Model Error Correction in Ensemble Data Assimilation

ACM 270 Report

Ritvik Teegavarapu
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
rteegava@caltech.edu

Miguel Liu-Schiaffini
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
mliuschi@caltech.edu

Hojjat Kaveh
Mechanical and Civil Engineering
California Institute of Technology
Pasadena, CA 91125
hkaveh@caltech.edu

Matthieu Darcy
Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
mdarcy@caltech.edu

June 22, 2024

Abstract

In numerical modeling, it is of interest to reduce the error incurred by a surrogate model, occurring from either incorrect assumptions of the underlying physics or unresolved processes. Learning this error step is of crucial importance to reduce the computational cost of repeated data assimilation calls. For this project, we consider the two-scale Lorenz-96 system, which has two types of variables acting at different (fast and slow) frequencies. In particular, we assimilate noisy observations from the slow scales of the true system into the single-scale Lorenz-96 system, thus incurring modeling error via the unresolved dynamics. We compare the effects of using a neural network to learn the increment step of the assimilation against the baseline ensemble Kalman filter (EnKF). We also use the auto-differentiable EnKF to directly learn the forecasting step. We find that the neural network-corrected EnKF tends to outperform the standard EnKF for most noise regimes. We find that the auto-differentiable EnKF generally underperforms compared to the neural network-correct EnKF but outperforms the baseline EnKF.

1 Introduction

When solving problems in numerical modeling, it is often useful to incorporate physical knowledge of the system to make more informed predictions about the system. As noted in Levine and Stuart [2022], data assimilation is the primary technique for predicting latent state variables using observations of the system, and theoretical frameworks have been posed in order to compare the estimated state to that of the true state. In prototypical applications of this methodology, such as climate modeling, it is often useful to cast the operators as physical transformations according to the physics of the system. However, incorrect assumptions on these physics or the underlying dynamics can yield to large model errors especially in chaotic dynamical systems, a topic explored in the paper Li et al. [2022].

In practice, this can be dealt with multiple ways. As noted in Farchi et al. [2021], there has been work done in alternating between the machine learning and data assimilation step to increase the accuracy of estimating not only the state, but also the surrogate model as whole. A drawback of this is the computational power needed to run repeated data assimilation calls, as well as sensitivity to the initialization of the surrogate model parameters. An appealing alternative has been to instead consider *hybrid* models, in which one uses the already existing knowledge-based model to inform the error of the surrogate model.

This paper will discuss the effects of implementing such a hybrid model in the context of the Lorenz-96 system, in which we have both slow-scale and fast-scale states. The problem with this system is that these scales interact with each other in the dynamical system, and our observations are furthermore noisy projections of these fast-scale states. Thus, it is of interest to see how incorporating a surrogate model that directly learns a mapping from the predicted state with no assimilated data to the error incurred (called the analysis increment) compares against a traditional data assimilation method (the ensemble Kalman filter for the paper).

The remainder of this report is organized as follows. In Section 2, we describe the basic data assimilation problem setup and the filtering problem of interest. We also describe our Lorenz-96 dataset and its details. In Section 3 we describe the three methods that we explore in this project: the ensemble Kalman filter, a neural network-corrected ensemble Kalman filter, and the auto-differentiable ensemble Kalman filter [Chen et al., 2022]. In Section 4 we discuss the main results of the project, and in Section 5, we provide some discussion of our results.

2 Problem Setting

In this project, we consider the assimilation of observations from a true fast-scale system into a slow-scale system. We consider an extension of the standard data assimilation problem.

We first define the standard *data assimilation setting*. Consider a dynamical system of the form

$$v_{j+1}^{\dagger} = \Psi\left(v_j^{\dagger}\right) + \xi_j^{\dagger} \tag{1}$$

$$v_0^{\dagger} \sim \mathcal{N}(m_0, C_0), \quad \xi_j^{\dagger} \sim \mathcal{N}(0, \Sigma), \tag{2}$$

where the sequence $\{\xi_j^{\dagger}\}$ is drawn i.i.d. and is independent of $v_0^{\dagger}$. Intuitively, Ψ corresponds to the evolution operator for the Markovian dynamics of the system $\{v_j^{\dagger}\}$, and $\xi_j^{\dagger}$ corresponds to additive Gaussian noise. From this true system, we consider the observation operator $h(\cdot)$, which defines the (potentially noisy) observations of the true system:

$$y_{j+1}^{\dagger} = h\left(v_j^{\dagger}\right) + \eta_{j+1}^{\dagger}, \tag{3}$$

$$\eta_{j+1}^{\dagger} \sim \mathcal{N}(0, \Gamma), \tag{4}$$

where $\eta_{j+1}^{\dagger}$ are drawn i.i.d. and $\{\eta_{j+1}^{\dagger}\}$ is independent of $v_0^{\dagger}$ and $\{\xi_j^{\dagger}\}$.

The classical data assimilation setting assumes that one has access to the true evolution operator Ψ. However, in practice this may not be the case. In this project, we assume that the true system evolves by a fast-scale system, but that the observations that we receive are noisy projections of this fast-scale system onto its slow-scale components. As such, we instead consider the hidden state to be v_{j+1}, the slow-scale components of the true fast-scale state $v_{j+1}^{\dagger}$. We also assume that we only have access to Ψ_s, the evolution operator of the slow-scale system.

As such, we would like to solve the *filtering problem* on this system. We assume that we are given the history of observations until time j, denoted $Y_j^{\dagger}$, and we wish to find the distribution $\mathbb{P}(v_j \,|Y_j^{\dagger})$.

