the Wasserstein distance is that it is real-valued for non-overlapping distributions. Indeed, the Kullback-Leibler (KL) divergence is infinite for disjoint distributions and using it as a loss function leads to vanishing gradient Arjovsky et al. [3]. The Wasserstein Distance does not exhibit vanishing gradients when distributions do not overlap, as did the KL divergence in the original GAN formulation.

Unfortunately, the formulation in Eq. 3.10 is intractable. A reformulation is necessary using the dual form discovered by Kantorovich [60]. Reframing the problem as a linear programming problem

yields

$$W(p_\theta, p_{\text{sys}}) = \sup_{f \in 1-\text{Lipshitzian}} \left[\mathbb{E}_{x\sim p_{\text{sys}}}\left[f(x)\right] - \mathbb{E}_{x\sim p_\theta}\left[f(x)\right]\right], \tag{3.12}$$

where $1-$Lipshitzian denotes the set of Lipshitzian functions $f : \mathbb{R}^n \to \mathbb{R}$ of coefficient 1 *i.e.*for any $(x_1, x_2) \in \mathbb{R}^n$, $|f(x_1) - f(x_2)| \leq ||x_1 - x_2||$, $||\cdot||$ being the Euclidian norm of $\mathbb{R}^n$. For any $1-$Lipshitzian function f the computation of Eq. 3.12 is simple: the first expectation can be approximated by :

$$\mathbb{E}_{x\sim p_{\text{sys}}}\left[f(x)\right] \approx \mathbb{E}_{x\sim p_{\text{data}}}\left[f(x)\right], \tag{3.13}$$

where the right-hand side is computed as the empirical mean over the database p_{data} that approximates p_{sys} in the weak sense Eq. (3.13). The second expectation can be computed from the equality

$$\mathbb{E}_{x\sim p_\theta}\left[f(x)\right] = \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f(g_\theta(z))\right], \tag{3.14}$$

where the expectation of the right-hand side can be approximated by the empirical mean computed from an ensemble of samples of z which are easy to sample due to the Gaussianity.

However, there is no simple way to characterize the set of $1-$Lipshitzian functions which limits the search of an optimal function in Eq. 3.12. Instead of looking at all $1-$Lipshitzian functions, a family of functions, $\{f_w\}$ parameterized by a set of parameters w, is introduced. In practice, it is engendered by a neural network with trainable parameters w, called the *critic*.

Finally, if the weights of the network are constrained to a compact space $\mathcal{W}$, which can be done by the weight clipping method described in Arjovsky et al. [3], then $\{f_w\}_{w\in\mathcal{W}}$ will be K-Lipschitzian with K depending only on $\mathcal{W}$ and not on individual weights of the network. This yields :

$$\begin{aligned}\max_{w\in\mathcal{W}} \left[\mathbb{E}_{x\sim p_{data}}\left[f_w(x)\right] - \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f_w(g_\theta(z))\right]\right] \leq \\ \sup_{f\in 1-\text{Lipshitzian}} \left[\mathbb{E}_{x\sim p_{data}}\left[f(x)\right] - \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f(g_\theta(z))\right]\right]\end{aligned} \tag{3.15}$$

which tells us that the critic tends to the Wasserstein distance when trained optimally *i.e.*, if we find the max in Eq. 3.15 and if f is in (or close) to $\{f_w\}_{w\in\mathcal{W}}$. The weight clipping method was replaced by the gradient penalty method in Gulrajani et al. [45] because it diminished the training quality as mentioned in Arjovsky et al. [3]. Because it results from a neural network, any function f_w is differentiable, so that the $1-$Lipshitzian condition remains to ensure a gradient norm bounded by 1 *i.e.*for any $x \in X$, $||\nabla f_w(x)|| \leq 1$. To do so, Gulrajani et al. [45] have proposed to compute the optimal parameter $\tilde{w}(\theta)$ as the solution of the optimization problem

$$\tilde{w}(\theta) = \text{argsup}_w L(\theta, w) \tag{3.16}$$

where L is the cost function

$$L(\theta, w) = \mathbb{E}_{x\sim p_{data}}\left[f_w(x)\right] - \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f_w(g_\theta(z))\right] + \lambda\mathbb{E}_{\hat{x}\sim\hat{p}}\left[\left(||\nabla f_w(\hat{x})|| - 1\right)^2\right] \tag{3.17}$$

with λ the magnitude of the gradient penalty and where $\hat{x}$ is uniformly sampled from the straight line between a sample from p_{data} to a sample from p_θ (line 8) of Algorithm 1. The optimal solution $\tilde{w}(\theta)$ is obtained from a sequential method where each step writes

$$w_{k+1} = w_k + \beta_k \nabla_w L(\theta, w_k), \tag{3.18}$$

where β_k is the magnitude of the step. In an adversarial way, Eq. 3.17 could be solved sequentially

e.g., by the steepest descent algorithm with an update given by :

$$\theta_{q+1} = \theta_q - \alpha_q \nabla_\theta W(p_{\theta_q}, p_{\text{sys}}), \tag{3.19}$$

where α_q is the magnitude of the step. We chose to use the two-sided penalty for gradient penalty method, as it was shown to work well in Gulrajani et al. [45]. At convergence, the Wasserstein distance is approximated by :

$$W(p_\theta, p_{\text{sys}}) \approx \mathbb{E}_{x\sim p_{data}}\left[f_{\tilde{w}(\theta)}(x)\right] - \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f_{\tilde{w}(\theta)}(g_\theta(z))\right]. \tag{3.20}$$

Hence, the solution of the optimization problem Eq. 3.11 is obtained from a sequential process composed of two steps, summarized in the Algorithm 1. In the first step, the weights of the generator are frozen with a given set of parameters θ_q and the critic neural network is trained in order to find the optimal parameter $\tilde{w}(\theta_q)$ solution Eq. 3.16 (lines 3 – 11 in Algorithm 1). In the second step, the critic is frozen and, the generator is set as trainable in order to compute θ_{q+1} from Eq. 3.19 (lines 12 – 17 in Algorithm 1). Note that in the Algorithm 1, the steepest descent is replaced by an Adam optimizer [63], a particular implementation of stochastic gradient descent which has shown to be efficient in deep learning.

Algorithm 1 WGAN training algorithm.

Require: Learning rate lr, batch size b, n_{critic} number of iteration of the critic per generator iteration.
Require: w_0 and θ_0 respectively the initial critic and generator parameters.
1: # Optimization cycle
2: **while** θ has not converged **do**
3: # 1. Computation of the Wasserstein distance by maximization over 1−Lipshitzian functions
4: **for** $t = 0, \ldots, n_{critic}$ **do**
5: # 1.1 Computation of the gradient for the 1−Lip. function.
6: Sample$\{x^{(i)}\}_{i=1}^b \sim P_{data}$ a batch from the real data.
7: Sample$\{z^{(i)}\}_{i=1}^b \sim P_\theta$ a batch from the generated data.
8: Sample$\{\hat{x}^{(i)}\}_{i=1}^b$ where $\hat{x} = \xi x + (1-\xi) g_\theta(z)$ where $\xi \sim \mathcal{U}[0,1]$
9: $grad_w \leftarrow \nabla_w \left[\frac{1}{b}\Sigma_{i=1}^b f_w(x^{(i)}) - \frac{1}{b}\Sigma_{i=1}^b f_w(g_\theta(z^{(i)})) + \frac{\lambda}{b}\Sigma_{i=1}^b \left(||\nabla f_w(\hat{x}^{(i)})|| - 1\right)^2\right]$
10: # 1.2 Update the parameter w to maximize Eq. 3.12
11: $w \leftarrow w + lr * Adam(w, grad_w)$
12: **end for**
13: # 2. Update the generator
14: # 2.1 Compute the gradient of the Wasserstein distance
15: Sample$\{z^{(i)}\}_{i=1}^b \sim P_\theta$ a batch from the generated data.
16: $grad_\theta \leftarrow \nabla_\theta \left[\frac{1}{b}\Sigma_{i=1}^b f_w(g_\theta(z^{(i)}))\right]$
17: # 2.2 Update the parameter θ to minimize the Wasserstein distance
18: $\theta \leftarrow \theta - lr * Adam(\theta, grad_\theta)$
19: **end while**

The reader should note that in the rest of this manuscript the GAN version will always be the Wasserstein GAN version described in this section but will be designate by GAN instead of WGAN for simplicity. However, between classical and Wasserstein version the name of the classifier network is referenced respectively as the discriminator and the critic this notation will be maintained.

3.6 Related work

The GAN method is an efficient way to learn the physical constraints that are present in the data. It can help to learn $p(x)$ in the Bayes theorem Eq. 2.6 which is equivalent to reduce the size of the research space for an inverse problem by generating only physically realistic states. Moreover, GANs are able to generate images with sharp gradients which is a recurrent problem in data assimilation methods. It is usually hard to correct the position of a sharp object while keeping the image realist. Being able to produce realistic states at a cheap computational cost is already an interesting asset.

3.6.1 Linear inverse problem

Since the introduction of GAN method in Goodfellow et al. [41], it was used to solve numerous application of inverse problem. Compressed sensing or linear inverse problem applied to images was realized using GAN prior such as in Bora et al. [11] where it uses a trained GAN that maps a latent space to the space of the unknown vector $x^* \in \mathbb{R}^n$ in the equation :

$$y = Ax^* + \eta \tag{3.21}$$

where $y \in \mathbb{R}^m$ is from linear measurement of x^*, $A \in \mathbb{R}^{n \times n}$ is the measurement matrix and $\eta \in \mathbb{R}^m$ is a noise. It was proved that using a GAN prior improves the results especially when there are few measurements. It was solved by using gradient descent on the GAN's input because it is differentiable by construction. Rick Chang et al. [94] developed a way to generalize the application of GANs to a group of linear problem close to compressed sensing such as inpainting and super resolution. The idea was to avoid training task specific GANs by using a projection operator. Neural networks able to perform multiple tasks are one of the grand challenges of deep learning, see Jaegle et al. [57].

3.6.2 Physically constrained inverse problem

Recently GAN has drawn attention of researchers that are tackling physically constrained inverse problem due to their universal approximator property mentioned in Sec. 3.1.1. The fact that data measurements in such problems are common, neural network in general can be an interesting tool for various physical applications because of the amount of available data to exploit. For example in earth science by Dueben and Bauer [25], or in reservoir characterization by Ertekin and Sun [28]. By zooming in the different machine learning methods applied in physics, GANs remain far from their optimal use, compared to the capability proven in other application domains such as in computer vision for example. Using a trained GAN in an inverse problem framework could be useful on several aspects.

An important source of model uncertainty in inverse problem comes from the approximation in mathematical parameterization *i.e.*, in the climate sense, scheme of subgrid processes, especially in the weather prediction application. First deterministic physical models were used to parameterize such processes and reduce model errors. One of the main progress was the stochastic methods to represent subgrid processes. Although new stochastic parameterization methods have improved prediction capability of NWPs, some of these methods still suffer from inaccurate distribution matching that cause biases and deteriorate the predictions [6]. Different stochastic parameterizations exists using for example, statistics of a model uncertainty and theoretical knowledge of the atmospheric processes [100]. Data driven approach can extract features in observations or model states and learn complex

non-linear subgrid processes in a computationally efficient manner, therefore machine learning are good candidates to represent subgrid processes.

Machine learning method for such application was first proposed by Krasnopolsky et al. [66]. Since it was investigated for different applications, Bolton and Zanna [10] used a CNN to replicate the spatio-temporal variability of the subgrid eddy momentum forcing. Gentine et al. [38] used a CNN to represent unresolved moist convection in coarse-scale climate models, Rasp et al. [93] used a CNN to represent all atmospheric subgrid processes in a climate model by learning from a multiscale model in which convection was treated explicitly. Yuval et al. [112] compares random forests neural networks methods for calculating subgrid terms through coarse graining.

Recently GANs were investigated to be used as stochastic parameterization of subgrid processes, in Gagne et al. [35] a GAN is used for the parameterization of subgrid processes in the Lorenz '96 dynamical system. Their main advantage is their ability to be trained in an unsupervised manner and to match a large variety of distributions.

GANs can also be used for Bayesian inversion. In Patel and Oberai [87], a GAN prior is applied in uncertainty quantification on a temperature field in heat conduction problem. Parameterization is used as a prior for distributions difficult to represent mathematically. The Bayesian inference is then done in the low dimensional latent space, and allows an efficient way for posterior sampling. Bayesian inversion in Adler and Öktem [2] is using a GAN prior and compare it with the state-of-the-art technique like Gibbs sampler applied to image reconstruction of 3D computed tomography.

Laloy et al. [68] used a Spatial GAN to parameterize the spatial distribution of subsurface rock facies. It was trained on a unique training image and can generate unconditional realizations after training. A posterior conditioning on static properties was demonstrated using MCMC inversion. Laloy et al. [67] realized a study of the dimensionality reduction using variational auto-encoder on a dataset of tens of thousands of images generated by multipoint statistics method. A comparison with PCA method and discrete cosine transform was done.

GAN prior was recently used in diverse physical inverse problem with the help of adjoint method, because gradient of GAN output with respect to input variables is available. In Mosser et al. [84], the author performs a stochastic seismic waveform inversion with a GAN prior. It uses an adjoint model to link the data mismatch to the GAN latent input parameters. It was also applied to history matching in reservoir characterization domain in Mosser et al. [83] but an important computational cost due to a high number of gradient descent iteration necessary to inverse the complete forward model. Moreover, the development of adjoint model is still in development and is costly as a consequence ensemble methods are still widely used and having a parameterization technique is valuable.

Finally, ensemble methods were coupled with neural network parameterization to alleviate one of the main limitation of these methods applied to subsurface history matching due to the non-Gaussianity distribution of parameters. Canchumuni et al. [13] used an auto-encoder coupled with PCA and ES-MDA for history matching. In following articles deep belief network [14] and variational auto-encoder [15] were used as parameterization of geological facies for history matching. Canchumuni et al. [16] realized a benchmark comparing different deep learning methods such as GAN, WGAN, convolutional VAE, PCA and proposed strategies in order to implement localization in those frameworks. Similarly, Bao et al. [5] coupled ES-MDA and GAN for history matching flow and transport data in hydrology. In the last 3 years multiple research papers were dedicated to the coupling of history matching and deep learning parameterization techniques with promising results, however some challenges remain open. First, application of such coupling remain in the subsurface characterization domain whereas data assimilation is widely used among diverse applications that could benefit of advantages offered

by such parameterization method *e.g.*, fire front assimilation, numerical weather forecast etc. Few of the previously mentioned studies tried their method on 3D cases, which is an important matter because of the increase of computational cost for 3D GANs due to the limited memory of GPUs for the moment.

The research objective of the current study is to first give a good understanding to the reader of data assimilation and deep learning domains. Underline the different limitation encountered in current operational data assimilation and emphasize the advantages of recent data driven strategies that could alleviate the previously mentioned limitations in data assimilation. The use of GAN as parameterization for ensemble methods is investigated in Chap. 5, its application for producing climate data is investigated in Besombes et al. [7] included in Chap. 6. Optimization methods for inverting the GAN function and recover conditioned generations in the latent space is tackled in Chap. 7.

Chapter 4

Generating realistic reservoir topologies

Basic concepts about neural networks and GANs can be found in Sec. 3, its application for generating realistic reservoir topologies will be tackled in this chapter. As mentioned in Sec. 1.1.5 several methods exist for the generation of realistic reservoir topologies but heterogeneities and especially channel heterogeneities remain hard to characterize. Object-based methods necessitate the creation of training images by experts combining the different information about the structure of the particular reservoir exploited. The realizations are usually very similar to the training image and then lack of variability. Whereas, pixel-based methods are computationally demanding and are not able to preserve the continuity of geological heterogeneities such as channels.

GANs are potential candidates due to their ability to generate realistic samples from a dataset. Moreover, it was specifically designed to be able to generate samples in parallel at a low computational cost. Our objective is to train a Wasserstein GAN on a dataset made of channel heterogeneities. The GAN has to be able to generate new samples with high diversity in the sense of preserving the different properties that characterize channelized heterogeneities without reproducing images from the dataset. This chapter describes the dataset used, the choice of the different hyper-parameters for the architectures and the training of the GAN, finally it tackles the definition of metrics to assess the quality of the generations.

4.1 Dataset

The dataset was created at Total with an industrial tool called Flexboolx, it is made of 10000 samples of 2D channelized reservoir (100 by 100 pixels) with two rock types, illustrated in Fig. 4.1. Channels are oriented in the West-East direction and made of a rock with a high permeability and porosity compared to the background facies (black). Channels' geometry is created with sinusoidal function controlled by the amplitude, width, thickness, wavelength hyper-parameters all drawn from a triangular distribution. The facies density *i.e.*, proportion of the white pixels compared to black pixels in Fig. 4.1 is variable following a triangular distribution visible in Fig. 4.4 computed from the dataset samples. As a preprocessing the dataset samples are normalized such as background material is indexed by -1 and heterogeneity material is indexed by 1.

The fluid flow inside the reservoir is largely driven by heterogeneities due to its physical properties but fluids can also flow through the black facies, at a slower speed. The creation of the dataset was done by creating 3D channelized blocks and slicing them horizontally (in the $x - y$ plane where z represents the height). A particular attention was given to the fact that a maximum of one slice was done for each 3D block. The reason for this constraint is that the different samples could be correlated if the constraint was not imposed. The consequence of the GAN training is the generation of similar

channels at the same position because they are more frequent in the dataset. The variability in the dataset is an important property that could influence history match results by outputting a realization with high certainty only because this realization was overfitted during training.

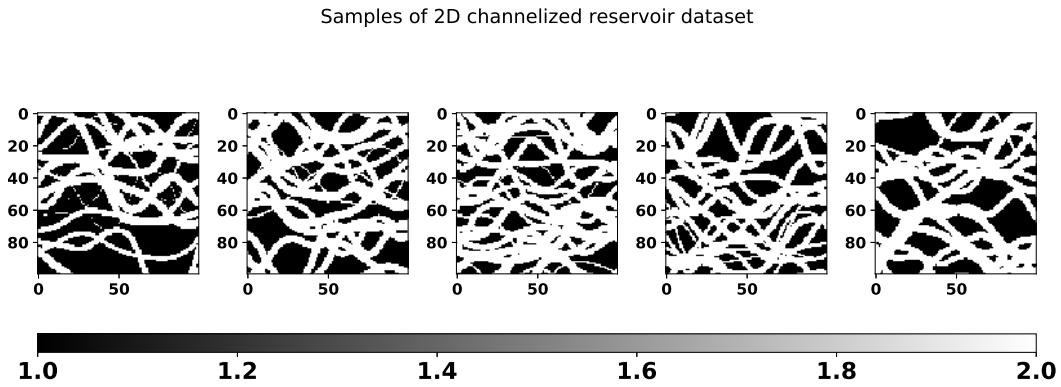


Figure 4.1 – Samples from 2D channelized reservoir dataset. 2 rock types are present in the reservoir. Background material (black pixels indexed by 1) has a low permeability and porosity. Heterogeneity material (white pixels indexed as 2) is highly porous and permeable.

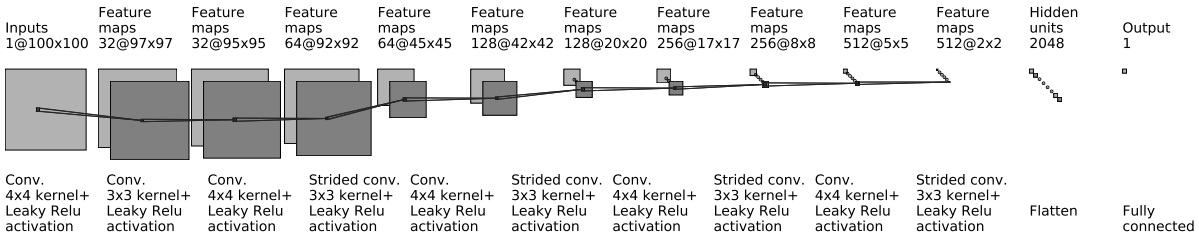


Figure 4.2 – Critic architecture as a convolutional network.

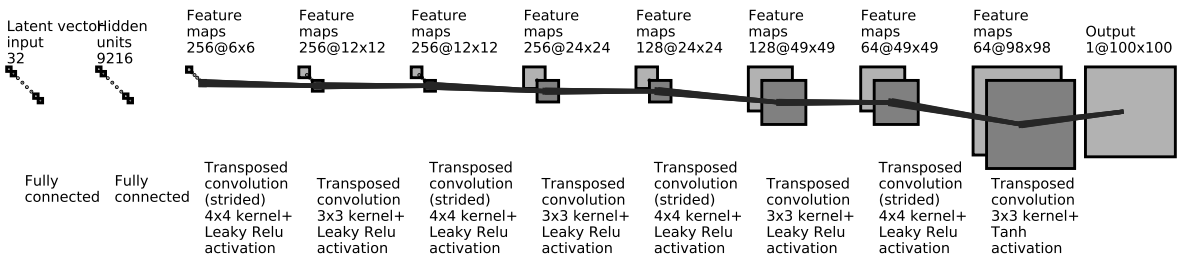


Figure 4.3 – Generator architecture as a convolutional network.

4.2 Architectures

Our first attempt to design a GAN in order to generate reservoir representations, was to design the generator and critic as convolutional neural networks CNN. It consists of numerous layers in order to increase or decrease the size of the image in respectively the generator or the critic. Most of the guidelines were taken from Gulrajani et al. [45].

The guidelines established in Arjovsky et al. [3] and Gulrajani et al. [45] for the networks architecture and training hyper-parameters were followed. GANs are known to be time-consuming to train, usually needing a high number of iterations due to the alternating aspect of the training algorithm between the critic and the generator. Our initial architecture used a simple convolutional network for both, with a high number of parameters. Figures 4.2 and 4.3 show the classical convolutional network that was used at the beginning of the study. The critic is made of a succession of convolutional layers. The leaky-Relu activation function is used in order to induce non-linearity in the network function. Leaky-Relu activation function is similar to Relu function except that it allows a small, positive gradient when the unit is not active, see Eq. 4.1. The generator is made of a succession of transposed convolutions with leaky Relu activation except for the last layer where a tangent hyperbolic function is used in order to output a binary image where pixel values are comprised between 1 or -1 to match the binary dataset images. The value -1 corresponds to the background material and 1 to the facies that constitute the channels. However, it proved difficult to train fitting multimodal distributions such as pixel value distribution for the images representing the rock types of a reservoir.

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ 0.01x & \text{otherwise.} \end{cases} \tag{4.1}$$

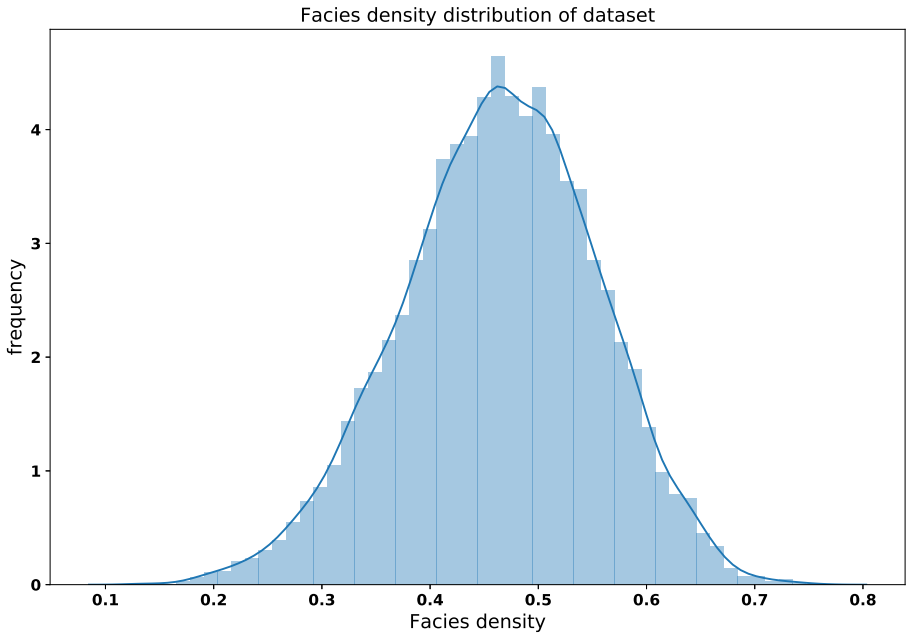


Figure 4.4 – Histogram of facies density for the 10000 samples dataset.

That is why for this study a ResNet-inspired architecture [50] was chosen. The goal of the Residual network is to reduce the number of parameters of the network and avoid gradient vanishing which is

Architecture	Number of layers		Number of trainable parameters	
	Generator	Discriminator	Generator	Discriminator
GAN-CNN	10	12	4.4M	5.9M
GAN-ResNet	20	38	1.6M	3.3M

Table 4.1 – Summary of the 2 GAN architectures. Only dense and convolutional layers are counted. (M is for millions)

a recurrent problem for deep networks that results in an even slower training.

A network is composed of a stack of layers. When a specific succession of layers is used several times we can refer to it as a *block*. The link between two layers is called a connection, a shortcut connection refers to a link between two layers that are not successive in the architecture. A residual block as in Fig. 4.6 for the critic or Fig. 4.5 is composed of a stacked convolution and a parallel identity shortcut connection. The idea is that it is easier to learn the residual mapping than all of it, so residual blocks can be stacked without observing a vanishing gradient.

Vanishing gradient is a well known limitation of deep neural networks, where information going backward during backpropagation is fading away due to chain rule properties multiplying low values of partial derivatives of the loss function with respect to the weights. As a result, layers at the top of the architecture do not receive significant updates and it can stop the learning of top layers. Skip connections in residual layers play the role of highways of information by linking top layers with shortcuts during backpropagation.

Moreover, during the building of an architecture a residual block can be added to an N layers network without reducing its accuracy. Because skip connections can easily learn the functional $F(x) = 0$ when the number of layers is already sufficient. Whereas on a sequential network without skip connections an added layer has to learn the identity function $F(x) = x$ if the number of layers was already optimal, which is a harder task. Residual blocks allow building deeper networks without loss in accuracy. The comparison of both architectures is visible in Table. 4.1, for the rest of the manuscript the stacked convolution architecture will be referenced as GAN-CNN and the residual one as GAN-ResNet.

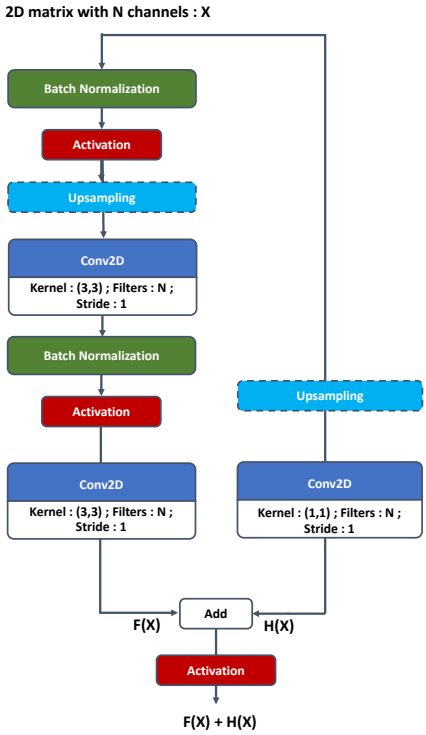


Figure 4.5 – Convolutional block for the generator.

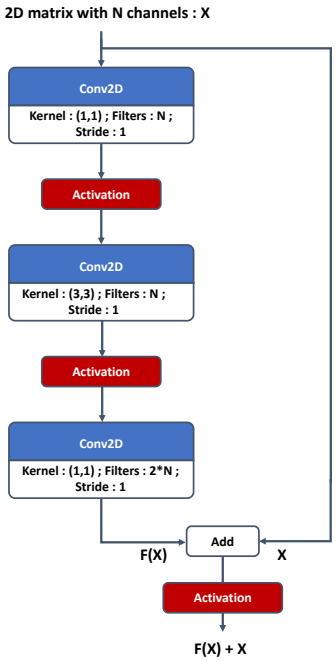
4.2.1 Critic network

$$\text{Critic} : \mathbb{R}^{Nx \times Ny \times 1} \mapsto \mathbb{R} \tag{4.2}$$

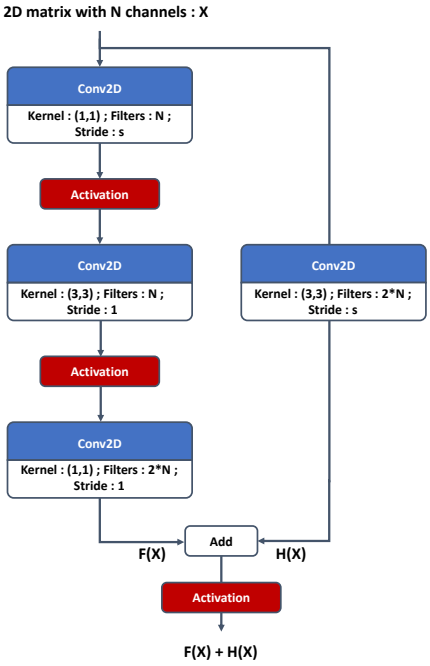
The critic network function Eq. 4.2, input has the shape of a sample from the dataset $X \in \mathbb{R}^{Nx \times Ny \times 1}$. Its output must be a real number because it is an approximation of the Wasserstein distance between the distribution of a batch of images from the dataset and the one from the generator that is being processed. The architecture ends with a dense layer of one neuron with linear activation. The architecture for GAN-CNN is taken from a convolutional classifier with appropriate modification to output binary images, illustrated Figures. 4.2.

The core of the structure for the GAN-ResNet is taken from the residual network and can be seen in Fig. 4.7a. The GAN-ResNet is a classical residual network, starting with a convolution with 7×7 kernels and a succession of convolutional and identity blocks Fig. 4.6a, 4.6b. At each strided convolutional block, $s = 2$ in Fig. 4.6b, the image size is divided by a factor 2. It is equivalent to a learnable pooling layer that can increase the quality of the result [101].

Finally, an average pooling is done, and the output is fed to a fully connected layer of 100 neurons, then to the output neuron. Batch normalization is not present in the critic's architecture following Gulrajani et al. [45]. The batch normalization changes the critic's problem by considering all the batch in the training objective whereas we are already penalizing the norm of the critic's gradient with respect to each sample in the batch.



(a) Residual identity block for the critic



(b) Residual convolutional block for the critic. If s is different from 1 it is referenced as strided convolutional block in Fig. 4.7a

Figure 4.6 – Identity (a) and convolutional (b) block for the critic.

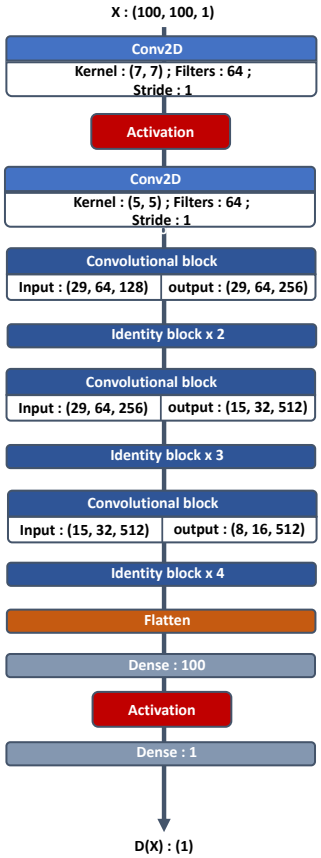
4.2.2 Generator network

The input of the generator network, illustrated in Fig. 4.7b, is an m-dimensional vector containing noise drawn from the normal distribution $\mathcal{N}(0, \mathbf{I}_m)$, for the numerical experiment $m = 32$, which was chosen after training GANs with different latent space dimensions : 8, 32 and 128. Deciding criteria was a compromise between the quality of the generations on metrics described in Sec. 4.4 and the necessity of a limited size of the dimension of parameter space for history matching purposes. Following the guidelines in Zhang et al. [113] and Brock et al. [12] the spectral normalization [82] was implemented in the convolutional layers of the generator. It was proven that it can improve image quality and training speed without increasing computational cost in Zhang et al. [113].

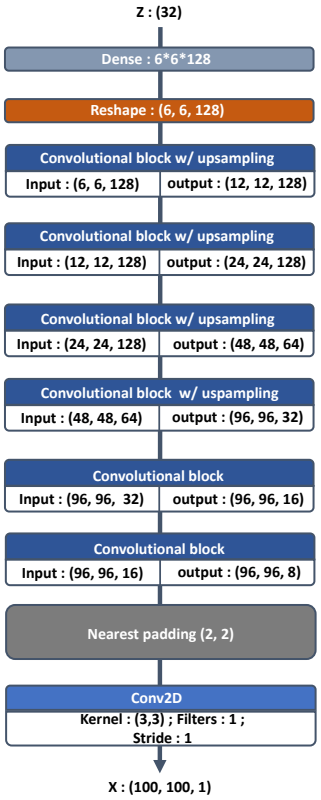
Figures. 4.8a and 4.8b shows the comparison of the 2 points correlation metric for 10000 realizations for different latent space dimensions. The mean of 2 points correlation is matched by the GAN32 and GAN128, even if the standard deviation is overestimated by the GAN32 it was chosen to keep a relatively low dimensional latent space for history matching purpose. The output of the generator has the shape of a sample of the dataset $X \in \mathbb{R}^{Nx \times Ny \times 1}$. The input is passed through a fully connected layer of output shape $(6 \times 6 \times 128)$ and fed to residual blocks. The rest of its architecture is also a residual network with a succession of modified convolutional blocks (relative to the one in the critic network). Modifications of the convolutional block are the following :

1. An up sampling layer is added to increase the image size in some convolutional blocks.
2. Nearest padding layers are added in residual blocks.
3. Spectral normalization is used in convolutional layers.

One could argue that the ReLU activation function is not differentiable in 0, but this is managed



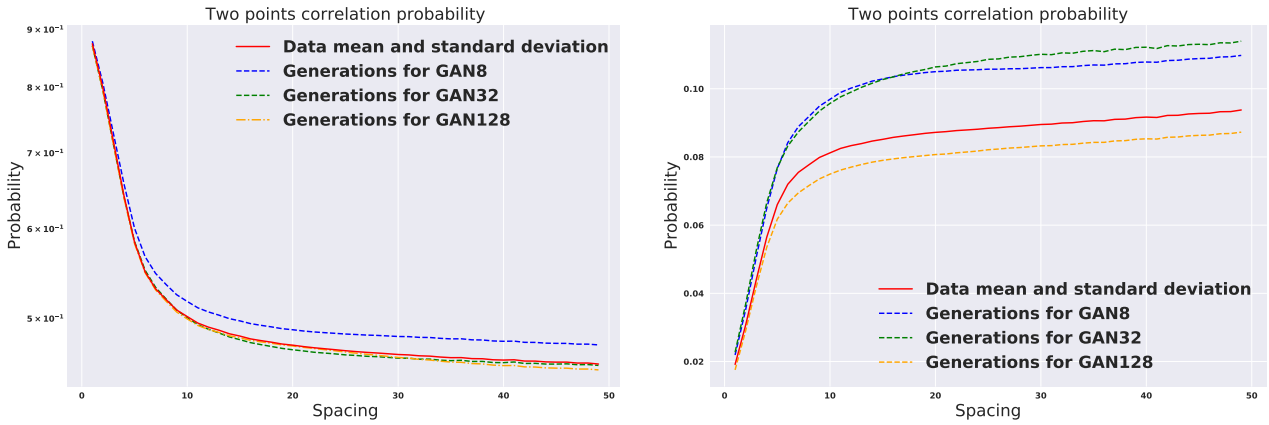
(a) Critic residual architecture.



(b) Generator residual architecture.

Figure 4.7 – GAN Residual network architecture.

by taking the left derivative in the software implementation. The study does not claim that the network architectures used are optimal, the computational burden (described in Sec. 4.3) was too high to run a parameter sensitivity study. Guidelines from Gulrajani et al. [45] were followed and the hyper-parameters were adapted to the current problem. It showcases an example of hyper-parameters producing interesting results, and inspired readers are encouraged to modify and improve this architecture.



(a) Mean of 2 points correlation with respect to their distance for 10000 samples from generated realization with GAN with different latent space dimensions and from dataset.

(b) Standard deviation of 2 points correlation with respect to their distance for 10000 samples from generated realization with GAN with different latent space dimensions and from dataset.

Figure 4.8 – 2 points correlation mean (left) and standard deviation (right) comparison for GANs with different dimensions of latent space. GAN8, GAN32, GAN128 refers to GANs with 8, 32 and 128-dimensional latent space.

4.3 GAN training

	Network	
Hyper parameters	Generator	Critic
Iterations	60 000	350 000
Batch Size	128	128
Optimizer	Adam	Adam
Initial learning rate (lr)	$1e^{-3}$	$1e^{-3}$
λ in Eq. eq:WGANLoss		10

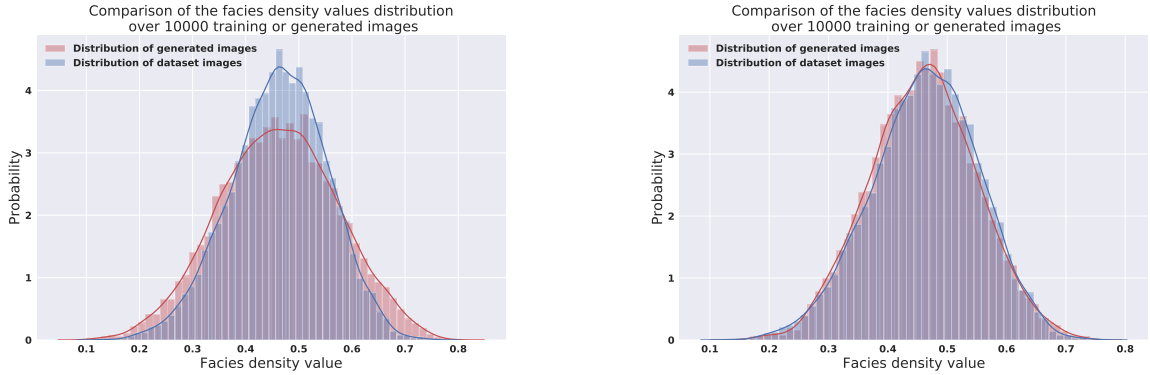
Table 4.2 – Hyper-parameters for training step.

The GAN was trained on an Nvidia Tesla V100 of 16GB of memory for 20 hours. Hyper-parameters are summarized in the Tab. 4.2. For Wasserstein GAN it is common to perform multiple training batches for the critic for each batch processed by the generator. The reader should note that GAN training usually requires many iterations. There is no precise stopping criteria and no direct way to assess the quality of the image without defining specific domain metrics, see Table 4.2. This explains the necessity of defining and testing GAN realizations at different steps of the training to assess the convergence of the GAN training.

4.4 Quality check

To assess the quality of the synthetic generation, it is necessary to define and choose metrics. In reservoir characterization, usual metrics are based on 2 points statistics. After training a GAN, different metrics were verified in order to compare the generations with dataset samples.

After the creation of the GAN-CNN architecture it was important to verify if the diversity of the dataset was retrieved in the generated samples. Figure 4.9a shows the facies density distribution over 10000 samples from the dataset (blue) and 10000 samples generated by the GAN-CNN (red). Facies density is an important parameter that controls the pressure in the reservoir. It appears that distributions are close, but some improvements could be done on the variance of the facies density for generated samples. Figure 4.9b shows the same quantity for GAN-ResNet architecture. It shows a better fit of the facies density distribution on the tails of the distribution and of the most frequent facies densities for the GAN-ResNet architecture.



