Variational Data Assimilation with a Learned Inverse Observation Operator

Thomas Frerix [1 2] **Dmitrii Kochkov** [1] **Jamie A. Smith** [1] **Daniel Cremers** [2]
Michael P. Brenner [1 3] **Stephan Hoyer** [1]

Abstract

Variational data assimilation optimizes for an initial state of a dynamical system such that its evolution fits observational data. The physical model can subsequently be evolved into the future to make predictions. This principle is a cornerstone of large scale forecasting applications such as numerical weather prediction. As such, it is implemented in current operational systems of weather forecasting agencies across the globe. However, finding a good initial state poses a difficult optimization problem in part due to the non-invertible relationship between physical states and their corresponding observations. We learn a mapping from observational data to physical states and show how it can be used to improve optimizability. We employ this mapping in two ways: to better initialize the non-convex optimization problem, and to reformulate the objective function in better behaved physics space instead of observation space. Our experimental results for the Lorenz96 model and a two-dimensional turbulent fluid flow demonstrate that this procedure significantly improves forecast quality for chaotic systems.

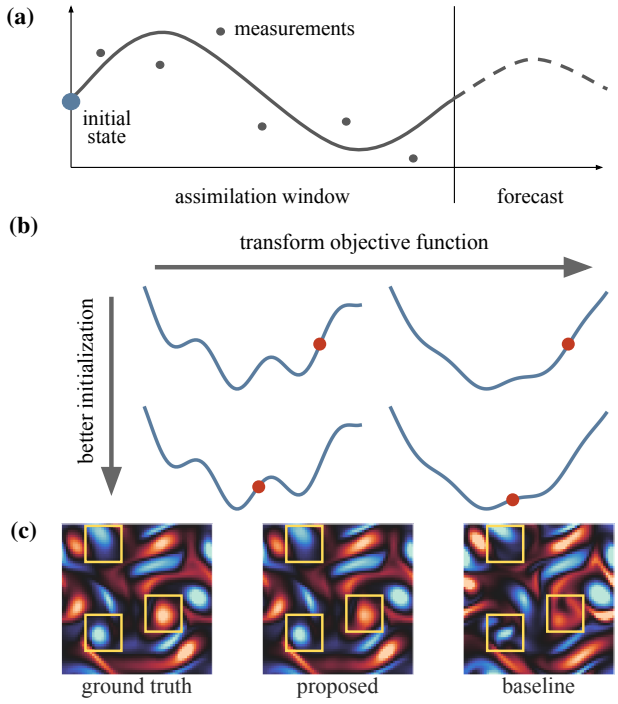

Figure 1. Overview of the proposed method. **(a)** The principle of variational data assimilation. The goal is to optimize for an initial state (large blue dot) of a physical system such that the evolution over an assimilation window (solid line) is close to measurements (small gray dots). The model is subsequently used to make predictions into the future (dashed line). **(b)** Improving optimizability of variational data assimilation. We use a learned inverse observation operator to better initialize the optimization problem (red dots) and to transform the non-convex objective function to be better behaved. **(c)** Data assimilation results of the proposed method compared with a traditional algorithm. Depicted is a vorticity prediction of a two-dimensional turbulent fluid flow. The proposed method more accurately captures vorticity features (yellow squares).

1. Introduction

Variational data assimilation provides the basis for numerical weather prediction (ECMWF, 2019), integrating the non-linear partial differential equations describing the atmosphere. The core algorithm is an optimization problem for the initial state of the system, such that when the equations of motion are evolved over time, the resulting trajectories are close to the measurements. Continuing to evolve the physical system into the future then yields a forecast (Figure 1a). Over the last decades, these algorithms have led to a steady improvement in forecast quality, though further improvements are limited by computational resources. Data assimilation accounts for a significant fraction of the computational cost for numerical weather prediction. This restricts the amount of data that can be assimilated and only a small volume of available satellite data is utilized for operational forecasts (Bauer et al., 2015; Gustafsson et al., 2018).

[1]Google Research [2]Technical University of Munich [3]Harvard University. Correspondence to: Thomas Frerix <thomas.frerix@tum.de>, Stephan Hoyer <shoyer@google.com>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

Weather forecasting systems are complex algorithmic pipelines (Bauer et al., 2015). Recent work has shown that in some cases forecasts can be improved by replacing the entire system with a machine learned prediction (see, e.g., (Sønderby et al., 2020; Ham et al., 2019)). This approach is very powerful, but physical models remain more accurate for global weather forecasting (Rasp et al., 2020). Moreover, they offer guarantees of generalization, interpretability and physical consistency because they are built upon well-known physical principles. In fact, some of the best pure machine learning approaches rely upon pre-training with simulation data due to insufficient historical observations (Rasp & Thuerey, 2020; Ham et al., 2019). Additionally, physical modeling facilitates the principled coupling of processes on different characteristic spatial and time scales, e.g., the atmosphere, ocean, and land surfaces, which is critical for complex forecasting applications (Bauer et al., 2015). Consequently, a more promising approach may be to *augment* physical models with machine learning (Watt-Meyer et al., 2021; Kochkov et al., 2021).