2.1 Lorenz-96 System

In particular, in this project we consider the two-scale Lorenz-96 system, with both a fast and slow scale. We assume that the two-scale system is the true system with states $\{v_j^{\dagger}\}$, and the projection of these states (plus noise) onto the slow-scale components are the observations $\{y_j^{\dagger}\}$. We assume that we have access to the slow-scale dynamics Ψ_s, and we let $\{v_j\}$ be the states that are derived from evolving the slow-scale dynamics. As such, in this case the observation operator $h(\cdot)$ is merely the identity map.

Consider the following equations for the Lorenz-96 (abbreviated L96 henceforth), which consist of two equations coupling the fast and slow scales as mentioned.

$$\frac{dX_k}{dt} = -X_{k-1} \cdot (X_{k-2} - X_{k+1}) - X_k + F - \left(\frac{hc}{b}\right) \cdot \sum_{j=0}^{J-1} Y_{j,k} \tag{5}$$

$$\frac{dY_{j,k}}{dt} = -cbY_{j+1,k} \cdot (Y_{j+2,k} - Y_{j-1,k}) - cY_{j,k} + \frac{hc}{b} X_k \tag{6}$$

Parameter	Description	Value
K	Number of global-scale variables	40
J	Number of local-scale variables per single global-scale variable	5
`nt`	Number of time-steps	5000
`si`	Sampling time interval	0.2
Δt	Integrator time interval	0.005
F	Forcing	8
h	Coupling coefficient	1
b	Ratio of amplitudes	10
c	Time-scale ratio	10

Table 1: Parameters used for data generation of the two-scale L96 data.

The $\{X_k\}_{k=1}^{K}$ denote the K slow-scale variables, and the $\{Y_{j,k}\}_{j=1}^{J}$ denote the total $J \cdot K$ fast-scale variables. The *coupling* of the contrasting variable occurs in the last term of the dynamical system. In the first equation, the fast-scale variables are summed over a particular choice of k in the slow-scale ODE. In the second equation, the fast-scale ODE has a forcing term that is governed by the slow-scale variable to the particular k.

Furthermore, we have some key parameters that are informative with respect to the physical dynamics being modeled.

- The constant b indicates the influence of the topology for non-linear interaction terms with respect to the fast-scale.
- The constant c indicates the rate of fluctuations in the fast-scale variables in comparison to that of the slow-scale variables.
- The constant h indicates the strength of the coupling between the fast and slow-scale variables.
- The constant F is a forcing term that determines the chaotic nature of the dynamical system.

The primary application of the L-96 system is to serve as a simplified version of the model for the atmosphere dynamics. As noted in Balwada et al. [2023], the real atmospheric system experiences effects on all scales, from a particulate level to patches of the Earth. In a computational sense, it is intractable to resolve this entire scope of scales (namely a global climate model), and the problem reduces to parameterizing the unresolved scales to understand their affects on the resolved ones.

As discussed, the L-96 system is a simplified version of the atmospheric system, in which we only consider the slow and fast scales, and the analogous global climate model in this setting would then be using the resolved slow scales to resolve the fast scales. Thusly, our problem of interest is to synthesize trajectories from the ground truth two-scale model, and assimilate these slow-scale noisy trajectories in hopes of recovering the full system.

2.2 Data Generation

As mentioned above, in this project we assume that the 2-scale L96 model is the ground truth, and use the slow-scale components of this true model (plus noise) as the observations.

To generate these synthetic observations, we used a fourth-order Runge-Kutta integrator on the two-scale system to generate the true model observations. We use the code and methodology presented in Balwada et al. [2023] to generate the data. The data-generation process on the true system is noise-less, but we later incorporate observation noise drawn from an isotropic Gaussian. Based on Balwada et al. [2023], we define the initial condition as

$$\begin{aligned} X_k^{(0)} &= s(k,K) \cdot (s(k,K) - 1) \cdot (s(k,K) + 1) + \epsilon_k \\ Y_{j,k}^{(0)} &= 0, \end{aligned}$$

where $s(k,K) = \frac{1+2k}{K} - 1$ and $\epsilon_k \sim \mathcal{N}(0, 0.05)$.

The values and descriptions of the various parameters of interest are shown in Table 1. Figure 1 shows a sample true 2-scale trajectories generated with these parameters and their corresponding projections onto the slow scales.

