Imperial College London Department of Computing

Encoder-Decoder to reduce space of Background Covariance Matrix for 3D-Variational Data Assimilation

Progress Report: June 2019

by
Julian Mack

Submitted in partial fulfilment of the requirements for the MSc Degree in Computing (Machine Learning) of Imperial College London.

Abstract

Data Assimilation (DA) is an uncertainty quantification technique used to reduce forecasting errors by incorporating model predictions with observations of the state space. Variational Data Assimilation and Ensemble Kalman Filters are the typical methods of solving this problem. In many applications, the large size of the physical domain necessitates working in a reduced space in order to achieve the real-time DA required for operational use. Several proposals to reduce the space have been proposed and employed. In this project, we propose to use Autoencoder (AE) technology to reduce the computational cost of solving 3D Variational data assimilation. In particular, the space will be reduced by applying AEs to the background error covariance matrix. We validate this approach with data from the MAGIC project on pollution modelling and the open-source fluid dynamic software Fluidity.

Contents

1 Introduction

There are a host of systems that we would like to predict more accurately. Often, for example with weather systems, we are making millions of sensor measurements per second and also have a good, but not perfect, forecast model. Data Assimilation is a method of merging our observed data with our forecasting models to make our predictions more accurate for longer into the future.

Data Assimilation is a mature technology but in its most general form, is unfeasible for many large systems at the speeds required for real-time assimilation. As such, there are a variety of simplifying assumptions that make the problem tractable that have differing impacts on performance. One such simplification is the reduction of the rank of the background covariance matrix using PCA or SVD. These methods both have high computational complexity and greatly limit the speed at which DA can occur. The aim of this work is to apply modern machine learning technologies, specifically autoencoders, to obtain reduced space implicit representations of the background covariance matrix. The hypothesis is that the AE method will perform better than existing approaches in terms of both speed and assimilation accuracy.

AEs are widely used in DA to embed the knowledge of the forecast model efficiently in a process referred to as Reduced Order Modelling (ROM) but, to our knowledge, this is the first work applying AEs to the background covariance matrix space reduction.

In order to test our hypothesis, I will use data from the MAGIC project [1], [2] [3]: an international collaboration with the aim of producing an operational DA model to map the fluid flow in a site in South London as a method of monitoring and controlling pollution levels. I will attempt to reproduce and improve upon the work in [3] which uses truncated SVD (TSVD) as a method of reducing the space of the background covariance matrix.

I will detail the problem formulation as well as review related works in Section 2. In Section 3, I will detail our approach to solving this problem, while in Section 4, I will briefly summarise the progress made thus far including technical challenges encountered and detail my aims for the final three months of the project.

2 Related Works

The traditional use-case of Data Assimilation (DA) [4] [5] is Numerical Weather Prediction (NWP) [6], [7], [8] but the technique is applicable in any scenario where both a forecasting model and observational data is available. As such, it has now been utilised in contexts as diverse as Oceanic Modelling [9], [10], Solar Wind prediction [11] and inner city pollution modelling [3], the latter of which forms the basis for this work.

Forecasting models introduce uncertainty from numerous sources. These include, but are not limited to, uncertainty in initial conditions, imperfect representations of the underlying physical processes and numerical errors. As a

result, a model without access to real-time data will accumulate errors until its predictions no longer correspond to reality [12]. Similarly, all observations will have an irreducible uncertainty as a result of imperfect measuring devices. The key idea in DA is that the overall uncertainty in a forecast can be reduced by producing a weighed average of model forecasts and observations. In DA, quantities with lower uncertainty are given a larger importance and spatial and temporal correlations between data is taken into account.

In DA we make observations at a small subset of the total points in our system meaning that even they were completely reliable, we would not be able to use them to set the initial conditions for our forecasting model. Since we assume we approximately know how the system develops (as we have a forecasting model), DA is an 'inverse' problem which can be summarised as:

> *"what set of initial conditions will seed the models to best predict the known observations?"*[13]

The problem ill-posed meaning we must introduce a priori information in the form of historical data and knowledge of the underlying physics. Broadly speaking, there are three methods to approximately solve the Data Assimilation problem [14]:

i Ensemble Kalman Filters (EnKFs). KFs [15] are a Bayesian method of incorporating multiple sources of Gaussian uncertainty. In DA, they are typically used in ensembles (EnKF) [9] to generate error statistics.

ii Variational Data Assimilation (VarDA) methods [7], [8]. VarDA approaches involve minimising a cost function to obtain a single maximum likelihood estimate. In VarDA errors are also assumed to be Gaussian, but unlike EnKF, VarDA does not produce uncertainty estimates.

iii Monte-Carlo methods allow assimilation of data from sources with non-Gaussian uncertainties. The most common type of method is the Particle Filter (PF) [16].