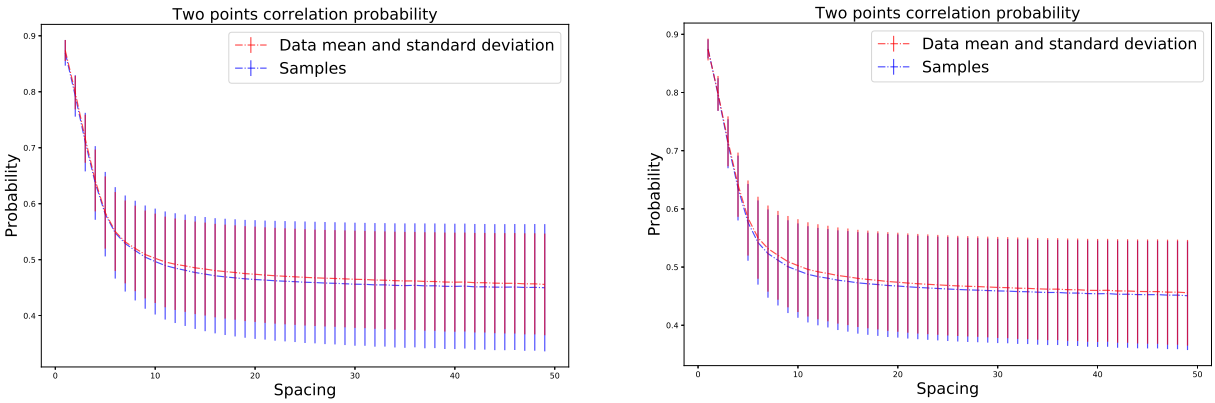
(a) Histograms of facies density for 10000 samples from the training dataset and from the generated samples using GAN-CNN.

(b) Histograms of facies density for 10000 samples from the training dataset and from the generated samples using GAN-ResNet.

Figure 4.9 – Comparison of facies density distribution for 10000 samples from dataset and from generation of GAN-CNN (left) and GAN-ResNet (right).

2 points correlation were also computed to compare the results between the two architectures of GANs. The mean and the standard deviation of this quantity computed on 10000 samples and superposed with the 10000 samples from the dataset samples (blue curve) is visible Fig. 4.10a and 4.10b. The reader can observe a slightly better fit of the standard deviation of the 2 point correlation for the GAN-ResNtet architecture.

It is also interesting to verify correlations between the input and the output of the GAN. Figure 4.11a and 4.11b respectively show the correlation between the facies density of a sample and one component of the latent vector of the corresponding sample, and the correlation between the facies density and the 2-norm of the latent vector. It appears that a strong correlation is present between the 2-norm of the latent vector and the facies density. This implies that a certain structure in the latent space is present without any enforcement during the training. It raises the following question : Is this structure useful for data assimilation algorithms ? These plots can be compared with those of the same quantities applied to samples from the GAN-ResNet Fig. 4.11a and 4.11b. It can be observed that the latent space structure is not present anymore. However, a light correlation can be seen between a latent vector component and the facies density of the corresponding image.



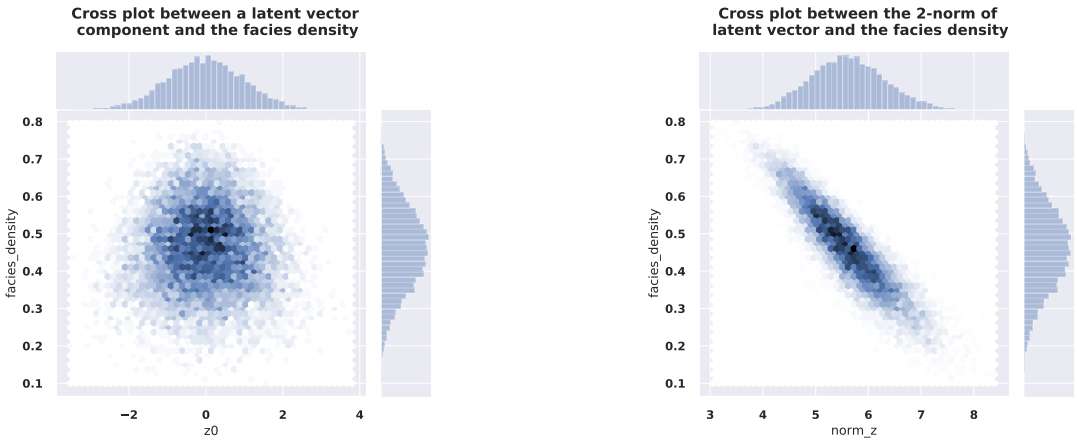
(a) Two points correlation metric and there standard deviation for 1000 samples from the dataset and 1000 samples generated by the GAN.

(b) Two points correlation metric and there standard deviation for 1000 samples from the dataset and 1000 samples generated by the GAN.

Figure 4.10 – Comparison of two points correlation metric and there standard deviation for 1000 samples from the dataset (blue) and 1000 samples generated by the GAN-CNN (red curve in left panel) and GAN-ResNet (red curve in right panel).

Given the different comparisons the GAN-ResNet architecture was chosen to be used as parameterization in the ES-MDA data assimilation techniques. It seems that the GAN-ResNet architecture better represents the dataset variability.

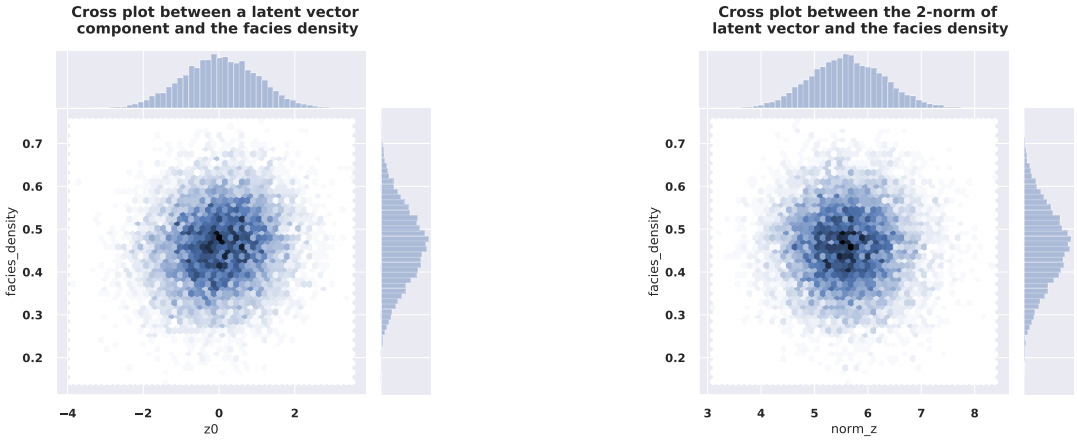
The author would like to emphasize the necessity of adding other metrics for assessing the quality of the generation. This is why the involvement of domain experts are necessary for the design of such parameterization techniques. The creation of score-based challenges competition is one of the most efficient ways to bring deep learning researchers into applied fields such as reservoir characterization and stimulate the improvement of data driven techniques by well-defined competitions.



(a) Correlation plot between the distribution of one component of the latent space and the facies density of the corresponding images.

(b) Correlation plot between the distribution of the 2-norm of latent vectors and the facies density of the corresponding images.

Figure 4.11 – Correlation of latent variables and properties of the corresponding generation for GAN-CNN.



(a) Correlation plot between the distribution of one component of the latent space and the facies density of the corresponding images.

(b) Correlation plot between the distribution of the 2-norm of latent vectors and the facies density of the corresponding images.

Figure 4.12 – Correlation of latent variables and properties of the corresponding generation for GAN-ResNet.

Chapter 5

History matching using GANs

In this chapter a twin experiment was designed to create observations from a true case to perform history matching, described in Sec. 2.3.3. The objective is to recover this true model from the GAN parameterization by assimilating the observation associated with the true case. The true case was taken from the dataset used to train the generative network. One could note that taking the true case from the dataset on which the GAN was trained is too easy, but it must be emphasized that generators never see the dataset images directly. Two test cases are tested, one with horizontal wells another one with a 5 wells disposition.

5.1 Reservoir simulation

This section deals with reservoir simulation with the objective to describe the different simplifications and models used in the history matching cases to simulate the different fluid phase in a porous media. The chosen reservoir simulator is Open Porous Media (OPM) which is a suite of software, an initiative that encourages open innovation and reproducible research on modelling and simulation of porous media processes. It was designed in collaboration by SINTEF, NORCE (formerly IRIS), Equinor,Ceetron Solutions, Poware Software Solutions, and Dr. Blatt HPC-Simulation-Software & Services. The suite is made of different modules, the OPM Flow reservoir simulator [92] is one of them.

5.1.1 Governing equations

5.1.1.1 Black-oil model

The simulator relies on a black-oil model which is the most widely used flow model in hydrocarbon reservoir simulation. It is based on the assumption that 3 pseudo-phases are represented : water w, oil o and gas g. The oil and gas phase represent all the hydrocarbon components at a given thermodynamic state. The main assumptions of a black-oil model are the following :

- The water phase neither dissolves in the oil phase, nor evaporates in the gas phase.
- The oil phase does not dissolve in the water phase, but evaporates in the gas phase.
- The gas phase does not dissolve in the water phase, but can dissolve in the oil phase.

The continuous set of equation is :

$$\begin{aligned}
\frac{\partial}{\partial t}\left[\phi\left(\frac{S_o}{B_o}+\frac{R_V S_g}{B_g}\right)\right]+\nabla\cdot\left(\frac{1}{B_o}\mathbf{u_o}+\frac{R_V}{B_g}\mathbf{u_g}\right)&=0\\
\frac{\partial}{\partial t}\left[\phi\left(\frac{S_w}{B_w}\right)\right]+\nabla\cdot\left(\frac{1}{B_w}\mathbf{u_w}\right)&=0\\
\frac{\partial}{\partial t}\left[\phi\left(\frac{R_S S_o}{B_o}+\frac{S_g}{B_g}\right)\right]+\nabla\cdot\left(\frac{R_S}{B_o}\mathbf{u_o}+\frac{1}{B_g}\mathbf{u_g}\right)&=0
\end{aligned} \tag{5.1}$$

Where ϕ is the porosity of the porous medium, S_w is the water saturation, S_o, S_g are saturations of oil and gas phases in the reservoir. $\mathbf{u_o}$,$\mathbf{u_g}$ and $\mathbf{u_w}$ are Darcy velocities of the oil, gas and water phases. B_o, B_g and B_w are respectively the oil, gas and water formation volume factor, which corresponds to the ratio of some volume of a liquid at reservoir conditions to the volume of the same liquid at standard conditions. Finally, R_S is the ratio of a solution of gas in the oil phase, and R_V is a ratio of a vaporized oil in the gas phase. The phase fluxes are given by Darcy's law :

$$\mathbf{u_\alpha} = -\lambda K\left(\nabla p_\alpha - \rho_\alpha g\right) \tag{5.2}$$

where K is the permeability of the porous media, α is the phase subscript (w, o or g), λ_α is the mobility of phase α, given by $\lambda_\alpha = k_{r,\alpha}/\mu_\alpha$. μ_α and ρ_α are respectively the viscosity and the density of the phase α, and $k_{r,\alpha}$ is the relative permeability of the phase α defined by $k_{r,\alpha} = k_\alpha/K$ with k_α the permeability of phase α.

$$\begin{aligned}
S_w + S_o + S_g &= 1,\\
p_{c,ow} &= p_o - p_w\\
p_{c,og} &= p_o - p_g
\end{aligned} \tag{5.3}$$

where $p_{c,\alpha\beta}$ is the capillary pressure between phase α and β.

5.1.1.2 Initial and boundary conditions

The initial conditions are defined by the initial values of pressures p, saturations S_α and mixing ratios R_S and R_V if present defined by the user or computed from an equilibrating procedure not detailed here (see [92]). Boundary conditions are set as no-flow Neumann *i.e.*the reservoir has no fluid communication with surrounding rocks. The fluid communication is with the wells model described in the next paragraph. Finally, the equations are discretized in space with an upwind finite-volume scheme using a two-point flux approximation and in time using an implicit (backward) Euler scheme.

5.1.1.3 Well models

Another boundary condition remains to be described, the well models. The production data that have to be matched, are measured at the wells, these are critical to the history match process. OPM provides two different well models : the "Standard" well model, and the multi-segment well model. The standard model describes the flow conditions in each well with a single set of primary variable

[53]. For a 3-phase black oil system the equation for well p is :

$$Q_t = \Sigma_{\alpha \in \{o,g,w\}} g_\alpha Q_\alpha, \quad F_w = \frac{g_w Q_w}{Q_t}, \quad F_g = \frac{g_g Q_g}{Q_t} \tag{5.4}$$

Where Q_t is the weighted total flow rate and Q_α the flow rate of component α, F_w and F_g the weighted fraction of water and gas in the well. The inflow rate for well W at reservoir conditions are calculated as :

$$q^r_{\alpha,j} = T_{W,j} M_{\alpha,j} \left[p_j - (p_{bhp,W} + h_{W,j}) \right] \tag{5.5}$$

where $q^r_{\alpha,j}$ is the flow rate of phase α through cell connection j, $T_{w,j}$ is the connection transmissibility factor, $M_{\alpha,j}$ is the mobility of phase α at cell connection j, $p_{bhp,w}$ is the bottom -hole pressure of the well W and $h_{W,j}$ is the pressure difference within the well bore between connection j and the well's bottom-hole datum depth. The flow rate is positive for flow from the reservoir to the well bore and negative in the other way. The system is closed with this last equations :

$$R_{\alpha,w} = \frac{A_{\alpha,W} - A^0_{\alpha,W}}{\Delta t} + Q_\alpha - \Sigma_{j \in C(W)} q_{\alpha,j} \tag{5.6}$$

here $C(W)$ is the set of connections of the well W, $A_{\alpha,W}$ is the amount of component α in the well bore, introduced for better stability of the well solution. Finally, two equations are needed to represent how the wells are controlled. A well can be controlled by an imposed bottom-hole pressure target :

$$R_{c,W} = p_{bhp,W} - p^{target}_{bhp,W} \tag{5.7}$$

Or it can be controlled by an imposed flow rate :

$$R_{c,W} = Q_\alpha - Q^{target}_\alpha \tag{5.8}$$

The other well model available in OPM, the multi-segment well model, can be used when the cross flow phenomenon is observed. The cross flow phenomenon occurs when fluids enter the well bore by some connections of the well bore and get reinjected into the reservoir into another connection of the same well. Because of the absence of multi segment wells in this study, it will not be detailed. The reader can refer to [92] for more information.

5.2 Results of ESMDA using GAN parameterization

5.2.1 Problem description

In the reservoir application, the objective is to find models of the reservoir's physical properties that output predictions that fit observations gathered during the reservoir exploitation. For now, the ES-MDA (Ensemble Smoother with Multiple Data Assimilation [27], described in Sec. 2.2 is used because it manages to assimilate high dimensional data in a reasonable amount of simulations. Because the problem is under constrained, many potential reservoir models fit our observations. The consequence is that a probability distribution is sought as an answer to the ill-posed inverse problem. To rephrase, the solution must have enough variability to quantify the probability of finding oil by drilling a well at a specific location or estimate the oil quantity a given reservoir will produce.

The GAN parameterization allows generating sharp geological heterogeneities such as channels compared to common techniques, and it uses continuous parameters drawn from a multi-dimensional

normal distribution to encode the set of realistic reservoir models. On the other hand, it adds another function, the generator neural network that is a source of non-linearity in the dynamical function Eq. 5.9. Some recent work showed results using the ES-MDA algorithm with a GAN parameterization of the reservoir models [16], [5].

$$\begin{aligned} \mathbb{R}^{32} &\mapsto \mathbb{R}^{N_o} \\ z &\mapsto OPM(GAN(z)) \end{aligned} \tag{5.9}$$

The question is : is the ensemble formulation adapted to the problem of history matching with parameterization by the GAN of the geological heterogeneities ?

The problem described here can be seen as a multi-modal problem to solve, because of the non-monotonic relation between the GAN parameters and the observation data. In this case, the ensemble method is not suited to get different solutions with enough variability and this is what is demonstrated in the toy experiment 2.5. However, the experiment is a simplification of reality and does not use all the possible observations available in a real case. The addition of other observations such as seismic observation increases the regularization imposed on the inverse problem and allows the problem to be well conditioned. The objective here is to find a solution and its associated uncertainty represented by the variance of the latent vector at the final iteration of ES-MDA.

5.2.2 Reservoir model description

In this experiment, history matching was used to determine the spatial distribution of different variables (porosity, permeability) that have an influence on the predicted variables, well bottom hole pressure (WBHP) computed at reference depth, well oil production rate (WOPR) at surface conditions and the well water cut (WWCT) that is defined by the ratio of the water rate at surface conditions and the fluid (oil and water) rate at surface conditions. These quantities are available for every well. The history matching is known to be difficult because of the high-dimensionality of the parameter space, the non-linear relation between parameters and observation spaces, and the non-Gaussian distribution of the parameters. The non-linearity and the high dimension can be managed by using particular methods such as iterative Kalman smoother. Specifically, Ensemble smoother for Multiple Data Assimilation (ES-MDA) is a popular method [1].

To deal with the non-Gaussian distribution of the facies parameter, independently of the present work Canchumuni et al. [16] proposed to use a generative adversarial network (GAN) as a parameterization. The idea is to perform the history matching of the static variable in the latent space where variability for the data assimilation algorithm can be obtained by drawing from known distributions.

5.2.2.1 Inverse problem formulation.

As a simplification, the permeability and the porosity are supposed to be constant for each rock type, also called facies. In this way, the GAN has to learn only the rock type spatial distribution. A twin experiment was designed by taking a geological model in the training set, running the porous media fluid flow simulator and retrieving the well logs of this run y_{obs}. Uncertainties are associated with the observation to represent the measurement error. Uncertainty values were chosen for representing typical errors found in reality : 3 bars for WBHP, 10% of the WOPR measure and 10% of the WWCT. An uncertainty of 0.01 was added for the WWCT when water is absent from the extracted fluid. The

exact data assimilation procedure Fig. 5.1, is as follows :

- We draw N vectors $(z_i)_{i=1...N} \in \mathcal{N}^{32}(0,1)$ in the input space of the GAN.
- Once z is passed through the generator function G, the output is a 2D reservoir modelisation X which is a 2D matrix of size 100x100 pixels. $G(z) = X$
- The reservoir modelisation X is then pass in a porous media fluid flow simulator, OPM. It outputs the well logs y_{pred}. $y_{pred} = OPM(x) = OPM \ o \ G(z)$ which is the dynamical function.
- Then error between the observations and the predictions of the ensemble is computed : $r = y_{obs} - y_{pred}$.
- Finally, ESMDA algorithm is used to correct the rock type spatial distribution parameters z.

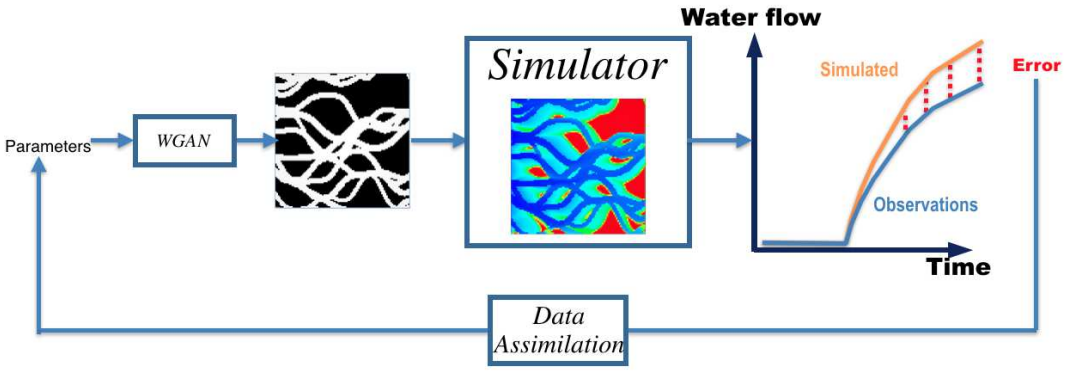


Figure 5.1 – Scheme of the data assimilation loop.

5.2.3 Horizontal wells test case

The first test case consists of two horizontal wells, named AI1 for the injector and P1 for the producer, respectively at the West and East sides of the reservoir. A horizontal well injects or extracts oil on all sides of the reservoir as visible in Fig. 5.2. This simplified test case allows us to see the behavior of the model without unnecessary complexity. One should note that the facies proportion is the most determinant parameter because of the horizontality of the well, the connectivity has a reduced influence on the predicted data. The reservoir fluid flow simulation parameters are :

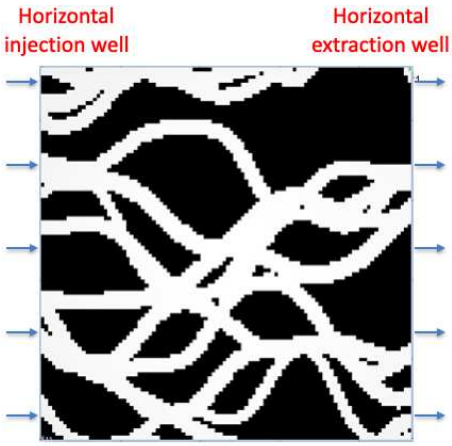


Figure 5.2 – Scheme of reservoir

- A black-oil model with 2-phase fluid (oil/water) with no gas.

Data assimilation parameters	
Hyper parameters	value
ESMDA iterations	5
Ensemble members	100
Stdd of noise added to observation	0.01

Table 5.1 – Hyper-parameters for horizontal wells experiment.

- The Corey model was used for relative permeability with no capillary pressure.
- Wells are controlled with historical control rates issued from forecast simulation of the same case.

The imposed fluid flow constraint is the same one as the one used to create the true case in the twin experiment. It means that the only parameters being estimated is the spatial distribution of facies, controlled by the latent space of the GAN. A fluid flow simulation using OPM was conducted in order to generate the well data such as WOPR, WWCT at well P1 and WBHP at well P1 and AI1. For the data assimilation process the ES-MDA algorithm [27] was used. Table 5.1 shows the set of parameters used for the experiment.

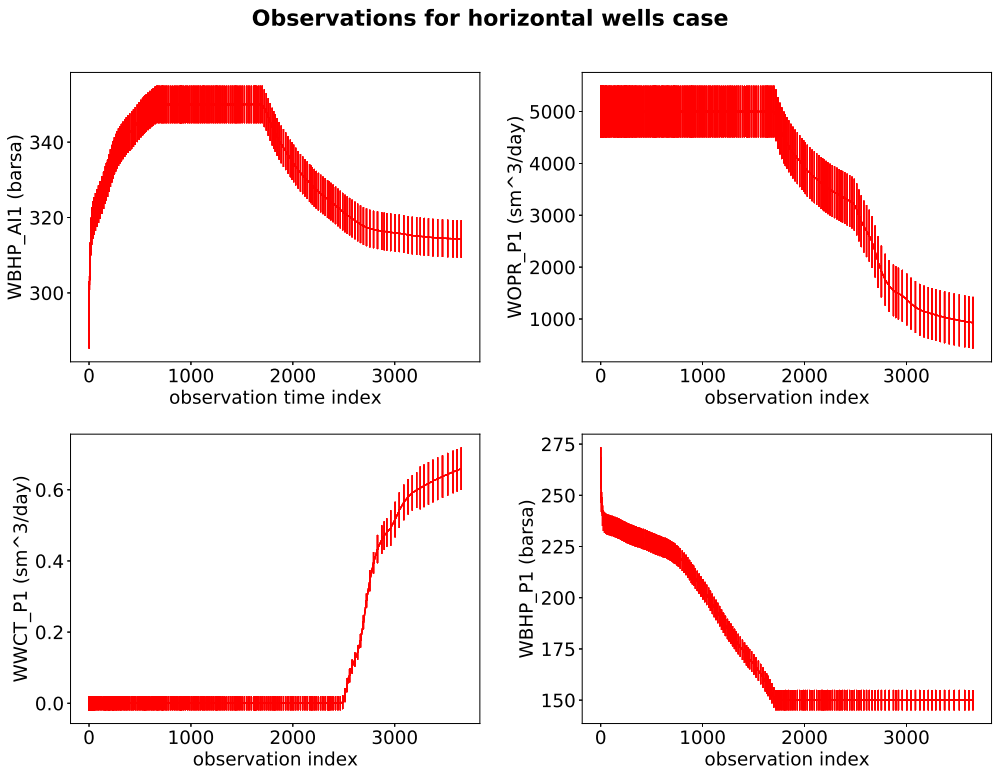


Figure 5.3 – Constraint on dynamical data for the horizontal well test case. Red curve represent the value of each variable assimilated. The bars represent the uncertainty on the measures.

5.2.4 Results for horizontal wells test case

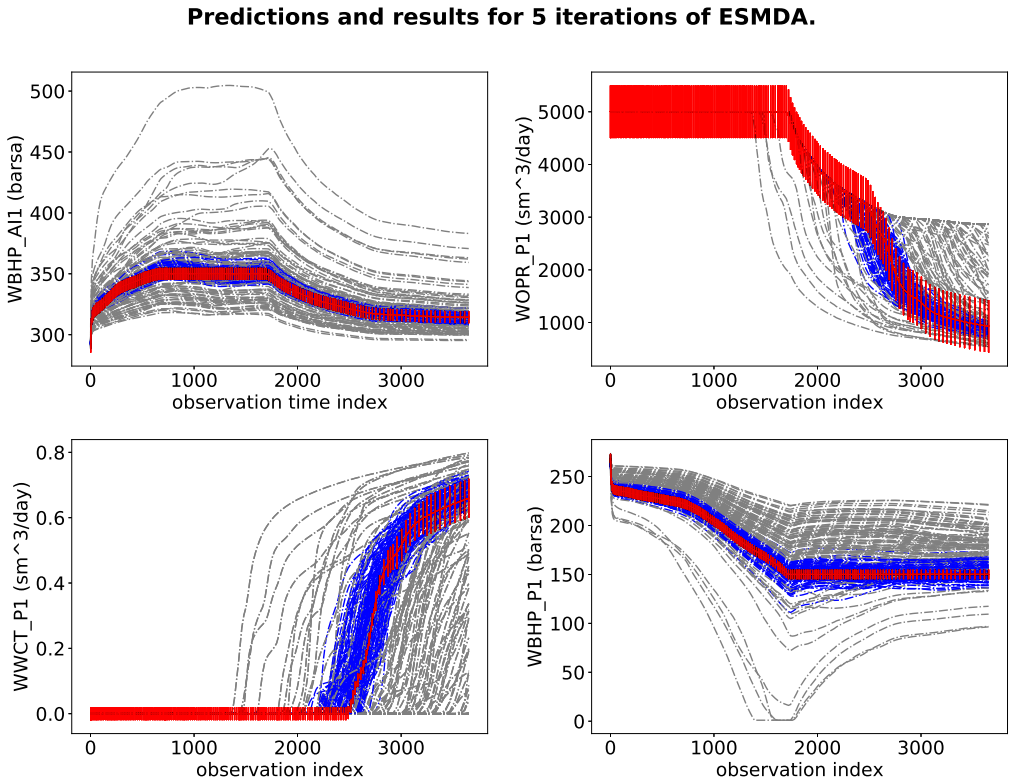


Figure 5.4 – Results of a history match on horizontal wells test case. ESMDA algorithm was used with 5 iterations and 100 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 5th iteration.

The results for ESMDA with 5 iterations and 100 ensemble members are shown in Fig. 5.4. A reduction of ensemble variance on the predictions is observed as the ESMDA iterations advance. The method converges to an ensemble that better fits the target distribution than the initial ensemble. Figure 5.5 shows the reduction of data mismatch, defined in Eq. 5.10, after each ESMDA iteration. It is also important to analyze the ensemble as numerical reservoir models.

Figure 5.6 represents the ensemble mean and standard deviation (std) at final iteration. One can see that channels are visible in the mean subplot, showing that a majority of ensemble members have the same channels. This means a low variability, visible in the pixel standard deviation plot, in the ensemble members whereas, for horizontal wells test cases many solutions are possible because mainly driven by the facies density in the reservoir model. The reservoir model corresponding to the final ensemble can be seen in the Fig. 5.8.

Finally, Fig. 5.7 shows the distribution of the parameters at initial and final ES-MDA iteration. These distributions seem to be close to Gaussian distributions. Moreover, the range of the parameter values are included in the range of a multivariate normal distribution. One should note that during the training the latent vectors are drawn from the multivariate normal distribution which implies that realistic generations are only guaranteed for latent vectors drawn within the range of this distribution.

The conclusion of this first test case is that heterogeneities in the final ensemble are perfectly

represented thanks to the GAN parameterization. Images are binary, and shows small differences on the channel's positions. Five ensemble iterations were sufficient for convergence under uncertainty bars of observations, other runs with different numbers of ESMDA iterations were done in Sec. 5.2.4.1 to study the influence of this hyper-parameter.

$$DM = \sum_{i=1}^{N_{obs}} \frac{|obs_i - pred_i|}{\sigma_{obs,i}} \tag{5.10}$$

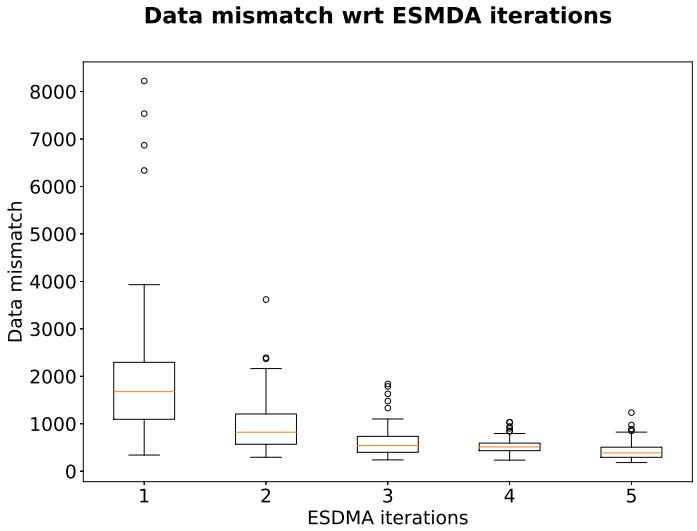


Figure 5.5 – Data mismatch distribution over the 100 ensemble members for each of the 5 ESMDA iterations.

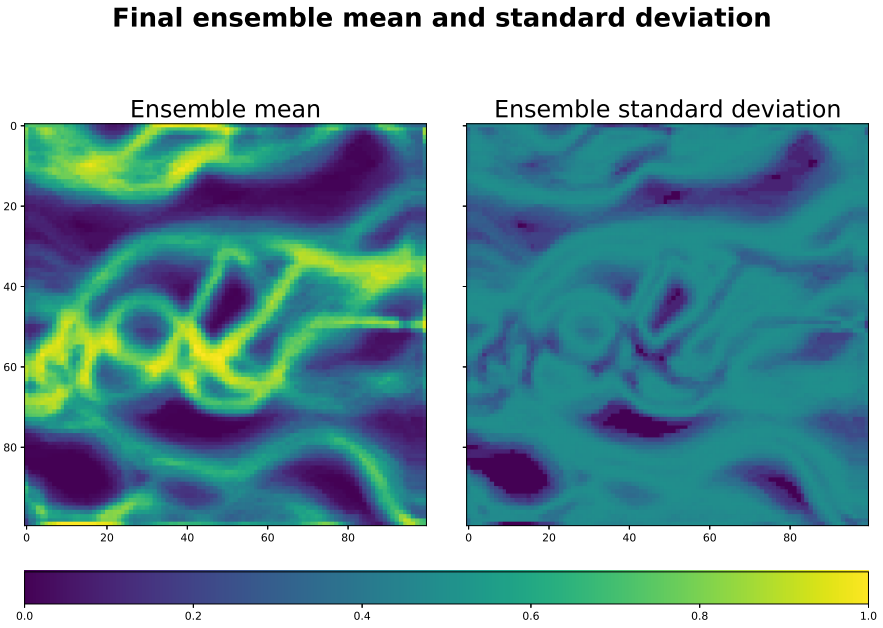


Figure 5.6 – Mean and std of the ensemble in the image space for a run with 5 iterations and 100 ensemble members.

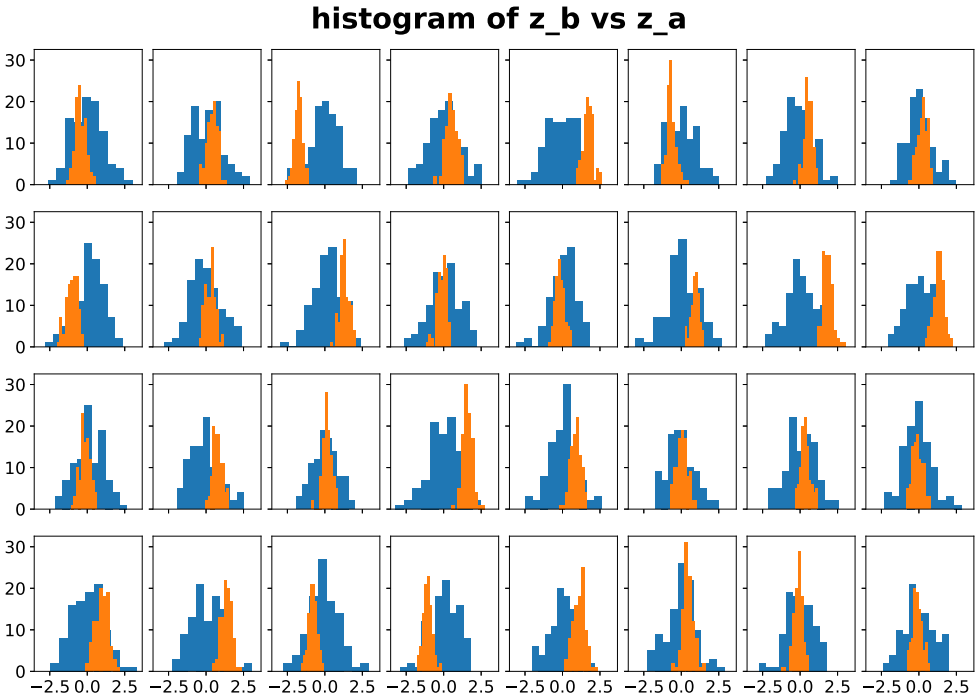


Figure 5.7 – Comparison of the distribution of each component of the latent vector of initial ensemble (blue) and final ensemble (orange) for a run with 5 iterations and 100 ensemble members.

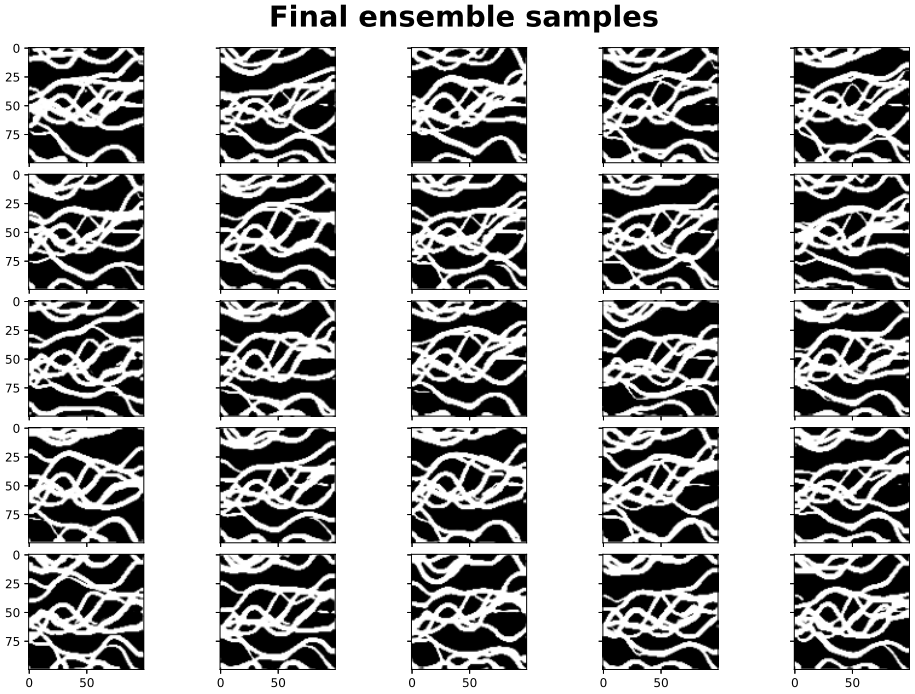


Figure 5.8 – Samples of the final ensemble for a run with 5 iteration and 100 ensemble members.

5.2.4.1 Number of ESMDA iterations

The number of ESMDA iterations is an important parameter that has to be defined a priori by the user. It is possible to see its influence by comparing a run with 5 and 30 ESMDA iterations, respectively illustrated Fig. 5.4 and Fig. 5.9. Figure 5.10 shows the data mismatch distribution of the final ensembles for runs with 5, 10, 20 and 30 ESMDA iterations. It shows that the more ESMDA iteration the more the uncertainty is reduced and predictions fit observations. However, it can also be seen in Fig. 5.11b and 5.13 where results for 5 ESMDA iterations are reminded for visual comparison, an ensemble collapse. In the sense that the variability of the ensemble is almost nonexistent due to a low variance in the ensemble of latent vectors. An ensemble collapse is not a satisfying solution knowing that the history match will be used to forecast the quantity of oil available in the future exploitation of this reservoir. The more variability obtained in the final ensemble the more the distribution of possible solutions is sampled. Largely different solutions are not possible to retrieve for an ensemble data assimilation algorithm because of the ill-posed problem. Our case does not aim at finding the multi-modal solution because additional constraints used in these cases such as seismic images are not used. Instead, our study aims at getting the maximum variance possible that remains in the uncertainty bars of observations. Mode collapse will be tackled in Sec. 5.2.4.2.

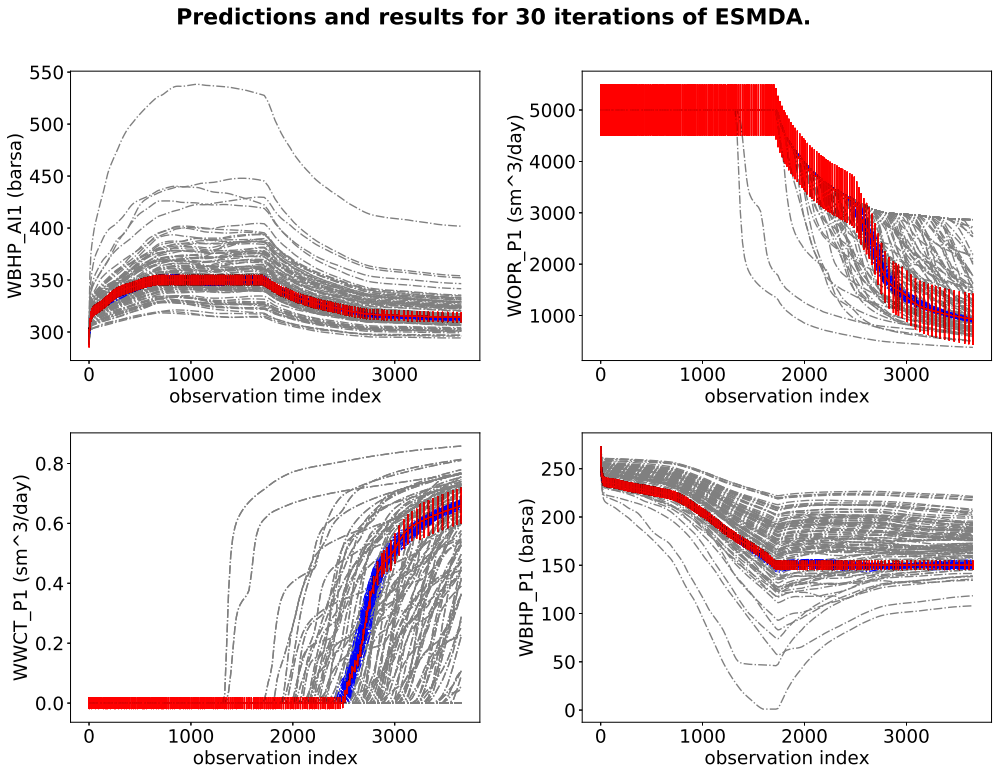


Figure 5.9 – Results of a history match on horizontal wells test case. ESMDA algorithm was used with 30 iteration and 100 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 5th iteration.