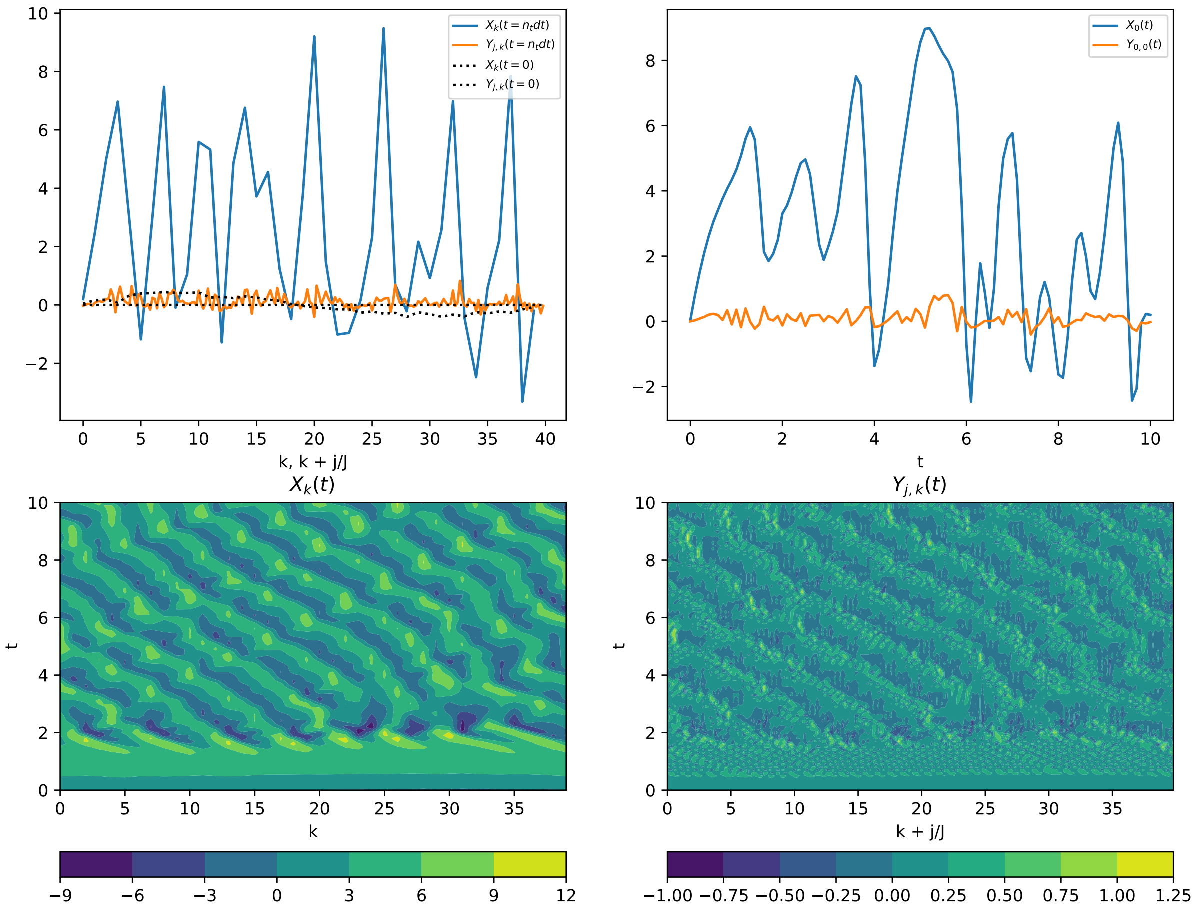

Figure 1: **Top-left:** Sample slow (X) and fast (Y) scale states at the intial condition and at time $t = 10$ seconds. **Top-right:** X_0 and $Y_{0,0}$ as a function of time for a particular trajectory. **Bottom-left:** States of the slow-scale components over time. **Bottom-right:** States of the fast-scale components over time. The code used to generate figures can be found in Balwada et al. [2023].

3 Methods

In this section, we describe the methods used in this project. We first review the ensemble Kalman filter. Then, we discuss a neural network correction of the ensemble Kalman filter. Finally, we review the auto-differentiable ensemble Kalman filter, introduced in Chen et al. [2022].

3.1 Ensemble Kalman Filter (EnKF)

In this section, we review the classical ensemble Kalman filter, henceforth abbreviated as EnKF. This filter is typically used in settings where the state operator Ψ is non-linear and our observation operator H is linear. This methodology reduces to doing a Kalman-style update on an ensemble of N particles at time step j denoted as $\{v_j^{(n)}\}_{n=1}^N$, with the covariance computed as the ensemble covariance. Thus, the filtering distribution can be approximated by the empirical measure as follows.

$$\pi_j^N(v_j) \approx \frac{1}{N} \cdot \sum_{n=1}^{N} \delta_{v_j - v_j^{(n)}}$$

Using the dynamical model, one can predict the states of the particles at time $j+1$ which is denoted as $\{\hat{v}_{j+1}^{(n)}\}_{n=1}^N$. The empirical covariance can then be used to construct the objective function and perform the analysis step. The algorithm below describes the implementation of EnKF.

```
1: **Input:** Ensemble size $N$
2: **for** $n = 1, \cdots, N$ **do**
3:     $v_0^{(n)} \sim \mathcal{N}(m_0, C_0)$                                    ▷ Initialize ensemble members
4: **end for**
5: **for** $j = 0, \cdots, J-1$ **do**
6:     $\xi_j^{(n)} \sim \mathcal{N}(0, \Sigma)$ i.i.d $n = 1, \cdots, N$                      ▷ Prediction Step
7:     $\hat{v}_{j+1}^{(n)} = \Psi\left(v_j^{(n)}\right) + \xi_j^{(n)}$ $n = 1, \cdots, N$
8:     $\hat{m}_{j+1} = \frac{1}{N} \cdot \sum_{n=1}^{N} v_{j+1}^{(n)}$
9:     $\hat{C}_{j+1} = \frac{1}{N} \cdot \sum_{n=1}^{N} \left(v_{j+1}^{(n)} - \hat{m}_{j+1}\right) \oplus \left(v_{j+1}^{(n)} - \hat{m}_{j+1}\right)$
10:    $\eta_{j+1}^{(n)} \sim \mathcal{N}(0, \Gamma)$ i.i.d $n = 1, \cdots, N$                     ▷ Analysis Step
11:    $y_{j+1}^{(n)} = y_{j+1} + \eta_{j+1}^{(n)}$ $n = 1, \cdots, N$
12:    $v_{j+1}^{(n)} = (I - K_{j+1}H)\hat{v}_{j+1}^{(n)} + K_{j+1}y_{j+1}^{(n)}$ $n = 1, \cdots, N$
13: **end for**
```