(iii) are a generalisation of (i) where a non-Gaussian distribution is represented with an ensemble of models. They have very high computational cost and are not currently in operational use [14]. As a result, I will not cover them here but the interested reader should see the review of PFs and other Monte-Carlo methods in [16].

In this project, I will be using (ii) in order to test our hypothesis rather than (i) - even though the latter are widely used in operational systems. The primary reason for this is that the VarDA framework is more easily extended to incorporate the non-linearities introduced by AEs than KFs are. However, if our hypothesis is validated, the AE approach will be applicable in (i) as well as in (ii).

In the following sections I will briefly summarise the theoretical work relevant to this project starting with definitions of a set of consistent notation in section

(2.1). In section (2.2) I will cover KFs so that I can discuss why they are less appropriate in this project and in the following section (2.3) I will detail the form of VarDA that I will be using here. In section (2.4) I will review the AE literature with a specific focus on DA applications so that I am in a position to detail my approach to the problem in section (3).

2.1 Data Assimilation Definitions

I will mostly follow the notation given in the review paper [14].

- Let $\boldsymbol{x}_t$ represent the state of the model at time t such that:

$$\boldsymbol{x}_t \in \mathbb{R}^n$$

  where n is the number of elements in the state vector. The state for T total time-steps can be given in a single matrix:

$$\boldsymbol{X} = [\boldsymbol{x}_0, \boldsymbol{x}_1, ..., \boldsymbol{x}_T] \quad \in \quad \mathbb{R}^{n \times T}$$

  In many practical problems, n is large: $\geq \mathcal{O}(10^6)$.

- Let $\boldsymbol{y}_t$ represent the observation space of the system where:

$$\boldsymbol{y}_t \in \mathbb{R}^M$$

  where typically $M << n$.

- Let $\boldsymbol{\mathcal{H}}_t$ be an observation operator such that:

$$\boldsymbol{\mathcal{H}}_t[\boldsymbol{x}_t] = \boldsymbol{y}_t + \boldsymbol{\epsilon}_t \tag{1}$$

  where $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_t)$ is the observation error. Often, observations are assumed to be uncorrelated meaning that $\boldsymbol{R}_t$ is diagonal. When all observations are of the same type and made with the same device we have:

$$\boldsymbol{R}_t = \sigma_0^2 \boldsymbol{I}$$

- Let $\boldsymbol{\mathcal{M}}_{t-1,t}$ be the forecast model that propagates the system forward from time-step $t-1$ to t such that:

$$\boldsymbol{x}_t = \boldsymbol{\mathcal{M}}_{t-1,t}[\boldsymbol{x}_{t-1}] + \boldsymbol{\eta}_t \tag{2}$$

  where $\boldsymbol{\eta}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_t)$ is the model error introduced over this interval.

- Let $\boldsymbol{x}_t^b \in \mathbb{R}^n$ be the background state at time-step t. All a priori information about the system is introduced in the first background state $\boldsymbol{x}_0^b$. Future background estimates are defined according to the free-running model (i.e. where $\boldsymbol{\eta}_t$ is assumed to be zero):

$$\boldsymbol{x}_t^b = \boldsymbol{\mathcal{M}}_{t-1,t}[\boldsymbol{x}_{t-1}^b]$$

- Let $\boldsymbol{B}_t$ represent the background state $\boldsymbol{x}_t^b$ covariance matrix. In theory, it is found by evaluating:

$$\boldsymbol{B}_t = (\boldsymbol{x}_t^b - \boldsymbol{x}_t^*)(\boldsymbol{x}_t^b - \boldsymbol{x}_t^*)^T \tag{3}$$

  where $\boldsymbol{x}_t^*$ is the unknown true state of the atmosphere. In practice, even if the true state were known, this matrix is too large to fit in memory as it is in n^2 which is $\geq \mathcal{O}(10^{12})$ for most practical problems. There are multiple ways of representing $\boldsymbol{B}_t$ with fewer than n^2 pieces of information, including localisation [17], Control Variable Transforms [18] and TSVD [3]. In this work we present a novel method of approximately representing this matrix using an Auto-Encoder framework.