In this paper, we augment a traditional variational data assimilation algorithm with machine learning. We use the equations of motion to evolve the dynamical system, while machine learning is used only to improve the optimization problem for calculating the initial state. To this end, we learn an approximate inverse to the observation operator. Using this mapping, we provide an effective initialization scheme for the non-convex optimization problem and transform the objective function for the variational data assimilation problem to be better behaved (Figure 1b). We generate observational data from two model problems, a classical model for data assimilation introduced by Lorenz (Lorenz, 1995), and a turbulent fluid flow in two spatial dimensions. We demonstrate that the algorithm enhanced with machine learning leads to a substantial performance improvements over the baseline. Figure 1c shows an example of an improved forecast.

2. Related Work

Data assimilation is a suitable formalism for combining physical modeling with machine learning since large scale applications are characterized by rich physics and large amounts of data. Both approaches can be viewed in the framework of Bayesian inference (Geer, 2020). Machine learning approaches to modify the physical model for data assimilation include a learned correction to an approximate model (Farchi et al., 2020; Brajard et al., 2020a), training a machine learning model to completely emulate the physics (Brajard et al., 2020b), and learning a forcing term within the weak-constraint 4D-Var formulation (Bonavita & Laloyaux, 2020). In contrast, we use an exact physical model and modify the representation of observations using machine learning. Mack et al. (2020) formulate variational data assimilation in a latent space derived by training an autoeconder. The dimensionality reduction allows for significantly faster optimization. However, this approach loses physical guarantees for decoded states.

Integration of dynamical systems is a central component of data assimilation. However, simulating high-resolution dynamics quickly becomes computationally intractable. To ameliorate this issue, several recent works combine traditional numerical solvers with machine learning to obtain high-resolution physics from coarser simulations. MeshfreeFlowNet (Jiang et al., 2020) continuously parameterizes the spatial domain by learning an interpolation function for each grid cell. Um et al. (2020) incorporate a correction operator directly into the numerical solver and train this function to nudge an inaccurate solution towards a more accurate one. The authors of (Bar-Sinai et al., 2019; Zhuang et al., 2020) learn a discretization scheme for PDEs that better captures the unresolved physics, leading to improvements over ad hoc finite difference discretization methods. Using a fully differentiable computational fluid solver, Kochkov et al. (2021) demonstrate that with this approach the grid resolution can be reduced by an order of magnitude without sacrificing accuracy. Similarly, we use a fully differentiable solver for our model systems and our approach may therefore be complemented by such ideas.

Variational data assimilation requires solving a difficult optimization problem. Our approach of improving optimizability of this problem with machine learning can be contextualized with other works that employ machine learning to transform a physics constrained optimization problem. In the context of simulating mechanical materials, Beatson et al. (2020) approximate the inner problem of a bi-level optimization problem by a learned function, thus crucially reducing the the computation cost. To optimize photonic device designs, Kudyshev et al. (2021) train for a compressed design space with an adversarial autoencoder. This space is then explored using an evolutionary algorithm. Ackmann et al. (2020) learn a preconditioner to improve the solution of a linear system arising during the integration of a shallow-water model. As with our approach, the preconditioned system does not suffer from generalization issues of the machine learning model. We can guarantee a certain performance level by defaulting to a classical method. Various works improve optimization problems not with a component learned from training data, but by reparameterizing the objective function with a neural network architecture. The neural network here acts as an overparameterization with a specific inductive bias, e.g., convolutional neural networks for building hierarchical, multi-scale representations (Hoyer et al., 2019; Ulyanov et al., 2018) or fully-connected networks for continuous representations (Mildenhall et al., 2020).

3. Variational Data Assimilation

The state of the art variational data assimilation algorithm is called 4D-Var (Bannister, 2017). It minimizes an objective function of the form

$$\begin{aligned} J(x_0) =& (x_0 - x^b)^T \mathbf{B}^{-1} (x_0 - x^b) \\ &+ \sum_{t=0}^{T} (\mathcal{H}(x_t) - y_t)^T \mathbf{R}^{-1} (\mathcal{H}(x_t) - y_t) \\ x_{t+1} =& \mathcal{M}(x_t). \end{aligned} \tag{1}$$