The Kalman gain matrix can be written as follows in terms of the empirical covariance matrix $\hat{C}_{j+1}$, observation operator H, and covariance matrix for the noise Γ.

$$K_{j+1} = \hat{C}_{j+1}H^\top \cdot \left(H\hat{C}_{j+1}H^\top + \Gamma\right)^{-1}$$

In our implementation of EnKF, we also consider *inflation* and *localization*. For some $\alpha \geq 1$, we consider multiplicative inflation, by which

$$\hat{v}_{j+1}^{(n)} \leftarrow \hat{m}_{j+1} + \alpha\left(\hat{v}_{j+1}^{(n)} - \hat{m}_{j+1}\right). \tag{7}$$

This has the effect of increasing the forecast covariance, which alleviates the underestimating of analysis covariance by sample error. We also consider localization, which scales the values of the forecast covariance by a Gaussian kernel. This has the effect of dampening spurious correlations between variables that are spatially distant. In particular, localization performs the following modification of the forecast covariance matrix:

$$\hat{C}_{j+1} \leftarrow L \odot \hat{C}_{j+1}, \tag{8}$$

where $\odot$ represents the Hadamard product and for some lengthscale ℓ, the localization matrix is defined as

$$L_{ik} = e^{-D_{ik}^2/\ell}, \tag{9}$$

where D_{ik} is the spatial distance between variables X_i and X_k.

3.2 Neural Network Correction of the Ensemble Kalman filter (NN-EnKF)

In this section, we describe our methodology for introducing a neural network correction to the EnKF, which we abbreviate NN-EnKF. Suppose that we have run an EnKF on a trajectory (or a subset of one), and we have access to both the forecast results $\{\hat{v}_j^{(n)}\}_j$ for all $n = 1, \ldots, N$ and the analysis results $\{v_j^{(n)}\}_j$ for all $n = 1, \ldots, N$.

The goal of the neural network correction is to train a neural network to correct the systematic error of the forward model. In the L-96 example, since we are assimilating noisy observations of the two-scale L-96 system into the one-scale system, the time-evolution operator Ψ_s is biased with respect to the true two-scale evolution operator Ψ.

As such, we train a simple feedforward neural network (although the particular architecture is not relevant) to predict the analysis increment from the forecast state; that is, the map

$$\hat{v}_j^{(n)} \mapsto v_j^{(n)} - \hat{v}_j^{(n)}, \tag{10}$$

using the full training set (i.e., results from the forecast and analysis steps for each particle in the enesemble) from the baseline EnKF.

The intuition behind this approach is that under the approximate dynamics Ψ_s, the EnKF must account for both the systematic error between Ψ_s and Ψ and the noise incurred from the observations. The neural network learns to map the forecast state to the analysis increment of the EnKF without the observations. The interpretation is thus that the neural network is learning to correct for this systematic error based only on the forecast state.

Given a trained neural network NN_θ parameterized by θ, we once again assimilate the observations. The NN-corrected analysis step is given by

$$v_{j+1}^{(n)} = \hat{v}_{j+1}^{(n)} + \text{NN}_\theta(\hat{v}_{j+1}^{(n)}) + K_{j+1}(y_{j+1}^{(n)} - \eta_{j+1}^{(n)} - H\hat{v}_{j+1}^{(n)}). \tag{11}$$

The intuition of the neural network correction is that NN_θ adjusts the systematic error, while the assimilation of the observations adjusts for the noise.

3.3 Auto-differentiable Ensemble Kalman Filter (AD-EnKF)

In this section, we review the auto-differentiable EnKF, as introduced in Chen et al. [2022]. To adapt the same notation as provided in the paper, we will note that $x_{0:T}^{1:N}$ represents the ensemble of N particles from time $j = 0$ to J. With respect to time, we have a homogeneous transition kernel $p(x_j|x_{j-1};\theta)$ is parameterized by θ which may encode information about the dynamical system or the surrogate neural network. The goal is to then learn θ given the assimilated observations $y^{1:J}$.

To do this, the AD-EnKF framework is to perform a gradient ascent algorithm on θ. First, this is done by approximating the log-likelihood of θ defined as $\mathcal{L}(\theta) := p_\theta(y^{1:J})$. We first assume that in our initial formulation of the filtering problem, the transition kernel p can be written as follows where $\theta = \{\alpha, \beta\}$.

$$p(x_j|x_{j-1};\theta) = \mathcal{N}(x_j; F_\alpha(x_{j-1}), Q_\beta) \tag{12}$$

We now have the parameterization of θ in terms of a deterministic mapping F_α and covariance matrix Q_β. By our typical definition of the forecast step, we know that the transition kernel can be approximated by the normal distribution centered around the empirical mean and with spread empirical covariance.

$$p(x_j|x_{j-1};\theta) \approx \mathcal{N}(\hat{m}_j, \hat{C}_j) \tag{13}$$

From this approximation, we know that the forecasted particles $\hat{v}_{j+1}^{(n)}$ will implicitly depend on the choice of θ through the initial particle $v_{j+1}^{(n)}$, as well as explicitly through the choice of α and β from the mapping written in 12. From all this, we can then write the log-likelihood of the assimilated data as follows.

$$\mathcal{L}(\theta) = \sum_{j=1}^{j} \mathcal{N}(y_j; H\hat{m}_j, H\hat{C}_j H^\top + \Gamma) \tag{14}$$

Since the forecast moments depend on the particles, which depend on θ, then the log likelihood is implicitly a function of θ. To this end, we can now differentiate this mapping using the standard auto-differentiation toolbox and update the θ iteratively to estimate the most likely value.