2.2 Kalman Filters, KFs

PLACEHOLDER: I'm afraid I was not able to write up my Kalman Filter notes in time for the progress report deadline. They are not critical for this work but will be present in the final literature review as they provide important context to the VarDA approach. I have read and analysed the following papers:

- [15] - Original paper
- [19] - Least Squares Derivation
- [9] - Ensemble Kalman Filters
- [20] - Non-linear KF for Oceanic Data Assimilation. There are very close links between this work and the work I am doing.
- [21] - Oceanic Data Assimilation with Neural Networks. Demonstrates the success of the NN approach with KF. Clear link between this and the work of this project.

2.3 Variational Data Assimilation, VarDA

VarDA involves minimising a cost function in order to find the most likely state values $\boldsymbol{X}^{DA}$ given the observations $\boldsymbol{y}_t$, model $\boldsymbol{\mathcal{M}}_{t-1,t}$ predictions and their uncertainties. The problem is to find the initial state $\boldsymbol{x}_0^{DA}$ that satisfies:

$$\boldsymbol{x}_0^{DA} = \arg\min_{\boldsymbol{x}_0} J(\boldsymbol{x}_0) \tag{4}$$

where the cost function $J(\boldsymbol{x}_0)$ is:

$$\begin{aligned} J(\boldsymbol{x}_0) = \frac{1}{2}\left\|\boldsymbol{x}_0 - \boldsymbol{x}_0^b\right\|^2_{\boldsymbol{B}_0^{-1}} + \frac{1}{2}\sum_{t=0}^{T}\left\|\boldsymbol{y}_t - \boldsymbol{\mathcal{H}}_t[\boldsymbol{x}_t]\right\|^2_{\boldsymbol{R}_t^{-1}} \\ + \frac{1}{2}\sum_{t=0}^{T}\left\|\boldsymbol{x}_t - \boldsymbol{\mathcal{M}}_{t-1,t}[\boldsymbol{x}_{t-1}]\right\|^2_{\boldsymbol{Q}_t^{-1}} \end{aligned} \tag{5}$$

In VarDA, this cost-function is explicitly differentiated and then approximately solved by first-order optimisation routines (see section 2.3.3). Note that this formulation is equivalent to the Kalman Filter result (see Appendix A).

The terms in (5) can be interpreted as follows:

- The first term is the background term J^b which measures the difference between our a priori expectation and the initial model state after DA has been performed.
- The second term is the observation term J^o which measure the deviation between the observations and the model predictions.
- The third term J^Q measures the size of the model error.

Note that we are assuming all errors are Gaussian by using this least-squares formulation. This problem specification is known as the 'weak constraint 4D-Var' [22] where:

- '4D' refers to the fact that we are considering three spatial dimensions as well as one temporal dimension. It is contrasted with 3D-Var which considers only one time-step.
- 'weak' refers to the fact that we are considering model errors and hence do not require the system to follow the model trajectory exactly. This is contrasted with *strong constraint* VarDA in which $J^Q = 0$.

In the current work, as a result of time restrictions, I am only expecting to implement strong constraint 3D-Var[1]. As such, we can drop the time subscripts (as we are only considering $t = 0$) and the cost function is given as:

$$J(\boldsymbol{x}) = \frac{1}{2} \left\| \boldsymbol{x} - \boldsymbol{x}^b \right\|^2_{\boldsymbol{B}^{-1}} + \frac{1}{2} \left\| \boldsymbol{y} - \boldsymbol{\mathcal{H}}[\boldsymbol{x}] \right\|^2_{\boldsymbol{R}^{-1}} \tag{6}$$

2.3.1 Incremental VarDA Formulation

The cost functions 6 (and 5) will be convex if $\boldsymbol{\mathcal{H}}_t$ (and $\boldsymbol{\mathcal{M}}_{t-1,t}$) are linear which they are not in general. We can approximately achieve this by linearising these operators about the background state $\boldsymbol{x}_0^b$ and formulating the problems in terms of perturbations to this background state in a method known as the incremental formulation [6][2]:

$$\delta \boldsymbol{x}_t \coloneqq \boldsymbol{x}_t - \boldsymbol{x}_t^b \tag{7}$$

[1] There is a possibility I will exceed these expectations. See section (4) for discussion of the order in which potential extensions will be attempted.

[2] Note: in theory, the perturbations can be made with respect to a general state $\boldsymbol{x}_0^g$ instead of $\boldsymbol{x}_0^b$ but this is rarely useful in practice.