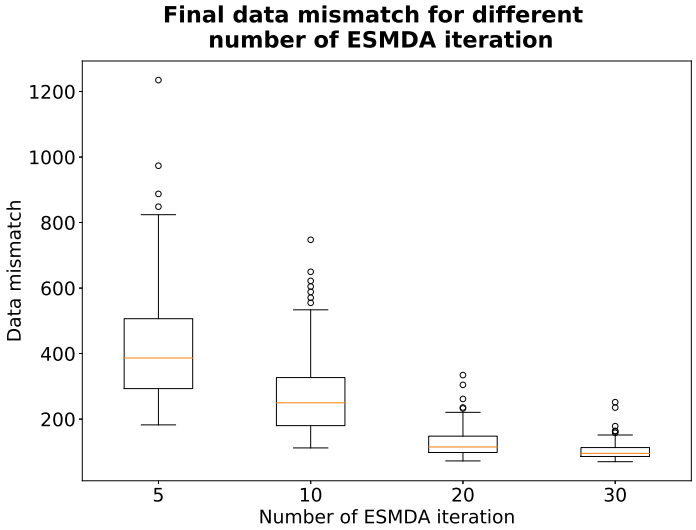
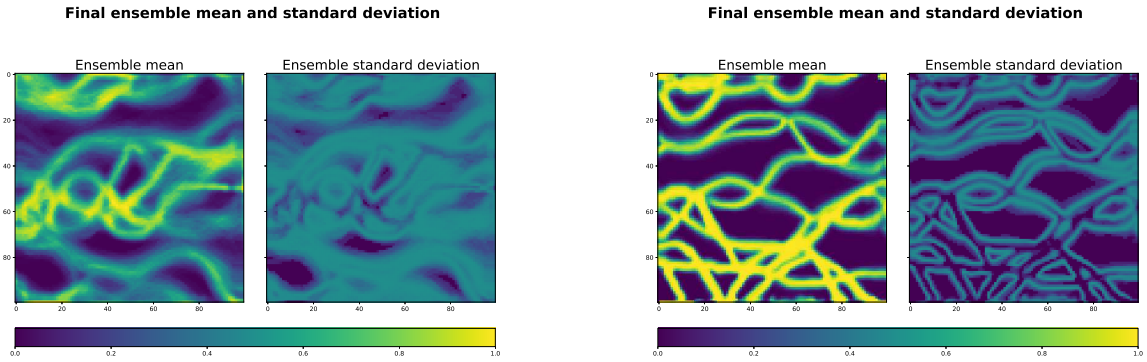


Figure 5.10 – Results of a history match on horizontal wells test case. ESMDA algorithm was used with 30 iteration and 100 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 5th iteration.



(a) Mean and std of the ensemble in the image space for a run with 5 iterations and 100 ensemble members.

(b) Mean and std of the ensemble in the image space for a run with 30 iterations and 100 ensemble members.

Figure 5.11 – Comparison of mean and std of analysis for a run with 100 ensemble members and 5 (left) and 30 (right) ESMDA iterations.

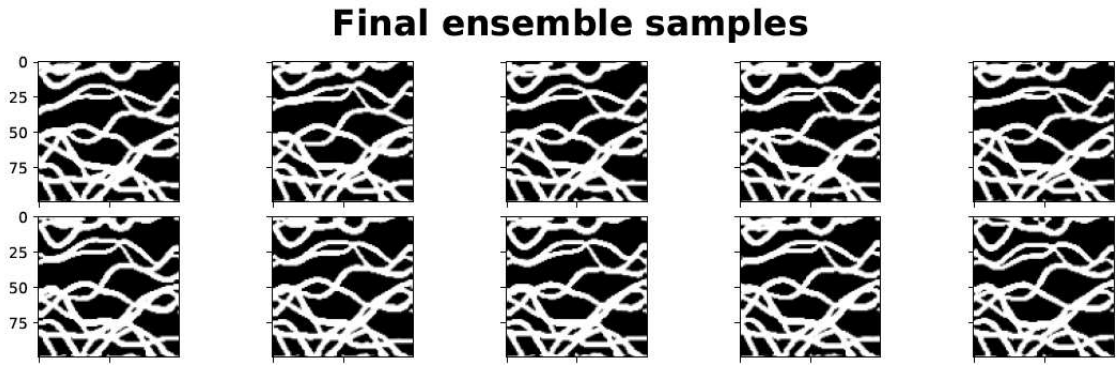


Figure 5.12 – Samples of the final ensemble for a run with 30 iteration and 100 ensemble members.

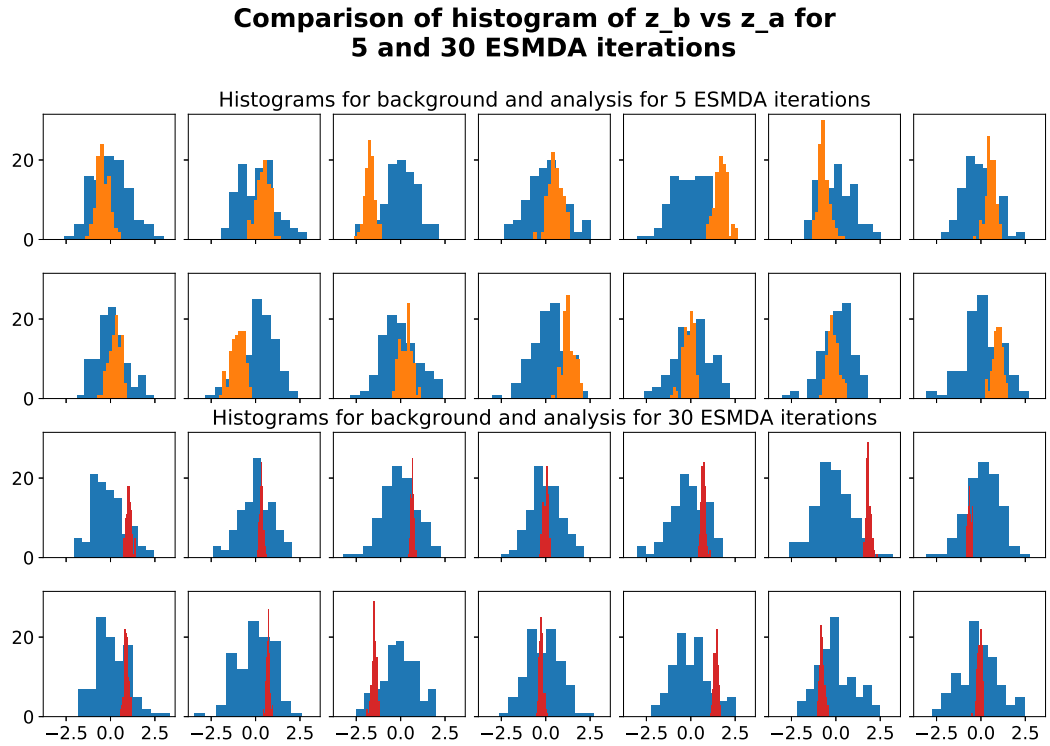


Figure 5.13 – Comparison of the distribution of each component of the latent vector of initial ensemble (blue) and final ensemble for a run with 5 ESMDA iterations (orange), and for a run with 30 iterations (red). Both run were done using 100 ensemble members.

5.2.4.2 Ways of avoiding ensemble collapse

The ensemble collapse is the consequence of the analysis rank deficiency described in Sec. 2.3.5. This rank deficiency happens in the case where $N_{ens} \leq N_{obs}$ [61]. To remove the ensemble collapse phenomenon it is necessary to increase the number of ensemble members or to use the subspace inversion described in Sec. 2.3.5.

Ensemble size The influence of the ensemble size is shown in this paragraph. Ensemble of 500 and 1000 members were used in this study, the results with 1000 ensemble members will be described for comparison. The cost of the total simulations is increased by a factor 10 compared to the cases in 5.2.4.1 for the same number of ESMDA iterations. 10 ESMDA iterations are necessary to have most of the predictions under uncertainty bars of observations. Figure 5.14 shows a data mismatch quality that have a better variability in the sense of fulfilling the observation uncertainties with spatially different realizations. The ensemble size can be a way to increase the variability of the ensemble as shown in Fig. 5.15, 5.16 and 5.17. Figure 5.15 underlines the fact that realizations do not have particular redundant channels over the entire final ensemble and can be interpreted as a better analysis that samples more efficiently the possible realizations that fit observation. It should be reminded that a high number of numerical reservoir model fit observation due to horizontal well, it will be demonstrated in Sec. 5.2.5 that for more complex test cases, mean of the ensemble in the image space must have more visible properties. However, it is not always possible to increase the ensemble size on a real case reservoir because of the important computing time due to the size of the reservoir numerical model, such as in the 5SPOTS case in Sec. 5.2.5.

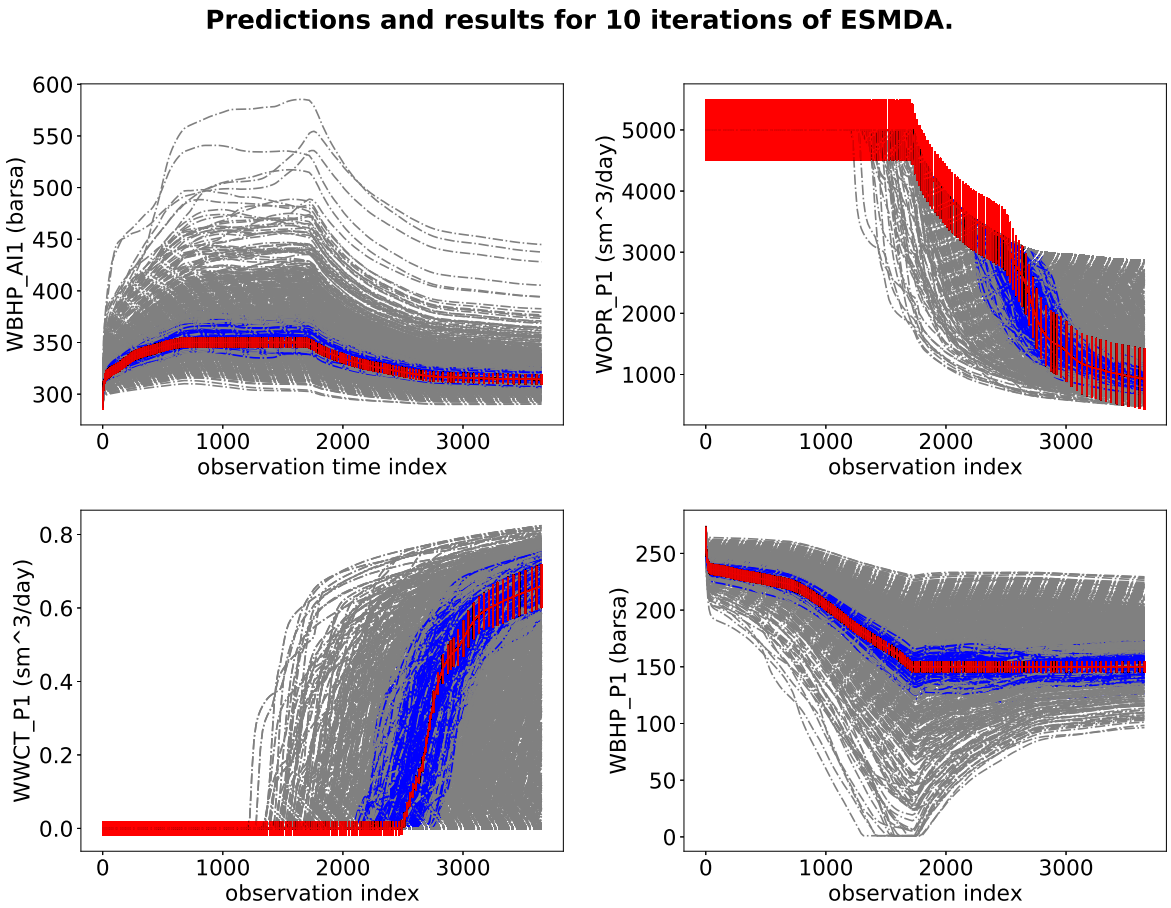


Figure 5.14 – Results of a history match on horizontal wells test case. ESMDA algorithm was used with 10 iteration and 1000 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 5th iteration.

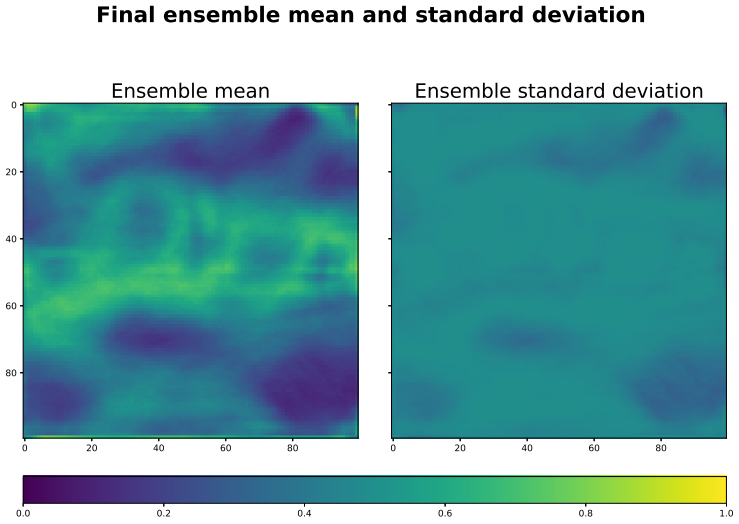


Figure 5.15 – Mean and std of the ensemble in the image space for a run with 30 iterations and 100 ensemble members.

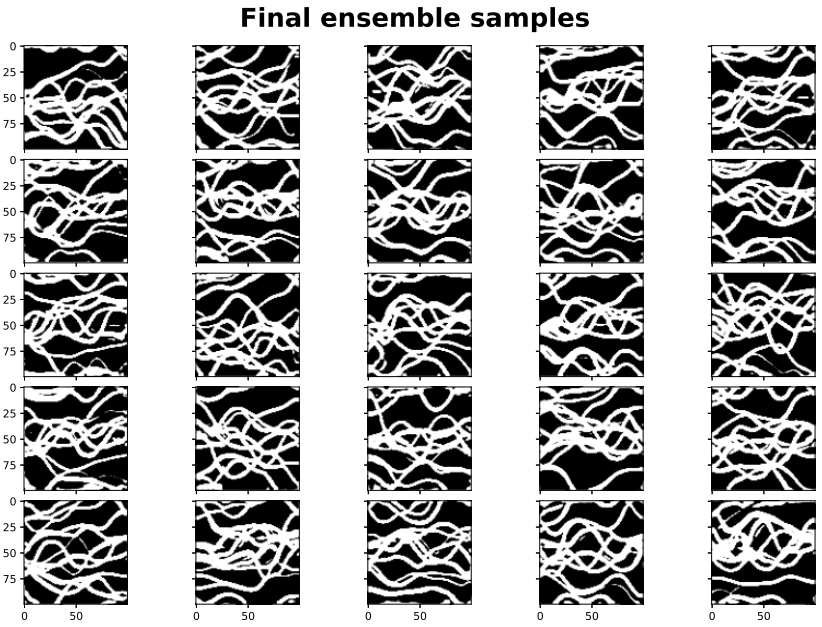


Figure 5.16 – Samples of the final ensemble for a run with 30 iteration and 100 ensemble members.

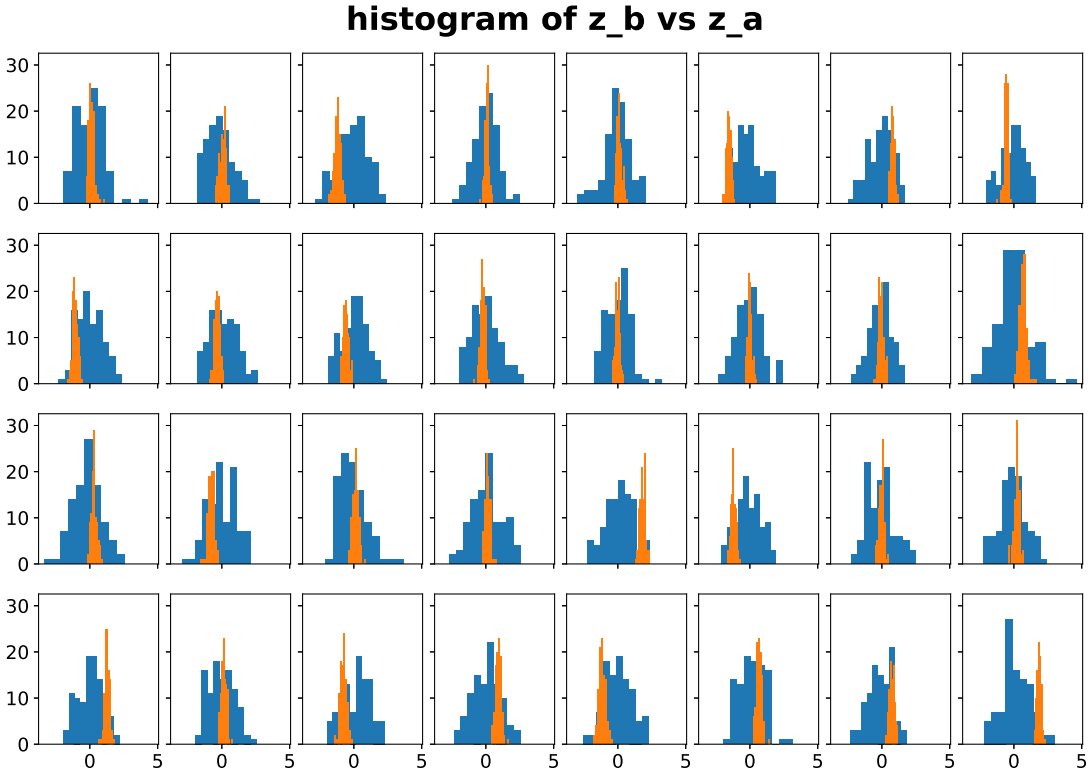


Figure 5.17 – Comparison of the distribution of each component of the latent vector of initial ensemble (blue) and final ensemble (orange) for a run with 10 iterations and 1000 ensemble members.

5.2.5 Five wells case (5SPOTS)

A second case was created with 5 different wells described in Fig. 5.18. At the center of the reservoir a well injector (AI1) is present, and 4 well producers (P1, P2, P3, P4) are placed around the producer. The objective of this case, referred to as the 5SPOTS case, is to give more importance to the connectivity of the heterogeneities, compared to the horizontal well case where the facies density was the main characteristic to match. Other properties such as porosity and permeability for each facies and reservoir model are unchanged. An important property to expect in the history matching is the facies at the wells' location at the end of the data assimilation routine that are necessary to allow close predictions from observations. Facies at well locations are known and determine the response of the reservoir when under constraint such as injection or extraction of a fluid at the well. Most of the time, permeable facies are present at well bore because fluids will be mostly advected by the most permeable and porous media.

First a run with 15 ESMDA iterations and 100 ensemble members will be performed. Observations are visible for each well Fig. 5.19, 5.20, 5.21 and 5.22. Uncertainties of observations are defined in the same way as in the horizontal case.

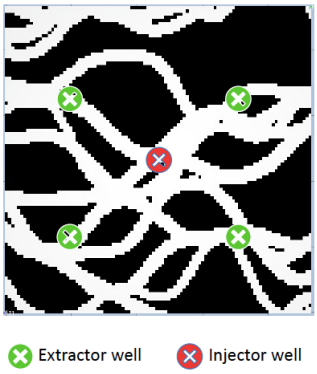


Figure 5.18 – Scheme of 5SPOTS case. At the center of the reservoir a well injector (AI1) is present, and 4 well producers (P1, P2, P3, P4) are placed around the producer.

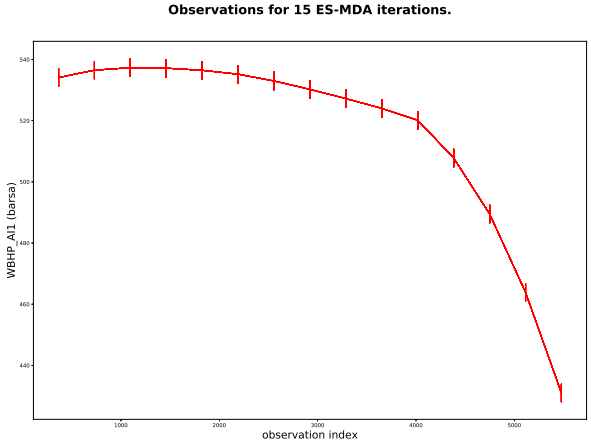


Figure 5.19 – Observations of WBHP for 5SPOTS case at injector well.

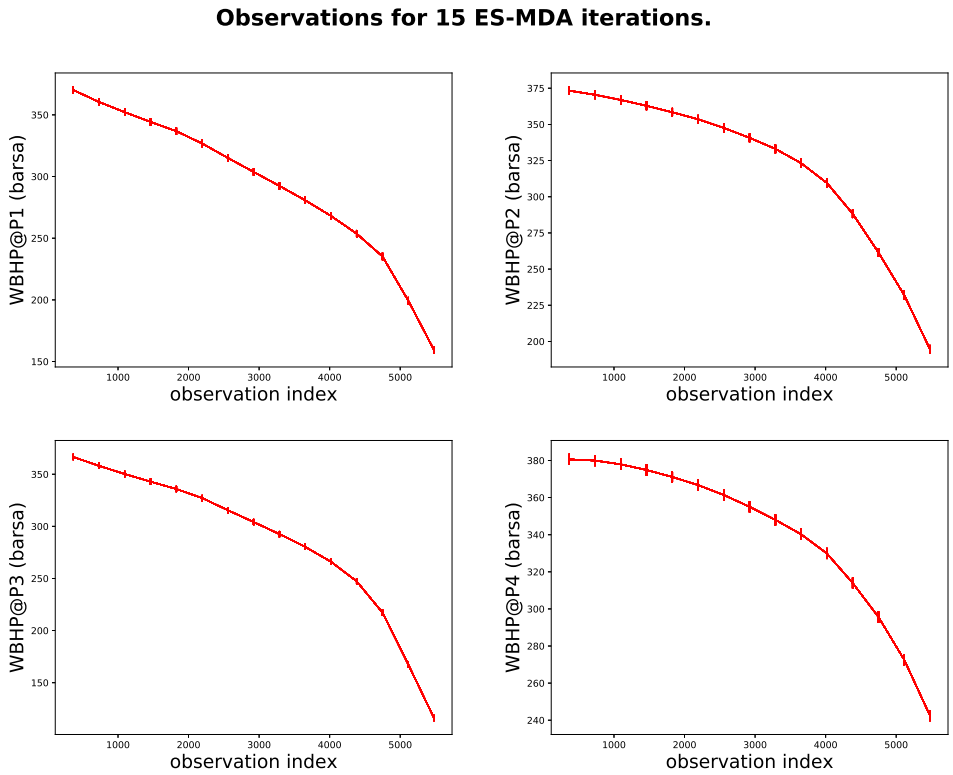


Figure 5.20 – Observations of WBHP for 5SPOTS case at producer wells.

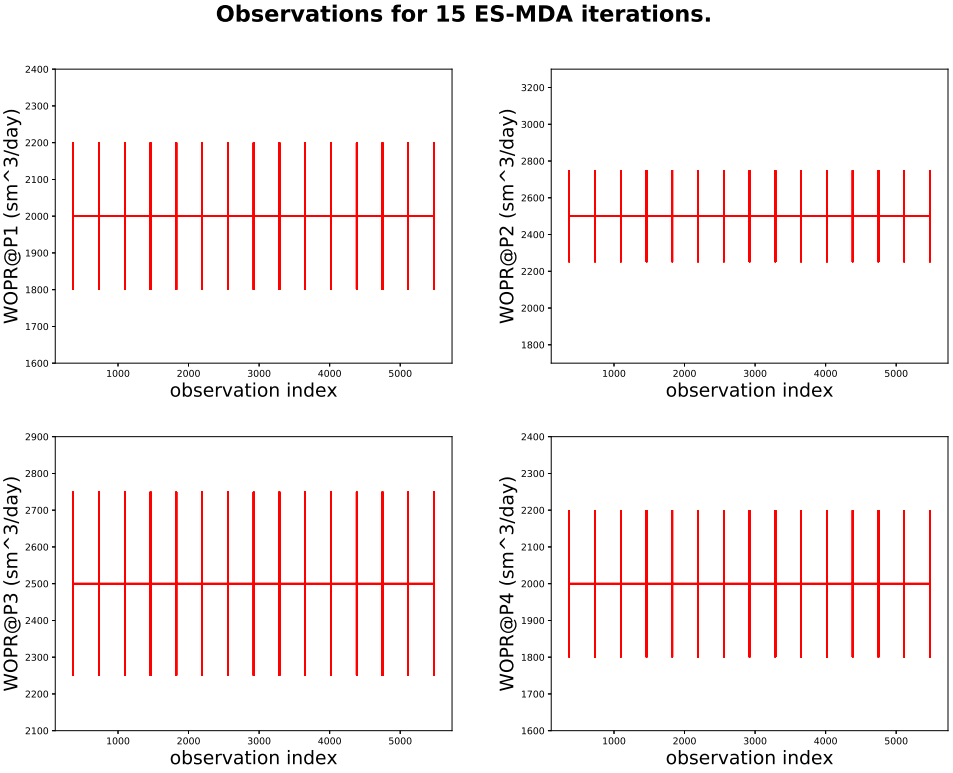


Figure 5.21 – Observations of WOPR for 5SPOTS case at producer wells.

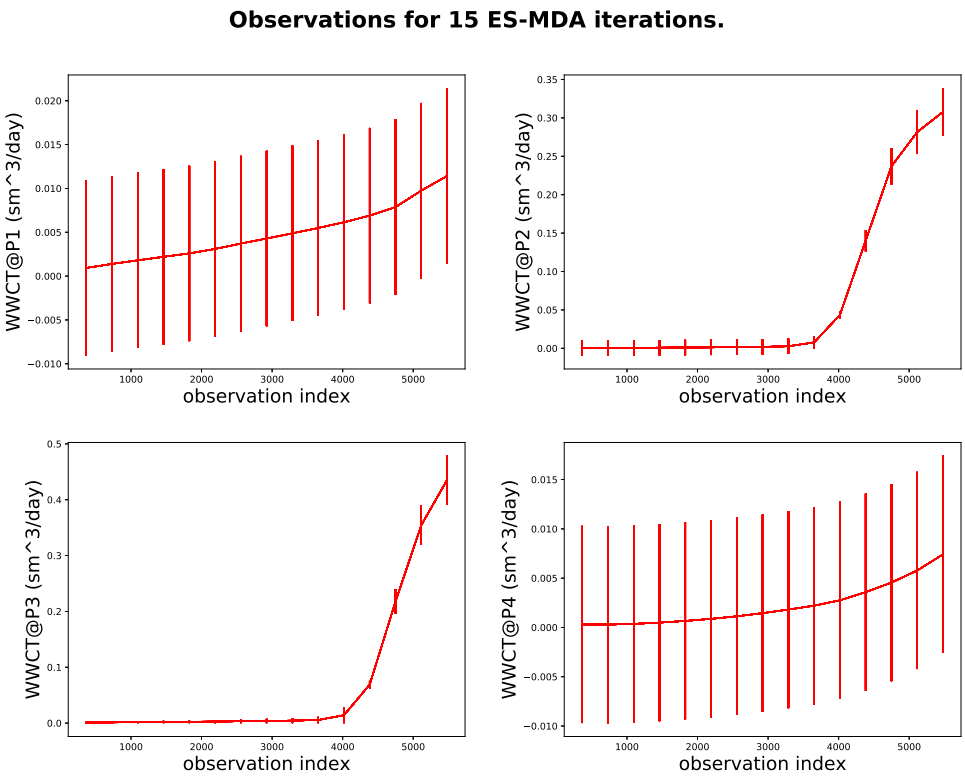


Figure 5.22 – Observations of WWCT for 5SPOTS case at producer wells.

5.2.5.1 Results for 5SPOTS case

A first run was performed with 15 ESMDA iterations for 100 ensemble members. The history matching results, illustrated Fig. 5.23, 5.24, 5.25 and 5.26 show a convergence to a satisfying solution regarding the uncertainty bars on observation. However, the Fig. 5.27 shows a similar ensemble collapse as in the previous case. The same experiment will be reproduced using the subspace inversion in order to alleviate the ensemble collapse and have an increased diversity in the final ensemble.

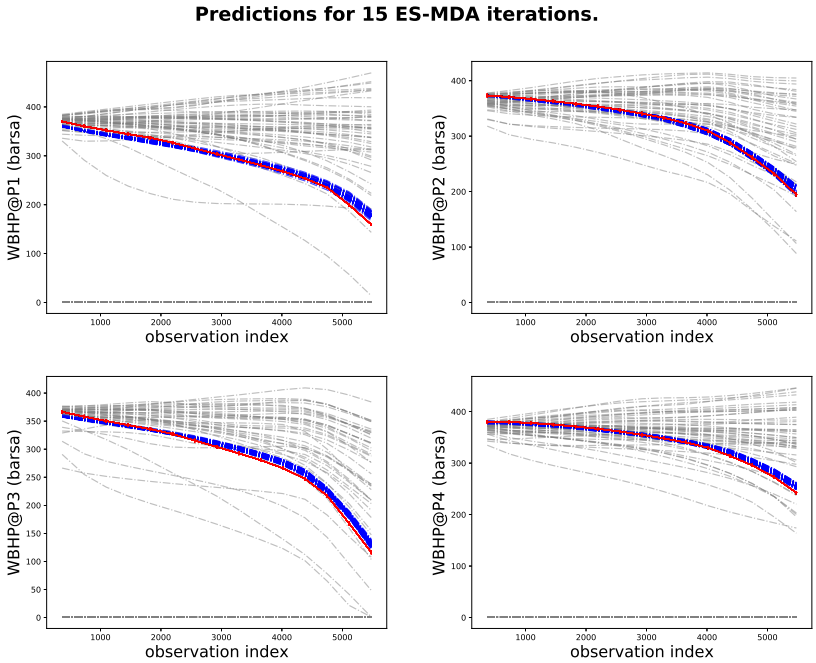


Figure 5.23 – Results of history match for WBHP at 4 producer wells for 5SPOTS case. ESMDA algorithm was used with 15 iterations and 100 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 15th iteration.

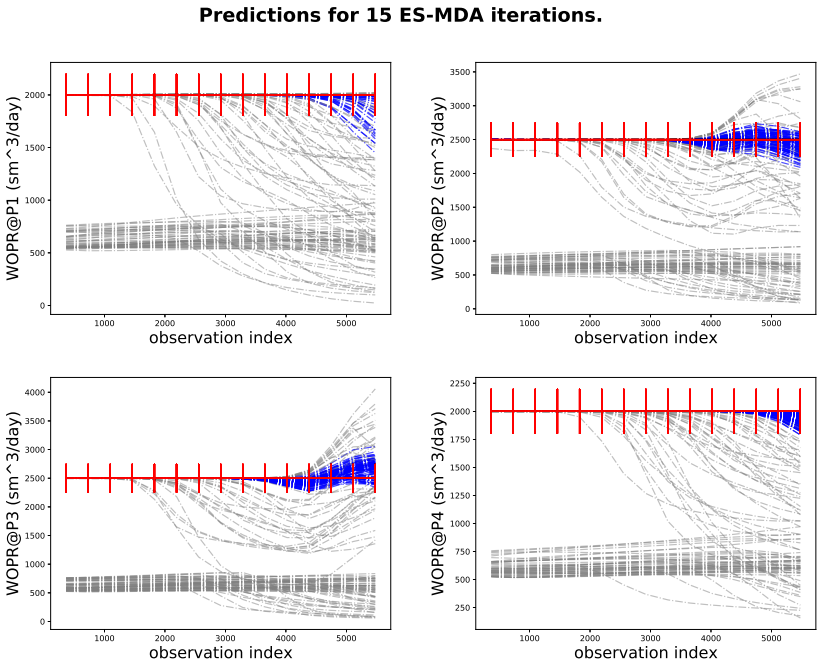


Figure 5.24 – Results of history match for WOPR at 4 producer wells for 5SPOTS case. ESMDA algorithm was used with 15 iterations and 100 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 15th iteration.

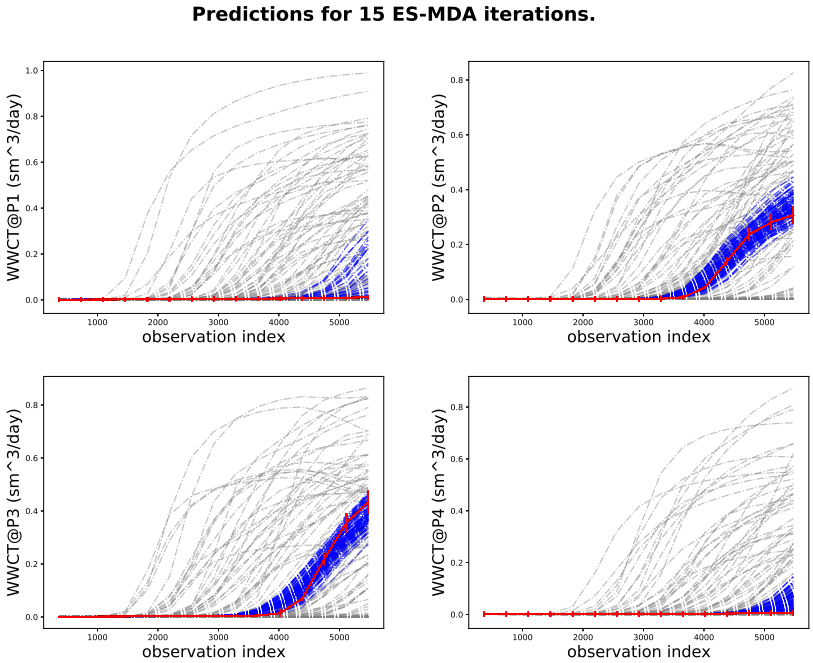


Figure 5.25 – Results of history match for WWCT at 4 producer wells for 5SPOTS case. ESMDA algorithm was used with 15 iterations and 100 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 15th iteration.

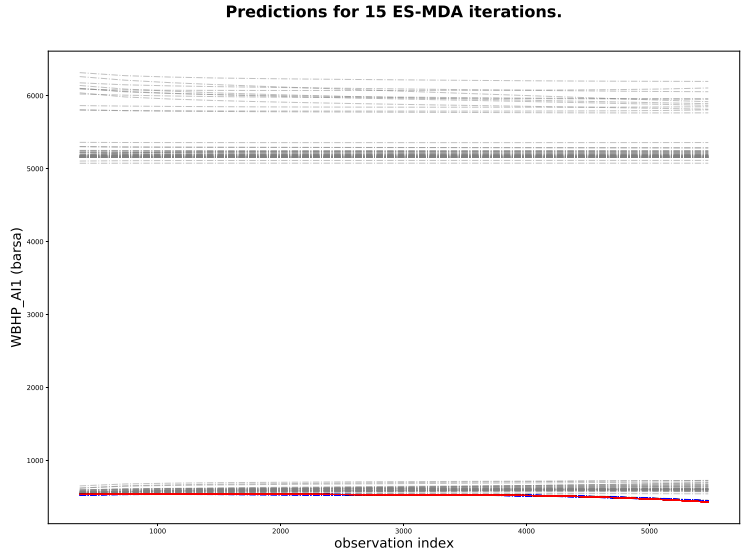


Figure 5.26 – Results of history match for WBHP at injector well for 5SPOTS case. ESMDA algorithm was used with 15 iterations and 100 ensemble members. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 15th iteration.

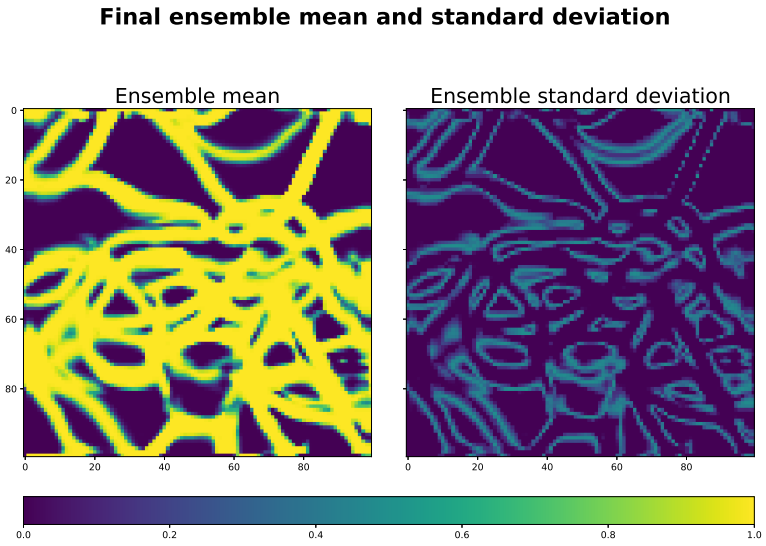


Figure 5.27 – Mean and std of the ensemble in the image space for a run with 15 iterations and 100 ensemble members.

5.2.5.2 5SPOTS results using subspace inversion

It was demonstrated that the increase of ensemble members could limit the ensemble collapse to an almost unique realization with very low variability in Sec. 5.2.4.2. But it comes with an important computational cost. Another way to avoid ensemble collapse phenomena is to use the subspace inversion method described in Sec. 2.3.5. The user has to choose the threshold value to truncate the eigen components of the singular value decomposition done on the Kalman gain. This method will be demonstrated on the 5SPOTS case because a higher number of observations are available due to the high number of wells in this case and subspace inversion method acts as a way to select observations that are the most correlated to parameters and removes others. The singular value decomposition (SVD) cut value has to be set by the user, and to the knowledge of the author there is no way to determine the cut value *apriori*. The cut value was set to 0.925.

Results are shown Fig. 5.28, 5.29 for WWCT and WOPR where predictions are not fitting observation as well as the case without the subspace inversion due to the absence of ensemble collapsed. Moreover, Fig. 5.30 shows an interesting result where variability is visible in the analysis except at well locations. This shows that the ensemble converged to solutions conditioned at the wells without any additional information. That kind of application usually requires static constraint to get such conditioning. Increasing SVD cut value can improve the history match while conserving the static conditioning. The subspace inversion method has also the advantage to reduce the computational cost of the assimilation algorithm by reducing the size of Kalman gain matrix when observations are highly dimensional.

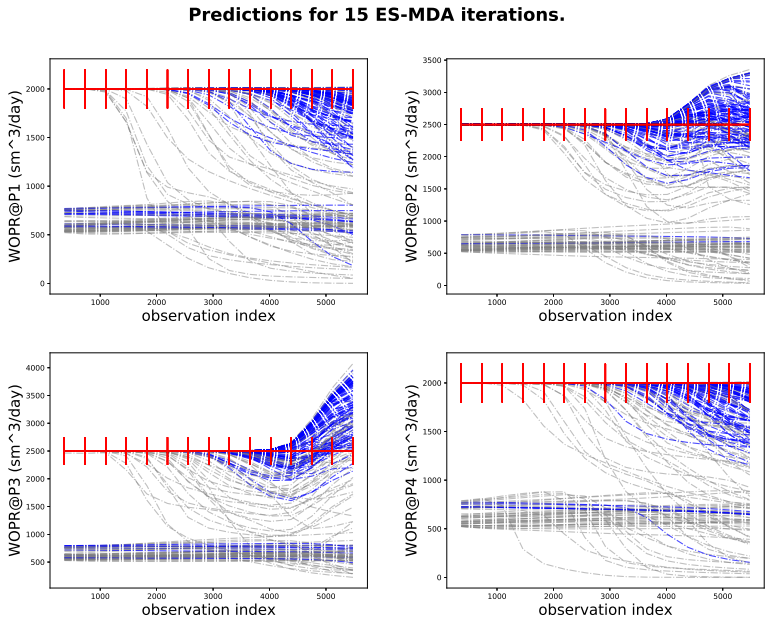


Figure 5.28 – Results of history match for WOPR at 4 producer wells for 5SPOTS case. ESMDA algorithm was used with 15 iterations and 100 ensemble members and a SVD cut at 0.925. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 15th iteration.