4 Experimental Results

In this section, we compare the performance of the three algorithms discussed in Section 3 on assimilating the two-scale L-96 system from noisy observations of the slow scales. In particular, we consider several different degrees of observation noise. The observation noise is modeled as an isotropic mean-zero Gaussian, and we perform several experiments varying the variance of this Gaussian. Our neural network-based models are trained on a trajectory of 5000 points and tested on a withheld test trajectory of 5000 points started from a different initial condition.

Our numerical results are presented in Tables 3 and 2, and Figure 5. We measure the accuracy of the filtering methods by measuring the 2-expected energy score (2-ESE) of the predictions from their analysis steps. Note that this is equivalent to the mean squared error.

We observe that EnKF + NN tends to outperform the baseline ensemble Kalman filter in terms of 2-ESE over the test trajectory. As observation noise increases, we observe that the EnKF + NN performs similarly compared to EnKF. We attribute this to overfitting of the noise in the training trajectory. As the observation noise increases, it may be difficult for the neural network to learn a useful signal to correct the systematic bias of the forward model.

We also observe that AD-EnKF consistently outperforms EnKF. Figure 2 illustrates the average RMSE over all variables as a function of time and Figures 3 and 4 show the absolute error between the prediction and analysis and the true state. We observe that in both cases, the errors are lower than the regular EnKF. However, the Autodifferentiable EnKF also tends to underperform compared to EnKF + NN, particularly in the forecast phase (which is the only phase of each method that contains any learning). We speculate that this may be caused by overfitting to the noise. AD-EnKF

Observation noise	Ensemble Kalman Filter	EnKF + NN	Autodifferentiable EnKF
0.1	9.318	3.088	**2.905**
0.5	7.767	**2.600**	6.300
1.0	8.279	**3.207**	8.090
2.0	11.424	**4.923**	11.420

Table 2: 2-expected energy scores between the ensemble **prediction** mean and the true slow-scale states of L-96 at a variety of observation noise levels. The squared error is averaged over a testing trajectory.

Observation noise	Ensemble Kalman Filter	EnKF + NN	Autodifferentiable EnKF
0.1	1.612	0.593	**0.557**
0.5	0.245	**0.234**	0.242
1.0	0.970	**0.969**	0.970
2.0	4.000	4.000	4.000

Table 3: 2-expected energy scores between the ensemble **analysis** mean and the true slow-scale states of L-96 at a variety of observation noise levels. The squared error is averaged over a testing trajectory.

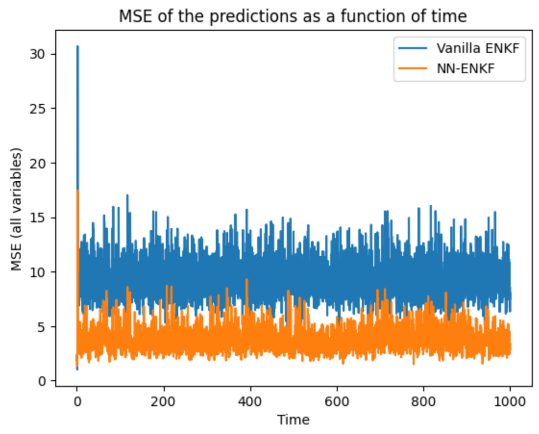

Figure 2: Average MSE across all components (Autodifferentiable EnKF and EnKF).

only outperforms EnKF for the low noise regime, although this regime appears slightly anomalous because all three models perform worse than expected (compared to noise scale 0.5) there. We speculate that perhaps the larger term in the covariance inverse may be regularizing.

From Figure 5, we see observe that the NN-EnKF forecast state is slightly closer to the true slow-scale components than the EnKF-predicted state. However, the AD-EnKF forecast state is much closer to the true slow-scale components than either the forecast state of EnKF or that of NN-EnKF. This is attributable to the neural network correction. This consequently leads to analysis results that are closer to the truth for NN-EnKF than for the baseline EnKF, and closer for AD-EnKF than for either of these.

4.1 Implementation details

In our experiments, we set the number of particles in every EnKF to be $N = 50$. We also set the model noise to be drawn from an isotropic Gaussian $\xi_j \sim \mathcal{N}(0, 0.1I)$, where I denotes the identity matrix. Given a particular initial condition v_0, we initialize the location of each particle in the ensemble by the isotropic Gaussian $\mathcal{N}(v_0, I)$. In our experiments, we set the length-scale of the localization matrix to be $\ell = 20$, and we set the inflation parameter to $\alpha = 1$. We discuss our observations with these hyperparameters in more detail in Section 5.

For NN-EnKF, we use a simple feedforward neural network with three layers: one mapping the input $\mathbb{R}^K \to \mathbb{R}^{4K}$, one mapping the hidden dimension $\mathbb{R}^{4K} \to \mathbb{R}^{4K}$, and one mapping the hidden dimension to the output dimension $\mathbb{R}^{4K} \to \mathbb{R}^K$. We use ReLU nonlinearities between the layers. We train the neural network for 3 epochs with a learning rate of 10^{-3} and weight decay of 10^{-2}.