The goal is to produce a maximum likelihood estimate of the initial state x_0 of a trajectory $(x_1, \ldots, x_T)$ that is evolved through a physics model $\mathcal{M}$, given a sequence of observations $(y_1, \ldots, y_T)$. The observation operator $\mathcal{H}$, maps physical states into the space of observations. As an example, the physics model could be the Navier Stokes equations for evolving a weather system, and the observation operator could measure the state of the atmosphere at discrete weather stations. The loss $J(x_0)$ models the initial condition and conditional distribution of observations as a multi-variate normal distribution. The first term incorporates a guess for the initial state x_0 (the so-called background state x^b), where $\mathbf{B}$ is the background covariance matrix, representing the uncertainty about this assumption, i.e., $x_0 \sim \mathcal{N}(x^b, \mathbf{B})$. Similarly, the matrix $\mathbf{R}$ models the observation error covariance, i.e., $y_t \sim \mathcal{N}(\mathcal{H}(x_t), \mathbf{R})$. The simplifying assumption of Gaussian background and observation errors may suffer from model misspecification when applied to real-world data (Bocquet et al., 2010). Altogether, this amounts to solving a non-linear least squares problem. We denote the initialization of this optimization problem by *initial condition* and refer to an *initial state* to describe the first state x_0 of a physics trajectory.

4D-Var minimizes objective functions of the form (1) via gradient based optimization. To forecast a trajectory $(x_1, \ldots, x_{T'})$, the objective (1) is minimized to estimate x_0 over a fixed-length window of observations, the so-called assimilation window, which is shifted in time. The forecast state from a previous assimilation window becomes the background state for the next assimilation window. Minimizing (1) is a difficult optimization problem for various reasons (Andersson et al., 2005): First, the physics model $\mathcal{M}$ is in general non-linear or even chaotic, so that small changes in the initial state can lead to large changes in an integrated state. Secondly, the observation operator $\mathcal{H}$ that reduces information from physics trajectories to observations is usually non-invertible and possibly non-linear.

In what follows, we focus on the difficulty posed by the observation operator $\mathcal{H}$, and learn an approximate inverse to $\mathcal{H}$ that maps the observational data to the space of physical states. For simplicity, we focus our analysis on a fixed time horizon without a shifting assimilation window. Moreover, we neglect prior modeling of the initial state, so that we omit the first term in (1). Finally, we neglect an explicit observation noise model, i.e., we set $\mathbf{R}$ to be the identity matrix. This amounts to studying the following simplified version of the 4D-Var model:

$$J(x_0) = \sum_{t=0}^{T} ||\mathcal{H}(x_t) - y_t||_2^2, \qquad x_{t+1} = \mathcal{M}(x_t) \tag{2}$$

The method presented in this paper is not restricted to this setting, but equally applies to the general 4D-Var problem. However, to study the effect of the observation operator $\mathcal{H}$ on the optimization problem, additional aspects of the problem are not necessary.

4. Learning an Inverse Observation Operator

To be precise, we distinguish the space $\mathcal{P}$ of physical states or *physics space* and the space $\mathcal{O}$ of observations or *observation space*. The observation operator $\mathcal{H} : \mathcal{P} \to \mathcal{O}$ maps the physics space $\mathcal{P}$ to the observation space $\mathcal{O}$. The variational data assimilation objective (2) is formulated in observation space. However, the non-invertibility (and potential non-linearity) of $\mathcal{H}$ makes minimizing this objective difficult. The key idea of this paper is to parameterize an approximate inverse $h_\theta : \mathcal{O} \to \mathcal{P}$ and to use machine learning to train the parameters θ. The training target is to map observations to corresponding physical states, in our notation to obtain $h_\theta(y_t) \approx x_t$. While in practice there is ample training data from historical observations, in this work we revert to simulations for generating training data.

In order to exploit both spatial and temporal correlations, we construct a fully-convolutional architecture in space and time. Fully-convolutional architectures are natural for several reasons: they use local filters and therefore enforce the locality of the underlying equations of motion. Additionally, the number of parameters in a convolutional layer does not increase with input size. This is vital because typical physics models $\mathcal{M}$ are discretized over large grids.

We implement our models for the approximate inverse in JAX (Bradbury et al., 2018) and use Flax as neural network library, with the Adam optimizer (Kingma & Ba, 2015) and learning rate of 10^{-3} for training[1]. We also use JAX to implement differentiable simulators for the physics model $\mathcal{M}$ that arises inside the objective function (2) (Kochkov et al., 2021). Operational numerical weather prediction models similarly make use of differentiable simulators, where they are known as adjoint models. All models can be trained and optimized on a single NVIDIA V100 GPU.