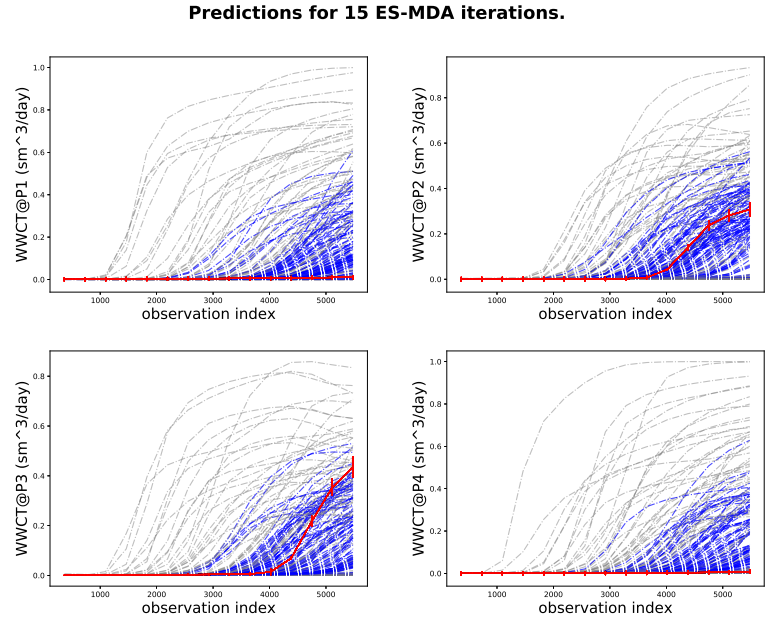


Figure 5.29 – Results of history match for WWCT at 4 producer wells for 5SPOTS case. ESMDA algorithm was used with 15 iterations and 100 ensemble members and a SVD cut at 0.925. Red curves are the observations, gray curves are the predictions of the initial ensemble and the blue curves are the prediction at the 15th iteration.

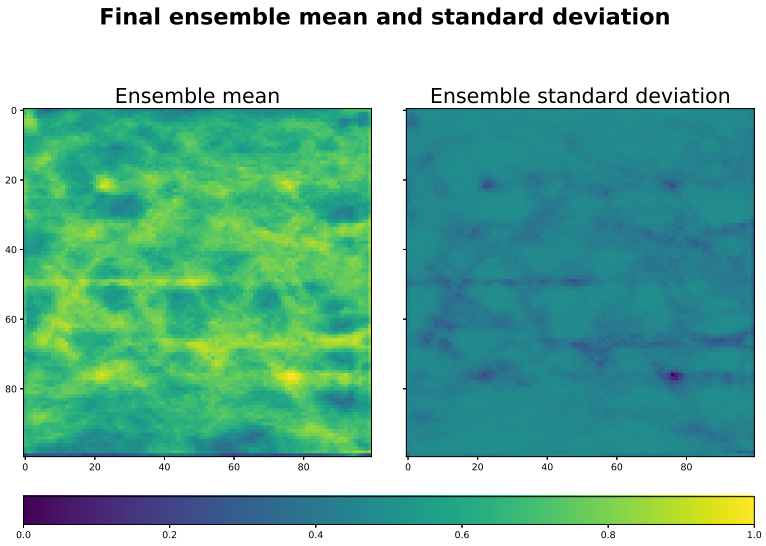


Figure 5.30 – Mean and std of the ensemble in the image space for a run with 15 iterations and 100 ensemble members and a SVD cut at 0.925.

5.2.6 Discussion

The results of this study support the evidence that GAN parameterization of numerical reservoir models by a continuous latent space that conserves the Gaussian assumption of Kalman theory is possible. It was shown that the use of ensemble data assimilation algorithm for history matching coupled with GAN parameterization conserves the spatial distribution of geological heterogeneities and allows conservation of variability in the predictions. Computational cost of our method is mostly during the training of GAN, cost of the inference during data assimilation algorithm is not significant compared to reservoir fluid flow simulations. GAN parameterization also allows an important dimension reduction of the parameter space. One of the limitations in our method is the necessity for the creation of a dataset, similarly to object-based methods and the creation of training images. However, the dataset properties seem less restrictive than the creation of training images that are case dependent for a given heterogeneity type such as channels. The dataset needs to represent the diversity of the different trends found in geological media. Another limitation is the availability of GPUs with enough memory to increase the dimension of numerical reservoir models parameterized by a GAN.

The author would like to emphasize that the architecture of the GAN and its hyper-parameters in general could be improved. Numerous variations and improvements have been done on GANs, GANs can be conditioned at inference or could be disentangled giving each latent direction a visual signification for example which could be interesting improvements in the method presented here. Readers are encouraged to try improved versions and perform sensitivity analysis to see how these results could be improved.

Chapter 6

Producing realistic climate data with generative adversarial network

6.1 Using balanced climate generator for data assimilation

The WGAN could be useful in this context due to its ability to learn the balance constraint of an atmospheric state. This study aims at showing how generative neural networks can be used to improve the spatial analysis of the observations gathered every day. This work focuses on precise problems encountered during the data assimilation cycle, but will not be applied in a complete data assimilation routine because of the complexity and the redundancy with the work applied to reservoir data assimilation.

In Sec. 1.2.2, the different methods employed for initialization in the sense of starting with a balanced atmospheric state before forecast step were mentioned. Houtekamer and Zhang [55] explains that the main source of imbalance could come from the localization step. Localization is a way to remove the sampling noise, due to the finite size of ensemble used in DA, and that creates spurious long range forecast error correlations. As a consequence, the localization limits the influence of observation information when it is geographically far from the corrected point. However, the localization implies imbalanced corrections, *e.g.*, it can increase the gradient in geopotential corrections which implies an over-estimation of the nearly geostrophic wind velocity. We think that increasing the ensemble size by producing realistic multivariate fields near a given situation would help to limit the use of localization.

Our idea was to use a Wasserstein GAN to learn from a global circulation model daily output to learn the manifold of balanced atmospheric state. This study has the objective to prove the concept of GAN application to climate data. Different development choices were made for simplicity of the implementation. The main challenges to apply it to climate models were the following :

- A decision was made to work on the projected atmospheric fields (using equirectangular projection) to keep samples as images for simplicity of the GAN implementation.
- Using a projection implies the use of boundary conditions such as periodic boundary conditions in the West-East direction. Which will be implemented and described in Besombes et al. [7] included in Sec. 6.2. The projection of a sphere on a plane also has a geometrical effect such as an increase in the size of meteorological objects located close to the poles, illustrated Fig. 6.1. In Fig. 6.1 a field of Gaussian noise was projected on spherical harmonics and projected back on the Cartesian grid. It illustrates the distortion of the meteorological objects when they are located close to the poles. The truncation effect is due to the non-bijectivity of the projection on the Cartesian grid of the spherical harmonics. This effect is the reason the field containing the latitude of each pixel was introduced in the data, because it has an important influence on the generation and it could facilitate the training.
- Dataset samples size : climate state is characterized by an important number of variables on a

numerical grid with an important number of cells. Moreover, this grid is replicated on different layers on the altitude direction. To alleviate the computational cost, especially the GPU memory cost during the GAN training the z direction was considered as image channels (such as RGB channels in colored images). This choice avoids the use of 3D convolutions that are very costly.

These solutions were chosen for their simplicity to prove the feasibility of the study and present the concept of GANs for the geoscience community. But the use of GAN in other domains led to important advances and derivations of deep learning and more precisely GAN models. For example Perraudin et al. [88] developed DeepSphere which is a package that implements CNNs for spherical data represented as a graph of connected nodes. This would be a way of reducing the error due to projection on Cartesian grids. The increasing memory capacity of GPUs could also allow the use of 3D convolutional layers for better vertical coherence. Other ways of improvement are discussed in the conclusion of Besombes et al. [7].

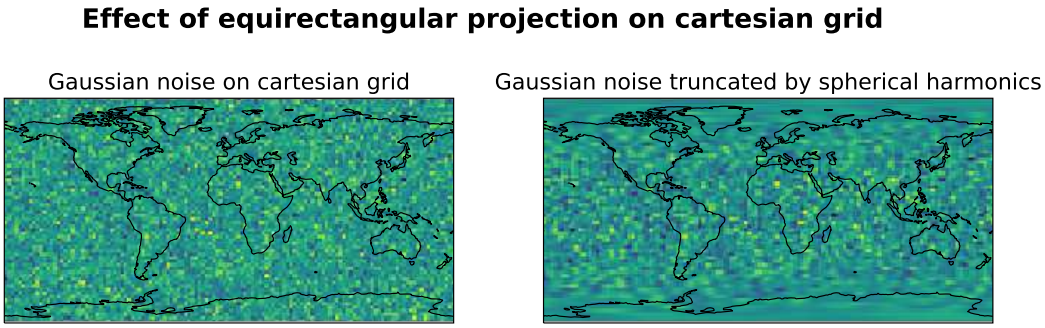


Figure 6.1 – Distortion and truncation effect for the equirectangular projection of spectral grid. A Gaussian field on a Cartesian grid (left) is projected onto the spherical harmonics and projected back on the Cartesian grid (right).

6.2 Producing realistic climate data with generative adversarial network

Nonlin. Processes Geophys., 28, 347–370, 2021
https://doi.org/10.5194/npg-28-347-2021
© Author(s) 2021. This work is distributed under
the Creative Commons Attribution 4.0 License.

Nonlinear Processes in Geophysics
Open Access
EGU

Producing realistic climate data with generative adversarial networks

Camille Besombes[1,4], **Olivier Pannekoucke**[2], **Corentin Lapeyre**[1], **Benjamin Sanderson**[1], **and Olivier Thual**[1,3]

[1]CERFACS, Toulouse, France
[2]CNRM, Université de Toulouse, Météo-France, CNRS, Toulouse, France
[3]Institut de Mécanique des Fluides de Toulouse (IMFT), Université de Toulouse, CNRS, Toulouse, France
[4]Institut National Polytechnique de Toulouse, Toulouse, France

Correspondence: Camille Besombes (besombes@cerfacs.fr), Olivier Pannekoucke (opannekoucke@cerfacs.fr), Corentin Lapeyre (lapeyre@cerfacs.fr), Benjamin Sanderson (sanderson@cerfacs.fr) and Olivier Thual (thual@cerfacs.fr)

Received: 9 February 2021 – Discussion started: 16 February 2021
Revised: 8 June 2021 – Accepted: 17 June 2021 – Published: 30 July 2021

Abstract. This paper investigates the potential of a Wasserstein generative adversarial network to produce realistic weather situations when trained from the climate of a general circulation model (GCM). To do so, a convolutional neural network architecture is proposed for the generator and trained on a synthetic climate database, computed using a simple three dimensional climate model: PLASIM.

The generator transforms a "latent space", defined by a 64-dimensional Gaussian distribution, into spatially defined anomalies on the same output grid as PLASIM. The analysis of the statistics in the leading empirical orthogonal functions shows that the generator is able to reproduce many aspects of the multivariate distribution of the synthetic climate. Moreover, generated states reproduce the leading geostrophic balance present in the atmosphere.

The ability to represent the climate state in a compact, dense and potentially nonlinear latent space opens new perspectives in the analysis and handling of the climate. This contribution discusses the exploration of the extremes close to a given state and how to connect two realistic weather situations with this approach.

1 Introduction

The ability to generate realistic weather situations has numerous potential applications. Weather generators can be used to characterize the spatio-temporal complexity of phenomena in order, for example, to assess the socio-economical impact of the weather (Wilks and Wilby, 1999; Peleg et al., 2018). However, in numerical weather prediction the dimension of a simulation can be very large: an order of 10^9 is often encountered (Houtekamer and Zhang, 2016). The small size of ensembles used in data assimilation, due to computational limitations, leads to a misrepresentation of the balance present in the atmosphere such as an increment in the geopotential height, resulting in an unbalanced incremented wind because of localization (Lorenc, 2003). Issues of small finite samples of weather forecast ensembles could be addressed by considering larger synthetic ensembles of generated situations. With current methods it is difficult to generate a realistic climate state at a low computational cost. This is usually done by using analogs or by running a global climate model for a given time (Beusch et al., 2020) but remains costly. Generators can also be used for super resolution so as to increase the resolution of a forecast leading to better results than interpolations (Li and Heap, 2014; Zhang et al., 2012).

The last decade has seen new kinds of generative methods from the machine-learning field using artificial neural networks (ANNs). Among these, generative adversarial networks (GANs) (Goodfellow et al., 2020), and more precisely Wasserstein GANs (WGANs) (Arjovsky et al., 2017), are effective data-driven approaches to parameterizing complex distributions. GANs have proven their power in unsupervised learning by generating high-quality images from complex distributions. Gulrajani et al. (2017) trained a WGAN on the ImageNet database (Russakovsky et al., 2015), which contains over 14 million images with 1000 classes, and suc-

cessfully learned to produce new realistic images. Several techniques developed for computer vision with GANs seem promising for domains in the geosciences. Notable examples of usage to date include Yeh et al. (2017) to do inpainting, where the objective is to recover a full image from an incomplete one, Ledig et al. (2017) to do super resolution, or Isola et al. (2017) to do image-to-image translation, where an image is generated from another one, e.g., translate an image that contains a horse into one with a zebra.

Data-driven approaches and numerical weather prediction are two domains that share important similarities. Watson-Parris (2021) explains that both domains use the same methods to answer different questions. This study and Boukabara et al. (2019) also show that numerical weather prediction contains lots of interesting challenges that could be tackled by machine-learning methods. It clarifies the growing literature about data-driven techniques applied to weather prediction. Scher (2018) used variational autoencoders to generate the dynamics of a simple general circulation model conditioned on a weather state. Weyn et al. (2019) trained a convolutional neural network (CNN) on gridded reanalysis data in order to generate 500 hPa geopotential height fields at forecast lead times up to 3 d. Lagerquist et al. (2019) developed a CNN to identify cold and warm fronts and a post-processing method to convert probability grids into objects. Weyn et al. (2020) built a CNN able to forecast some basic atmospheric variables using a cubed-sphere remapping in order to alleviate the task of the CNN and impose simple boundary conditions.

While there is a growing interest in using deep-learning methods in weather impact or weather prediction (Reichstein et al., 2019; Dramsch, 2020), few applications have been described using GANs applied to physical fields in recent years (Wu et al., 2020). Notable examples include application to subgrid processes (Leinonen et al., 2019), to simplified models such as the Lorenz '96 model (Gagne et al., 2020) or to data processing like satellite images (Requena-Mesa et al., 2018). In particular, little is known about the feasibility of designing and training a generator that would be able to produce multivariate states of a global atmosphere. A first difficulty is to propose an architecture for the generator, with the specific challenge of handling the spherical geometry. Most of the neural network architectures in computer vision handle regular two-dimensional images instead of images representing projected spherical images. Boundary conditions of these projections are not simple, as the spherical geometry also influences the spread of the meteorological object as a function of its latitude. These effects have to be considered in the neural network architecture. Another difficulty is to validate the climate resulting from the generator compared with the true climate.

In this study, in order to evaluate the potential of GANs applied to the global atmosphere, a synthetic climate is computed using the PLASIM global circulation simulator (Fraedrich et al., 2005a), a simplified but realistic implementation of the primitive equations on the sphere. An architecture is proposed for the generator and trained using an approach based on the Wasserstein distance. A multivariate state is obtained by the transformation of a sample from a Gaussian random distribution in 64 dimensions by the generator. Thanks to this sampling strategy, it is possible to compute a climate as represented by the generator. Different metrics are considered to compare the climate of the generator with the true climate and to assess the realism of the generated states. Because the distribution is known, the generator provides a new way to explore the climate, e.g., simulating the intensification of a weather situation or interpolating two weather situations in a physically plausible manner.

The article is organized as follows. The formalism of WGAN is first introduced in Sect. 2 with the details of the proposed architecture. Then, Sect. 3 evaluates the ability of the generator to reproduce the climate of PLASIM with assessment of the climate states that are produced by the generator. The conclusions and perspectives are given in Sect. 4.

2 Wasserstein generative adversarial network to characterize the climate

2.1 Parameterizing the climate of the Earth system

The Earth system is considered to be the solution of an evolution equation

$$\partial_t \chi = \mathcal{M}(\chi), \tag{1}$$

where χ denotes the state of the system at a given time and $\mathcal{M}$ characterizes the dynamics including the forcing terms, e.g., the solar annual cycle. While χ should stand for continuous multivariate fields, we consider its discretization in a finite grid so that $\chi \in X$ with $X = \mathbb{R}^n$, where n denotes the dimension. Equation (1) describes a chaotic system. The *climate* is the set of states of the system along its time evolution. It is characterized by a distribution or a probability measure, denoted p_{clim}.

Obtaining a complete description of p_{clim} is intractable due to the complexity of natural weather dynamics and because a climate database, p_{data}, is limited by numerical resources and is only a proxy for this distribution.

For instance, in the present study, the true weather dynamics $\mathcal{M}$ are replaced by the PLASIM model that has been time-integrated over 100 years of 6 h forecasts. Accounting for the spinup, the first 10 years of simulation are ignored. Thus, the climate p_{clim} is approximated from the resulting climate database of 90 years, p_{data}. The synthetic dataset is presented in detail in Sect. 3.1.

Thus, p_{data} lives in the n-dimensional space X, but it is non-zero only on an m-manifold $\mathbb{M}$ (where $m \ll n$) that can be fractal. The objective is to learn a mapping

$$g : Z \longmapsto X \tag{2}$$

from $Z = \mathbb{R}^m$, the so-called latent space, to X. Moreover, g must transform a Gaussian $\mathcal{N}(0, \mathbf{I}_m)$ to $p_{\text{data}} \subset \mathbb{M}$.

The main advantage of such a formulation is to have a function g that maps a low-dimensional continuous space Z to $\mathbb{M}$. This property could be useful in the domain of the geosciences, notably in the climate sciences, where a high-dimensional space is ruled by important physical constraints and parameters.

Here the generator is a good candidate for learning the physical constraints that make a climate state realistic without the need to run a complete simulation. The construction of the generator is now detailed.

2.2 Background on Wasserstein generative adversarial networks

To characterize the climate, we first introduce a simple Gaussian distribution $p_z = \mathcal{N}(0, \mathbf{I}_m)$ of zero mean and covariance the identity matrix $\mathbf{I}_m$, defined on the space $Z = \mathbb{R}^m$, called the latent space. The objective of an adversarial network is to find a nonlinear transformation of this space Z to X as written in Eq. (2) so that the Gaussian distribution maps to the climate distribution, i.e., $g_\#(p_z) = p_{\text{clim}}$, where $g_\#$ denotes the push forward of a measure by the map g, defined here as follows: for any measurable set E of X, $g_\#(p_z)(E) = p_z(g^{-1}(E))$, where $g^{-1}(E)$ denotes the measurable set of Z that is the pre-image of E by g. The latent space, Z, can be seen as an encoded climate space where each point drawn from p_z corresponds to a realistic climate state and where the generator is the decoder. Looking for such a transformation is non-trivial.

The search is limited to a family of transformations $\{g_\theta\}$ characterized by a set of parameters θ. Thus, for each θ, the nonlinear transform of the Gaussian p_z by g_θ is a distribution p_θ. The goal is then to find the best set of parameters θ^* such that $\theta^* = \text{argmin}_\theta \text{di}(p_\theta, p_{\text{clim}})$, where di is a measure of the discrepancy between the two distributions, so that p_{θ^*} approximates p_{clim}. This method is known as generative learning, where g_θ is implemented as a neural network of trainable parameters θ. Note that, being a neural network, the resulting g_θ is then a differentiable function.

Even with this simplified framework, the search for an optimal θ is not easy. One of the difficulties is that the differentiability of g_θ requires the comparison of continuous distribution p_θ with p_{clim}, which is not necessarily a density on a continuous set. To alleviate this issue, Arjovsky et al. (2017) introduced an optimization process based on the Wasserstein distance defined for the two distributions p_{clim} and p_θ by

$$W(p_\theta, p_{\text{clim}}) = \inf_{\gamma \in \Pi(p_\theta, p_{\text{clim}})} \mathbb{E}_{(x,y)}\left[\|x - y\|\right], \tag{3}$$

where $\Pi(p_\theta, p_{\text{clim}})$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are, respectively, $\int_y \gamma(\cdot, \mathrm{d}y) = p_\theta$ and $\int_x \gamma(\mathrm{d}x, \cdot) = p_{\text{clim}}$. The Wasserstein distance, also called the Earth mover distance (EMD), comes from optimal transport theory and can be seen as the minimum work required (in the sense of mass × transport) to transform the distribution p_θ into the distribution p_{clim}. Thus, the set $\Pi(p_\theta, p_{\text{clim}})$ can be seen as all the possible mappings, also called couplings, to transport the mass from p_θ to p_{clim}. The Wasserstein distance is a weak distance: it is based on the expectation, which can be estimated whatever the kind of distribution. Hence, the optimization problem is stated as

$$\theta^* = \text{argmin}_\theta W(p_\theta, p_{\text{clim}}), \tag{4}$$

which leads to the WGAN approach.

One of the major advantages of the Wasserstein distance is that it is real-valued for non-overlapping distributions. Indeed, the Kullback–Leibler (KL) divergence is infinite for disjoint distributions, and using it as a loss function leads to a vanishing gradient (Arjovsky et al., 2017). The Wasserstein distance does not exhibit vanishing gradients when distributions do not overlap, as did the KL divergence in the original GAN formulation.

Unfortunately, the formulation in Eq. (3) is intractable. A reformulation is necessary using the dual form discovered by Kantorovich (Kantorovich and Rubinshtein, 1958). Reframing the problem as a linear programming problem yields

$$W(p_\theta, p_{\text{clim}}) = \sup_{f \in 1-\text{Lipshitzian}} \left[\mathbb{E}_{x \sim p_{\text{clim}}}\left[f(x)\right] - \mathbb{E}_{x \sim p_\theta}\left[f(x)\right]\right], \tag{5}$$

where $1-$Lipshitzian denotes the set of Lipshitzian functions $f : \mathbb{R}^n \to \mathbb{R}$ of coefficient 1, i.e., for any $(x_1, x_2) \in \mathbb{R}^n$, $|f(x_1) - f(x_2)| \leq ||x_1 - x_2||$, $||\cdot||$ being the Euclidian norm of $\mathbb{R}^n$. For any $1-$Lipshitzian function f the computation of Eq. (5) is simple: the first expectation can be approximated by

$$\mathbb{E}_{x \sim p_{\text{clim}}}\left[f(x)\right] \approx \mathbb{E}_{x \sim p_{\text{data}}}\left[f(x)\right], \tag{6}$$

where the right-hand side is computed as the empirical mean over the climate database p_{data} that approximates p_{clim} in the weak sense Eq. (6). The second expectation can be computed from the equality

$$\mathbb{E}_{x \sim p_\theta}\left[f(x)\right] = \mathbb{E}_{z \sim \mathcal{N}(0; \mathbf{I}_m)}\left[f(g_\theta(z))\right], \tag{7}$$

where the expectation of the right-hand side can be approximated by the empirical mean computed from an ensemble of samples of z which are easy to sample due to the Gaussianity.

However, there is no simple way to characterize the set of $1-$Lipshitzian functions, which limits the search for an optimal function in Eq. (5). Instead of looking at all $1-$Lipshitzian functions, a family of functions $\{f_w\}$ parameterized by a set of parameters w is introduced. In practice, it is engendered by a neural network with trainable parameters w, called the *critic*.

Finally, if the weights of the network are constrained to a compact space $\mathcal{W}$, which can be done by the weight-clipping

method described in Arjovsky et al. (2017), then $\{f_w\}_{w\in\mathcal{W}}$ will be K-Lipschitzian with K depending only on $\mathcal{W}$ and not on individual weights of the network. This yields

$$\max_{w\in\mathcal{W}}\left[\mathbb{E}_{x\sim p_{\text{data}}}\left[f_w(x)\right]-\mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f_w(g_\theta(z))\right]\right] \leq \sup_{f\in 1-\text{Lipshitzian}}\left[\mathbb{E}_{x\sim p_{\text{data}}}\left[f(x)\right] - \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f(g_\theta(z))\right]\right], \tag{8}$$

which tells us that the critic tends to the Wasserstein distance when trained optimally, i.e., if we find the max in Eq. (8) and if f is in (or close to) $\{f_w\}_{w\in\mathcal{W}}$. The weight-clipping method was replaced by the gradient penalty method in Gulrajani et al. (2017) because it diminished the training quality as mentioned in Arjovsky et al. (2017). Because it results from a neural network, any function f_w is differentiable, so that the $1-$Lipshitzian condition remains to ensure a gradient norm bounded by 1, i.e., for any $x \in X$, $||\nabla f_w(x)|| \leq 1$. To do so, Gulrajani et al. (2017) have proposed computing the optimal parameter $\tilde{w}(\theta)$ as the solution of the optimization problem

$$\tilde{w}(\theta) = \text{argsup}_w L(\theta, w), \tag{9}$$

where L is the cost function

$$L(\theta, w) = \mathbb{E}_{x\sim p_{\text{data}}}\left[f_w(x)\right] - \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f_w(g_\theta(z))\right] + \lambda\mathbb{E}_{\hat{x}\sim\hat{p}}\left[\left(||\nabla f_w(\hat{x})|| - 1\right)^2\right], \tag{10}$$

with λ the magnitude of the gradient penalty and where $\hat{x}$ is uniformly sampled from the straight line between a sample from p_{data} to a sample from p_θ (line 8) of Algorithm 1. The optimal solution $\tilde{w}(\theta)$ is obtained from a sequential method where each step is written as

$$w_{k+1} = w_k + \beta_k \nabla_w L(\theta, w_k), \tag{11}$$

where β_k is the magnitude of the step. In an adversarial way, Eq. (10) could be solved sequentially, e.g., by the steepest descent algorithm with an update given by

$$\theta_{q+1} = \theta_q - \alpha_q \nabla_\theta W(p_{\theta_q}, p_{\text{clim}}), \tag{12}$$

where α_q is the magnitude of the step. We chose to use the two-sided penalty for the gradient penalty method, as it was shown to work well in Gulrajani et al. (2017). At convergence, the Wasserstein distance is approximated by

$$W(p_\theta, p_{\text{clim}}) \approx \mathbb{E}_{x\sim p_{\text{data}}}\left[f_{\tilde{w}(\theta)}(x)\right] - \mathbb{E}_{z\sim\mathcal{N}(0;\mathbf{I}_m)}\left[f_{\tilde{w}(\theta)}(g_\theta(z))\right]. \tag{13}$$

Hence, the solution of the optimization problem Eq. (4) is obtained from a sequential process composed of two steps, summarized in Algorithm 1. In the first step, the weights of the generator are frozen with a given set of parameters θ_q and the critic neural network is trained in order to find the optimal parameter $\tilde{w}(\theta_q)$ solution Eq. (9) (lines 3–11 in Algorithm 1). In the second step, the critic is frozen and the generator is set as trainable in order to compute θ_{q+1} from Eq. (12) (lines 12–17 in Algorithm 1). Note that in Algorithm 1, the steepest descent is replaced by an Adam optimizer (Kingma and Ba, 2014), a particular implementation of stochastic gradient descent which has been shown to be efficient in deep learning.

The following sections will aim to create a climate data generator from the WGAN method. The next section will describe the architecture of the network adapted to the complexity of the dataset used.

2.3 Neural network implementation

WGANs are known to be time-consuming to train, usually needing a high number of iterations due to the alternating aspect of the training algorithm between the critic and the generator. Our initial architecture used a simple convolutional network for both, with a high number of parameters, but it proved difficult to train a fitting multimodal distribution such as green distributions in the left panels in Fig. 15. That is why for this study a ResNet-inspired architecture (He et al., 2016) was chosen. The goal of the residual network is to reduce the number of parameters of the network and avoid gradient vanishing, which is a recurrent problem for deep networks that results in an even slower training.

A network is composed of a stack of layers; when a specific succession of layers is used several times, we can refer to it as a *block*. The link between two layers is named a connection; a shortcut connection refers to a link between two layers that are not successive in the architecture. A residual block (Figs. 2 and 3) is composed with stacked convolution and a parallel identity shortcut connection. The idea is that it is easier to learn the residual mapping than all of it, so residual blocks can be stacked without observing a vanishing gradient. Moreover, a residual block can be added to an N-layer network without reducing its accuracy because it is easier to learn $F(x) = 0$ by setting all the weights to 0 than it is to learn the identity function. Residual blocks allow building of deeper networks without loss of accuracy.

One should note that the PLASIM simulator is a spectral model run on a Gaussian grid that consequently enforces the periodic boundary condition. In order to impose the periodic boundary condition in the generated samples, it was necessary to create a *wrap padding layer*, which takes multiple columns at the eastern side and concatenates them to the western side and vice versa. In the critic, the wrap padding is only after the input, since the critic will discriminate the images from the generator that are not continuous in the west–east direction. In the generator, the *wrap padding layer* is in every residual block; it is necessary because the reduced size of the convolution kernel compared to the image size makes it more difficult for the network to extract features from both sides of the image simultaneously. The north–south bound-

Algorithm 1 WGAN training algorithm.

Require: Learning rate lr, batch size b, n_{critic} number of iteration of the critic per generator iteration.
Require: w_0 and θ_0 respectively the initial critic and generator parameters.
# Optimization cycle
while θ has not converged **do**
  # 1. Computation of the Wasserstein distance by maximization over 1−Lipshitzian functions
  for $t = 0, \ldots, n_{critic}$ **do**
    # 1.1 Computation of the gradient for the 1−Lip. function.
    Sample$\{x^{(i)}\}_{i=1}^{b} \sim P_{\text{data}}$ a batch from the real data.
    Sample$\{z^{(i)}\}_{i=1}^{b} \sim P_\theta$ a batch from the generated data.
    Sample$\{\hat{x}^{(i)}\}_{i=1}^{b}$ where $\hat{x} = \xi x + (1-\xi) g_\theta(z)$ where $\xi \sim \mathcal{U}[0,1]$
    $grad_w \leftarrow \nabla_w \left[\frac{1}{b} \Sigma_{i=1}^{b} f_w(x^{(i)}) - \frac{1}{b} \Sigma_{i=1}^{b} f_w(g_\theta(z^{(i)})) + \frac{\lambda}{b} \Sigma_{i=1}^{b} \left(||\nabla f_w(\hat{x}^{(i)})|| - 1 \right)^2 \right]$
    # 1.2 Update the parameter w to maximize Eq. (5)
    $w \leftarrow w + lr * Adam(w, grad_w)$
  end for
  # 2. Update the generator
  # 2.1 Compute the gradient of the Wasserstein distance
  Sample$\{z^{(i)}\}_{i=1}^{b} \sim P_\theta$ a batch from the generated data.
  $grad_\theta \leftarrow \nabla_\theta \left[\frac{1}{b} \Sigma_{i=1}^{b} f_w(g_\theta(z^{(i)})) \right]$
  # 2.2 Update the parameter θ to minimize the Wasserstein distance
  $\theta \leftarrow \theta - lr * Adam(\theta, grad_\theta)$
end while

ary is padded by repeating the nearest line, called the *nearest padding layer*. In Figs. 1–5 padding layer arguments have to be understood as (longitude direction, latitude direction), where the integer means the number of columns or rows to be taken from each side and placed next to the other one; e.g., *Wrappadding* $(0, 3)$ means the output image is six columns larger than the input. If the argument is not mentioned, then the arguments for wrap and nearest padding are $(0, 1)$ and $(1, 0)$, respectively.

2.3.1 Critic network

The critic network input has the shape of a sample from the dataset $X \in \mathbb{R}^{\text{nlat} \times \text{nlon} \times \text{nfield}}$.

Its output must be a real number because it is an approximation of the Wasserstein distance between the distribution of the batch of images from the dataset and the one from the generator that is being processed. The architecture ends with a dense layer of one neuron with linear activation. The core of the structure is taken from the residual network and can be seen in Fig. 1. After the custom padding layers mentioned previously, the critic architecture is a classical residual network, starting with a convolution with 7×7 kernels, followed by a maximum pooling layer to reduce the image size and a succession of convolutional and identity blocks (Figs. 2 and 3). At each strided convolutional block, $s = 2$ in Fig. 3, the image size is divided by a factor 2. It is equivalent to a learnable pooling layer that can increase the result (Springenberg et al., 2014). Finally, an average pooling is done, and the output is fed to a fully connected layer of 100 neurons and then to the output neuron. Batch normalization is not present in the critic architecture following Gulrajani et al. (2017); the batch normalization changes the discriminator's problem by considering all of the batch in the training objective, whereas we are already penalizing the norm of the critic's gradient with respect to each sample in the batch.

2.3.2 Generator architecture

The input of the generator network (see Fig. 4) is an m-dimensional vector containing noise drawn from the normal distribution $\mathcal{N}^m(0, \mathbf{I}_m)$ for the numerical experiment $m = 64$. The output of the generator has the shape of a sample of the dataset $X \in \mathbb{R}^{\text{nlat} \times \text{nlon} \times \text{nfield}}$. The input is passed through a fully connected layer of output shape $(8, 16, 128)$ and fed to residual blocks. The rest of its architecture is also a residual network with a succession of modified convolutional blocks (relative to the one in the critic network). Modifications of the convolutional block are the following.

1. An upsampling layer is added to increase the image size for some convolutional blocks.

2. Wrap and nearest padding layers are added in, respectively, the west–east and north–south directions.

3. A batch normalization layer is present after convolutional layers.

One could argue that the ReLU activation function is not differentiable in 0, but this is managed by taking the left derivative in the software implementation. The study does not claim that the network architectures used are optimal: the

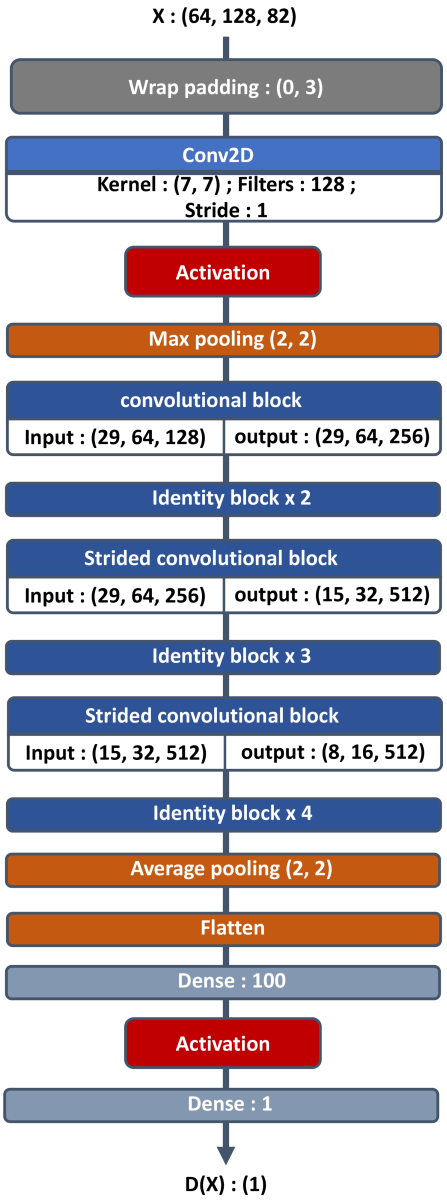


Figure 1. Critic architecture.

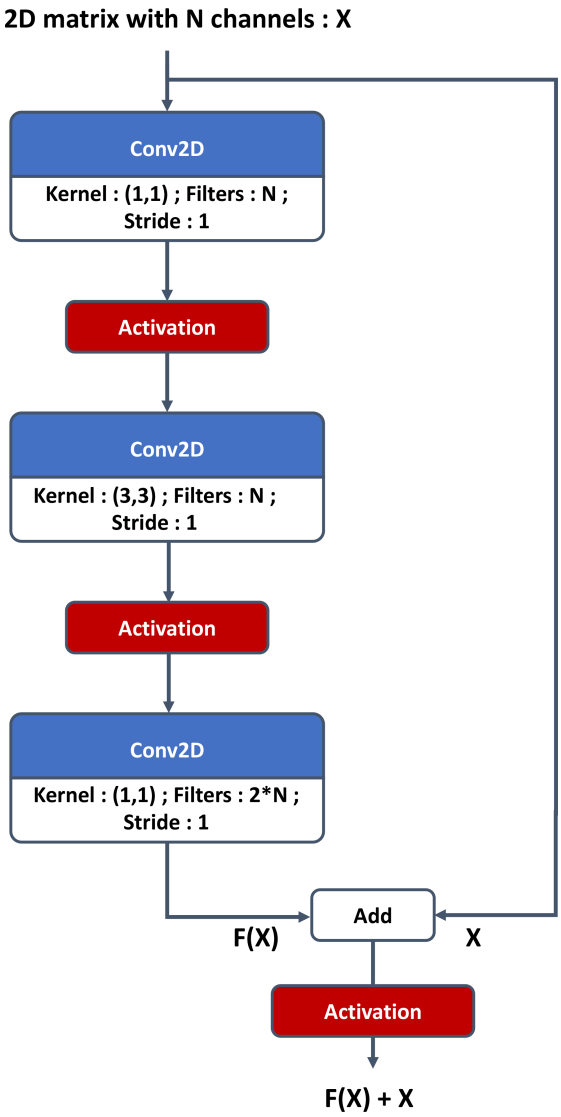


Figure 2. Residual identity block for the critic.

computational burden was too high to run a parameter sensitivity study. Guidelines from Gulrajani et al. (2017) were followed, and the hyperparameters were adapted to the current problem. It showcases an example of hyperparameters producing interesting results, and inspired readers are encouraged to modify and improve this architecture.

2.3.3 Training parameters

For the training phase, the neural network's hyperparameters are summarized in Table 1. The training was performed on an Nvidia Tesla V100-SXM2 with 32 GB of memory over 2 d. The choice of the optimizer, initial learning rate, weight of gradient penalty (λ in Eq. 10) and ratio between critic and

Table 1. Hyperparameters for training step.

	Network	
Hyperparameters	Generator	Critic
Iterations	30 000	150 000
Batch size	128	128
Optimizer	Adam	Adam
Initial learning rate (lr)	$1e^{-3}$	$1e^{-3}$
Learning rate decay every 3000 iterations	0.9	0.9
Number of trainable weights	$1.5e^{6}$	$4e^{6}$
λ in Eq. (10)		10

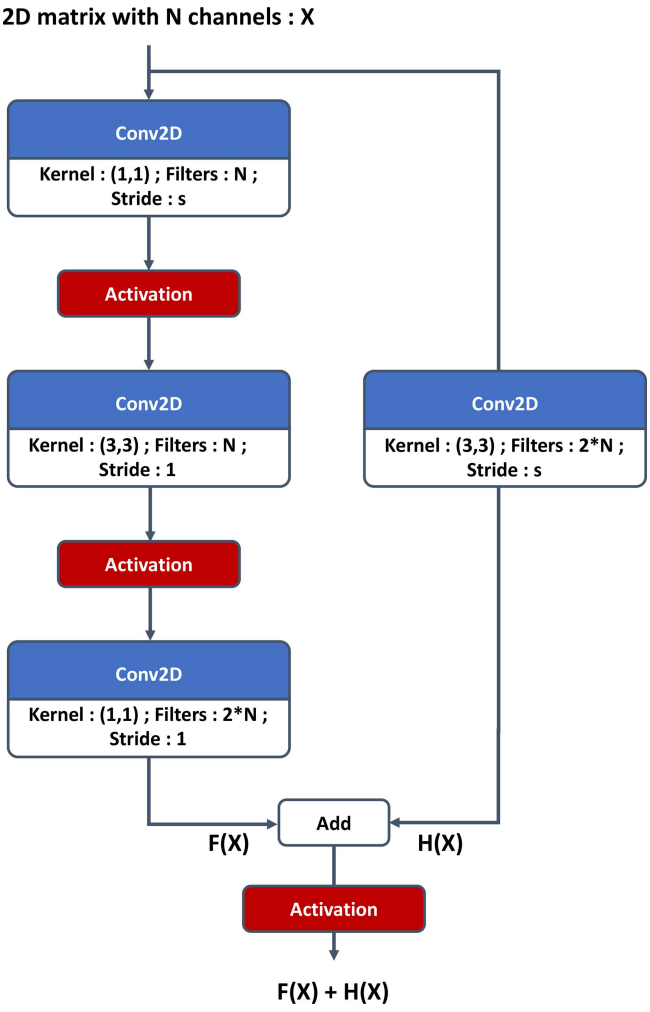


Figure 3. Residual convolutional block for the critic. If s is different from 1, it is referenced as a strided convolutional block in Fig. 1.