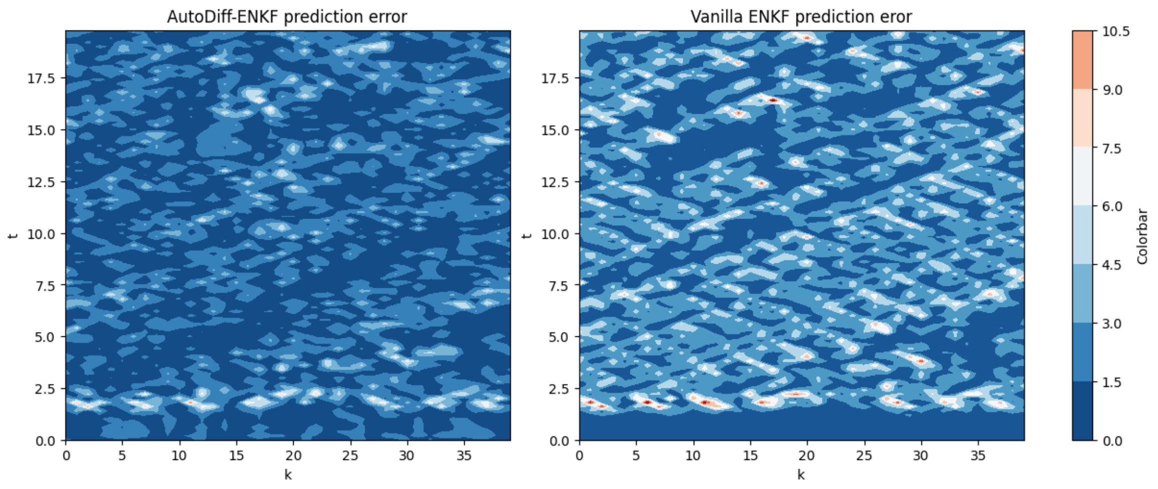

Figure 3: Absolute error of the prediction step (Autodifferentiable EnKF and EnKF).

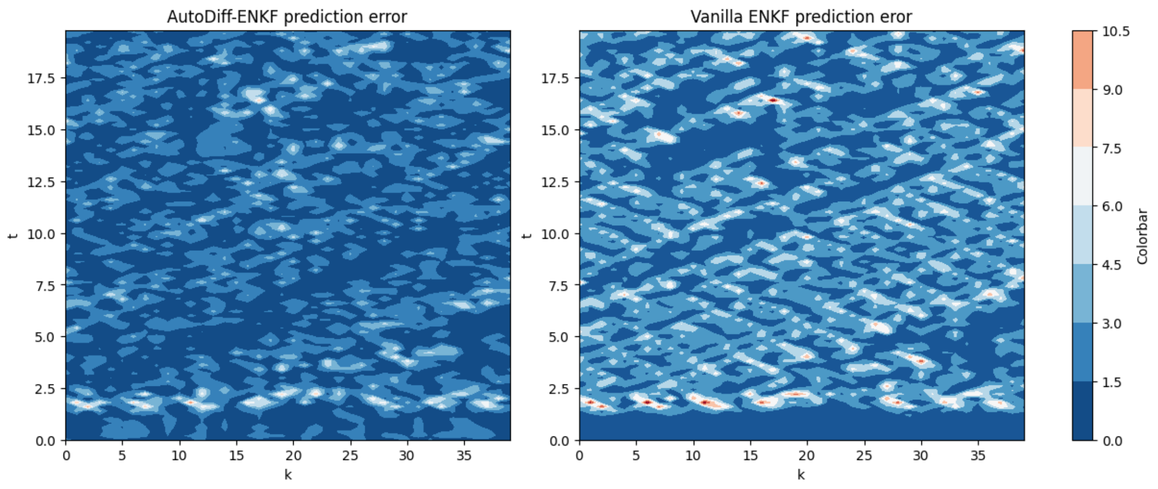

Figure 4: Absolute error of the analysis step (Autodifferentiable EnKF and EnKF).

For AD-EnKF, following Chen et al. [2022], we use a simple feedforward neural network with the same architecture as for NN-EnKF to learn a correction term to the slow-scale L-96 forward model. This is trained end-to-end by backpropagating through the log-likelihood. We train using the same learning rate and weight decay as for NN-EnKF. The model for noise scales $0.1, 0.5, 1.0, 2.0$ are trained for 25, 17, 12, and 12 epochs, respectively.

5 Discussion and Conclusion

In this project, we compared two machine learning data assimilation methods with a baseline ensemble Kalman filter at correcting systematic error in the forward model for the two-scale Lorenz-96 system. We also compared the efficacy of these models under variations in observation noise. Our numerical results can be found in Section 4. In our experiments, we observed a variety of empirical nuances that we discuss in more detail in this section.

In the past, several works have explored the problem of learning evolution operators of Markovian systems [Levine and Stuart, 2022, Li et al., 2022]. In particular, the effect of learning the time-evolution operator of chaotic systems under different temporal sampling rates has been explored [Levine and Stuart, 2022, Li et al., 2022]. These works have empirically observed that, fixing a learning method, there is an optimal time step that minimizes the error in learning the evolution operator of chaotic systems. Intuitively, for time steps that are too small, data-driven models often converge to the identity map, a subotpimal local minimum. For large time steps, data-driven models struggle to learn the evolution of chaotic systems due to their large Lyapunov exponents.