For strong constraint incremental 3D-Var the problem is then:

$$\delta \boldsymbol{x}^{DA} = \arg\min_{\delta \boldsymbol{x}} J(\delta \boldsymbol{x}) \tag{8}$$

$$J(\delta \boldsymbol{x}) = \frac{1}{2} \left\| \delta \boldsymbol{x} \right\|^2_{\boldsymbol{B}^{-1}} + \frac{1}{2} \left\| \boldsymbol{d} - \boldsymbol{H} \delta \boldsymbol{x} \right\|^2_{\boldsymbol{R}^{-1}} \tag{9}$$

where $\boldsymbol{H}$ is the observation operator linearized about the background state and $\boldsymbol{d} = \boldsymbol{y} - \boldsymbol{H}[\boldsymbol{x}]$ is the 'misfit' between observation and prediction. The more general 4D-Var weak constraint incremental formulation is given in [6] and reviewed in a modern context in [14].

2.3.2 Control Variable Transform (CVT) in 3D VarDA

As discussed, to deal with the fact that $\boldsymbol{B}$ is in $\mathcal{O}(n^2)$ we must represent it implicitly. A common way of doing this is by using the formulation in [18] which states that:

$$\delta \boldsymbol{x} = \boldsymbol{V} \mathbf{w} \tag{10}$$

$$\boldsymbol{B} = \boldsymbol{V} \boldsymbol{V}^T \tag{11}$$

In this case the problem can be written as:

$$\mathbf{w}^{DA} = \arg\min_{\mathbf{w}} J(\mathbf{w})$$

$$J(\mathbf{w}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{1}{2} \left\| \boldsymbol{d} - \boldsymbol{H} \boldsymbol{V} \mathbf{w} \right\|^2_{\boldsymbol{R}^{-1}} \tag{12}$$

$\boldsymbol{V}$ can be estimated in a number of ways by:

i Averaging over a time-series of T' historical time-steps.

ii Averaging over a time-series of T' forecast time-steps.

iii Averaging over an ensemble of S forecasting models at the same time-step (see section 2.2)

Note that $T', S << n$ so in all three cases the implicit matrix $\boldsymbol{B}$ is not full rank and as a result of sampling errors these methods will overestimate the covariance between states that are spatially distant. One of the solutions to this is to use localisation methods [17] that attenuate the covariance between states that are spatially distant to one another. These approaches are outside the scope of this project but would be expected to improve on any results found here.

2.3.3 CVT VarDA optimisation

There is a large amount of research into VarDA optimisation strategies that are beyond the scope of the work here but it is necessary to highlight a few points about our optimisation approach:

- The problem in (12) is ill-conditioned as the matrix $\boldsymbol{V}$ has a large condition number[3] in most practical contexts (for example see [3]). Conjugate Gradient (CG) methods are well suited to minimisation of poorly conditioned functions [23].
- One variation of CG methods is L-BFGS which has been show by [24] and [25] and recently [3] to give faster convergence than their counterparts for problems of the size encountered in DA. I will use L-BFGS here.
- In L-BFGS, the rate of convergence is dependent on the condition number of the Hessian [25] (the larger the condition number, the slower convergence will be). As an aside, note that CVT confers an optimisation advantage over the standard incremental formulation (9) as (12) has the Hessian:

$$(\boldsymbol{I} + \boldsymbol{V}^T \boldsymbol{H}^T \boldsymbol{R}^{-1} \boldsymbol{H} \boldsymbol{V}) \tag{13}$$

  which has a minimum eigenvalue of 1 and is dominated by the condition number of $\boldsymbol{V}$ [26] while the non-CVT equation (9) has Hessian:

$$(\boldsymbol{B}^{-1} + \boldsymbol{H}^T \boldsymbol{R}^{-1} \boldsymbol{H}) \tag{14}$$

  which has a minimum eigenvalue of 0 and is dominated by the condition number of $\boldsymbol{B}$. This difference in condition-number and subsequent convergence rate has been extensively observed in the literature including in [3].

2.3.4 Reduced Space VarDA

Even with the CVT Transform, optimisation methods are slow. As such, it is common in DA to use an eigen-analysis technique such as SVD or Empirical Orthogonal Functions (EOFs) [27][4] to remove low-variance modes from $\boldsymbol{V}$. This has the dual benefit of:

i Reducing the condition number of the problem (as lower variance modes are removed) and hence reducing the number of iterations required in the minimisation routine.

ii Reducing the complexity of the matrix multiplications (as the factorised form is available).