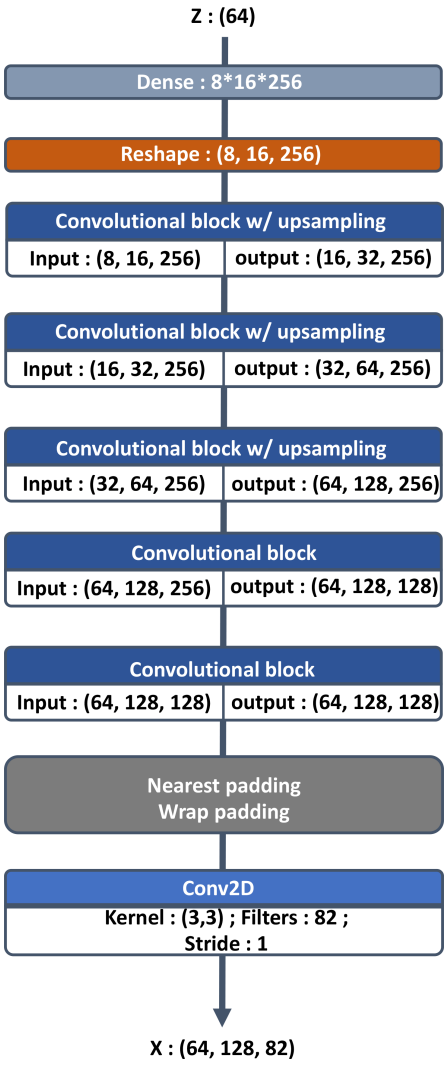


Figure 4. Generator architecture.

generator iteration was directly taken from Gulrajani et al. (2017). The iterations mentioned in Table 1 are the number of batches seen by each neural network.

The training loss in Fig. 6 was smoothed using exponential smoothing:

$$s_t = \alpha y_t + (1-\alpha)\, s_{t-1}, \tag{14}$$

where y_t is the value of the original curve at index t, s_t is the smoothed value at index t and α is the smoothing factor (equal to 0.9 here). An initial spinup of the optimization process tends to exhibit an increase in the loss of the first steps of the training phase before decreasing. This can be explained by the lack of useful information in the gradient due to the initial random weights in the network. A decrease in the Wasserstein distance can be seen in Fig. 6, which indicates a convergence during the training phase, although it is possible to use the loss of the critic as a convergence criterion because the Wasserstein loss is used and has a mathematical meaning such as the distance between synthetic and real data distributions and should converge to 0. However, WGAN-GP is not yet proven to be locally convergent under proper conditions (Nagarajan and Kolter, 2017); the consequence is that it can cycle around equilibrium points and never reach a local equilibrium. Condition on loss derivative is also difficult because of the instability of the GAN training procedure. Consequently, a quality check using metrics adapted to the domain on which the GAN is applied is still necessary. Moreover, at the end of the training, a first experiment was conducted to see whether the generations are present in the dataset. The histogram of the Euclidian distance divided by the number of pixels in one sample between one generation and all of the dataset can be seen in Fig. 7. Here, one can see that the minimum is around 0.8, which shows that the generated image is not inside the dataset. This experiment shows that the generator is able to generate samples without reproducing the dataset. It should be noted that in the WGAN

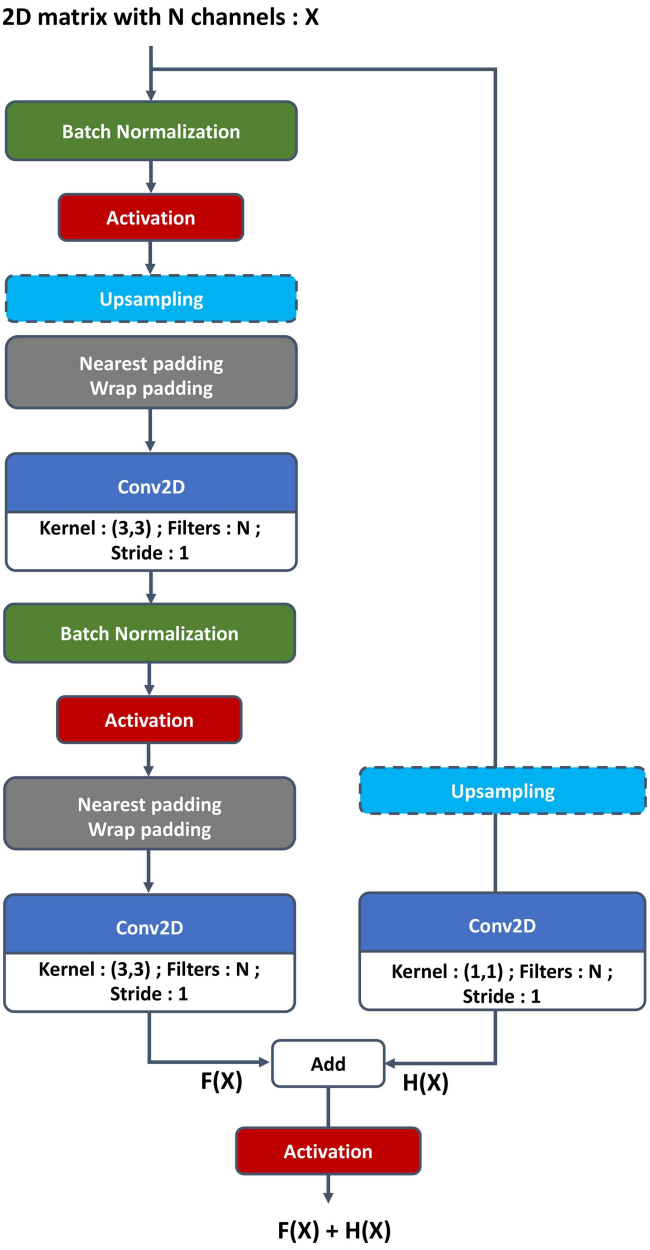


Figure 5. Residual convolutional block for the generator. The upsampling layer can be removed if not necessary and is mentioned when used in Fig. 4.

framework, the generator never directly sees a sample from the dataset.

There are no stopping criteria for the training, and it was stopped after 35 000 iterations in the interest of computational cost. It should be highlighted that the performance of generative networks and especially GANs is difficult to evaluate. In the deep-learning literature, the quality of the images generated is assessed using a reference image dataset such as ImageNet (Russakovsky et al., 2015) and computing the inception score (IS) or the Fréchet inception distance (FID).

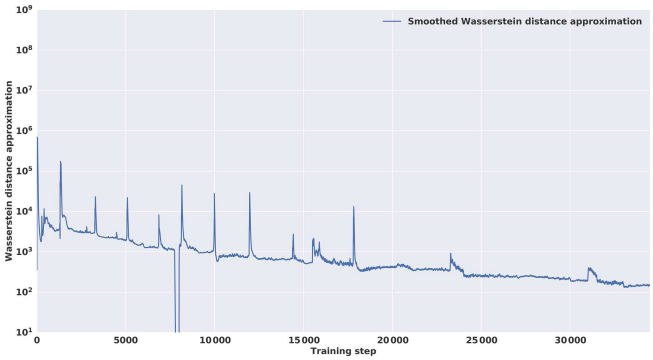


Figure 6. Smoothed version of the Wasserstein distance computed during the training. The vertical axis is in log scale.

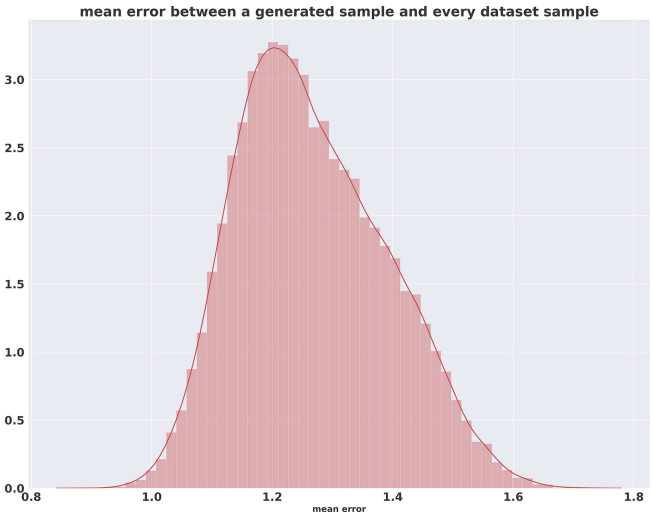


Figure 7. Two-norm distance between a generated sample and all the dataset samples.

Table 2. Variables used in the dataset.

Variables			
Name	Short name	Prognostic	Diagnostic
Temperature (K)	ta	×	
Eastward wind ($m s^{-1}$)	ua		×
Northward wind ($Pa s^{-1}$)	va		×
Relative humidity (frac.)	hus	×	
Vertical velocity ($Pa s^{-1}$)	wap		×
Vorticity (s^{-1})	ζ	×	
Divergence (s^{-1})	d	×	
Geopotential height (gpm)	zg	×	
ln(surface pressure)	P		×
Latitude (degree)	lat		×

Both use the inception network trained on ImageNet: the IS measures the quality and diversity of the images by classifying them and measuring the entropy of the classification, while the FID computes a distance between the features extracted by the inception network and is more robust to GAN-mode collapse.

Because our study does not apply to the ImageNet dataset, it is necessary to compute our own metrics. Section 3 proposes an approach for this kind of method in the domain of geosciences and more precisely the study of atmospheric fields. Our main objective is to assess the fitting quality of the dataset climate distribution.

3 Evaluation and exploration of the generator

The metrics by which the results will be analyzed are visual aspects, capacity to generate atmospheric balances and statistics of the generations compared to climate distribution. For the latter, the chosen metric is the Wasserstein distance. Because it is the same metric the generator has to minimize during the training step, it seems a good candidate to assess the training quality. One could argue that the network is overly trained on this metric; that is why we use other metrics such as mean and standard deviation differences and singular value decomposition to complete our analysis. Finally, because no trivial stop criteria are available, it is interesting to see where the magnitude of the Wasserstein distance is large so as to diagnose some limitations of the trained generator that would provide some ideas of improvements.

3.1 Description of the synthetic dataset

To create synthetic data, a climate model known as PLASIM (Fraedrich et al., 2005a) was used, which is a general circulation model (GCM) of medium complexity based on a simplified general circulation model PUMA (Portable University Model of the Atmosphere) (Fraedrich et al., 2005b). This model based on primitive equations is a simplified analog for operational numerical weather prediction (NWP) models. This choice facilitates the generation of synthetic data thanks to its low resolution and reasonable computational cost. Different components can be added to the model in order to improve the circulation simulation such as the effect of ocean with sea ice, orography with the biosphere or annual cycle.

A 100-year daily simulation was run at a T42 resolution (an approximate resolution of 2.8°). We used orography and annual cycle parameterization; ocean and biosphere modelization were turned off in order to keep the dataset simple enough for our exploratory study. We removed the first 10 years in order to keep only the stationary part of the simulation. These resulting 90 years of simulation constitute the sampling of the climate distribution that we aim to reproduce. As preprocessing, each of the channels was normalized.

Each database sample is an 82-channel (nfield) two-dimensional matrix of size 64 (nlat) by 128 (nlon) pixels. The channels represent seven physical three-dimensional variables: the temperature (ta), the eastward (ua) and northward (va) wind, relative humidity (hus), vertical velocity (wap), the relative vorticity (ζ), divergence (d) and geopotential height (zg) at 10 pressure levels from 1000 to 100 hPa, plus the surface pressure (ps). Another channel was added to represent the latitude: it is an image going from -1 at the top of the image (North Pole) to 1 at the bottom (South Pole) in every column. It was found that hard coding the latitude in the data improved the learning of physical constraints, allowing the network to be sensitive to the fact that the data are represented by the equirectangular projection of the atmospheric physical fields, and, for example, the size of meteorological objects increases closer to the poles. Finally, the choice of having diagnostic variables in the dataset was to help the post-processing, and assessment of their necessity requires further research.

3.2 Comparison between climate dataset and generated climate

Our study aims to have a generator able to reproduce the climate distribution present in the dataset made from the low-resolution GCM PLASIM. This section proposes a way to assess the quality of the distribution learned by the WGAN.

The first required property for a weather generator is a low computational cost compared to the GCM that produced the data. Here the simulation with the GCM PLASIM took 50 min for a 100-year simulation in parallel on 16 processors, whereas the generator took 3 min to generate 36 500 samples equivalent to a 100-year simulation on an NVIDIA Tesla V-100.

Each generated sample is compared with dataset samples. Figures 8 and 9 show a sample where only the pressure levels 1000, 500 and 100 hPa are represented for readability. It should be noted that the generated fields seem to be spatially noisy compared to the dataset. The periodic boundary is respected knowing that in the dataset the borders are located at the longitude 0° where no discontinuities can be observed. In the figures, the image is translated in order to have Europe at the center of the image and to see whether some discontinuities remain.

In order to quantitatively assess the generator quality, Figs. 10 and 11 show the mean and standard deviation pixel-wise differences over 10 800 samples (equivalent to 30 years of data) between normalized dataset and generations. It appears that fields where small-scale patterns are present are the most difficult to fit for the generator.

In order to go further in the analysis of the generated climate states, a singular value decomposition (SVD) was performed over 30 years of the dataset (renormalized over the 30 years). Then the same number of generated data was considered and projected onto the five first principal components

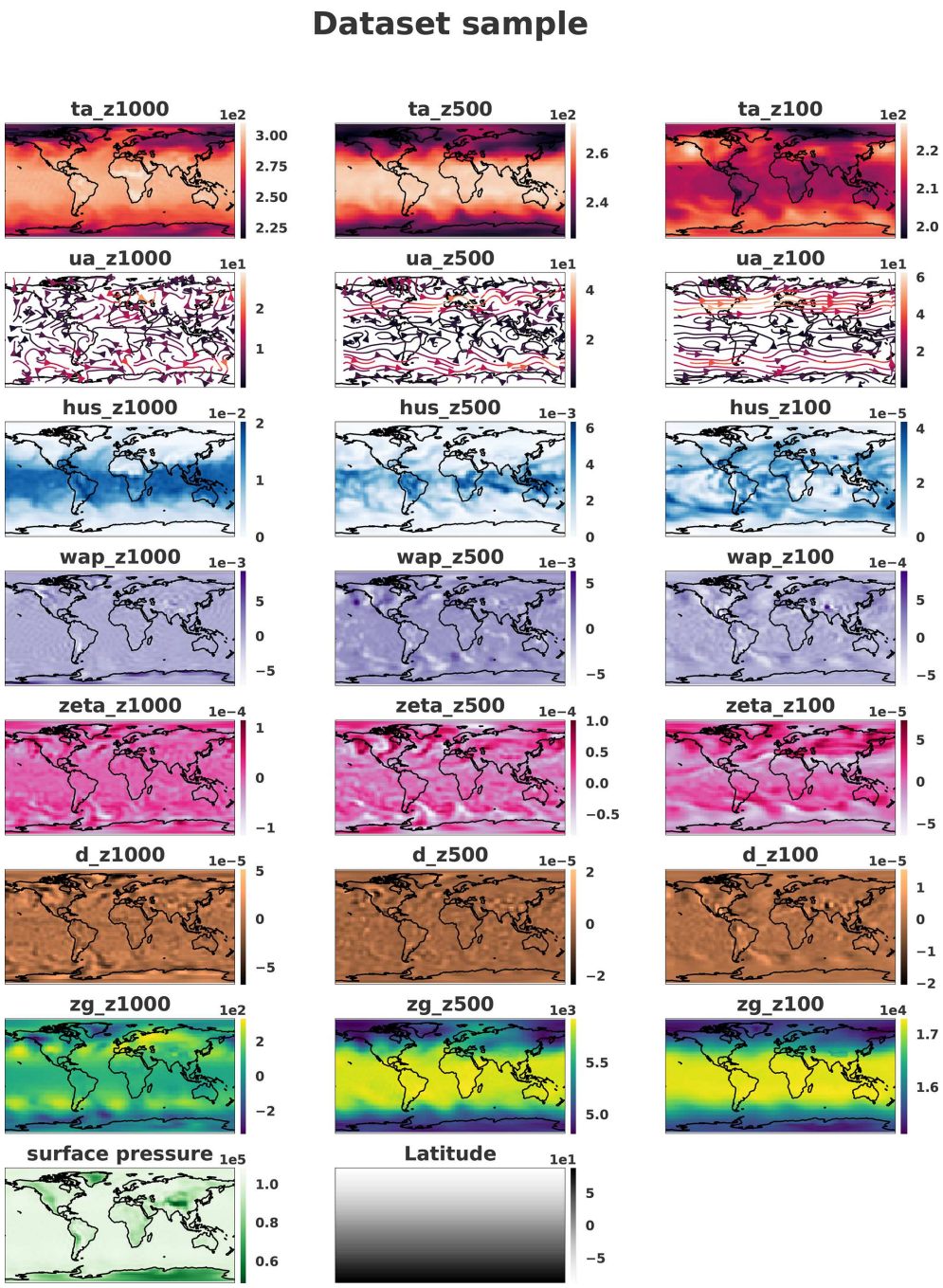


Figure 8. Sample on three different pressure levels (1000, 500 and 100 hPa) taken from the dataset. The samples were horizontally transposed in order to have Europe at the center of the images. Coastlines were added a posteriori for readability. Units available in Table 2.

of the SVD that represent 75 % of explained variance of the dataset. In Fig. 12 the dot product is represented between SVD components derived from the dataset $(u_i)_{i\in\{0,\dots,4\}}$ and another one derived from the generated data $(v_i)_{i\in\{0,\dots,4\}}$. Figure 12 represents the cross-covariance matrix defined by $s_{ij} = u_i \cdot v_j$. Values close to 1 or -1 show that the eigenvectors for both datasets (original and generated) are similar. This is another way of assessing whether the covariance structure of the original data is being preserved, and Fig. 12 shows that the five eigenvectors are similar. One should note that the SVD algorithm used from Pedregosa et al. (2011) suffers from sign indeterminacy, meaning that the signs of SVD components depend on the random state and the algorithm. For this reason, we consider the dot product close to both 1 and -1. One should note that an inversion remains between the components with indexes 3 and 4, which could

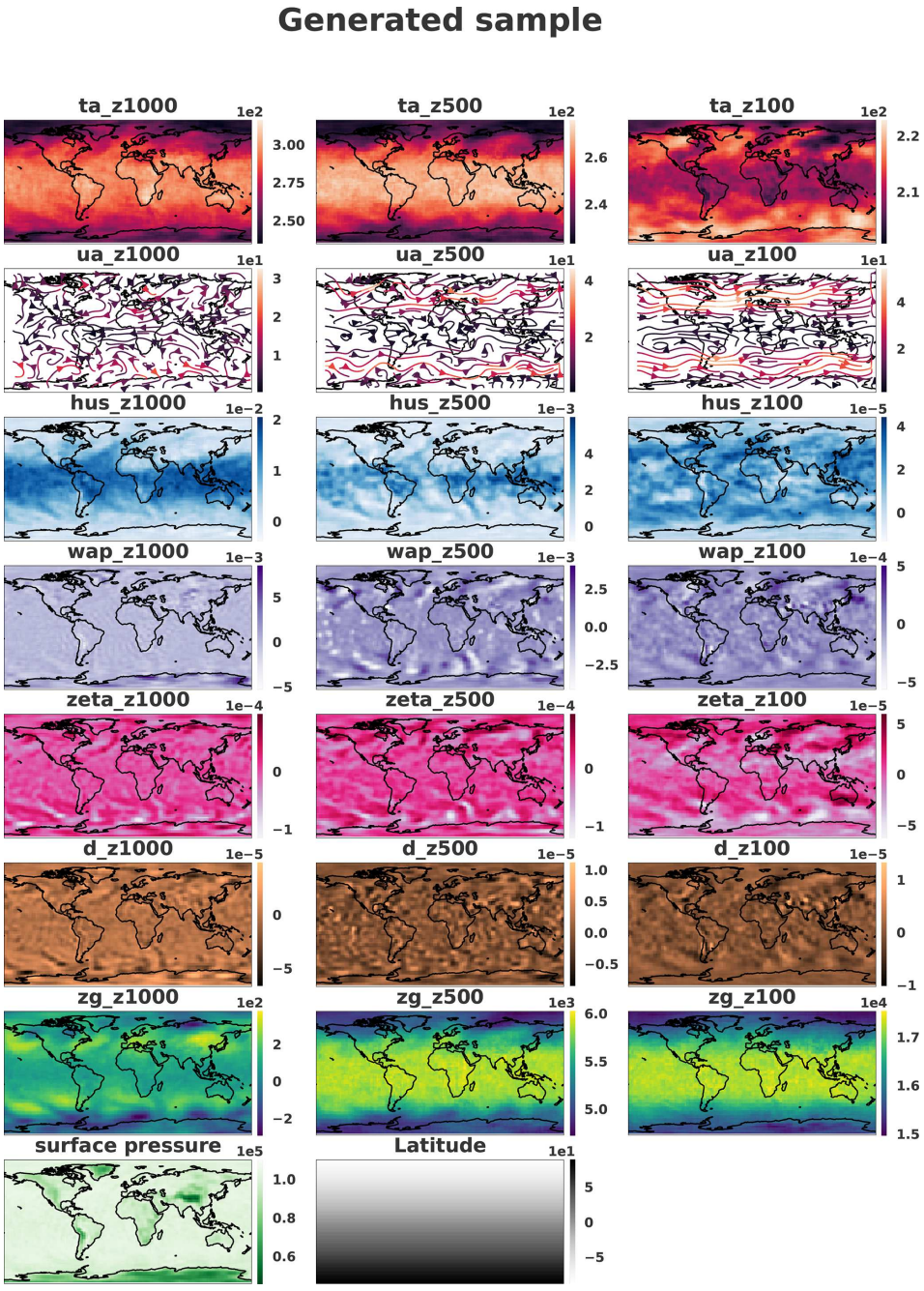


Figure 9. Sample on three different pressure levels (1000, 500 and 100 hPa) generated by the network. The samples were horizontally transposed in order to have Europe at the center of the images to verify the quality of the periodic boundary. Coastlines were added a posteriori for readability.

be explained by a difference of eigenvalue order (sorted in decreasing order) in each dataset that determines the order of eigenvectors. The fourth principal direction (index 3 in the figure) of the generated data represents more variation of the generated dataset than the same direction explains variation in the original dataset. Figure 13 shows clearly the inversion of the last principal components between the dataset and generations. This suggests a way of improving our method in future work.

Figure 15 shows the temperature (at the pressure level 1000 hPa) distribution at different pixel locations corresponding to the red dots in Fig. 14. Different latitudes (42, −2 and −70°) were chosen to represent diverse distributions. A value of Wasserstein distance is associated with each plot, representing the distance between the two normalized

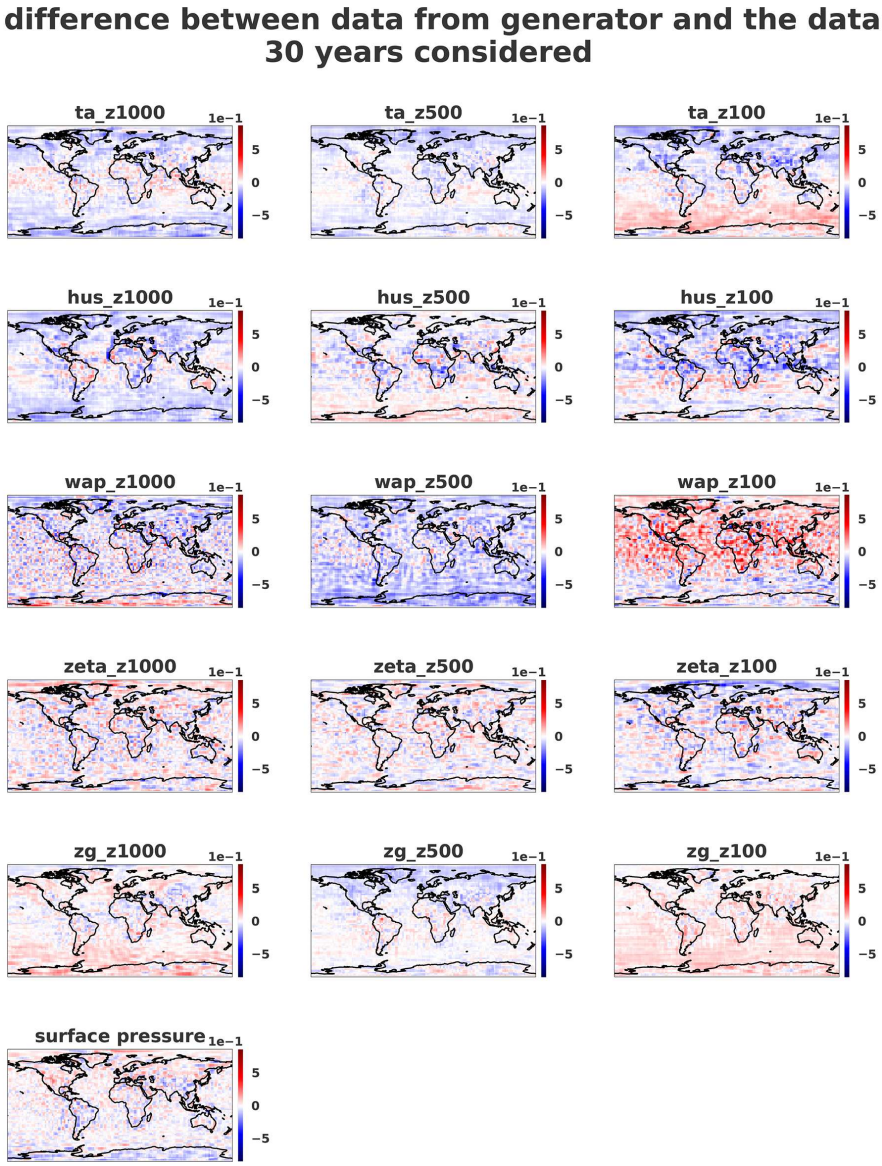


Figure 10. Mean error over 30 years of the normalized dataset and the same number of normalized generated samples on three different pressure levels (1000, 500 and 100 hPa). The samples were horizontally transposed in order to have Europe at the center of the images. Coastlines were added a posteriori for readability.

distributions. It is notable that the Wasserstein distance in the context of GAN training was introduced by Arjovsky et al. (2017) in order to avoid the mode collapse phenomenon where the generated samples produced by the GAN are representing only one mode of the distribution. In Fig. 15, even if the figure shows that some bimodal distributions remain approximated by a unimodal distribution, the span of these distributions covers the multiple modes of the targeted distribution. This explains why the higher Wasserstein distance in the figure is in the top-left panel, since despite the bimodal-generated distribution the high temperature values do not seem to be represented by the generated samples.

It follows that a good way to see the general statistics learned by the generator is to plot the Wasserstein distance for every pixel and for every variable. This result can be visualized spatially in Fig. 16, where we observe that certain variables are better fitted by the generator than others. The figure also shows that areas with more variability such as land areas and more precisely mountainous areas are the most difficult to fit. As a way to better interpret this metric, Fig. 17 represents the distributions corresponding to the minimum and maximum values of the metric. The distribution of the Wasserstein distance can also be visualized grouped by pressure level and type of variable in Fig. 18. The wap variable

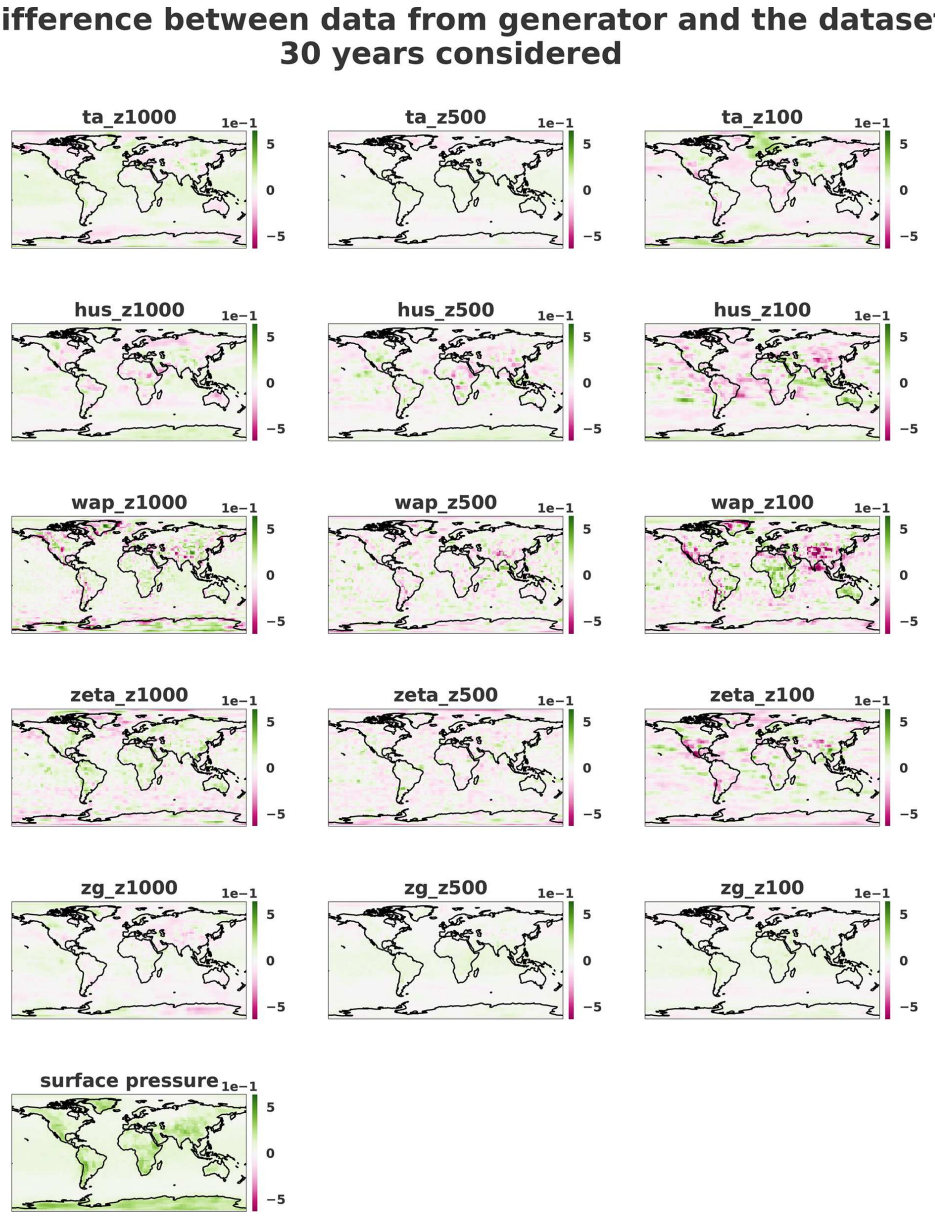


Figure 11. Standard deviation error over 30 years of the normalized dataset and the same number of normalized generated samples on three different pressure levels (1000, 500 and 100 hPa). The samples were horizontally transposed in order to have Europe at the center of the images. Coastlines were added a posteriori for readability.

that represents the vertical velocity seems to be the one with the higher Wasserstein distance value.

3.3 Analysis of the atmospheric balances

The previous subsection has shown the ability of the generator to engender weather situations and climate similar to those of the simulated weather. However, geophysical fluids are featured by multivariate fields that present known balance relations. Among these balances, the simplest ones are the geostrophic and thermal wind balances (see, e.g., Vallis, 2006). The next two sections assess the ability of the generator to reproduce the geostrophic and thermal wind balances.

3.3.1 Geostrophic balance

The geostrophic balance occurs at a low Rossby number when the rotation dominates the nonlinear advection term. Two forces are in competition: the Coriolis force, $f\boldsymbol{k}\times\boldsymbol{u}$, where $\boldsymbol{k}$ denotes the unit vector normal to the horizontal (f is the Coriolis parameter and $\boldsymbol{u}$ is the wind) and the pressure term $-\nabla_p\Phi$, where Φ is the geopotential and where ∇_p denotes the horizontal gradient in the pressure coordinate.

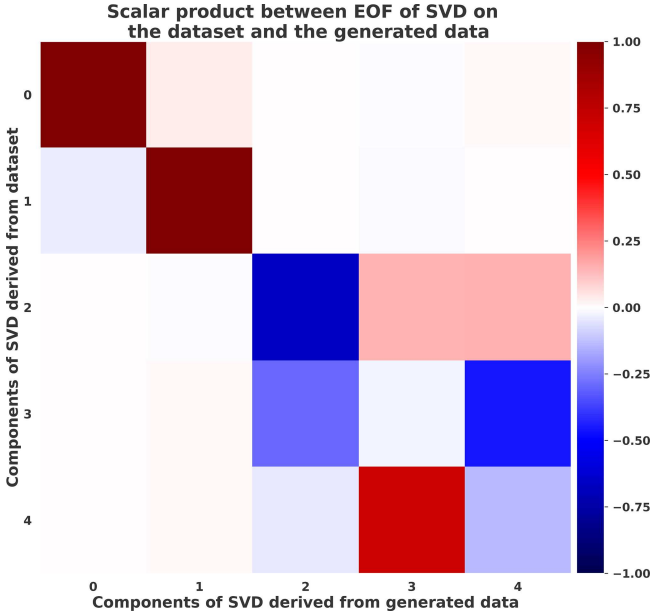


Figure 12. Scalar product of SVD components derived from a dataset and generated data.

Asymptotically, the Coriolis force is then balanced by the pressure term which leads to the geostrophic wind:

$$\boldsymbol{u}_{\mathrm{g}} = \frac{1}{f}\boldsymbol{k} \times \nabla_p \Phi. \tag{15}$$

The geostrophic flow is parallel to the line of constant geopotential, and it is counterclockwise (clockwise) around a region of low (high) geopotential. The magnitude of the geostrophic wind scales with the strength of the horizontal gradient of geopotential Vallis (2006, Sect. 2.8.2, p. 92).

This asymptotic balance Eq. (15) is verified to within 10 % of error at mid latitude, that is, $\boldsymbol{u} = \boldsymbol{u}_{\mathrm{g}} + \boldsymbol{u}_{\mathrm{ag}}$, where the magnitude of the ageostrophic wind, $\boldsymbol{u}_{\mathrm{ag}}$, is less than 0.1 of the magnitude of the real wind $\boldsymbol{u}$.

Figure 19a illustrates a particular boreal winter situation from the PLASIM dataset, focusing on the mid latitude and presenting a low area of geopotential in the southwest of Iceland. It appears that the wind is well approximated by the geostrophic wind, which is quantitatively verified in Fig. 20a that shows the norm of the ageostrophic wind normalized by the norm of the wind (that is, the relative error when approximating the wind by the geostrophic wind): the order of magnitude of the error is around 20 %. Properties of the geostrophic flow are visible, with a counterclockwise flow around the low geopotential. The wind is maximum where the horizontal gradient of geopotential is maximum, while its change in direction follows the trough.

A similar behavior can be observed in Fig. 19b, which illustrates a weather situation selected from the render by the generator of some samples in the latent space, so as to represent a boreal winter situation. This time, a low geopotential is found in the north of Europe. While the geopotential field is noisy (it is less smooth than in Fig. 19a), the wind is again found to be nearly geostrophic, verifying the geostrophic flow properties to within an error of 35 % (see Fig. 20b). The geopotential and wind fields were projected onto the solved dynamic truncation in order to remove the subgrid component due to the noise in the output of the generator. Despite the truncation, the geostrophic approximation seems to not be respected everywhere and could be a quantitative metric to monitor in order to improve our method.

We find that weather situations generated from samples in the latent space reproduce the geostrophic balance at an order of approximation that is similar to the one of the real dataset. This means that the generator is able to produce the realistic multivariate link between the wind and the geopotential. This property is essential in operational weather forecasting, e.g., in producing balanced fields in the ensemble Kalman filter.

3.3.2 Thermal wind balance

The thermal wind balance arises by combining the geostrophic wind Eq. (15) and the hydrostatic approximations, $\frac{\partial \Phi}{\partial p} = -\frac{1}{\rho}$, where ρ is the density (Vallis, 2006, Sect. 2.8.4, p. 95): taking the derivative of Eq. (15) with respect to the pressure p makes the hydrostatic approximation appear, so that the vertical derivative of the geostrophic wind can be written as

$$\frac{\partial \Phi}{\partial p} = -\frac{R}{pf}\boldsymbol{k} \times \nabla_p T, \tag{16}$$

where the ideal gas equation, $p = \rho R T$, has been used. Equation (16) is the thermal wind balance that relates the vertical shear of the horizontal wind to the horizontal gradient of temperature. In particular, when the temperature falls in the poleward direction, the thermal wind balance predicts an eastward wind that increases with height.

Figure 21a and b show the vertical cross section of the zonal average of temperature and of the zonal wind for a particular weather situation in the dataset, corresponding to a boreal winter situation of the same weather situation represented in Fig. 21: the temperature is higher in the Southern Hemisphere than in the Northern Hemisphere, with a strong horizontal gradient of temperature in latitude ranges $[-80°, -40°]$ and $[40°, 80°]$. At the vertical of the horizontal gradient of temperature, the wind is eastward and increases with the height: this illustrates the thermal wind balance which produces a strong curled jet at the vertical of the strong horizontal gradient of temperature as shown in Fig. 22a that illustrates, for the same weather situation, the temperature at the bottom (800 hPa) with the horizontal wind at the top (200 hPa) of the troposphere.