In this project, we observed similar behavior when varying the sampling time interval `si`. For small values of `si`, such as 0.1, we observed that the neural network-corrected EnKF under-performed compared to the baseline EnKF. For large values of `si`, such as 1.0, we observed difficulties in model convergence and nearly identical 2-expected energy scores between NN-EnKF and the baseline EnKF. Figure 6 qualitatively compares the predicted states after forecasting and analysis for both EnKF and NN-EnKF for a large temporal sampling time step of 1.0.

Observe that the NN-EnKF predicted state from forecasting is very far from the true state of the slow-scale components of the system. In particular, it is farther from the truth than the EnKF predicted state using the slow scale dynamics Ψ_s. This suggests that the neural network is unable to correct the dynamics of the chaotic (and partially-observed) system, due to the large time step on which it is learning. As such, in our experiments, we set $\texttt{si} = 0.2$, a middle-ground value that is farther from the pathological edge cases.

We also observed the importance of using a sufficiently long trajectory for training. We found that using training trajectories that did not contain sufficient points led to overfitting in the neural network. As such, we made both the training and testing trajectories sufficiently long to avoid such issues. Note that this issue is not very serious in practice because one can always increase the number of EnKF particles to obtain more training data for the neural network.

Next, we also observe that localization plays a significant role in improving the performance of these models. Using an observation noise scale of 1.0, the 2-ESE error of EnKF decreased from approximately 3.5 to approximately 0.5 by introducing localization. In contrast, we observed that inflation did not provide a significant improvement in performance.

In our preliminary experiments we observe that the neural network for EnKF + NN tends to overfit to the training trajectory. We were able to address this issue by performing *early stopping* and only training for 3 epochs. We also incorporated weight decay as a regularizer. For AD-EnKF, we noticed that training for more epochs led to better performance.

References

D. Balwada, R. Abernathey, S. Acharya, A. Adcroft, J. Brener, V. Balaji, M. A. Bhouri, J. Bruna, M. Bushuk, W. Chapman, et al. Learning machine learning with lorenz-96. *Authorea Preprints*, 2023.

Y. Chen, D. Sanz-Alonso, and R. Willett. Autodifferentiable ensemble kalman filters. *SIAM Journal on Mathematics of Data Science*, 4(2):801–833, 2022.

A. Farchi, P. Laloyaux, M. Bonavita, and M. Bocquet. Using machine learning to correct model error in data assimilation and forecast applications. *Quarterly Journal of the Royal Meteorological Society*, 147(739):3067–3084, 2021. doi:https://doi.org/10.1002/qj.4116. URL `https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.4116`.

M. Levine and A. Stuart. A framework for machine learning of model error in dynamical systems. *Communications of the American Mathematical Society*, 2(07):283–344, 2022.

Z. Li, M. Liu-Schiaffini, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Learning chaotic dynamics in dissipative systems. *Advances in Neural Information Processing Systems*, 35:16768–16781, 2022.