[3] The condition number is the ratio of the largest to the smallest eigenvalues.

[4] Note that EOFs is an equivalent term to PCA which is used more widely in the DA literature. The approaches are mathematically equivalent in that they find a unique orthogonal basis that maximises the variance. Unlike the majority of computing, 'EOFs' is used instead of 'PCA' in the DA literature to reference the fact that one of the dimensions is time.

In this work, we propose a novel method to achieve all of the above using autoencoders.

2.4 Autoencoders

Autoencoders are an unsupervised machine learning method first proposed in [28]. They consist of an encoder $f(\boldsymbol{x})$ and decoder $g(\mathbf{w})$ such that:

$$f(\boldsymbol{x}) = \mathbf{w} \tag{15}$$
$$g(\mathbf{w}) = \boldsymbol{x'} \tag{16}$$

where $f(\cdot)$ and $g(\cdot)$ are non-linear neural networks and:

$$\boldsymbol{x}, \boldsymbol{x'} \in \mathbb{R}^n \tag{17}$$
$$\mathbf{w} \in \mathbb{R}^\tau \tag{18}$$
$$\tau < n \tag{19}$$

In other words, AEs compress the input data $\boldsymbol{x}$ to find a a latent representation $\boldsymbol{z}$ that is of lower dimensionality. The L1 or L2 of the reconstruction error is used as the training loss function which are respectively:

$$J_1(\boldsymbol{x}, \boldsymbol{x'}) = \ |\boldsymbol{x} - \boldsymbol{x'}| \tag{20}$$
$$J_2(\boldsymbol{x}, \boldsymbol{x'}) = \|\boldsymbol{x} - \boldsymbol{x'}\|^2 \tag{21}$$

Baldi refers to the AE training process as an attempt to find 'a low rank approximation to the identity function' [29].

2.4.1 Comparison with TSVD and PCA

Eigenanalysis techniques like PCA and TSVD are also methods of representing data in a reduced space. The key difference is that, unlike with these methods, the latent representation obtained by a general AE will be tailored to the input data distribution. Concretely, AEs are theoretically equivalent to a clustering algorithm [29] in the sense that the encoder learns to map commonly co-occurring inputs to a single internal representation. Moreover, PCA is a special case of an AE with linear activations (see Appendix B for derivation) [30].

There are three key advantages of using non-linear AEs for dimensionality reduction over either SVD or PCA:

i Calculating PCA and SVD for an $n \times M$ matrix where $M \leq n$ have complexities $\mathcal{O}(M^3)$ and $\mathcal{O}(M^2 n)$ respectively. In comparison, AE inference is linear in the input size which in the case of this matrix is $\mathcal{O}(Mn)$.

ii SVD and PCA both require that the latent basis is orthogonal which is only optimal when the data is drawn from a Gaussian distribution. In comparison, AEs produce non-linear combinations of the inputs meaning that arbitrary basis functions can be constructed.

iii Although matrix multiplications involving TSVD factorised matrices have low computational cost, the latent representation of the matrix $\boldsymbol{V}_{svd}$ is of the same size as the original matrix $\boldsymbol{V} \in \mathbb{R}^{n \times M}$. In comparison, the latent representation $\mathbf{w}$ in the AE system is $\in \mathbb{R}^{\tau}$. In most cases $\tau << nM$ so there will be a vast reduction in the size of the reduced space.

2.4.2 AEs in Data Assimilation

AEs are used extensively in DA, particularly in the task of Reduced Order Modelling (ROM). ROM is the method of embedding the knowledge in the forecast model $\boldsymbol{\mathcal{M}}_{t-1,t}, \forall t$ in a neural network. The rationale behind this is that the majority of the models are very expensive to run whilst neural networks are very fast at inference time. The first generation of ROMs used Proper Orthogonal Decomposition (POD). There are many examples of this approach but the following is typical [31]). Like EOFs, PODs are another way of referring to the modes generated by PCA. Modern ROMs are end-to-end neural networks that learn the physics with enough fidelity to be useful. The most comprehensive example in the literature is [2] although the authors of [32] produced a Neural network based ROM back in 2007. The MAGIC authors used a hybrid approach in which they used a neural network that was trained to find approximate PODs [2]. This method will successfully reduce inference time but loose the non-linear benefit of the neural network approach.