The same illustrations are shown in Fig. 21c and d when considering a generated situation, selected to correspond to a boreal winter situation: the characteristics related to the thermal wind balance as observed before are found again. This

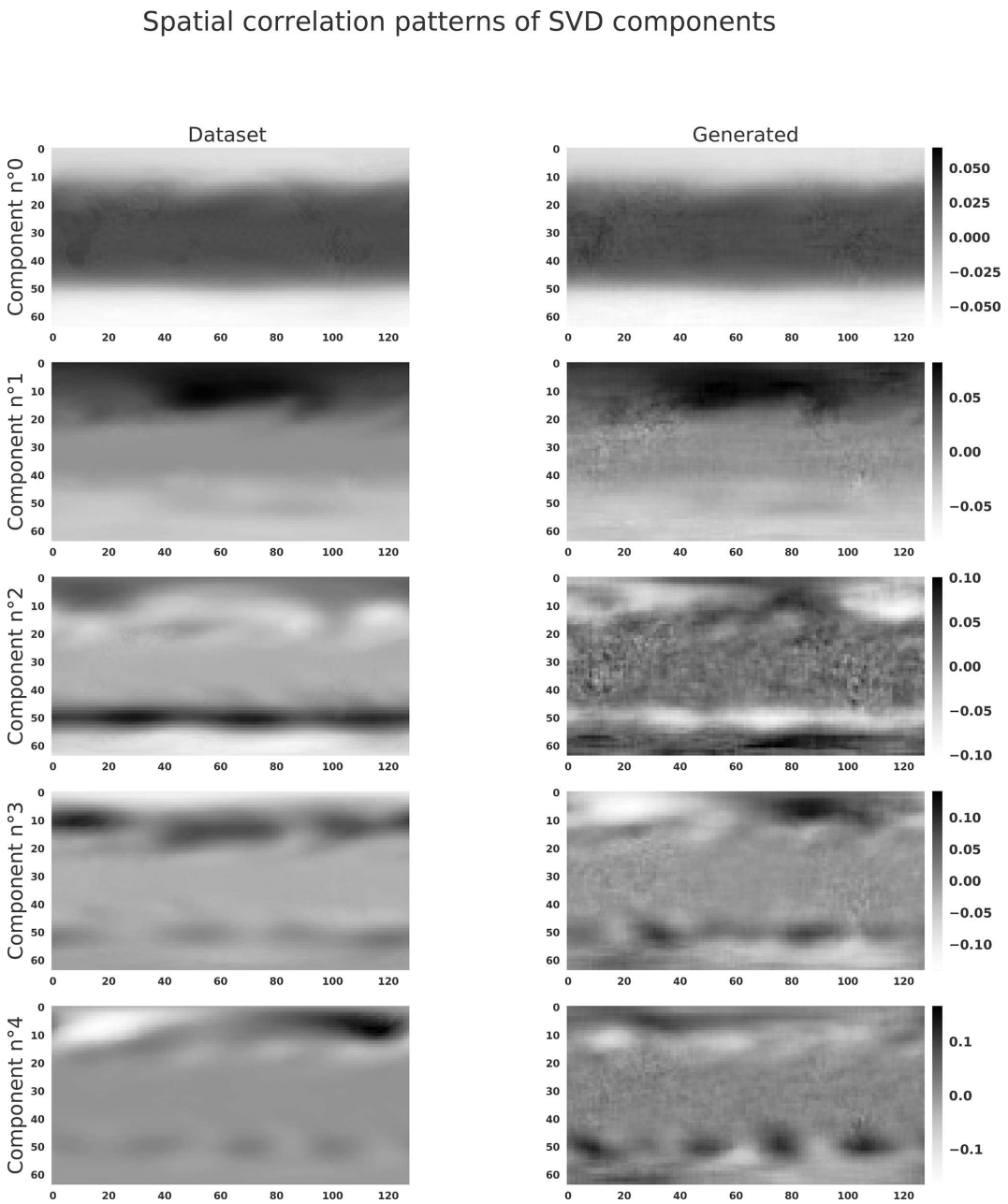


Figure 13. Spatial components corresponding to principal components of SVDs applied to the dataset and the generated samples.

results in the generator being able to render a weather situation that reproduces the thermal wind balance. Moreover, Fig. 23 shows the thermal wind balance averaged on 30 years for the dataset (Fig. 23a) and generations (Fig. 23b); both are very similar.

This section has shown the ability of the generator to reproduce some important balances present in the atmosphere. In particular, the generator is able to produce mid-latitude cyclones whose velocity field is in accordance with the geostrophic balance. The authors emphasize that it is necessary to conduct more analysis of the weather situations outputted by the generator, which is beyond the scope of this study. For example, it would be interesting to assess whether other inter-variable balances are present, such as the ω equation or vertical structures. Note that adding advanced diagnostic fields in the output of the generator could be investigated to improve the realism.

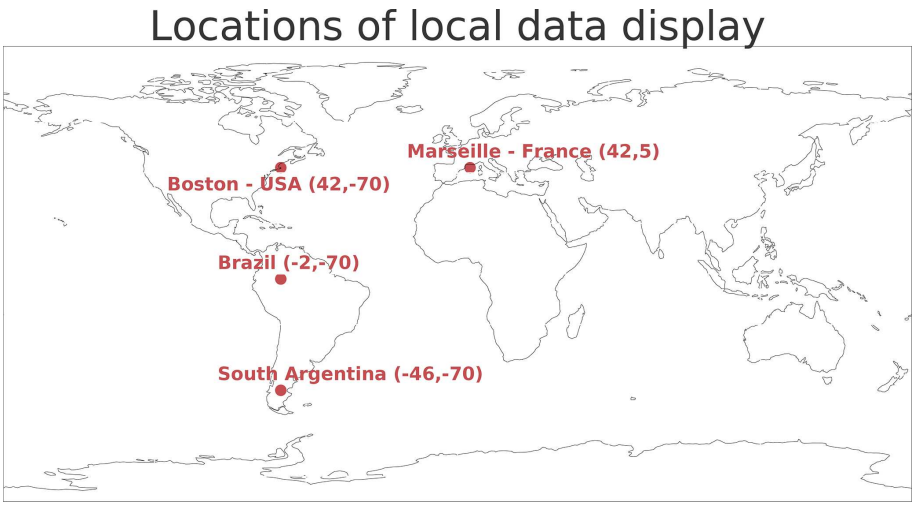


Figure 14. Location from where the temperature distributions are plotted in Fig. 15. The Wasserstein distance value associated for each plot was computed on normalized data.

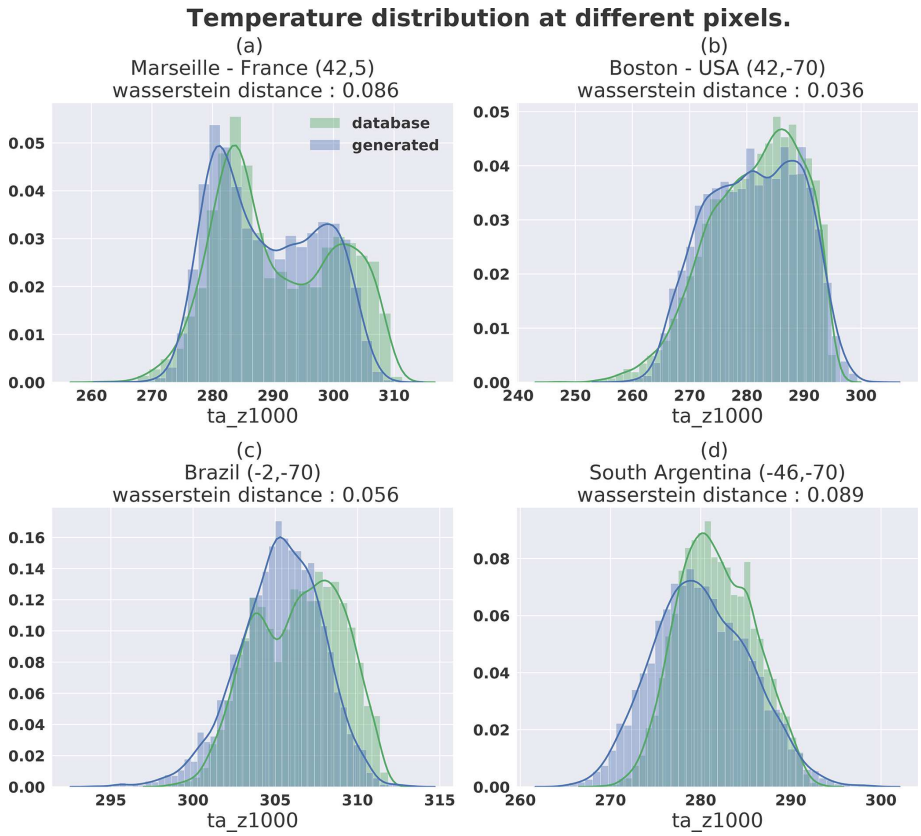


Figure 15. Temperature distribution at different locations for 5000 samples from dataset (green) and generated (blue).

3.4 Exploration of the latent space structure and its connection to the climate

An exploratory study was done on the property of the latent space and its consequence in the climate space in regard to climate domain problematics. If the generator is perfectly trained, then each sample generated with it should represent a typical weather situation. It is hard to figure out what the attractor of the climate is. However, the geometry of the Gaussian in high dimension being known, it is easy to characterize the climate in the latent space.

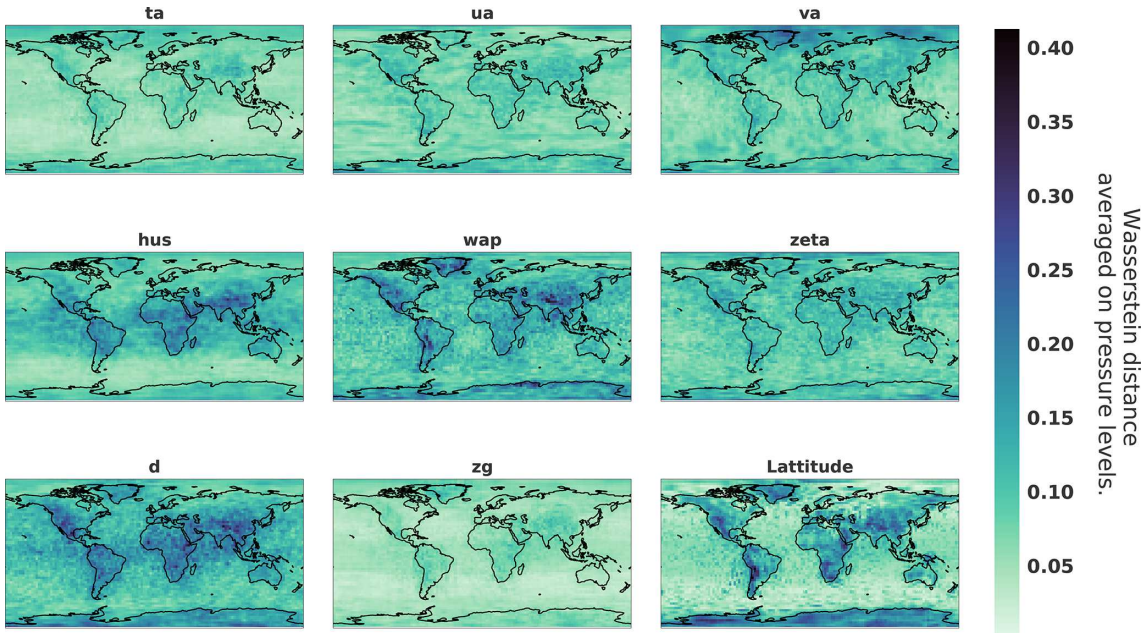


Figure 16. Wasserstein distance between 5000 datasets and generated samples on each pixel and each channel.

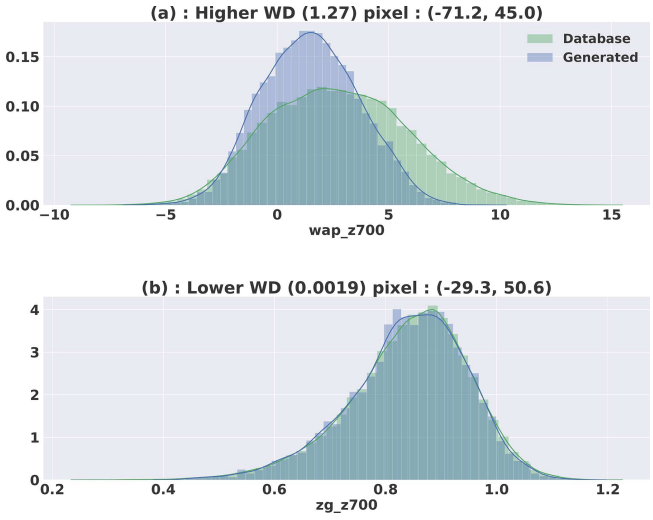


Figure 17. Distributions with the higher **(a)** and lower **(b)** Wasserstein distances computed on normalized data. The coordinates of corresponding pixels are, respectively, in latitude and longitude.

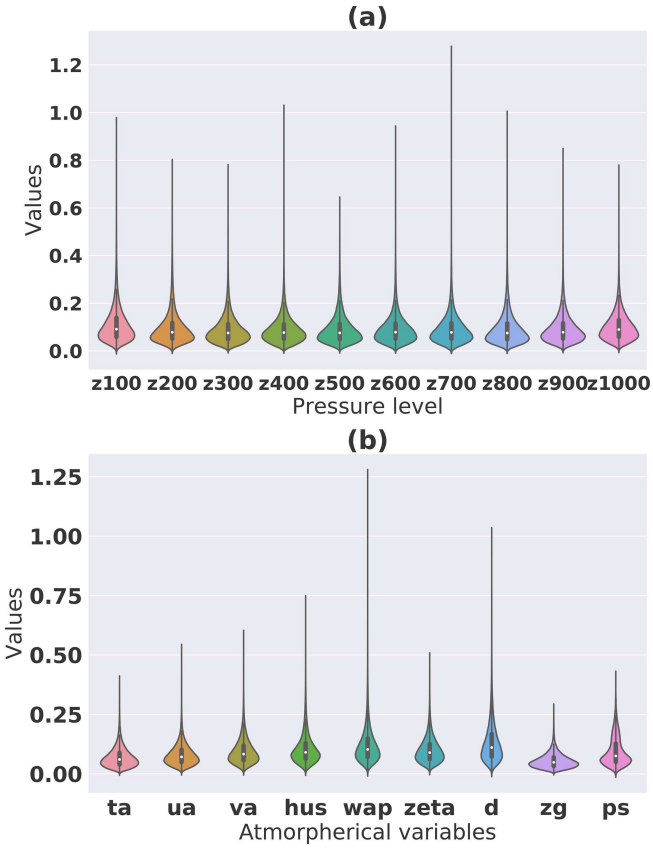


Figure 18. Wasserstein distance between 5000 datasets and generated samples on each pixel grouped by pressure height **(a)** or variables **(b)**.

3.4.1 Geometry of the normal distribution

For a normal law in the high dimension space $Z = \mathbb{R}^m$, i.e., with m larger than 10, the distributions of the samples are all located in a spherical shell of radius $\sqrt{m}$ and of thickness on order $\frac{1}{\sqrt{2}}$ (see, e.g., Pannekoucke et al., 2016). Because the covariance matrix $\mathbf{I}_m$ is a diagonal of constant variance, no direction of $\mathbb{R}^m$ is privileged, leading to an isotropic distribution of the direction of the sampled vectors: their unit directions uniformly cover the unit sphere. Another property

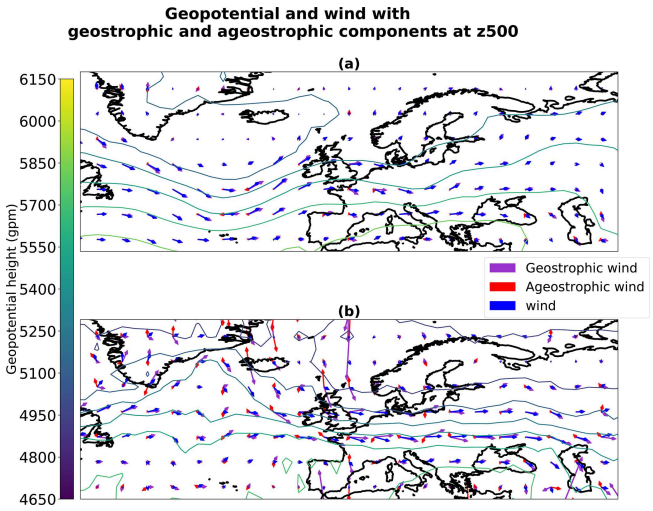


Figure 19. Geostrophic and ageostrophic wind derived from geopotential at 500 hPa. Situation taken from dataset **(a)** and generated **(b)**.

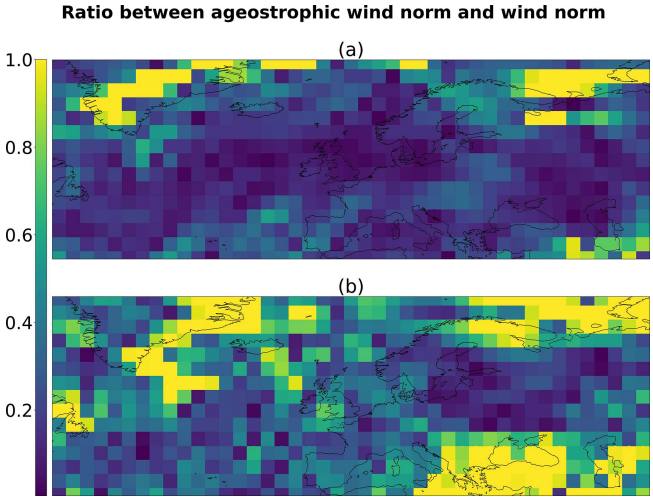


Figure 20. Relative error in the norm between geostrophic wind and normal wind shown in Fig. 19 for the situation taken from dataset **(a)** and generated **(b)**.

is that the angle formed by two sampled vectors is approximately of magnitude $\frac{\pi}{2}$: two random samples are orthogonal. These are simple consequences of the central limit theorem which predict, for instance, that the distance of a sample to the center of the sphere is asymptotically the Gaussian $\mathcal{N}(\sqrt{m}, \frac{1}{2})$.

Considering these properties, one can introduce a two dimensional pseudo-representation which preserves the isotropy of the distribution as well as the distribution to the origin: a random sample vector $\boldsymbol{x} = (x_1, x_2, \cdots, x_m)$ in $\mathbb{R}^m$ is represented by the projection $P_2(\boldsymbol{x}) = ||\boldsymbol{x}|| \frac{1}{\sqrt{x_1^2+x_2^2}}(x_1, x_2)$, where $||\cdot||$ stands for the Euclidian norm in $\mathbb{R}^m$.

Figure 24 illustrates this low-dimensional representation of an ensemble of 10 000 samples of the normal law in dimension $m = 64$. For instance, points A and B represent two independent samples: their distance to the origin is closed to $\sqrt{m} = 8$, and their angle is closed to $\frac{\pi}{2}$. While $m = 64$ can be considered a very small dimension, it appears that the distribution of the point's distance to the origin is well fit by the Gaussian $\mathcal{N}(\sqrt{64}, \frac{1}{2})$ (see inset figure in Fig. 24). Hence, it results that for this dimension, the interpretation of a Gaussian distribution as a spherical shell applies, with interesting consequences for extremes or typical states. A typical sample of this normal law is a point near the sphere of radius $\sqrt{64}$, while an extreme sample has a norm lying in the tails of the distribution $\mathcal{N}(\sqrt{64}, \frac{1}{2})$.

This suggests evaluating whether the extremes of the latent space correspond to those of the meteorological space.

3.4.2 Connection between extremes in the latent and physical spaces

Knowing what are the extremes in the latent space might be helpful to determine what are the extremes of the climate, at least to determine what are extreme situations closed to a given state.

For any sample in the latent space, say point A, we can construct the point on the sphere $\sqrt{m}$ along the same direction of A, $\overline{A}$, which can be considered the most likely typical state near A. Along the same direction of A, we can also construct the extreme situations $A^{\pm}$ whose distances to the origin, $\sqrt{m} \pm \frac{3}{\sqrt{2}}$, lay, respectively, in the left and right tails of the Gaussian distribution $\mathcal{N}(\sqrt{m}, \frac{1}{2})$.

Figure 25 represents the weather situation generated from a randomly drawn latent vector from a 64-dimensional Gaussian $\mathcal{N}(0, 1)$ sample A (Fig. 25a). Panel (a) represents a latent vector with a Euclidian norm equal to 7.69, close to the mean of the radial distribution of the hypersphere mentioned in Sect. 3.4.1. In the climate space this sample shows a meteorological object above northern Europe in the shape of a geopotential minimum which can be interpreted as a storm. This sample is the same as the one represented in Figs. 19b, 21b, and 22b.

The most likely typical state $\overline{A}$ (Fig. 25b) is the radial projection of the latent vector $\boldsymbol{A}$ onto the mean of the radial distribution; thus, its Euclidian norm is equal to 8. Because sample A has a norm close to sample B, the weather situations are very similar at the geopotential height at z1000. This is an expected effect because by construction of the generator the input space is continuous, so two points in the latent space must be similar. Extreme situation $A^{\pm}$ along the direction of A is represented in Fig. 25c and d. Both panels shows clear differences in the geopotential height. First the panel (Fig. 25c) shows a decrease in the storm located above northern Europe; the same effect is visible in the south of South America. However, the weather situation is very similar to Fig. 25a. By contrast, Fig. 25d represents a deeper geopoten-

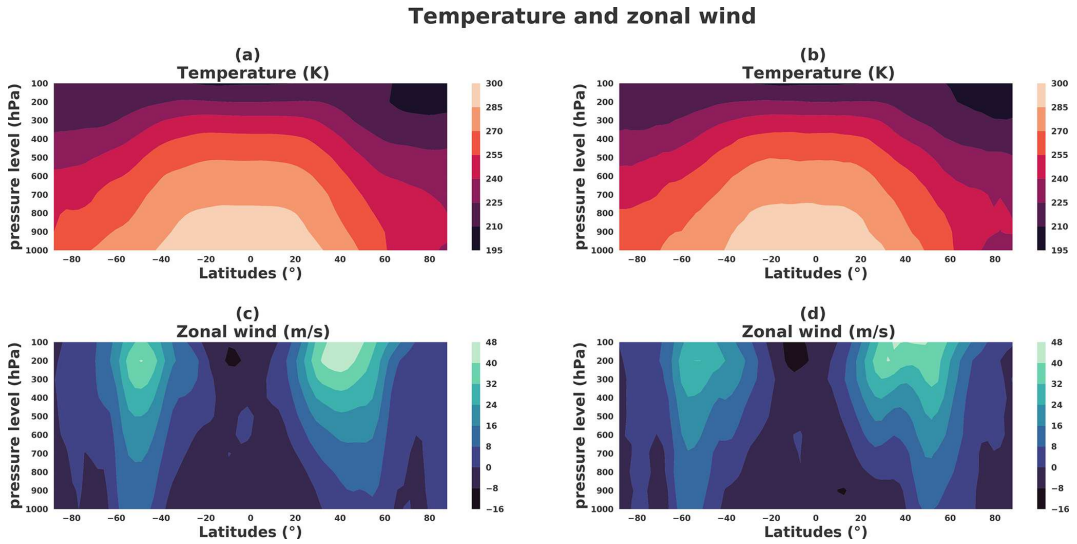


Figure 21. Temperature (K) and zonal wind ($\mathrm{m\,s^{-1}}$) latitude zonals from a boreal winter situation: the thermal wind balance. Left panels correspond to a situation taken from the dataset. **(a)** Zonal temperature and **(c)** zonal wind. Right panels correspond to a situation taken from the generator. **(b)** Zonal temperature and **(d)** zonal wind.

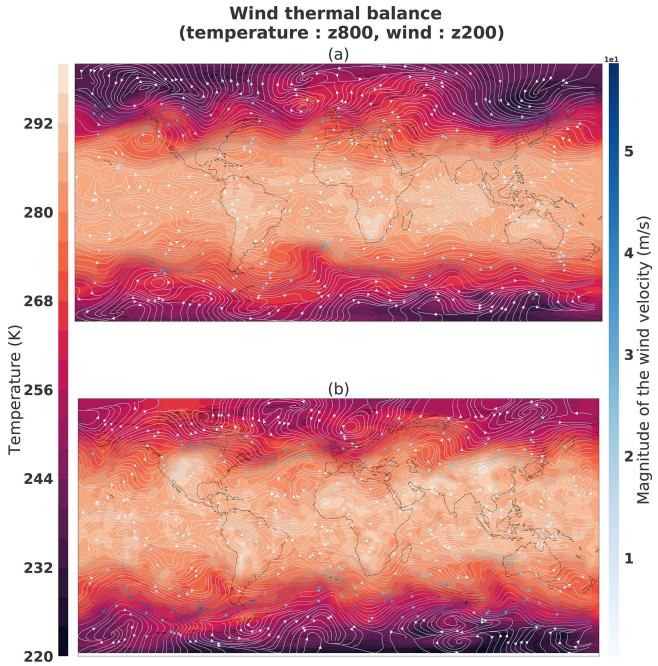


Figure 22. Thermal wind balance from the boreal winter situation shown in Fig. 21: **(a)** sample from the dataset; **(b)** sample generated by the generator. The temperature (K) is from pressure level 800 hPa and the wind ($\mathrm{m\,s^{-1}}$) from 200 hPa.

tial height minimum at the pre-existing storm of sample A. Thus, Fig. 25 seems to show a certain structure of the latent space generator where the radial direction could represent the strength of the meteorological objects such as storms above Europe, for example. It could be explained by the fact that the generator aims to map a distribution (64-dimensional Gaussian in the latent space) to another (weather distribution in the PLASIM physical space). Rare events exist in the latent space on the tails of the Gaussian distribution's potentially extreme weather situations. One of the ways to do a such mapping is to use the radial direction to represent high- or low-probability states of the climate. An important conclusion is that, for a given situation, the most likely state and the extremes are interesting physical states. This could open new possibilities to study an extreme situation close to a given one, which is an important topic, e.g., for insurance or to improve the study of high weather impact in ensemble forecasting.

The link of the animation of such interpolation is available on GitHub[1] of the project.

3.4.3 Interpolation in the latent space

Even if there are no dynamics in the latent space, which makes it impossible to construct a prediction from this space, we can consider how to interpolate two latent states. A naive answer is to compute the linear interpolation between two samples of the latent space A and B,

$$M_t = G\left((1-t)A + tB\right), \tag{17}$$

which results in the red chordal illustrated in Fig. 24. The chordal interpretation highlights a major drawback of the linear interpolation: middle points of the chordal are extremes; these intermediate points should not correspond to typical (or even physically realizable) weather situations.

[1] https://github.com/Cam-B04/Producing-realistic-climate-data-with-GANs (last access: 15 January 2021)

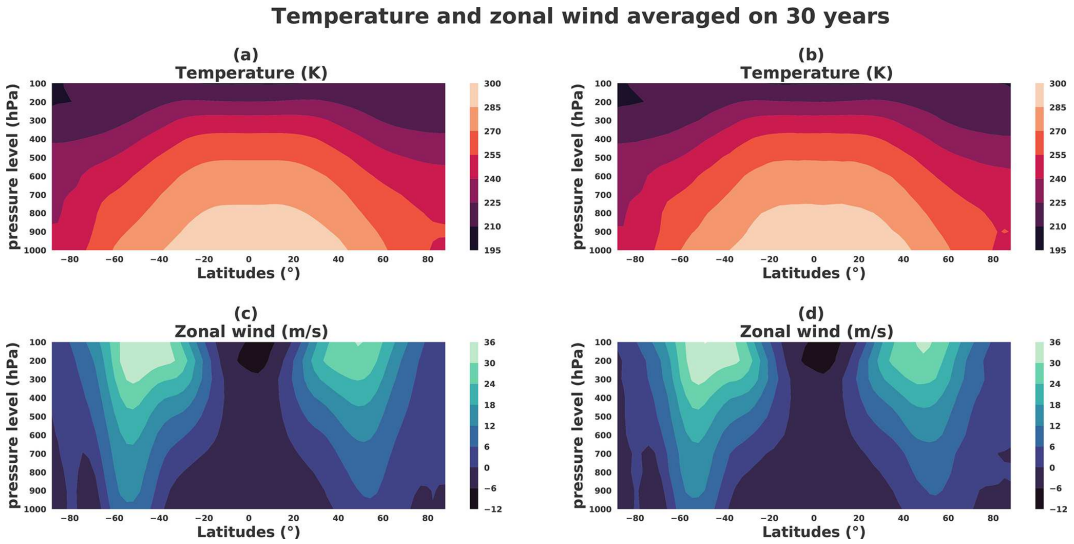


Figure 23. Temperature (K) and zonal wind ($\mathrm{m\,s^{-1}}$) latitude zonals averaged on the 30-year subsample. Left panels correspond to a situation taken from the dataset: **(a)** zonal temperature and **(c)** zonal wind. Right panels correspond to a situation taken from the generator: **(b)** zonal temperature and **(d)** zonal wind.

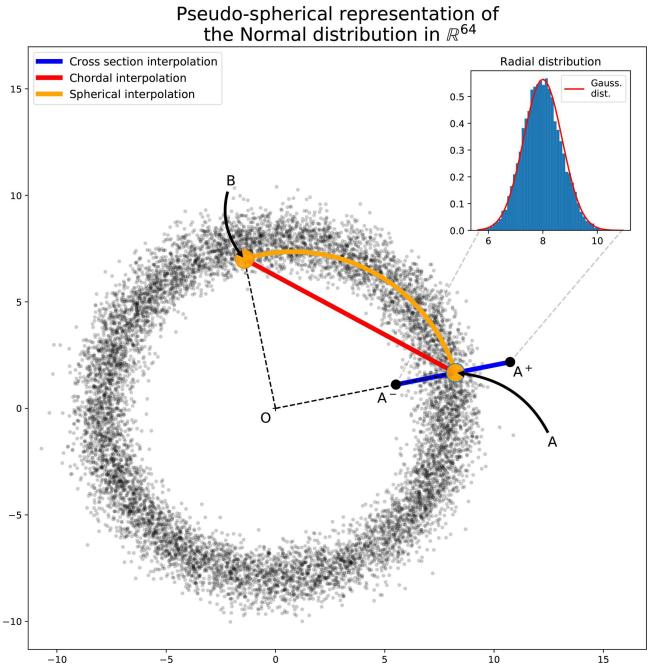


Figure 24. Pseudo-spherical metaphorical representation of 10 000 samples of the normal distribution in $\mathbb{R}^m$ with $m = 64$ and the distribution of the distance of samples to the center of the spherical shell. For a sample A, $A^{\pm}$ denotes two extreme situations along the direction of A. Any second sample B, typical of the distribution, appears orthogonal to A. The inset figure represents the radial distribution compared with the asymptotic central limit theorem (CLT) Gaussian distribution $\mathcal{N}(\sqrt{m}, \frac{1}{2})$ (thin red curve).

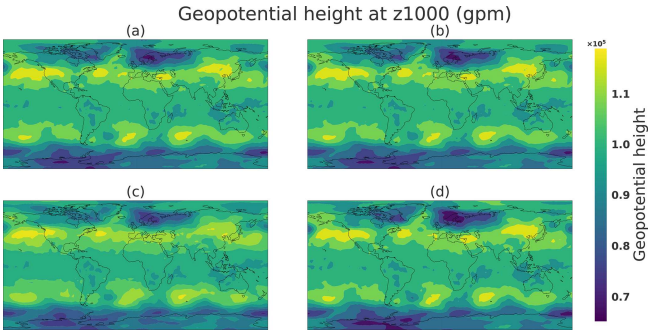


Figure 25. Generations obtained by radial interpolation in the latent space. Panel **(a)** is the image corresponding to a randomly drawn latent vector $\boldsymbol{A}$ (two-norm: 7.69), **(b)** is its projection onto the mean of the same direction $\overline{A}$ (two-norm: 8.0), and **(c)** and **(d)** are the projection onto, respectively, inferior A^- (two-norm: 5.87) and superior A^+ (two-norm: 10.12) 1 % quantile (see Fig. 24).

So as to preserve the likelihood of the interpolated weather situations, it is better to introduce a spherical interpolation. This kind of interpolation has also been used in image processing, where, e.g., White (2016) uses the formula

$$M_t = G\left(\frac{\sin((1-t)\theta)}{\sin\theta}A + \frac{\sin(t\theta)}{\sin\theta}B\right), \tag{18}$$

where θ is the angle $\widehat{A,B}$ and for $t \in [0,1]$ such as $M_0 = G(A)$ and $M_1 = G(B)$.

This interpolation will connect point A to point B within the spherical shell of typical states, as illustrated by the orange curve line in Fig. 24. Figure 26 shows snapshots of the climate generated from a spherical interpolation in the latent space between sample A and another random sample

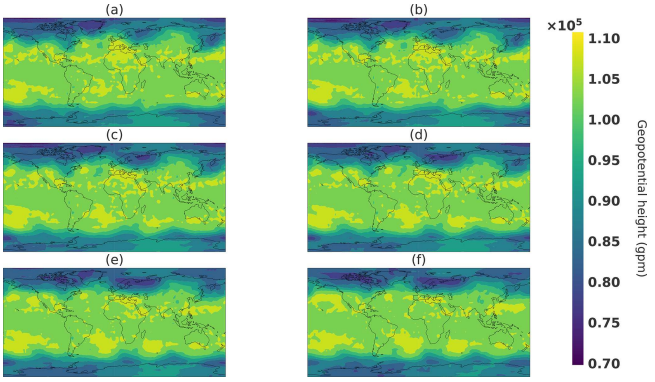


Figure 26. Spherical interpolation snapshots. Respectively, panels **(a–f)** correspond to values of t in Eq. (18) of 0, 0.2, 0.4, 0.6, 0.8, and 1.

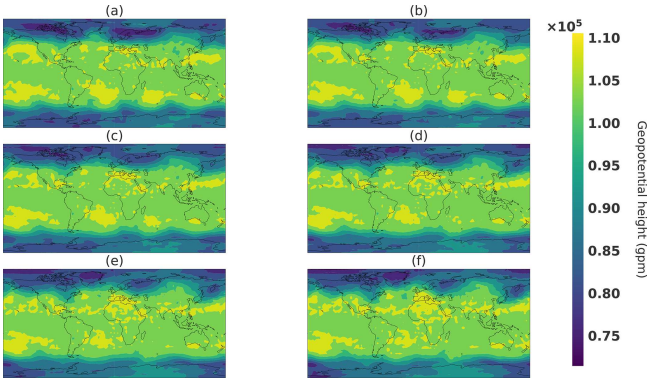


Figure 27. Linear interpolation in the latent space interpolation snapshots. Respectively, panels **(a–f)** correspond to values of t in Eq. (17) of 0, 0.2, 0.4, 0.6, 0.8, and 1.

B. For the sake of comparison, Figs. 27 and 28 are, respectively, snapshots of a linear interpolation in the latent space described in Eq. (17) and in the image space using the following equation:

$$M_t = (1-t)G(A) + tG(B). \tag{19}$$

The objective of this experience is to be able to produce realistic intermediate states. This can be visible in Fig. 26, where the storm above Europe emerges by first a smaller minimum in geopotential height that increases in size, whereas in both linear interpolations, in the latent and image spaces, the storm appears first as a long and thin geopotential minimum and then broadens in the latitude direction. Such a property can be helpful in the context of fluid dynamics for initial and boundary conditions of a local area model to avoid error correlated with user-defined parameters such as in lateral boundary conditions (Davies, 2014). An interesting generator property would be able to choose some characteristics of the generated climate such as meteorological objects at certain locations. In the next section, an experiment is conducted to see whether it is possible to change the location of such meteorological objects.

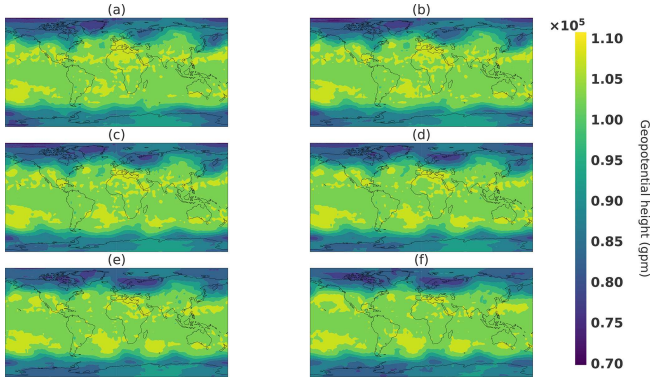


Figure 28. Linear interpolation in the image space. Respectively, panels **(a–f)** correspond to values of t in Eq. (19) of 0, 0.2, 0.4, 0.6, 0.8, and 1.

3.4.4 Coherent structure perturbation from the latent space

In this section, the goal is to study the difference between two climate states coming from close latent points. In this experiment, sample $G(A)$ will be the reference climate state, and we added noise to A such as $A = A + \epsilon_i$ with ϵ_i taken from $\mathcal{N}(0, 0.1)$.

Figure 29 shows the different climate states corresponding to $G(A)$ and $G(A+\epsilon_i)$ in the first column and the difference with the reference in the climate states $G(A) - G(A+\epsilon_i)$ in the second column. The second column shows dipoles that represent the movement of meteorological structures, for example, in the South American area of panel d. We remarked that the perturbation of one latent vector is translated in the climate state by a dipole creation when the difference is done between the reference and perturbed versions. This shows the possibility of moving the meteorological object by remaining on the manifold of the realistic climate state. This is an interesting asset for the climate domain, where it is complicated to interpolate between two states where a storm is at two different locations as mentioned in Hergenrother et al. (2002). The WGAN could be a way to propose realistic intermediate states.

4 Conclusions

Our study shows that it is possible to map the climate distribution output of a GCM to a much simpler low-dimensional distribution using a highly nonlinear neural-network-based generator. It also proposes ways to assess the quality of the generator by evaluating statistical quantities as well as with respect to physical balance properties.

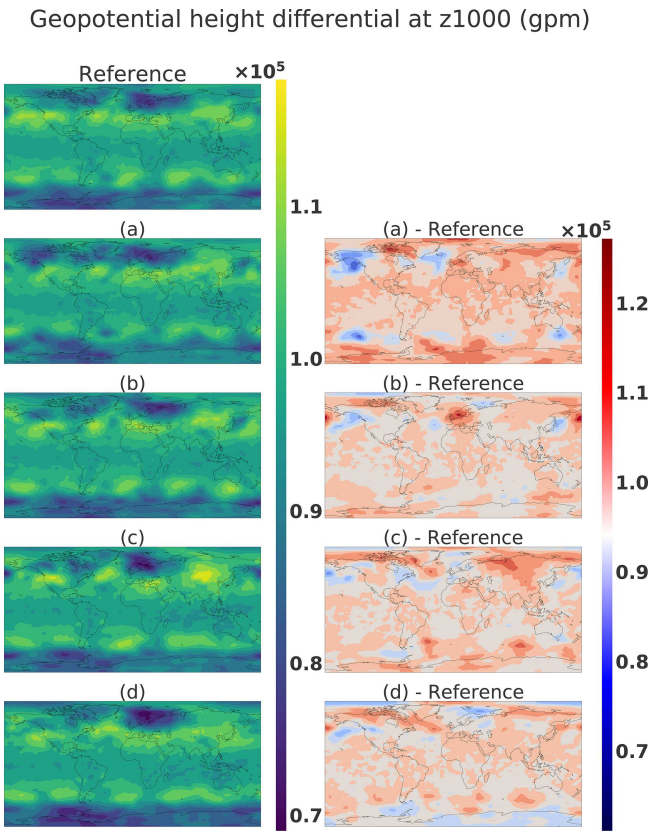


Figure 29. Geopotential height: the first column reference corresponds to $G(A)$, and panels **(a–d)** correspond to $G(A+\epsilon_i)$ and the second column $G(A)-G(A+\epsilon_i)$.

In this article, a weather generator based on the WGAN method able to produce realistic states of the atmosphere was created. Metrics such as SVD principal component comparison, Wasserstein distance on pixel value distribution and mean and standard deviation comparison were used in order to be compared to other future proposed methods.

A comparison of the atmospheric balance was realized between samples and averaged over 30 years of data, showing promising results. Coherence between variables as well as spatial coherence were also shown to be promising.

Interesting properties of such a generator were discussed with regard to possible applications in insurance, weather simulation and data assimilation. The generator is able to generate intermediate realistic climate states with coherent structures, interpolate between two defined states with other plausible states, and create realistic perturbations around a climate state, all at a low computational cost compared to a GCM.

A study was also done on the interpretability of the latent space and the connections between the extreme events in the data space and the latent space. It highlighted the radial direction as the direction of the intensity of climate events.

Our results highlight the ability of the method to handle the mapping of a high-dimensional distribution onto a multivariate Gaussian. We believe this is an important step that opens many opportunities for climate data exploration. Some extensions of this work as well as potential application are highlighted in the following.

First, the WGAN could be conditioned by the season or by the day in the year. Such conditioning would give access to other quantitative methods to assess the quality of the weather generator. It would be also an important step towards application in the risk assessment area, for example.