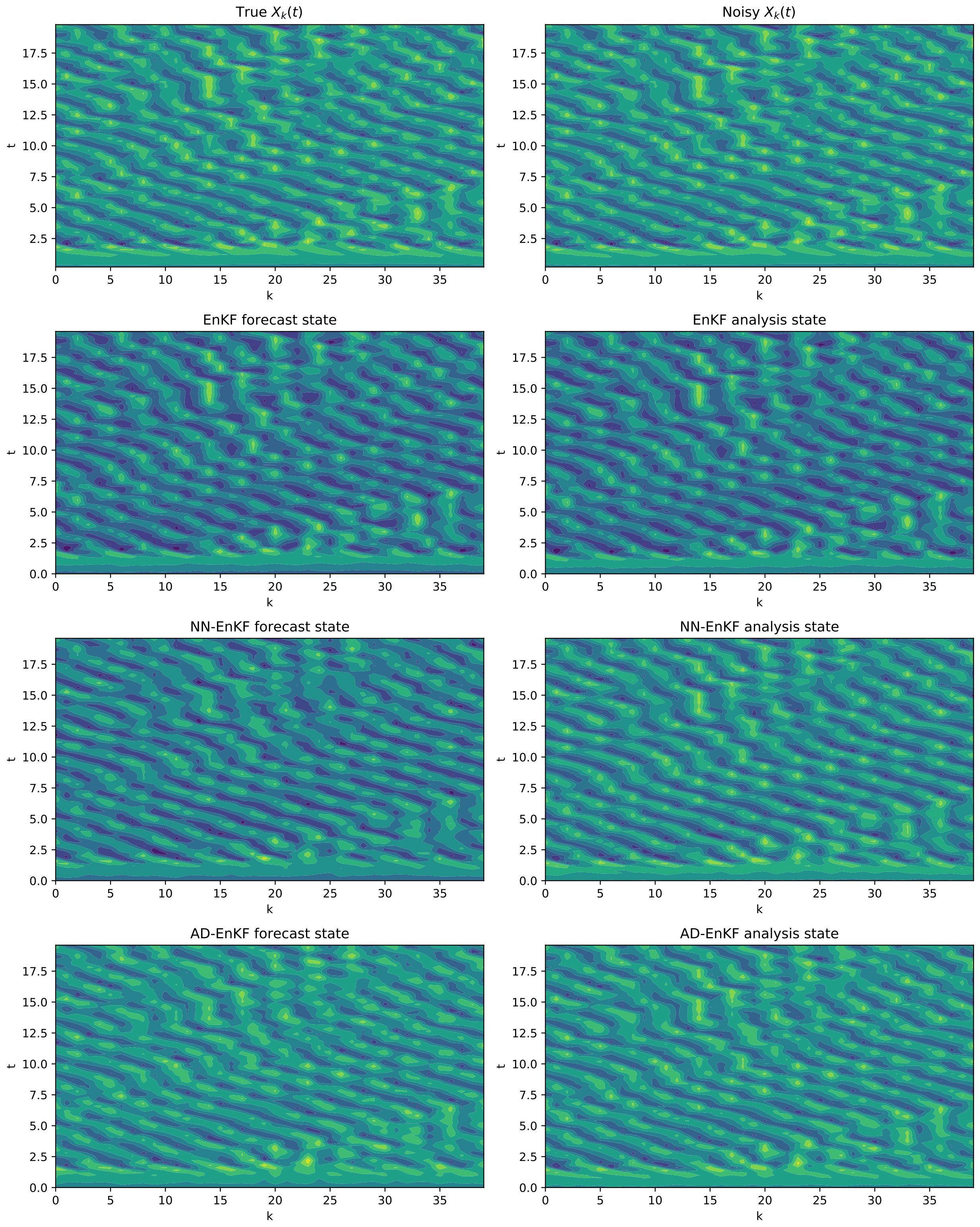

Figure 5: Visualizations of the slow-scale states of a testing trajectory, along with the predictions and analysis results of the methods explored. The noise in the noisy observations are generated from an isotropic standard Gaussian.

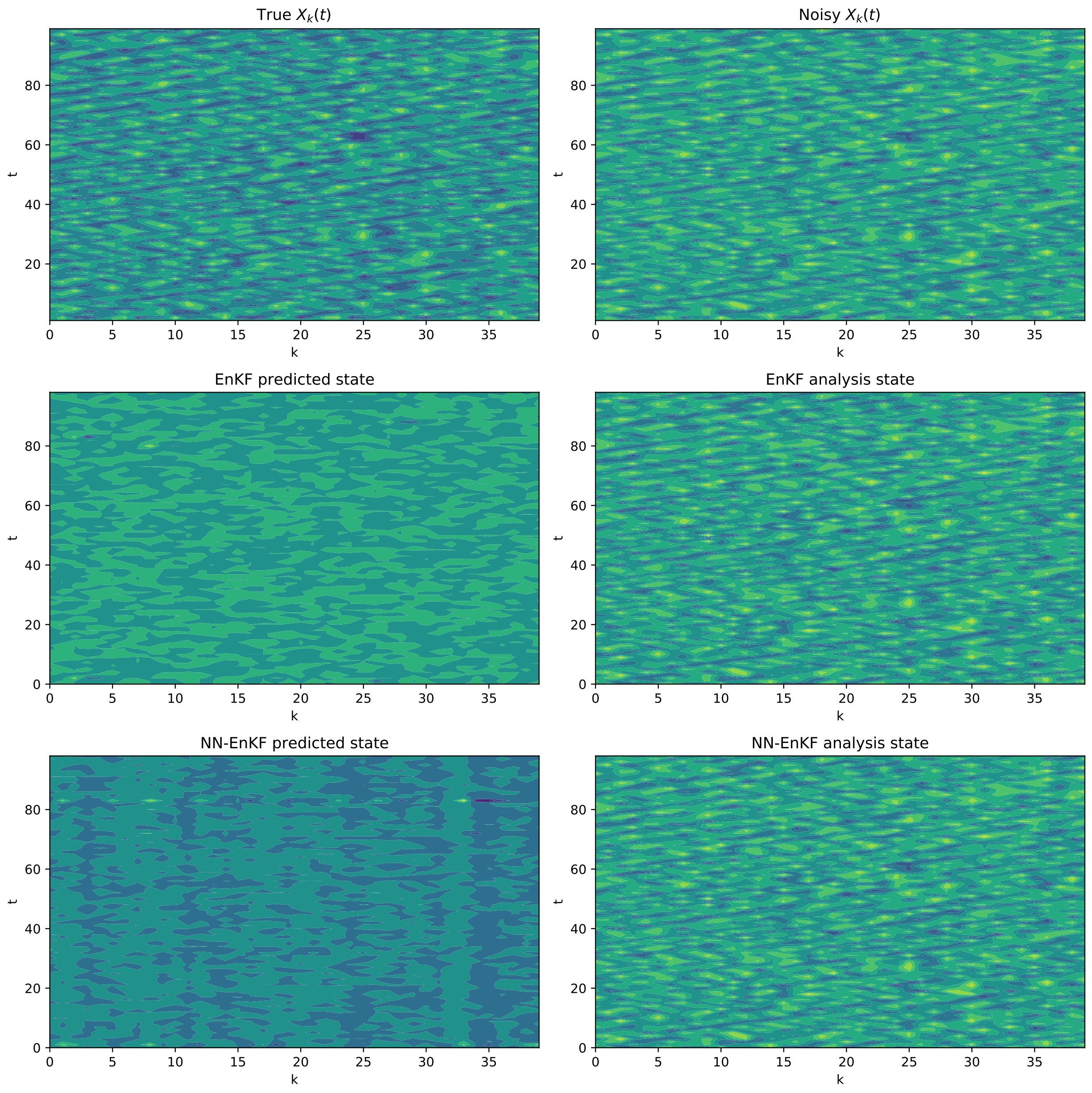

Figure 6: Comparison between baseline EnKF and neural network-corrected EnKF for a large temporal sampling rate of $\texttt{si} = 1.0$. Observe that the NN-EnKF predicted state after forecasting does not resemble the true state.