As far as I am aware, there are far fewer documented uses of AEs for Data Assimilation outside of ROM[5]. Of note is is [21] in which the authors used an autoencoder to produce a non-linear extension to the Kalman Filter for Oceanic Data Assimilation.

Similarly, I cannot find any work on the use of AEs for covariance matrix space reduction, although the following paper from 2017 [33] used AEs as a method of removing noise from covariance matrices.

3 Project Problem statement

In this project I aim to solve the problem in (12) with a small alteration: the $\boldsymbol{V}$ matrix in (12) is replaced with the non-linear decoder $g(\mathbf{w})$ neural network. $g(\mathbf{w})$ is half of the autoencoder framework:

$$f(\delta \boldsymbol{x}) = \mathbf{w}$$
$$g(\mathbf{w}) = \delta \boldsymbol{x'}$$

[5] I admit my reading on this topic is not extensive. I would appreciate any suggestions of relevant work.

where the latent representation is $\mathbf{w} \in \mathbb{R}^{\tau}$.

This gives the following formulation:

$$\mathbf{w}^{DA} = \arg\min_{\mathbf{w}} J(\mathbf{w})$$

$$J(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{1}{2}\left\|\boldsymbol{d} - \boldsymbol{H}g(\mathbf{w})\right\|^2_{\boldsymbol{R}^{-1}} \tag{22}$$

I will train and test the AE $f(\cdot)$, $g(\cdot)$ using simulated data from the open-source, finite-element, fluid dynamic software (`http://fluidityproject.github.io/`) in the MAGIC [1] domain in South London. For comparison, I will also replicate the work in [3] which solves (12) in the same location directly using TSVD.

4 Report: Progress and Challenges

The AE approach has many advantages over TSVD which have been covered at length above but it also raises two key challenges not present in existing DA methods:

i The minimisation of (22) requires the calculation of decoder network Jacobian matrix as, unlike the matrix $\boldsymbol{V}$, $g(\cdot)$ is a non-linear mapping.

ii A large amount of model simulation data is required to train the AE and the training process is expensive.

I will discuss these in turn before summarising my progress so far.

4.1 Jacobian Calculation

This challenge is considerable and was unforeseen at the start of the project so has impacted my rate of progress. The issue is that the Jacobian must be calculated in an efficient way so as not to invalidate one of the two hypotheses of the project: that AE based DA will be faster than TSVD based DA. The differential of (22) is:

$$\nabla J(\mathbf{w}) = \mathbf{w} - g'(\mathbf{w})^T\boldsymbol{H}^T\boldsymbol{R}^{-1}\big(\boldsymbol{d} - \boldsymbol{H}g(\mathbf{w})\big) \tag{23}$$

where $g'(\mathbf{w}) \in \mathbb{R}^{n\times\tau}$ is the Jacobian of the decoder network outputs $\delta\boldsymbol{x'}$ with respect to its inputs $\mathbf{w}$.

While there are many Python software frameworks that automatically accumulate gradients including Autograd, Torch and Tensorflow, they do not, as far as I am aware, calculate multi-dimensional Jacobians directly as the software is primarily used to calculate gradients with respect to weights rather than inputs. Instead they calculate 'Jacobian-Vector products' [34] which are single columns in the full $n \times \tau$ Jacobian. These tools can be used to construct full Jacobian but finding a method of avoiding expensive for loops has proved elusive.

My current solution is to differentiate the networks by hand and then write explicit functions for the gradients of each network. This is easy enough for simple feed-forward decoders but I foresee difficulties in implementing it this way for convoloutional AEs. Better suggestions are warmly welcomed.

4.2 Training Data

My current AE is trained on a single run of the Fluidity model. In order to increase the DA performance it will be desirable to do one of the following:

i Run an ensemble of Fluidity models in the same domain with perturbed initial conditions. Fluidity automatically corrects for noisy inputs in order to conserve physical quantities such as mass so this may not be as straightforward as it sounds.

ii Train a CAE on Fluidity data from a range of locations so that it learns underlying structure governing background covariance attenuation.

4.3 Progress

So far I have completed the following tasks:

i Conducted a wide-ranging review of the Data Assimilation literature - I had no background in this topic before commencing this project.

ii Completed an implementation of the TSVD approach in [3] to a standard high enough for open-source release.

iii Trained a simple feedforward AE in Torch. Unfortunately the latent representation is currently poor in comparison with TSVD.

iv Hand implemented the Jacobian of simple decoder networks above.

v Implementation of equation (22) - i.e. 3D-VarDA within an AE framework.

vi Good test coverage of all of the above.

vii Conducted a number of experiments to diagnose the issues with the current AE.