Optimization can be done to find specific states in the latent space by defining an objective function such as Euclidian distance in the climate space. The network gradient with respect to its inputs being accessible, direct minimization can be used to find climate states that fit observations in data assimilation problems. More advanced strategies, such as training a separate inference network (Chan and Elsheikh, 2019), are also possible to apply Bayes' rule without using a particle filter. It is also possible to condition the generations to a specific date in the annual cycle with slight modifications in the network architecture. One could think to condition the output of the generator by a forcing field in input such as forcing fields like SST fields for data assimilation application, which should be possible but with more important modifications of the network architecture and a possible impact on the speed of the training procedure.

A more sophisticated dataset could be used, such as a true climate reanalysis, to see the effect of the dataset complexity on the method's performance. The optimization of the network's architecture and a sensitivity study on the hyperparameters such as the dimension of the latent space, for example, would be useful. Moreover, it would be interesting to see whether it is possible to take advantage of the GAN trained in PLASIM to facilitate the training of a GAN on the reanalysis.

The structure of the latent space and its interpretability is also a critical way to exploit the specificities of the method. The opportunity to find similar climate states with extreme events is also something not possible with other weather generators and could have lots of application for risk assessment applications.

The definition of additional metrics to assess the quality of the generator should be the main focus following this study to identify improvement of the method and facilitate the participation from diverse researcher communities.

Finally, we could consider restarting the GCM from a generated state to assess how well balanced the generated fields are, which could have important implications in data assimilation methods.

The study is a first step towards deep-learning weather generation; while many challenges remain to be solved, it shows several potential applications of GANs to improve the effectiveness of current approaches.

Code availability. The code and the weights of the trained neural network are available at the following GitHub repository in v0.1: https://github.com/Cam-B04/Producing-realistic-climate-data-with-GANs.git (last access: January 2021). The repository is associated with the following DOI: https://doi.org/10.5281/zenodo.4442450 (Besombes, 2021).

Data availability. The dataset used is available on demand. The GitHub repository explains how to recreate it from a PLASIM simulation.

Author contributions. The authors contribute to the design of the neural network architecture and the experiments. CB implemented the neural network architecture, performed the PLASIM simulation and trained the WGAN. The analysis of the results has been made by the authors.

Competing interests. The authors declare that they have no conflict of interest.

Disclaimer. Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Acknowledgements. This research paper was written during a thesis in partnership with Total. We would like to thank Philippe Berthet, Anahita Abadpour, Daniel Busby and Tatiana Chugunova for their support in the application of our method in different fields of the geosciences. We would like to thank Rabeb Selmi for her help and for sharing her expertise.

Review statement. This paper was edited by Takemasa Miyoshi and reviewed by two anonymous referees.

References

Arjovsky, M., Chintala, S., and Bottou, L.: Wasserstein gan, arXiv [preprint], arXiv:1701.07875, 26 January 2017.

Besombes, C.: Producing realistic climate data with GANs, Zenodo [data set], https://doi.org/10.5281/zenodo.4442450, 2021 (data available at: https://github.com/Cam-B04/Producing-realistic-climate-data-with-GANs.git, last access: January 2021).

Beusch, L., Gudmundsson, L., and Seneviratne, S. I.: Emulating Earth system model temperatures with MESMER: from global mean temperature trajectories to grid-point-level realizations on land, Earth Syst. Dynam., 11, 139–159, https://doi.org/10.5194/esd-11-139-2020, 2020.

Boukabara, S.-A., Krasnopolsky, V., Stewart, J. Q., Maddy, E. S., Shahroudi, N., and Hoffman, R. N.: Leveraging modern artificial intelligence for remote sensing and NWP: Benefits and challenges, B. Am. Meteorol. Soc., 100, ES473–ES491, 2019.

Chan, S. and Elsheikh, A. H.: Parametric generation of conditional geological realizations using generative neural networks, Computat. Geosci., 23, 925–952, https://doi.org/10.1007/s10596-019-09850-7, 2019.

Davies, T.: Lateral boundary conditions for limited area models, Q. J. Roy. Meteor. Soc., 140, 185–196, 2014.

Dramsch, J. S.: 70 years of machine learning in geoscience in review, Adv. Geophys., 61, 1–55, 2020.

Fraedrich, K., Jansen, H., Kirk, E., Luksch, U., and Lunkeit, F.: The Planet Simulator: Towards a user friendly model, Meteorol. Z., 14, 299–304, 2005a.

Fraedrich, K., Kirk, E., Lunkeit, F., Luksch, U., and Lunkeit, F.: The portable university model of the atmosphere (PUMA): Storm track dynamics and low-frequency variability, Meteorol. Z, 14, 735–746, 2005b.

Gagne, D. J., Christensen, H. M., Subramanian, A. C., and Monahan, A. H.: Machine learning for stochastic parameterization: Generative adversarial networks in the Lorenz'96 model, J. Adv. Model. Earth Sy., 12, e2019MS001896, https://doi.org/10.1029/2019MS001896, 2020.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y.: Generative adversarial networks, Communications of the ACM, 63, 139–144, 2020.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C.: Improved training of wasserstein gans, in: Advances in neural information processing systems, arXiv [preprint], 5767–5777, arXiv:1704.00028v3, 2017.

He, K., Zhang, X., Ren, S., and Sun, J.: Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, CVPR 2016, 770–778, 2016.

Hergenrother, E., Bleile, A., Middleton, D., and Trembilski, A.: The abalone interpolation: A visual interpolation procedure for the calculation of cloud movement, in: Proceedings. XV Brazilian Symposium on Computer Graphics and Image Processing, Fortaleza, Brazil, 10 October 2002, 381–387, IEEE, 2002.

Houtekamer, P. L. and Zhang, F.: Review of the ensemble Kalman filter for atmospheric data assimilation, Mon. Weather Rev., 144, 4489–4532, 2016.

Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A.: Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, Honolulu, HI, USA, 21–26 July 2017, 1125–1134, 2017.

Kantorovich, L. V. and Rubinshtein, S.: On a space of totally additive functions, Vestnik of the St. Petersburg University: Mathematics, 13, 52–59, 1958.

Kingma, D. P. and Ba, J.: Adam: A Method for Stochastic Optimization, arXiv [preprint], arXiv:1412.6980, 22 December 2014.

Lagerquist, R., McGovern, A., and Gagne II, D. J.: Deep learning for spatially explicit prediction of synoptic-scale fronts, Weather Forecast., 34, 1137–1160, 2019.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., and Shi, W.: Photo-realistic single image super-resolution using a generative adversarial network, in: Proceedings of the IEEE conference

on computer vision and pattern recognition, Honolulu, HI, USA, 21–26 July 2017, 4681–4690, 2017.

Leinonen, J., Guillaume, A., and Yuan, T.: Reconstruction of cloud vertical structure with a generative adversarial network, Geophys. Res. Lett., 46, 7035–7044, 2019.

Li, J. and Heap, A. D.: Spatial interpolation methods applied in the environmental sciences: A review, Environ. Modell. Softw., 53, 173–189, 2014.

Lorenc, A. C.: The potential of the ensemble Kalman filter for NWP–a comparison with 4D-Var, Q. J. Roy. Meteor. Soc., 129, 3183–3203, 2003.

Nagarajan, V. and Kolter, J. Z.: Gradient descent GAN optimization is locally stable, arXiv [preprint], arXiv:1706.04156, 13 June 2017.

Pannekoucke, O., Cebron, P., Oger, N., and Arbogast, P.: From the Kalman Filter to the Particle Filter: A geometrical perspective of the curse of dimensionality, Adv. Meteorol., 2016, 9372786, https://doi.org/10.1155/2016/9372786, 2016.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E.: Scikit-learn: Machine Learning in Python, J. Mach. Learn. Res., 12, 2825–2830, 2011.

Peleg, N., Fatichi, S., Paschalis, A., Molnar, P., and Burlando, P.: An advanced stochastic weather generator for simulating 2-D high-resolution climate variables, J. Adv. Model. Earth Sy., 9, 1595–1627, 2017.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., and Prabhat: Deep learning and process understanding for data-driven Earth system science, Nature, 566, 195–204, 2019.

Requena-Mesa, C., Reichstein, M., Mahecha, M., Kraft, B., and Denzler, J.: Predicting landscapes as seen from space from environmental conditions, in: IGARSS 2018 – 2018 IEEE International Geoscience and Remote Sensing Symposium, Valencia, Spain, 22–27 July 2018, 1768–1771, IEEE, 2018.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Li, F.-F.: Imagenet large scale visual recognition challenge, Int. J. Comput. Vision, 115, 211–252, 2015.

Scher, S.: Toward data-driven weather and climate forecasting: Approximating a simple general circulation model with deep learning, Geophys. Res. Lett., 45, 12616–12622, 2018.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M.: Striving for simplicity: The all convolutional net, arXiv [preprint], arXiv:1412.6806, 21 December 2014.

Vallis, G. K.: Atmospheric and Oceanic Fluid Dynamics, Cambridge University Press, Cambridge, UK, https://doi.org/10.2277/0521849691, 2006.

Watson-Parris, D.: Machine learning for weather and climate are worlds apart, Philos. T. R. Soc. A, 379, 20200098, https://doi.org/10.1098/rsta.2020.0098, 2021.

Weyn, J. A., Durran, D. R., and Caruana, R.: Can machines learn to predict weather? Using deep learning to predict gridded 500-hPa geopotential height from historical weather data, J. Adv. Model. Earth Sy., 11, 2680–2693, 2019.

Weyn, J. A., Durran, D. R., and Caruana, R.: Improving Data-Driven Global Weather Prediction Using Deep Convolutional Neural Networks on a Cubed Sphere, J. Adv. Model. Earth Sy., 12, e2020MS002109, https://doi.org/10.1029/2020MS002109, 2020.

White, T.: Sampling Generative Networks, arXiv [preprint], arXiv:1609.04468, 14 September 2016.

Wilks, D. S. and Wilby, R. L.: The weather generation game: a review of stochastic weather models, Prog. Phys. Geog., 23, 329–357, 1999.

Wu, J.-L., Kashinath, K., Albert, A., Chirila, D., Prabhat, and Xiao, H.: Enforcing statistical constraints in generative adversarial networks for modeling chaotic dynamical systems, J. Comput. Phys., 406, 109209, https://doi.org/10.1016/j.jcp.2019.109209, 2020.

Yeh, R. A., Chen, C., Yian Lim, T., Schwing, A. G., Hasegawa-Johnson, M., and Do, M. N.: Semantic image inpainting with deep generative models, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017, 5485–5493, 2017.

Zhang, X.-C., Chen, J., Garbrecht, J., and Brissette, F.: Evaluation of a weather generator-based method for statistically downscaling non-stationary climate scenarios for impact assessment at a point scale, T. ASABE, 55, 1745–1756, 2012.

6.3 Conclusion

This chapter demonstrates the use of GANs for generating realistic climate data. Several metrics and tests were used to assess the quality of the generated states. It is a first step toward a way to increase ensemble size in NWP. Such a generator seems promising for other applications such as in risk assessment by being able to reproduce the climate distribution for agricultural simulation that needs a generator able to generate rapidly realistic atmospheric fields. The future study should focus on developing more metrics to compare other GANs architectures and the use of different hyperparameters. Defining a framework to assess the quality of deep learning models is an efficient way to stimulate competition and resulting scientific advances. Analysis of the use of the generated atmospheric state in AGCM simulation as the initial state to assess the balancing quality of the generations could be insightful. The *a priori* conditioning of the generation is also an interesting aspect of GANs but requires important modification in the architecture.

However, *a posteriori* conditioning and latent space exploration is already showing promising properties. The ability to interpolate between different climate states and consequently modify the position of meteorological objects might find application in different data assimilation frameworks. As a complementary study, the next chapter will describe two different ways to perform *a posteriori* conditioning with a GANs such as the one developed in the present work. It will be applied for numerical reservoirs for sake of visualization but it is transferable to any application domain using a similar GAN.

Chapter 7

Posterior sampling in WGAN latent space

One of the main advantages of the WGAN parameterization is to be able to generate constrained images, by using an optimization method to identify the latent space areas that suit the imposed constraint. A posteriori sampling is not the only method to generate constrained images with GANs : conditional GAN is a very active research field. The conditional GAN is trained directly to generate a dataset distribution with a constraint in its input. However, it usually requires a labeled dataset, which is not always available, especially since unsupervised learning is one of the main assets of the GAN framework. For this reason, constrained generation in our case is tackled with *a posteriori* sampling. It is easier to use across different application domains and has a low computational cost. It could have a direct application in data assimilation to search efficiently for realistic generations. In this chapter, the example will be in the reservoir domain, for sake of simplicity of visualization, and will demonstrate the possibilities of posterior sampling using GAN parameterization for data assimilation application. A test case is set up to show the efficiency of optimization in the latent space with different methods, such as derivative-free optimizers and the inference network.

The current chapter aims at underlining an interesting property brought by the GAN framework. This property is described as one of the future directions of this thesis work. Consequently, this chapter will give two preliminary studies to convince the reader of the ability of the GAN methods to generate conditioned samples which could be an important way to reduce the control space when performing data assimilation. Conditioning could also be useful when GAN is simply used for its encoding task.

7.1 Derivative-free methods

The GAN function is differentiable by construction and continuous on a bounded domain. Inversion by optimization of the GAN function is then possible to generate samples that are conditioned by an imposed constraint. Knowing the different limitations of numerical weather data assimilation methods like uncertainty quantification due to a low number of ensemble members, some applications are made available by *a posteriori* sampling of GANs such as ensemble augmentation. The first optimization that could be thought of is gradient descent (GD) due to the availability of its gradient. However, GD is an efficient method for convex or quasi-convex functions. Tests were done using this method and were not conclusive. The reason for the non conclusiveness is the lack of an exploration step in the algorithm of the gradient descent. Gradient computation is not suitable for highly non-linear functions because of the very local information brought by gradient computation.

Volz et al. [109] showed that derivative-free optimization methods such as Covariance Matrix Adaptation Evolution Strategy (CMA-ES) could be applied for *a posteriori* conditioning of GANs trained to

generate levels of the Mario video game. By analogy this method applied to condition the generation of subsurface reservoir models.

7.1.1 Covariance Matrix Adaptation Evolution Strategy (CMA-ES)

The CMA-ES algorithm is part of the evolutionary algorithms used for derivative-free numerical optimization. It is inspired from biological evolution, in the sense that each candidate solution is deduced from its parental individuals. It relies on the concept of recombination and mutation : at each iteration (also called generation), the candidates that best minimize the objective (or fitness) function are chosen as parents and modified to create the next generation. This next generation is sampled from a multivariate normal distribution.

Algorithm 2 [48] describes the different steps for $(\mu/\mu_w, \lambda)$-CMA-ES. A weighted combination of the μ best candidates among the λ new candidates of the current generation are used to update the statistical moments of the distribution that rules the candidate generation. First, λ candidates, noted $x_i \in \mathbb{R}^n$ with $i = 1, \ldots, \lambda$ are sampled from the distribution $\mathcal{N}(m, \sigma^2 C)$, where m is the best estimation of the solution so far. The variance corresponds to a perturbation (or mutation) following the covariance C, that is initialized as the identity matrix for the first iteration? σ is the step-size of the perturbation. These candidates are then evaluated on the objective function $\mathcal{L}$, and sorted in the increasing order of their objective function value (line 7 of Algo. 2). Then (line 9), a new mean value $m\check{r}k$ is computed from the μ best candidates :

$$m_k = \sum_{i=1}^{\mu} w_i x_i \tag{7.1}$$

where k is the index of the current generation. The weights w_i are constrained such that $\sum_{i=1}^{\mu} w_i = 1$ and $w_1 \geq \ldots \geq w_\mu$ and :

$$\mu_w = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \approx \lambda/4 \tag{7.2}$$

the choice of constant weights or decreasing weights only changes the speed of convergence. The conjugate evolution path p_σ is then updated according to :

$$p_{\sigma;k+1} = (1 - c_\sigma) p_{\sigma;k} + \sqrt{1 - (1 - c_\sigma)^2} \sqrt{\mu_w} C_k^{-1/2} \frac{m_{k+1} - m_k}{\sigma_k} \tag{7.3}$$

where $c_\sigma^{-1} \approx n/3$ and represents the learning rate for the cumulation for the step size control. The step size is then updated following :

$$\sigma_{k+1} = \sigma_k \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{||p_\sigma||}{E||\mathcal{N}(0, I)||} - 1\right)\right) \tag{7.4}$$

where d_σ is the damping factor for the step size update. $E||\mathcal{N}(0, I)||$ is the 2-norm of the candidates and is approximately equal to $\sqrt{n}$. Finally, the evolution path of the covariance and the covariance are updated :

$$p_{c;k+1} = (1 - c_c) p_c + \mathbf{1}_{[0, \alpha\sqrt{n}]}(||p_\sigma||) \sqrt{1 - (1 - c_\sigma^2)} \sqrt{\mu_w} \frac{m_{k+1} - m_k}{\sigma_k} \tag{7.5}$$

$$C_{k+1} = (1 - c_1 - c_\mu + c_s)C_k + c_1 p_c p_c^T + c_\mu \sum_{i=1}^{\mu} w_i \frac{x_i - m_k}{\sigma_k} \left(\frac{x_i - m_k}{\sigma_k} \right)^T \tag{7.6}$$

where $c_c^{-1} \approx n/4$ is the backward time horizon for the evolution path p_c, $c_1 \approx 2/n^2$ is the learning rate for cumulation for the rank-one update of the covariance matrix, $c_s = 1 - \mathbf{1}_{[0,\alpha]}(||p_\sigma||)^2 c_1 c_c(2 - c_c)$, $c_\mu \approx \mu_w/n^2$ is the learning rate for the rank-μ update of the covariance matrix and $\alpha = 1.5$. The value of the parameter is set to be effective in most of the cases. To go deeper in the meaning of the parameters value such as learning rates for example the reader can see [47–49].

Algorithm 2 CMA-ES algorithm.

Require: λ number of samples per iteration.
Initialize:
    m, σ, $C = I$, p_σ, p_c
while number of function execution $\leq$ budget **do**
    for i in $\{1, \ldots, \lambda\}$ **do**
        $x_i \sim \mathcal{N}(m, \sigma^2 C)$
        $L_i = L(x_i)$
    end for
    $x_{\{i,\ldots,\lambda\}} = x_{s(1),\ldots,s(\lambda)}$ with $s(i) = argsort(L_{1,\ldots,\lambda})$
    $m' = m$
    $m = \text{update\_m}(x_1, \ldots, x_\lambda)$
    $p_\sigma = \text{update\_ps}(p_\sigma, \sigma^{-1} C^{-1/2}(m - m'))$
    $p_c = \text{update\_pc}(p_c, \sigma^{-1}(m - m'), ||p_\sigma||)$
    $C = \text{update\_C}(C, p_c, (x_1 - m')/\sigma, \ldots, x_\lambda - m'/\sigma)$
    $\sigma = \text{update\_}\sigma(\sigma, ||p_\sigma||)$
end while

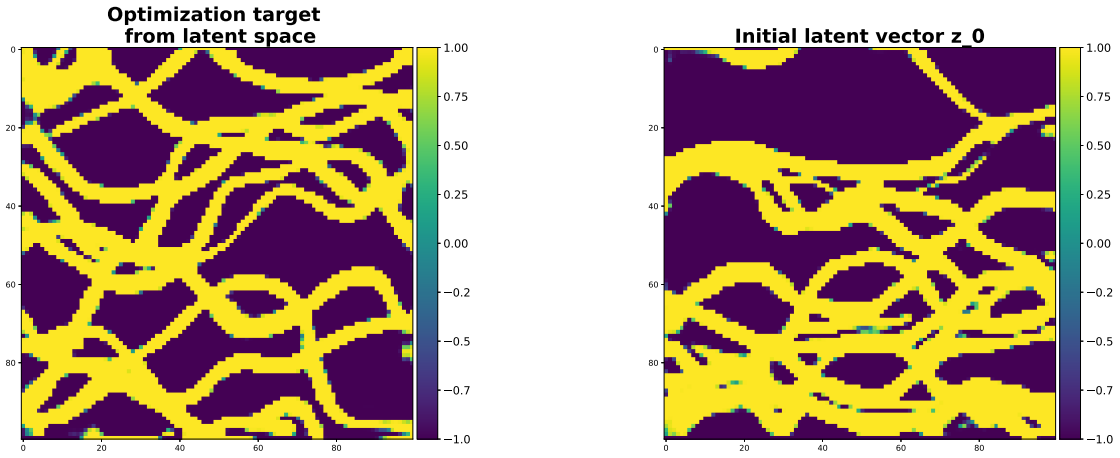
7.1.2 Test cases

The first test case aims at showing the ability to retrieve a latent vector in the latent space corresponding to a particular physical state, here a particular reservoir topology. In order to be able to monitor the distance of the solution compared to the target, the target is taken directly from a vector in the latent space. In this way it is assured that the reservoir topology exists in the latent space, the second step would be to retrieve a reservoir topology taken from the dataset instead of one generated by the GANs. This would necessitate the assurance of the existence in the latent space of the target reservoir topology in the latent space which is another active research field not tackled in the current work.

Let us starts by generating one sample from the GAN from a random vector in the latent space, called the target z_{true}. Then, another latent vector as the initial point of the optimization z_0, visible in Fig. 7.1a. The objective is to retrieve z_{true} starting from z_0 with an optimization algorithm. The loss function to minimize by the CMA-ES algorithm implementation from the package *nevergrad* is defined as the mean square loss over the image :

$$\mathcal{L} = \sum_{i,j=0}^{N_x, N_y} \left[G(z_{true})_{i,j} - G(z_{pred})_{i,j} \right]^2 \tag{7.7}$$

where N_x and N_y are the size of the image, and l is the optimization iteration index.



(a) Image corresponding to $G(z_{true})$.

(b) Image corresponding to $G(z_0)$.

Figure 7.1 – Target (left) and initial (right) reservoir models for CMA-ES optimization.

The experiment is done with a budget of 30000 executions of the GAN function by the optimization algorithm. Figure 7.2 shows the result of the optimization method. Figures 7.3a and 7.3b show that the loss of the prediction is low and converges asymptotically to 0, the distance of the predicted latent vector from z_{true} as well. It can be concluded that the real latent vector is retrieved.

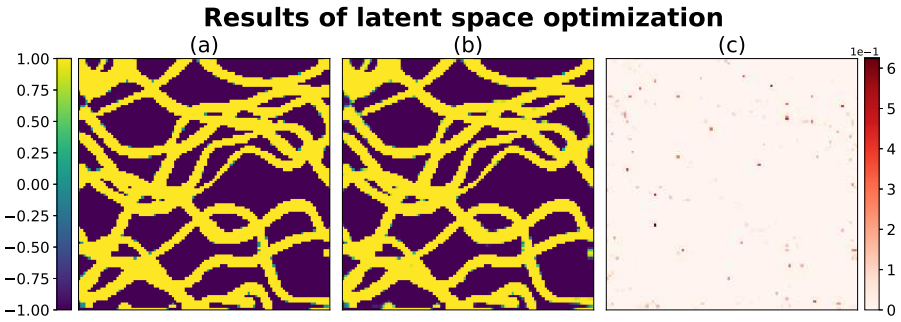
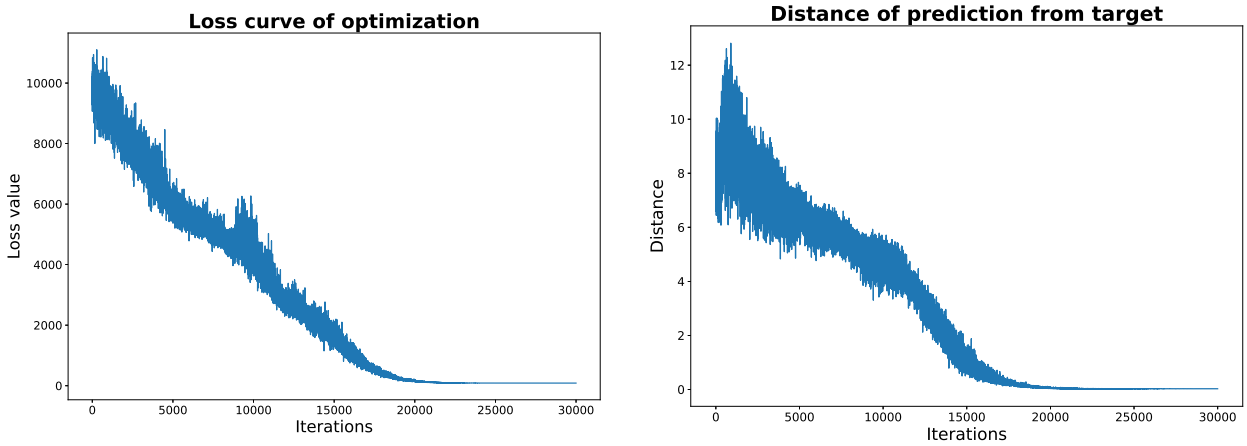


Figure 7.2 – Comparison between z_{true} (a), z_{pred} (b) and quadratic error (c).

This test case is among the most difficult ones concerning *a posteriori* sampling because the conditioning is imposed on the complete image. Usually *a posteriori* sampling, applied to data assimilation, consists in looking for images that have a defined property. As an example one could search for realizations that only have the same top right corner of the image which is a simpler task due to the multiple solutions in the latent space. Or in a real data assimilation case, one could use observations gathered by already drilled wells to know which facies is at specific positions in the image. Then, perform data assimilation on the subset of all the generations that match observation which is easier due to the reduced control space. The last example is not realizable with derivative free methods such as CMA-ES because it does not output the distribution of the latent space you should use in order to stay in this subset. However, another method of *a posteriori* sampling should be explored for this kind of application that is called Inference Network.



(a) Loss function with respect to optimization iterations.

(b) Distance of z_{pred} from z_{true} with respect to optimization iterations.

Figure 7.3 – Loss curve and distance of z_{pred} from z_{true} for CMA-ES optimization in GAN latent space.

7.2 Inference neural network

The idea behind Inference neural network (INN) [18] is to extend the generator by adding a small neural network at the input of the already trained generator such that $G \circ I = G_{cond}$. Suppose some measurements d_{obs} are available, the objective is to find $z*$ that maximizes its posterior probability knowing observations :

$$z* = \text{argmax}\, p(z|d_{obs}) \tag{7.8}$$

using Bayes' rule and applying negative logarithm to shape the problem as a minimization problem :

$$\begin{aligned} p(z|d_{obs}) &\propto p(d_{obs}|z)p(z) \\ -\log p(z|d_{obs}) &= -\log p(d_{obs}|z) - \log p(z) + const \end{aligned} \tag{7.9}$$

The prior $p(z)$ is the normal distribution which the generator was trained on : $p(z) \propto \exp(-\frac{1}{2}||z||^2)$. For the likelihood, the assumption of i.i.d Gaussian measurement noise yields to $p(d_{obs}|z) \propto \exp(-\frac{1}{2\sigma^2}||d(z) - d_{obs}||^2)$ with σ the variance of the measurement noise. Finally, an optimization can be used to minimize the following loss function :

$$\begin{aligned} L(z) &:= -\log p(z|d_{obs}) \\ &= ||d(z) - d_{obs}||^2 + \lambda||z||^2 \\ &= ||G(z)_{obs} - d_{obs}||^2 + \lambda||z||^2 \end{aligned} \tag{7.10}$$

where λ is equal to σ^2, the constant term was removed. Using directly an optimizer to minimize Eq. 7.10 can be computationally costly if a high number of conditioned samples need to be generated. The objective is to get a parameterization of conditioned samples to sample an important number of conditioned samples at low computational cost. Such a parameterization has a great value when performing data assimilation using a parameterization to induce additional constraints. The idea is to use another neural network at the input of the generator to perform *a posteriori* conditioning with the INN Fig. 7.4, where a small neural network was added in order to sample the Bayesian posterior $p(z|d_{obs})$ by minimizing $L(z)$ using a local optimizer.

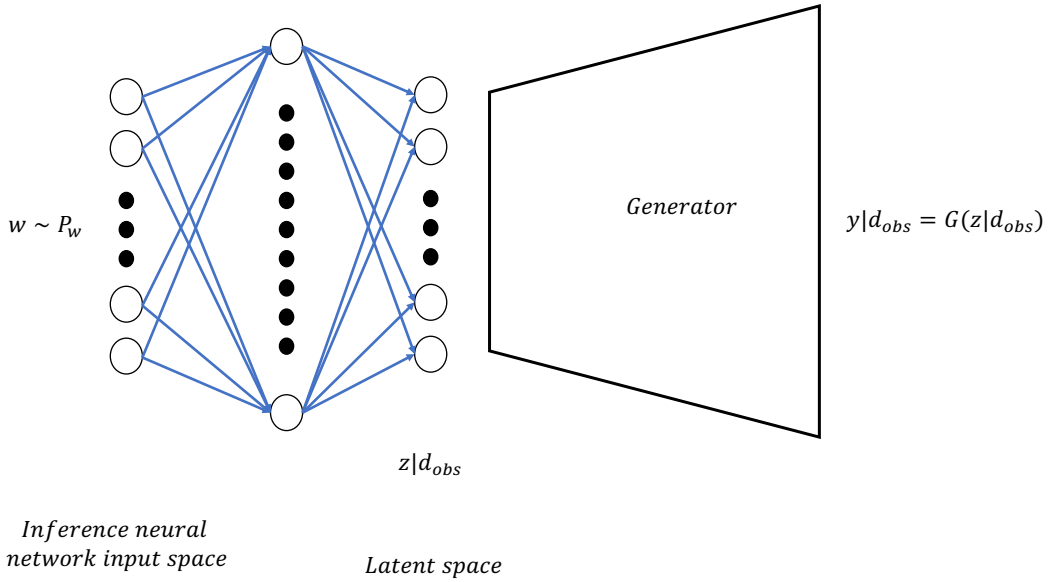


Figure 7.4 – Inference neural network framework.

The INN function can be written such as :

$$\begin{aligned} \mathcal{I}_\phi &: \mathbb{R}^{n_w} \mapsto \mathbb{R}^{n_z} \\ w \sim p_w &\mapsto z \sim q_\phi \end{aligned} \tag{7.11}$$

where ϕ represents the trainable weights of the INN, n_w and n_z are the size of the input space and the output space, the latter corresponds to the latent space. p_w is the distribution used to sample conditioned realization in the input space of the INN, and q_ϕ is the distribution induced by the INN which depends on its weights. The objective here is that $q_\phi = p(z|d_{obs})$. The Kullback-Leibler divergence yields :

$$\begin{aligned} D_{KL}(q_\phi \,||\, p(z|d_{obs})) &= \mathbb{E}_{z\sim q_\phi} \log \frac{q_\phi(z)}{p(z|d_{obs})} \\ &= \mathbb{E}_{z\sim q_\phi} - \log p(z|d_{obs}) + \mathbb{E}_{z\sim q_\phi} q_\phi(z) \\ &= \mathbb{E}_{z\sim q_\phi} - \log L(z) + \mathbb{E}_{z\sim q_\phi} q_\phi(z) + const \end{aligned} \tag{7.12}$$

the first term of equation Eq. 7.12 corresponds to the Eq. 7.10 induced by the distribution q_ϕ that can be written :

$$\mathbb{E}_{\sim q_\phi} L(z) \backsimeq \frac{1}{M} \sum_{i=1}^{M} L(I_\phi(w_i)) \tag{7.13}$$

by sampling M realizations $(w_1, \ldots, w_M)$ from p_w. The second term is more difficult to compute due to the intractable distribution q_ϕ. It is called the negative entropy of q_ϕ noted $H(q_\phi)$, Chan and

Elsheikh [18] use the Kozachenko-Leonenko estimator [43, 65] :

$$\hat{H}((z_1, \ldots, z_M)) = \frac{n_z}{M} \sum_{i=1}^{M} \log\ \rho(z_i) + const \tag{7.14}$$

where $\rho(z_i)$ is the distance between z_i and its k^{th} nearest neighbor. Goria et al. [43] says that a good rule of thumb is $k = \sqrt{M}$. The entropy measures how spread the samples induced by q_ϕ are, without this term nothing restrains the inference network to constantly output the same z for all $w \sim p_w$. This term controls the diversity of solutions that respect the constraint, and helps to approximate the full posterior of $p(z|d_{obs})$.

The architecture of the INN is made of 5 dense layers of 256 neurons each with batch normalization and leaky Relu activation function. The size of the input n_w is the same as the size of the latent space *i.e.*, $n_w = n_z = 32$. During training the input space of INN is sampled using the normal distribution.

7.2.1 5 SPOTS test case

One of the direct applications of this framework is to parameterize the set of plausible spatial distribution of subsurface properties with a manifold matching also the static constraint given by observations. To demonstrate the efficiency of the method, a test case was defined on the 5SPOTS case already studied in the chapter 5. The objective is to find a parameterization of plausible realizations matching the constraint of facies at well locations. Figure 7.5 shows the constraint of having a good facies imposed at the five wells locations.

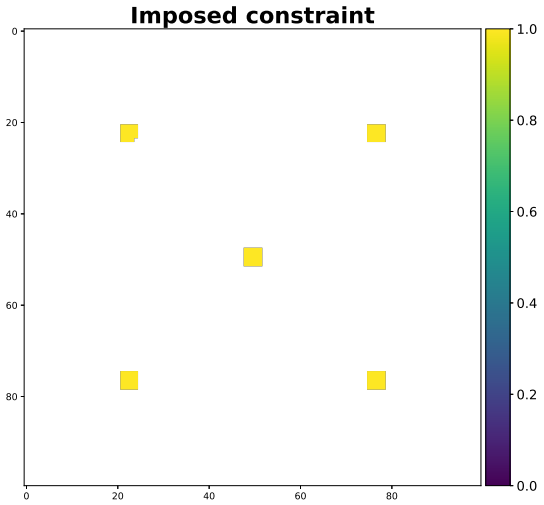


Figure 7.5 – Inference neural network framework.

Early results of the training are visible in Fig. 7.6, where a reduction of the different loss can be seen. However, Fig. 7.7 and 7.8 show a conditioning that seems to be respected but low variance and quality of the image is impacted. These results can be explained by the loss of the 2-norm of z in Fig. 7.6 it decreases to 1, which does not correspond to the normal distribution the generator was

trained with. The 2-norm of vectors from a multivariate normal distribution of dimension $n = 32$ must be around $E||\mathcal{N}(0, I^{32})|| = \sqrt{32} \approx 5.65$. q_ϕ is too far from the normal distribution and outputs latent vectors that are outside the area corresponding to realistic realizations of the latent space.

This can be corrected by modifying the weights of the different losses. It was decided to increase the weight in front of the entropy loss term from 1. to 5.. Another training was done with this change, the results are presented in Fig. 7.11 and 7.10. The quality of the image is increased and the diversity outside the constrained areas is important, Fig. 7.9 shows that the 2-norm of latent vectors draw from q_ϕ is around 5, which is corresponds to a 32-dimensional normal distribution.

The cost of the training of the INN is between 5 and 10 minutes on an Nvidia V100 16GB GPU due to the ability of the GAN to do fast inference in parallel. After training the result is a new latent space that can generate statically conditioned realizations. Finally, the parameterization can be used in a data assimilation framework by using the generator extended by the inference neural network to explore the manifold of realistic realizations statically constrained.

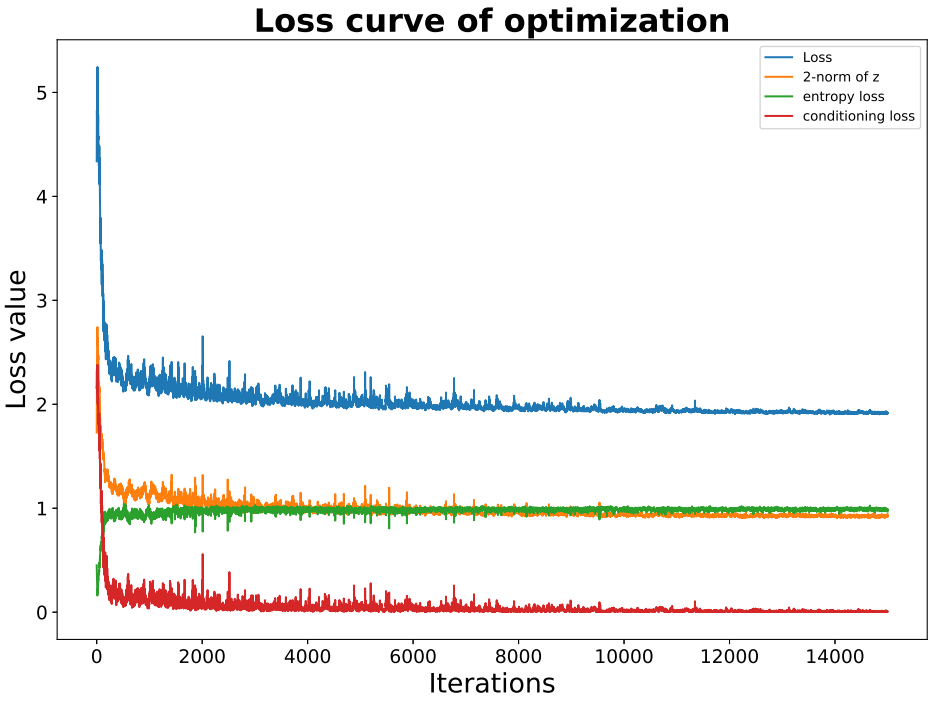


Figure 7.6 – Curve loss for INN with 5 spots constraint.

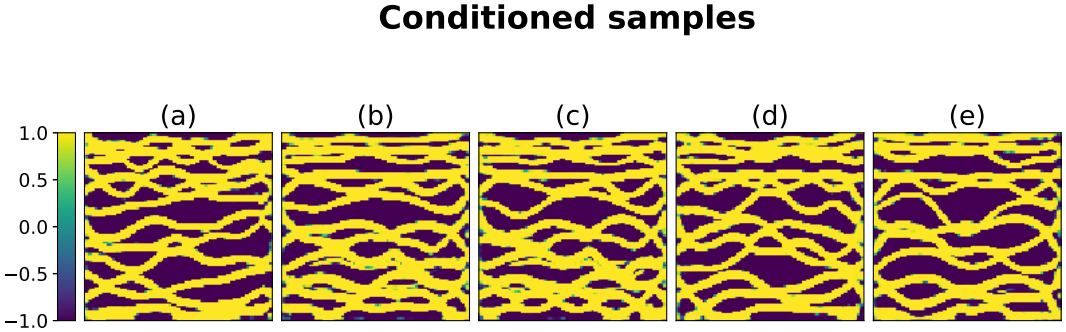


Figure 7.7 – Conditioned samples

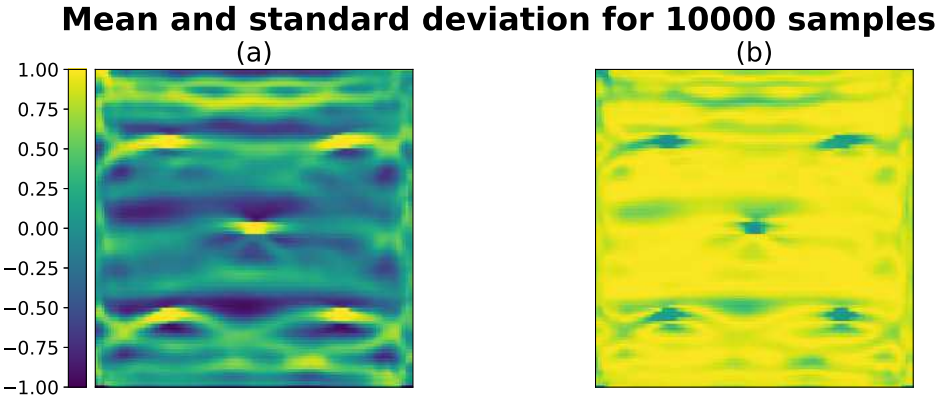


Figure 7.8 – Mean and standard deviation of 10000 conditioned samples

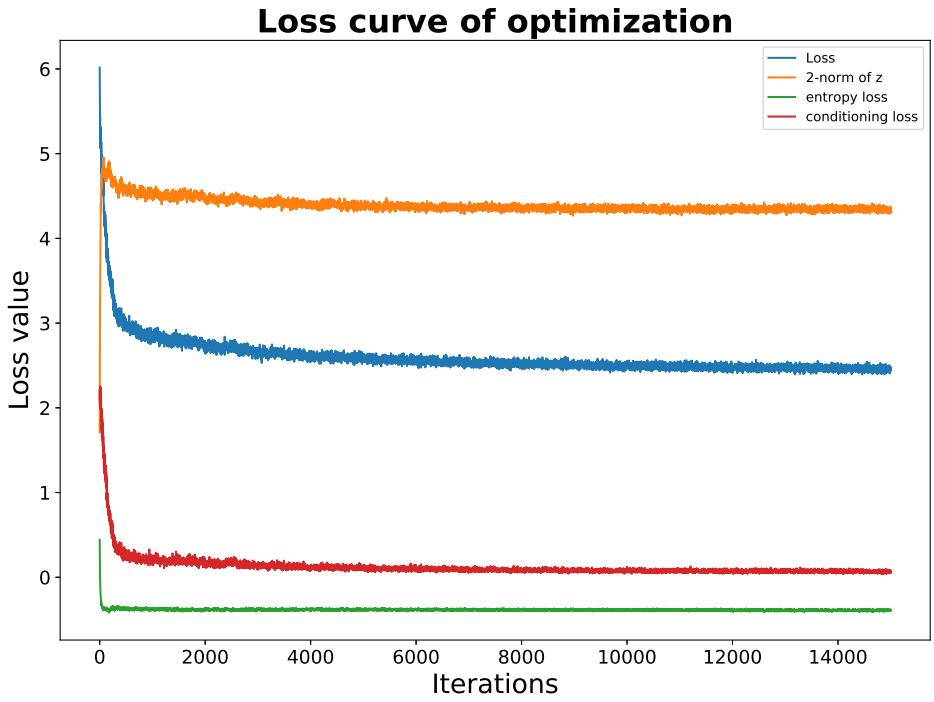


Figure 7.9 – Curve loss for INN with 5 spots constraint.