The implementation is at `https://github.com/julianmack/Data_Assimilation`[6]
Over the next three months, at minimum, I would like to achieve the following:

i Generate more fluidity data and use this for training.

ii Obtainin a model that performs better than TSVD. Only then conduct an extensive hyperparameter search.

[6]Please do not distribute until the project is further along.

iii Conduct experiments to compare the speed and accuracy of DA with AE and TSVD.

iv Expand literature review to include (and be more critical of) ML literature.

v Write up missing KF section and Appendices.

If time allows I would also like to attempt the following (in order of preference):

i Implement a 3D Convolutional AE rather than feedforward AE. This will likely greatly improve the quality of the latent representation. Fluidity uses a non-cubic grid so interpolation may be necessary. In current implementation, this will require hand differentiation of any decoder network used.

ii Add localisation techniques to this pipeline [17].

iii Implement 4D-Var.

iv Extend the above using an LSTM CAE to allow for flow-dependency in the representation of $\boldsymbol{B}$.

v Extend the above with a NN based ROM that is trained end-to-end at the same time as the background covariance matrix AE.

vi Apply this work to an alternative forecasting model.

It is unlikely that I will achieve the majority of the above in the time available.

References

[1] J. Song, S. Fan, W. Lin, L. Mottet, H. Woodward, M. Davies Wykes, R. Arcucci, D. Xiao, J. E. Debay, H. ApSimon, E. Aristodemou, D. Birch, M. Carpentieri, F. Fang, M. Herzog, G. R. Hunt, R. L. Jones, C. Pain, D. Pavlidis, A. G. Robins, C. A. Short, and P. F. Linden, "Natural ventilation in cities: the implications of fluid mechanics," *Building Research and Information*, vol. 46, no. 8, pp. 809–828, 2018.

[2] Z. Wang, D. Xiao, F. Fang, and C. Pain, "Model identification of reduced order fluid dynamics systems using deep learning Model identification of reduced order fluid dynamics systems using deep learning," no. October, 2017.

[3] R. Arcucci, L. Mottet, C. Pain, and Y. K. Guo, "Optimal reduced space for Variational Data Assimilation," *Journal of Computational Physics*, vol. 379, pp. 51–69, 2019.

[4] A. C. Lorenc, "Analysis methods of numerical weather prediction," *Quarterly Journal of the Royal Meteorological Society*, vol. 112, no. 474, pp. 1177–1194, 1986.

[5] A. C. Lorenc, "Optimal Nonlinear objective analysis," *Quarterly Journal of the Royal Meteorological Society*, vol. 114, no. 479, pp. 205–240, 1988.

[6] P. Courtier, J.-N. Thépaut, and A. Hollingsworth, "A strategy for operational implementation of 4D-Var, using an incremental approach," *Quarterly Journal of the Royal Meteorological Society*, vol. 120, no. 519, pp. 1367–1387, 1994.

[7] P. Courtier, E. Anderson, W. Heckley, D. Vasiljevec, M. Hamrud, A. Hollingsworth, F. Rabier, M. Fisher, and J. Pailleux, "The ECMWF implementation of three-dimensional variational assimilation (3D-Var). I: Formulation," *Quarterly Journal of the Royal Meteorological Society*, vol. 124, no. 550, pp. 1783–1807, 1998.

[8] W. Huang, A. J. Bourgeois, Q. N. Xiao, D. M. Barker, and Y.-R. Guo, "A Three-Dimensional Variational Data Assimilation System for MM5: Implementation and Initial Results," *Monthly Weather Review*, vol. 132, no. 4, pp. 897–914, 2004.

[9] G. Evensen, "The Ensemble Kalman Filter: Theoretical formulation and practical implementation," *Ocean Dynamics*, vol. 53, no. 4, pp. 343–367, 2003.

[10] S. Dobricic and N. Pinardi, "An oceanographic three-dimensional variational data assimilation scheme," *Ocean Modelling*, vol. 22, no. 3-4, pp. 89–105, 2008.

[11] M. Lang and M. J. Owens, "A Variational Approach to Data Assimilation in the Solar Wind," *Space Weather*, vol. 17, no. 1, pp. 59–83, 2019.

[12] J. Tribbia and D. P. Baumhefner, "Scale Interactions and Atmospheric Predictability: An Updated Perspective," *Monthly Weather Review*, vol. 132, no. 3, pp. 703–713, 2004.

[13] R. N. Bannister, "Elementary 4d-VAR," *DARC Technical Report No. 2*, no. 2, pp. 1–16, 2001.

[14] R. N. Bannister, "A review of operational methods of variational and ensemble-variational data assimilation," *Quarterly Journal of the Royal Meteorological Society*, vol. 143, no. 703, pp. 607–633, 2017.

[15] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[16] P. J. van Leeuwen, "Particle Filtering in Geophysical Systems," *Monthly Weather Review*, vol. 137, no. 12, pp. 4089–4114, 2009.

[17] T. Montmerle, Y. Michel, É. Arbogast, B. Ménétrier, and P. Brousseau, "A 3D ensemble variational data assimilation scheme for the limited-area AROME model: Formulation and preliminary results," *Quarterly Journal of the Royal Meteorological Society*, vol. 144, no. 716, pp. 2196–2215, 2018.

[18] R. N. Bannister, "A review of forecast error covariance statistics inatmospheric variational data assimilation. II: Modelling the forecast error covariance statistics," vol. 1996, no. November, p. 496, 2008.

[19] A. Lacey and N. Thacker, "Tutorial: The Kalman filter," *Imaging Science and Biomedical . . .*, pp. 133–140, 1998.

[20] S. Frolov, A. M. Baptista, T. K. Leen, Z. Lu, and R. van der Merwe, "Fast data assimilation using a nonlinear Kalman filter and a model surrogate: An application to the Columbia River estuary," *Dynamics of Atmospheres and Oceans*, vol. 48, no. 1-3, pp. 16–45, 2009.

[21] C. A. Quilodran Casas, N. Sparks, and R. Toumi, "Fast ocean data assimilation using a neural-network reduced-space regional ocean model of the North Brazil Current," *Progress in Oceanography*, 2019.

[22] D. Zupanski, "A General Weak Constraint Applicable to Operational 4DVAR Data Assimilation Systems," *Monthly Weather Review*, vol. 125, no. 9, pp. 2274–2292, 2002.

[23] L. D. Navon, I. M., "Conjugate Gradient Methods for Large-Scale Minimization in Meteorology," 1987.

[24] A. K. Alekseev, I. M. Navon, and J. L. Steward, "Comparison of advanced large-scale minimization algorithms for the solution of inverse ill-posed problems," *Optimization Methods and Software*, vol. 24, no. 1, pp. 63–87, 2009.

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

[26] R. Arcucci, L. D. Amore, J. Pistoia, and R. Toumi, "On the Variational Data Assimilation Problem Solving and Sensitivity Analysis," *Journal of Computational Physics*, 2017.

[27] E. N. Lorenz, "Empirical Orthogonal Functions and Statistical Weather Prediction," 1956.

[28] D. E. Rumelhart, G. Hinton, and R. J. Williams, "Learning Internal Representations by Error Propagation," *Cognitive Science*, no. V, 1986.

[29] P. Baldi, "Autoencoders, Unsupervised Learning, and Deep Architectures," pp. 37–50, 2012.

[30] P. Baldi and K. Hornik, "Neural networks and principal component analysis: Learning from examples without local minima," *Neural Networks*, vol. 2, no. 1, pp. 53–58, 1989.

[31] L. Cordier, B. R. Noack, G. Tissot, G. Lehnasch, J. Delville, M. Balajewicz, G. Daviller, and R. K. Niven, "Identification strategies for model-based control Topics in Flow Control. Guest editors J.P. Bonnet and L. Cattafesta," *Experiments in Fluids*, vol. 54, no. 8, 2013.

[32] R. van der Merwe, T. K. Leen, Z. Lu, S. Frolov, and A. M. Baptista, "Fast neural network surrogates for very high dimensional physics-based models in computational oceanography," *Neural Networks*, vol. 20, no. 4, pp. 462–478, 2007.

[33] S. Hayou, "Cleaning the correlation matrix with a denoising autoencoder," pp. 1–11, 2017.

[34] R. Grosse, "Lecture 10: Automatic Differentiation," in *Lecture Slides*, 2013.

Appendices

A Kalman Filter Derivation via cost function formulation

PLACEHOLDER: Due to time restraints - I have not been able to write up this derivation. Apologies for this. Please see [19] for the majority of the derivation. The final steps are my own.

B Equivalence of Linear AE and PCA

PLACEHOLDER: Due to time restraints - I have not been able to write up this derivation. Please see [30].