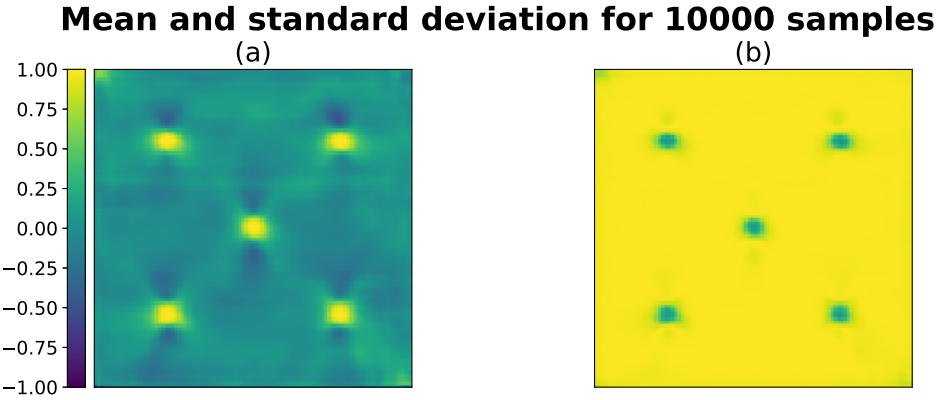


Figure 7.10 – Mean and standard deviation of 10000 conditioned samples

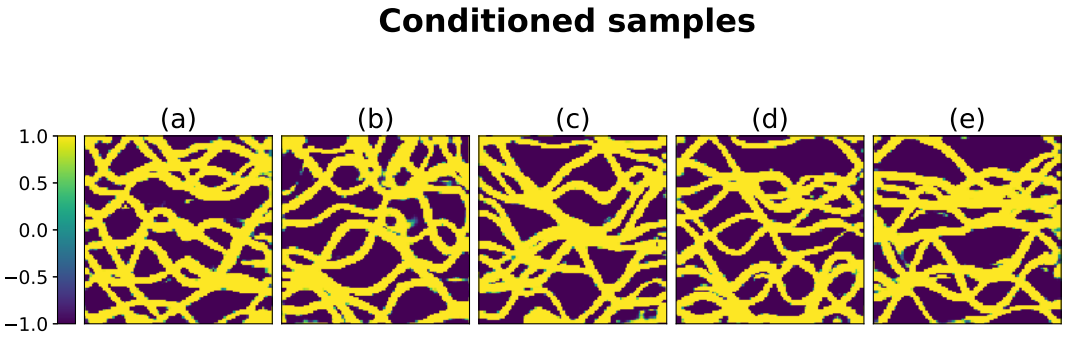


Figure 7.11 – Mean and standard deviation of 10000 conditioned samples

7.3 Discussion

These two methods constitute different ways to use the GAN a posteriori depending on the tackled problem properties. CMA-ES can be used to find realistic generations that look like a defined target. One of the applications in the context of climate could be after analysis to find the closest initialized *i.e.*, balanced atmospheric state of the analysis. The inference network is a way to introduce more constraints in the latent space at a low computational cost. The loss function can be modified to retrieve only generations with a particular facies density for example. It offers a way to reduce even more the dimension of the control space by choosing a lower dimension for INN input space for example. Lots of possibilities are accessible thanks to these two methods and these could be used in any other context than climate and hydrocarbon reservoir. These examples do not constitute a complete study but in the author opinion it is important to identify the promising properties of the GAN method. Especially for future researchers not familiar with this kind of generative networks who would like to continue to explore the possibility offered by this new data driven method in a different application domain.

Conclusion and Perspectives

Discussion

The present work gives an introduction to the two applied domains for readers not familiar with these particular applications, to identify limitations of the current methods and if our parameterization technique is applicable to each application domain. It underlines the necessity of balanced atmospheric state to avoid non-physical gravity waves in the atmospheric circulation simulation. Similarly, it also explains the limitations encountered in reservoir characterization where it is difficult to model the shape of certain geological heterogeneities.

We describe the chosen framework to solve the inverse problem of reservoir characterization. The theory of data assimilation and ensemble methods are explained to underline the similarities between the two application domains. Ensemble methods are suited for using GAN parameterization due to the assumption of Gaussian distribution of errors.

Next, the document introduces classical deep learning concepts and theoretical results to show the possibilities and limitations of neural networks notably GANs. It aims to give the reader an understanding of the principles of neural networks to show how they are now a tool on the shelf that can be used as any other mathematical method. This manuscript does not go too in-depth into the details of training general neural networks however, as the literature and set of online tools for this are numerous. The large variety of models available and their variation are mentioned, and the Wasserstein derivation of the GAN is described. In the section on limitations the rapid development and the under-exploitation of this progress will be discussed.

Our study shows a careful selection of hyper-parameters for the GAN training. The choice of certain hyper-parameters such as the networks' architecture and the different adaptations to the reservoir application domain are described. The definition of metrics for the assessment of the quality of the generations is presented. These metrics allowed the validation of the ResNet architecture compared to a classical CNN. A sufficient representation of the dataset was achieved regarding these metrics.

We show the effectiveness of the GAN parameterization coupled with the ES-MDA data assimilation algorithm for reservoir characterization for a horizontal test case and a 5 spots test case. The method achieved a satisfying match of the observation with a controllable variability allowed by the subspace inversion method. The different solutions of the ES-MDA algorithm represented realistic facies distributions (plausible geological shape of the channelized heterogeneities) thanks to the GAN parameterization.

We also demonstrate the applicability of the method for generating global balanced atmospheric states. The architectures and other hyper-parameters are presented. It proposes a data structure to avoid high memory consumption when GANs are applied to sizable data. It presents a way to enforce periodic boundary conditions to generations and metrics for quality assessment of the generation. Finally, it also presents a latent space exploration for future work with interesting properties linked to numerical weather prediction applications.

Finally, we present two different ways to enforce more constraint to the generation *a posteriori* that grant conditioning at a low computational cost. CMA-ES for optimization of the latent space, that is adapted thanks to its exploration property that allows projection of physical state in the latent space, which can be useful for balancing climate states. Inference neural network to recreate a conditioned latent space to improve optimization speed and quality.

The first objective of the current thesis was to demonstrate how generative adversarial networks could be used as a parameterization technique for ensemble-based data assimilation methods. What is more, this deep learning based parameterization method is applied to two different domains demonstrating the wide potential of the technique in an interdisciplinary context. Data assimilation is used in numerous scientific domains, which suggests that the GAN method could have many more applications, as long as there are datasets representing the constraints and the variability of the problem. We believe that our work can be useful to any application domain where physical constraints are difficult to represent mathematically, that are not Gaussian distributed, highly dimensional or lack realism when estimated, could be interested in GAN parameterization. This work is directly transferable to image-like data, but could be extended to many other data types such as audio or data on approachable meshes by leveraging the profusion of neural network architecture under active development (*e.g.*, graph neural network [110]).

This approach is in line with the work aimed at unifying separate scientific fields using very similar principles. Data assimilation is built on several decades of scientific development and is one of the domains that collects the most data. The recent development of data driven methods based on deep and machine learning could be a useful insight for the development of non-linear stochastic physical parameterization in climate study for example. These large amounts of data have to be processed efficiently and data driven are the perfect tools for it. In the other way, taking advantage of uncertain, sparse data is one of the current challenges in deep learning research whereas data assimilation experts have been using it for almost a century. Our study is a first step towards the unification of these domains. The importance of the clear definition of simple and accessible benchmark cases is one of the most efficient ways to create new innovative methods, examples are given in perspectives. Our research aims at giving an example of such test cases where promising results are demonstrated and on which future work should be built up.

Perspectives

Limitations

There are at least 2 potential limitations concerning the present study. First, the GAN remains a data driven method and relies by definition on the quality of the dataset used. The GAN parameterization method is only applicable to problems with access to large datasets, real or synthetic. One of the main properties needed in the dataset is variability which must sufficiently map that of real cases. The GAN can hardly extrapolate, meaning that for a given variable the dataset must represent in the best manner the range of possible values the variable can take.

A Second limitation is the absence of comparison with current parameterization methods of channelized reservoirs due to the lack of easily accessible parameterization methods. Current complex parameterization methods are under expensive licenses that are hard to reimplement.

Future work

Despite these limitations, this research can be seen as a first step towards integrating two lines of research, deep generative networks and data assimilation. The perspectives offered by this work are numerous. First, experts from applied domains should identify the different important metrics required to compare and improve the method. Following the same idea, the definition of precise context, parameters, case study to stimulate comparison and challenges would be a great playground for researchers of both communities. This showed very good results in the machine and deep learning community where precise metrics led the competition to important research advances. The MeteoNet initiative [46] from Meteo France, CASP competition tackling the folding protein problem where deep learning team improved significantly the results [106] and the 10 years roadmap [24] are three perfect examples of what should be done in every domain that wants to make the most out from these emerging data-driven methods.

GAN derivation and architectures

Future work should focus on the different advances made in neural network layers such as attention layers that introduce long distance covariance estimation on data samples by neural networks and improve the quality of the generations. Disentangled GAN could give more physical sense to the different latent vector components. The conditioned version of GANs could also be a great way to condition generation a priori. This implies important changes in the GAN architecture but could allow the conditioning using sea surface temperature for climate application for example. In reservoir application the implementation of multiple image channels including not only rock facies but also variable porosity and permeability fields conditioned by some latent vector components would be useful.

Increased complexity test cases

The use of GAN parameterization on more complex cases such as multiple heterogeneity types in one reservoir numerical model should be a secondary focus of the future work. Train GANs on more

complex climate data such as data from regional circulation model AROME which is currently studied at Meteo France, is of the challenge that needs to be solved to make the method applicable on real cases. Implementation of 3D cases using 3D convolutional layers to compare the coherence of the generations on the z-direction with the method used in Chap. 6 would be an interesting advance to plan the resource cost of the methods.

Conclusion et Perspectives (french)

Discussion

Ce travail introduit deux domaines d'application pour les lecteurs qui ne sont pas familiers avec ces derniers afin d'identifier les limites des méthodes actuelles et l'application de notre approche de paramétrisation à chacun d'eux. Il souligne la nécessité d'un état atmosphérique équilibré pour éviter les ondes de gravité non physiques au sein de la simulation de la circulation atmosphérique. De même, elle explique les limites rencontrées dans la caractérisation des réservoirs où il est difficile de modéliser la forme de certaines hétérogénéités géologiques.

Nous décrivons le cadre choisi pour résoudre le problème inverse de la caractérisation des réservoirs. La théorie de l'assimilation de données et les méthodes d'ensemble sont expliquées pour souligner les similitudes entre ces deux domaines d'application. Les méthodes ensemblistes sont adaptées à l'utilisation de la paramétrisation GAN en raison de l'hypothèse d'une distribution Gaussienne des erreurs.

Le document introduit également les concepts classiques d'apprentissage profond et les résultats théoriques pour montrer les possibilités et les limites des réseaux de neurones, notamment des GANs. Il vise à donner au lecteur une compréhension des principes des réseaux de neurones pour montrer comment ils sont maintenant un outil qui peut être utilisé comme toute autre méthode mathématique. Ce manuscrit n'entre cependant pas dans les détails de l'entraînement des réseaux de neurones, car la littérature et les outils en ligne sont nombreux. La grande variété de modèles disponibles et leurs variations sont mentionnées, et la dérivation de Wasserstein du GAN est décrite. Dans la section sur les limitations, le développement rapide et la sous-exploitation de ces progrès seront discutés.

Notre étude montre une sélection minutieuse des hyper-paramètres pour l'entraînement du GAN. Le choix de certains hyper-paramètres tels que l'architecture des réseaux et les différentes adaptations au domaine d'application des réservoirs sont décrits. Nous présentons la définition de métriques pour l'évaluation de la qualité des générations. Ces métriques ont permis de valider l'architecture du ResNet par rapport à un CNN classique. Une représentation suffisante du jeu de données a été obtenue en ce qui concerne ces métriques.

Nous montrons l'efficacité de la paramétrisation du GAN couplée à l'algorithme d'assimilation de données ES-MDA pour la caractérisation de réservoirs pour un cas test horizontal et un cas test à 5 puits. La méthode a permis d'obtenir une correspondance satisfaisante des observations avec une variabilité contrôlable grâce à la méthode de "subspace inversion". Les différentes solutions de l'algorithme ES-MDA ont représenté des distributions de faciès géologiques réalistes (forme géologique plausible des hétérogénéités canalisées) permise par la paramétrisation du GAN.

Nous démontrons également l'applicabilité de la méthode pour générer des états atmosphériques globaux équilibrés. Nous présentons les architectures et autres hyper-paramètres. Nous proposons une structure de données pour éviter une forte consommation de mémoire lorsque les GANs sont appliqués à des données de taille importante. Nous présentons également un moyen d'imposer des

conditions limites périodiques aux générations et des métriques pour l'évaluation de la qualité de la génération. Enfin, nous présentons une exploration de l'espace latent pour des travaux futurs avec des propriétés intéressantes liées à l'application de la prévision météorologique numérique.

Enfin, nous présentons deux façons différentes d'imposer plus de contraintes à la génération *a posteriori* qui permettent le conditionnement à un faible coût de calcul. Premièrement, l'algorithme CMA-ES pour l'optimisation de l'espace latent, qui est adapté grâce à sa propriété d'exploration et permet la projection d'un état physique dans l'espace latent. Cela peut être utile pour équilibrer les états climatiques. Ensuite l'"Inference network" pour recréer un espace latent conditionné afin d'améliorer la vitesse et la qualité de l'optimisation.

Le premier objectif de la présente thèse était de démontrer comment les réseaux adversariaux génératifs pouvaient être utilisés comme technique de paramétrisation pour les méthodes d'assimilation de données basées sur des ensembles. De plus, cette méthode basée sur l'apprentissage profond est appliquée à deux domaines différents démontrant le large potentiel de la technique dans un contexte interdisciplinaire. L'assimilation de données est utilisée dans de nombreux domaines scientifiques, ce qui suggère que la méthode GAN pourrait avoir de nombreuses autres applications, tant qu'il existe des ensembles de données représentant les contraintes et la variabilité du problème. Nous pensons que notre travail peut être utile à tout domaine d'application où les contraintes physiques sont difficiles à représenter mathématiquement. De même lorsque les paramètres ne sont pas distribués de manière Gaussienne, hautement dimensionnels ou manquant de réalisme. La paramétrisation des GANs est directement transférable à des données de type image, mais pourrait être étendue à de nombreux autres types de données comme l'audio ou les données sur des mailles en exploitant la profusion d'architecture de réseaux de neurones en développement continu (*e.g.*, graph neural network [110]).

Cette approche s'inscrit dans la lignée des travaux visant à unifier des domaines scientifiques distincts utilisant des principes similaires. L'assimilation de données repose sur plusieurs décennies de recherche scientifique et constitue l'un des domaines qui collectent le plus de données. Le développement récent de méthodes basées sur l'apprentissage profond et automatique pourrait être utile pour la création de paramétrisation physique stochastique et non linéaire utilisée par exemple dans l'étude du climat. Ces grandes quantités de données doivent être traitées efficacement. Les méthodes basées sur les données sont les outils parfaitement adaptés pour cela. D'autre part, tirer parti de données incertaines et éparses est l'un des défis actuels de la recherche sur l'apprentissage profond, alors que les experts en assimilation de données l'utilisent depuis près d'un siècle. Notre étude est un premier pas vers l'unification de ces domaines. L'importance de la définition claire de cas de référence simples et accessibles est l'un des moyens les plus efficaces pour créer de nouvelles méthodes innovantes, des exemples sont donnés dans les perspectives. Notre recherche vise à donner un exemple de tels cas de référence où des résultats prometteurs sont démontrés et sur lesquels les travaux futurs devraient être construits.

Perspectives

Limitations

Il existe au moins 2 limitations potentielles concernant la présente étude. Premièrement, le GAN reste une méthode basée sur les données et dépend par définition de la qualité du jeu de données utilisé. La méthode de paramétrisation du GAN n'est applicable qu'aux problèmes ayant accès à de grands ensembles de données, réels ou synthétiques. L'une des principales propriétés requises dans le

jeu de données est la variabilité, qui doit correspondre suffisamment à celle des cas réels. Le GAN peut difficilement extrapoler, ce qui signifie que pour une variable donnée, le jeu de données doit représenter au mieux la gamme des valeurs possibles de la variable.

Une deuxième limite est l'absence de comparaison avec les méthodes actuelles de paramétrisation des réservoirs canalisés en raison du manque de méthodes facilement accessibles. Les méthodes de paramétrisation complexes sont sous licence coûteuses et difficiles à réimplémenter.

Travaux futurs

Malgré ces limites, cette recherche peut être considérée comme un premier pas vers l'intégration de deux lignes de recherche, les réseaux génératifs profonds et l'assimilation de données. Les perspectives offertes par ce travail sont nombreuses. Tout d'abord, les experts des domaines appliqués devraient identifier les différentes métriques importantes requises pour comparer et améliorer la méthode. Suivant la même idée, la définition d'un contexte précis, de paramètres, d'une étude de cas pour stimuler la comparaison et les défis serait un grand terrain de jeu pour les chercheurs des deux communautés. Cela a donné de très bons résultats dans la communauté de l'apprentissage automatique et de l'apprentissage profond, où des métriques précises ont conduit la compétition à d'importantes avancées de la recherche. L'initiative MeteoNet [46] de Météo France, le concours CASP s'attaquant au problème du pliage des protéines où l'équipe de chercheur en apprentissage profond a considérablement amélioré les résultats [106] et la feuille de route pour les 10 prochaines années [24] sont trois exemples pertinents de ce qui devrait être fait dans chaque domaine qui souhaite tirer le meilleur parti de ces méthodes émergentes basées sur les données.

Dérivation et architectures des GAN

Les travaux futurs devraient se concentrer sur les différentes avancées réalisées dans les couches des réseaux neuronaux, telles que les couches d'attention qui introduisent l'estimation de la covariance à longue distance sur les échantillons de données par les réseaux neuronaux et améliorent la qualité des générations. Un disentangled GAN pourrait donner un sens plus physique aux différentes composantes du vecteur latent. La version conditionnée des GANs pourrait également être un excellent moyen de conditionner la génération *a priori*. Cela implique des changements importants dans l'architecture du GAN mais pourrait permettre le conditionnement en utilisant la température de surface de la mer pour une application climatique par exemple. Dans les applications de réservoir, la mise en œuvre de plusieurs canaux d'images comprenant non seulement les faciès rocheux mais aussi des champs de porosité et de perméabilité variables conditionnées par certaines composantes vectorielles latentes serait utile.

Cas d'essai de complexité accrue

L'utilisation de la paramétrisation des GANs sur des cas plus complexes tels que des types d'hétérogénéités multiples dans un modèle numérique de réservoir devrait être un objectif secondaire des travaux futurs. L'entraînement des GANs sur des données climatiques plus complexes telles que les données du modèle de circulation régionale AROME qui est actuellement étudié à Météo France, est un des défis à relever pour rendre la méthode applicable sur des cas réels. L'implémentation de cas 3D utilisant des couches convolutionnelles 3D pour comparer la cohérence des générations sur la direction z avec

la méthode utilisée dans le Chap. 6 serait une avancée intéressante pour planifier le coût en ressources des méthodes.

Appendices

List of Figures

List of Tables

List of Algorithms

Bibliography

[1] Anahita Abadpour, Moyosore Adejare, Tatiana Chugunova, Helene Mathieu, and Norman Haller. Integrated geo-modeling and ensemble history matching of complex fractured carbonate and deep offshore turbidite fields, generation of several geologically coherent solutions using ensemble methods. In Abu Dhabi International Petroleum Exhibition & Conference. OnePetro, 2018.

[2] Jonas Adler and Ozan Öktem. Deep bayesian inversion. arXiv preprint arXiv:1811.05910, 2018.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.

[4] G Burc Arpat and Jef Caers. Conditional simulation with patterns. Mathematical Geology, 39 (2):177–203, 2007.

[5] Jichao Bao, Liangping Li, and Fleford Redoloza. Coupling ensemble smoother and deep learning with generative adversarial networks to deal with non-gaussianity in flow and transport data assimilation. Journal of Hydrology, 590:125443, 2020.

[6] Judith Berner, Ulrich Achatz, Lauriane Batte, Lisa Bengtsson, Alvaro De La Camara, Hannah M Christensen, Matteo Colangeli, Danielle RB Coleman, Daan Crommelin, Stamen I Dolaptchiev, et al. Stochastic parameterization: Toward a new view of weather and climate models. Bulletin of the American Meteorological Society, 98(3):565–588, 2017.

[7] Camille Besombes, Olivier Pannekoucke, Corentin Lapeyre, Benjamin Sanderson, and Olivier Thual. Producing realistic climate data with generative adversarial networks. Nonlinear Processes in Geophysics, 28(3):347–370, 2021.

[8] Christopher M Bishop et al. Neural networks for pattern recognition. Oxford university press, 1995.

[9] Vilhelm Bjerknes. Dynamical meteorological and hidrography. New York: Carnegie Institute, Gibson Bros, 1911.

[10] Thomas Bolton and Laure Zanna. Applications of deep learning to ocean data inference and subgrid parameterization. Journal of Advances in Modeling Earth Systems, 11(1):376–399, 2019.

[11] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. In International Conference on Machine Learning, pages 537–546. PMLR, 2017.

[12] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096, 2018.

[13] Smith Arauco Canchumuni, Alexandre A Emerick, and Marco Aurelio Pacheco. Integration of ensemble data assimilation and deep learning for history matching facies models. In OTC Brasil. OnePetro, 2017.

[14] Smith WA Canchumuni, Alexandre A Emerick, and Marco Aurélio C Pacheco. History matching geological facies models based on ensemble smoother and deep generative models. Journal of Petroleum Science and Engineering, 177:941–958, 2019.

[15] Smith WA Canchumuni, Alexandre A Emerick, and Marco Aurélio C Pacheco. Towards a robust parameterization for conditioning facies models using deep variational autoencoders and ensemble smoother. Computers & Geosciences, 128:87–102, 2019.

[16] Smith WA Canchumuni, Jose DB Castro, Júlia Potratz, Alexandre A Emerick, and Marco Aurelio C Pacheco. Recent developments combining ensemble smoother and deep generative networks for facies history matching. Computational Geosciences, 25(1):433–466, 2021.

[17] Alberto Carrassi, Marc Bocquet, Laurent Bertino, and Geir Evensen. Data assimilation in the geosciences: An overview of methods, issues, and perspectives. Wiley Interdisciplinary Reviews: Climate Change, 9(5):e535, 2018.

[18] Shing Chan and Ahmed H Elsheikh. Parametric generation of conditional geological realizations using generative neural networks. Computational Geosciences, 23(5):925–952, 2019.

[19] Jules G Charney, Ragnar Fjörtoft, and John Von Neumann. Numerical integration of the barotropic vorticity equation. In The Atmosphere—A Challenge, pages 267–284. Springer, 1990.

[20] Francois Chollet. Deep learning with Python. Simon and Schuster, 2017.

[21] George Cybenko. Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 2(4):303–314, 1989.

[22] Roger Daley. Atmospheric data analysis. Number 2. Cambridge university press, 1993.

[23] Clayton Vernon Deutsch. Annealing techniques applied to reservoir modeling and the integration of geological and engineering (well test) data. PhD thesis, stanford university, 1992.

[24] Peter Dueben, Umberto Modigliani, Alan Geer, Stephan Siemen, Florian Pappenberger, Peter Bauer, Andy Brown, Martin Palkovič, Baudouin Raoult, Nils Wedi, et al. Technical memo. 2021.

[25] Peter D Dueben and Peter Bauer. Challenges and design choices for global weather and climate models based on machine learning. Geoscientific Model Development, 11(10):3999–4009, 2018.

[26] Alexandre A Emerick and Albert C Reynolds. History matching time-lapse seismic data using the ensemble kalman filter with multiple data assimilations. Computational Geosciences, 16(3): 639–659, 2012.

[27] Alexandre A Emerick and Albert C Reynolds. Ensemble smoother with multiple data assimilation. Computers & Geosciences, 55:3–15, 2013.

[28] Turgay Ertekin and Qian Sun. Artificial intelligence applications in reservoir engineering: a status check. Energies, 12(15):2897, 2019.

[29] Geir Evensen. Using the extended kalman filter with a multilayer quasi-geostrophic ocean model. Journal of Geophysical Research: Oceans, 97(C11):17905–17924, 1992.

[30] Geir Evensen. Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics. Journal of Geophysical Research: Oceans, 99 (C5):10143–10162, 1994.

[31] Geir Evensen. Sampling strategies and square root analysis schemes for the enkf. Ocean dynamics, 54(6):539–560, 2004.

[32] Geir Evensen and Peter Jan Van Leeuwen. An ensemble kalman smoother for nonlinear dynamics. Monthly Weather Review, 128(6):1852–1867, 2000.

[33] Brendan J Frey, Geoffrey E Hinton, Peter Dayan, et al. Does the wake-sleep algorithm produce good density estimators? In Advances in neural information processing systems, pages 661–670. Citeseer, 1996.

[34] Brendan J Frey, J Frey Brendan, and Brendan J Frey. Graphical models for machine learning and digital communication. MIT press, 1998.

[35] David John Gagne, Hannah M Christensen, Aneesh C Subramanian, and Adam H Monahan. Machine learning for stochastic parameterization: Generative adversarial networks in the lorenz'96 model. Journal of Advances in Modeling Earth Systems, 12(3):e2019MS001896, 2020.

[36] A Galli, H Beucher, G Le Loc'h, B Doligez, et al. The pros and cons of the truncated gaussian method. In Geostatistical simulations, pages 217–233. Springer, 1994.

[37] AJ Geer. Learning earth system models from observations: machine learning or data assimilation? Philosophical Transactions of the Royal Society A, 379(2194):20200089, 2021.

[38] Pierre Gentine, Mike Pritchard, Stephan Rasp, Gael Reinaudi, and Galen Yacalis. Could machine learning break the convection parameterization deadlock? Geophysical Research Letters, 45(11): 5742–5751, 2018.

[39] Matheron Georges. Principles of geostatistics. Economic geology, 58:1246–4, 1963.

[40] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. arXiv preprint arXiv:1701.00160, 2016.

[41] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Advances in neural information processing systems, pages 2672–2680, 2014.

[42] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. MIT press, 2016.

[43] Mohammed Nawaz Goria, Nikolai N Leonenko, Victor V Mergel, and Pier Luigi Novi Inverardi. A new class of random vector entropy estimators and its applications in testing statistical hypotheses. Journal of Nonparametric Statistics, 17(3):277–297, 2005.

[44] Felipe B Guardiano and R Mohan Srivastava. Multivariate geostatistics: beyond bivariate moments. In Geostatistics Troia'92, pages 133–144. Springer, 1993.

[45] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In Advances in neural information processing systems, pages 5767–5777, 2017.

[46] Larvor Gwennaëlle, Berthomier Léa, Chabot Vincent, Le Pape Brice, Pradel Bruno, and Perez Lior. Meteonet, an open reference weather dataset by meteo france. 2020.

[47] Nikolaus Hansen. The cma evolution strategy: a comparing review. Towards a new evolutionary computation, pages 75–102, 2006.

[48] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. Evolutionary computation, 9(2):159–195, 2001.

[49] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). Evolutionary computation, 11(1):1–18, 2003.

[50] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770–778, 2016.

[51] Von K Hinkelmann. Der mechanismus des meteorologischen lärmes. Tellus, 3(4):285–296, 1951.

[52] By A Hollingsworth, AC Lorenc, MS Tracton, K Arpe, G Cats, S Uppala, and P Kållberg. The response of numerical weather prediction systems to fgge level iib data. part i: Analyses. Quarterly Journal of the Royal Meteorological Society, 111(467):1–66, 1985.

[53] JA Holmes et al. Enhancements to the strongly coupled, fully implicit well model: wellbore crossflow modeling and collective well control. In SPE Reservoir Simulation Symposium. Society of Petroleum Engineers, 1983.

[54] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. Neural networks, 4(2):251–257, 1991.

[55] Peter L Houtekamer and Fuqing Zhang. Review of the ensemble kalman filter for atmospheric data assimilation. Monthly Weather Review, 144(12):4489–4532, 2016.

[56] P Jacquard et al. Permeability distribution from field pressure data. Society of Petroleum Engineers Journal, 5(04):281–294, 1965.

[57] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention. arXiv preprint arXiv:2103.03206, 2021.

[58] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[59] Eugenia Kalnay. Atmospheric modeling, data assimilation and predictability. Cambridge university press, 2003.

[60] Leonid Vasilevich Kantorovich and SG Rubinshtein. On a space of totally additive functions. Vestnik of the St. Petersburg University: Mathematics, 13(7):52–59, 1958.

[61] Jeffrey D Kepert. On ensemble representation of the observation-error covariance in the ensemble kalman filter. Ocean Dynamics, 54(6):561–569, 2004.

[62] Jack Kiefer and Jacob Wolfowitz. Stochastic estimation of the maximum of a regression function. The Annals of Mathematical Statistics, pages 462–466, 1952.

[63] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. ArXiv, 2014.

[64] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[65] LF Kozachenko and Nikolai N Leonenko. Sample estimate of the entropy of a random vector. Problemy Peredachi Informatsii, 23(2):9–16, 1987.

[66] Vladimir M Krasnopolsky, Michael S Fox-Rabinovitz, and Dmitry V Chalikov. New approach to calculation of atmospheric model physics: Accurate and fast neural network emulation of longwave radiation in a climate model. Monthly Weather Review, 133(5):1370–1383, 2005.

[67] Eric Laloy, Romain Hérault, John Lee, Diederik Jacques, and Niklas Linde. Inversion using a new low-dimensional representation of complex binary geological media based on a deep neural network. Advances in Water Resources, 110:387–405, 2017.

[68] Eric Laloy, Romain Hérault, Diederik Jacques, and Niklas Linde. Training-image based geostatistical inversion using a spatial generative adversarial neural network. Water Resources Research, 54(1):381–406, 2018.

[69] Yann Le Cun, Lionel D Jackel, Brian Boser, John S Denker, Henry P Graf, Isabelle Guyon, Don Henderson, Richard E Howard, and William Hubbard. Handwritten digit recognition: Applications of neural network chips and automatic learning. IEEE Communications Magazine, 27(11):41–46, 1989.

[70] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. Efficient backprop. In Neural networks: Tricks of the trade, pages 9–48. Springer, 2012.

[71] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural networks, 6(6):861–867, 1993.

[72] Yujia Li, Kevin Swersky, and Rich Zemel. Generative moment matching networks. In International Conference on Machine Learning, pages 1718–1727. PMLR, 2015.

[73] Andrew C Lorenc. Modelling of error covariances by 4d-var data assimilation. Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography, 129(595):3167–3182, 2003.

[74] Edward N Lorenz. Deterministic nonperiodic flow. Journal of atmospheric sciences, 20(2): 130–141, 1963.

[75] Edward N Lorenz. A study of the predictability of a 28-variable atmospheric model. Tellus, 17 (3):321–333, 1965.

[76] Edward N Lorenz. The predictability of a flow which possesses many scales of motion. Tellus, 21(3):289–307, 1969.

[77] EN Lorenz. (1963a). the predictability of hydrodynamic flow. Trans. New York Acad. Sci., II, 25:409–452, 1963.

[78] C Manwart, S Torquato, and R Hilfer. Stochastic reconstruction of sandstones. Physical Review E, 62(1):893, 2000.

[79] G Matheron, H Beucher, Ch De Fouquet, A Galli, D Guerillot, Ch Ravenne, et al. Conditional simulation of the geometry of fluvio-deltaic reservoirs. In Spe annual technical conference and exhibition. Society of Petroleum Engineers, 1987.

[80] Robert N Miller, Everett F Carter, and Sally T Blue. Data assimilation into nonlinear stochastic models. Tellus A: Dynamic Meteorology and Oceanography, 51(2):167–194, 1999.

[81] K Miyakoda and RW Moyer. A method of initialization for dynamical weather forecasting. Tellus, 20(1):115–128, 1968.

[82] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. arXiv preprint arXiv:1802.05957, 2018.

[83] Lukas Mosser, Olivier Dubrule, and Martin J Blunt. Deepflow: history matching in the space of deep generative models. arXiv preprint arXiv:1905.05749, 2019.

[84] Lukas Mosser, Olivier Dubrule, and Martin J Blunt. Stochastic seismic waveform inversion using generative adversarial networks as a geological prior. Mathematical Geosciences, 52(1):53–79, 2020.

[85] Dean S Oliver and Yan Chen. Recent progress on reservoir history matching: a review. Computational Geosciences, 15(1):185–221, 2011.

[86] RG Owens and TD Hewson. Ecmwf forecast user guide. Reading: ECMWF, 10:m1cs7h, 2018.

[87] Dhruv Patel and Assad A Oberai. Bayesian inference with generative adversarial network priors. arXiv preprint arXiv:1907.09987, 2019.

[88] Nathanaël Perraudin, Michaël Defferrard, Tomasz Kacprzak, and Raphael Sgier. Deepsphere: Efficient spherical convolutional neural network with healpix sampling for cosmological applications. Astronomy and Computing, 27:130–146, 2019.

[89] NA Phillips. Principles of large scale numerical weather prediction. In Dynamic meteorology, pages 1–96. Springer, 1973.

[90] Allan Pinkus. Approximation theory of the mlp model in neural networks. Acta numerica, 8: 143–195, 1999.

[91] D Rahon, PF Edoa, and M Masmoudi. Inversion of geological shapes in reservoir engineering using well-tests and history matching of production data. In SPE Annual Technical Conference and Exhibition. OnePetro, 1997.

[92] Atgeirr Flø Rasmussen, Tor Harald Sandve, Kai Bao, Andreas Lauser, Joakim Hove, Bård Skaflestad, Robert Klöfkorn, Markus Blatt, Alf Birger Rustad, Ove Sævareid, et al. The open porous media flow reservoir simulator. Computers & Mathematics with Applications, 81:159–185, 2021.

[93] Stephan Rasp, Michael S Pritchard, and Pierre Gentine. Deep learning to represent subgrid processes in climate models. Proceedings of the National Academy of Sciences, 115(39):9684–9689, 2018.

[94] JH Rick Chang, Chun-Liang Li, Barnabas Poczos, BVK Vijaya Kumar, and Aswin C Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection models. In Proceedings of the IEEE International Conference on Computer Vision, pages 5888–5897, 2017.

[95] Herbert Robbins and Sutton Monro. A stochastic approximation method. The annals of mathematical statistics, pages 400–407, 1951.

[96] José Roberto P Rodrigues. Calculating derivatives for automatic history matching. Computational Geosciences, 10(1):119–136, 2006.

[97] Fabien Roquet, Jean-Benoit Charrassin, Stephane Marchand, Lars Boehme, Mike Fedak, Gilles Reverdin, and Christophe Guinet. Delayed-mode calibration of hydrographic data obtained from animal-borne satellite relay data loggers. Journal of Atmospheric and Oceanic Technology, 28 (6):787–801, 2011.

[98] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. nature, 323(6088):533–536, 1986.

[99] Shah Shah, GR Gavalas, JH Seinfeld, et al. Error analysis in history matching: The optimum level of parameterization. Society of Petroleum Engineers Journal, 18(03):219–228, 1978.

[100] GJ Shutts and TN Palmer. Convective forcing fluctuations in a cloud-resolving model: Relevance to the stochastic parameterization problem. Journal of climate, 20(2):187–202, 2007.

[101] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806, 2014.

[102] Sebastien B Strebelle and Andre G Journel. Reservoir modeling using multiple-point statistics. In SPE Annual Technical Conference and Exhibition. OnePetro, 2001.

[103] Pejman Tahmasebi. Structural adjustment for accurate conditioning in large-scale subsurface systems. Advances in Water Resources, 101:60–74, 2017.

[104] Pejman Tahmasebi. Multiple point statistics: a review. Handbook of mathematical geosciences, pages 613–643, 2018.

[105] Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural networks for machine learning, 4(2):26–31, 2012.

[106] Kathryn Tunyasuvunakool, Jonas Adler, Zachary Wu, Tim Green, Michal Zielinski, Augustin Žídek, Alex Bridgland, Andrew Cowie, Clemens Meyer, Agata Laydon, Sameer Velankar, Gerard J. Kleywegt, Alex Bateman, Richard Evans, Alexander Pritzel, Michael Figurnov, Olaf Ronneberger, Russ Bates, Simon A. A. Kohl, Anna Potapenko, Andrew J. Ballard, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Ellen Clancy, David Reiman, Stig Petersen, Andrew W. Senior, Koray Kavukcuoglu, Ewan Birney, Pushmeet Kohli, John Jumper, and Demis Hassabis. Highly accurate protein structure prediction for the human proteome. Nature, 596(7873):590–596, 2021. doi: 10.1038/s41586-021-03828-1. URL https://doi.org/10.1038/s41586-021-03828-1.

[107] Peter Jan Van Leeuwen and Geir Evensen. Data assimilation and inverse methods in terms of a probabilistic formulation. Monthly Weather Review, 124(12):2898–2913, 1996.

[108] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017.

[109] Vanessa Volz, Jacob Schrum, Jialin Liu, Simon M Lucas, Adam Smith, and Sebastian Risi. Evolving mario levels in the latent space of a deep convolutional generative adversarial network. In Proceedings of the genetic and evolutionary computation conference, pages 221–228, 2018.

[110] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, et al. Deep graph library: A graph-centric, highly-performant package for graph neural networks. arXiv preprint arXiv:1909.01315, 2019.

[111] CLY Yeong and Salvatore Torquato. Reconstructing random media. Physical review E, 57(1): 495, 1998.

[112] Janni Yuval, Chris N Hill, and Paul A O'Gorman. Use of neural networks for stable, accurate and physically consistent parameterization of subgrid atmospheric processes with good performance at reduced precision. arXiv preprint arXiv:2010.09947, 2020.

[113] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In International conference on machine learning, pages 7354–7363. PMLR, 2019.