We use the trained inverse observation operator for two aspects of the optimization problem. First, we map the

[1] https://github.com/googleinterns/invobs-data-assimilation

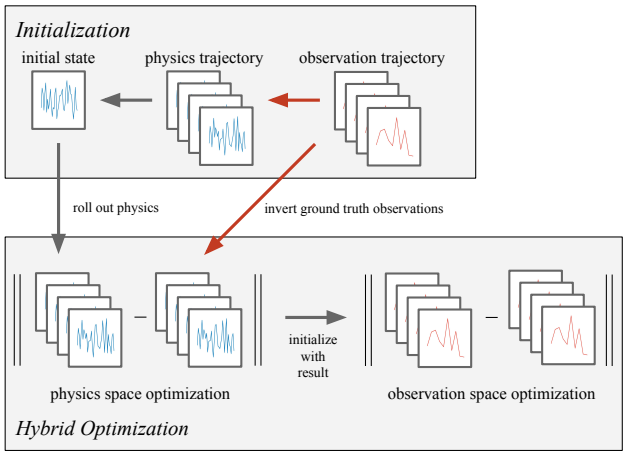

Figure 2. Variational data assimilation with a learned inverse observation operator. The learned inverse observation mapping is denoted by red, hollow arrows. We approximately invert an observation trajectory and choose its first state as an *initialization* of the non-convex optimization problem. The *hybrid optimization* approach first minimizes (3) in physics space and subsequently uses the optimization result to initialize refinement minimization of (2) in observation space.

earliest trajectory of observations to a trajectory in physics space and then use its first state as an initialization to the optimization problem. Secondly, we substitute (2) with a reformulated objective function in physics space:

$$\tilde{J}(x_0) = \sum_{t=0}^{T} ||x_t - h_\theta(y_t)||_2^2, \qquad x_{t+1} = \mathcal{M}(x_t) \quad (3)$$

An overview of this method is depicted in Figure 2. The objective functions (2) and (3) are not equivalent, rather we use (3) as a proxy for (2). As we will show in Section 5, minimizing (3) is a more benign optimization problem compared to minimizing (2). However, the caveat with minimizing (3) is that we can only expect h_θ to be an approximation that does not even guarantee to map to a physical state, i.e., a state that one could encounter under the statistically stationary dynamics of the model. As a consequence, we adopt a *hybrid* approach where we first minimize (3) and use the optimization result to initialize minimizing (2) for further refinement.

5. Results

We demonstrate this approach on two chaotic dynamical systems, the Lorenz96 model (Lorenz, 1995) and Kolmogorov flow (Chandler & Kerswell, 2013). As our baseline, we follow a common approach (Bannister, 2008) and assimilate in observation space over a set of uncorrelated variables, i.e., we minimize (2) after a whitening transformation to $\xi = C^{-1/2}x$, where C is the empirical covariance matrix over a set of 10^6 independent samples from the stationary distribution of the respective dynamical systems. The empirical covariance matrix might not be positive definite, an issue that is often encountered in applications (Tabeart et al., 2020). To ensure positive definiteness, we threshold the spectrum of C at 10^{-6}. To be precise, we solve

$$\min_{\xi_0} \sum_{t=0}^{T} ||\mathcal{H}(C^{1/2}\xi_t) - y_t||_2^2, \; \xi_{t+1} = C^{-1/2}\mathcal{M}(C^{1/2}\xi_t) \; . \quad (4)$$

We use L-BFGS (Nocedal & Wright, 2006) as an optimizer for assimilation, retaining a history of 10 vectors for the Hessian approximation. An optimization step in physics and observation space incurs a comparable computational cost, since the inverse observation operator is applied *prior* to optimization to modify the fitting targets.

We measure the quality of forecasts by an L_1 point-wise error metric ε between two states z_1, z_2:

$$\varepsilon(z_1, z_2) \coloneqq \|z_1 - z_2\|_1/\gamma \; , \quad (5)$$

where we scale this metric to a *relative* error by dividing by a mean error γ of random independent states sampled from the stationary distribution of the dynamical system. This metric can be easily interpreted: an order unity error implies the average performance of a random evolution of the system. We compare the optimized forecasts with the evolution from a ground truth initial state on a set of 100 test trajectories.

5.1. Lorenz96 Model

The single-level Lorenz96 model (Lorenz, 1995) is a periodic, one-dimensional model where each grid point is evolved according to the equation of motion

$$\frac{dX_k}{dt} = -X_{k-1}(X_{k-2} - X_{k+1}) - X_k + F \; . \quad (6)$$

Here, the first term models advection, the second term represents a linear damping, and F is an external forcing. We choose a grid of size $K = 40$ and an external forcing $F = 8$, parameters where the system is chaotic with a Lyapunov time of approximately 0.6 time units. For an observation operator, we use subsampling. We integrate trajectories over an assimilation window of $T = 10$ time steps with a time increment of $\Delta t = 0.1$ time units starting from an initial condition in the statistically stationary regime, i.e., where $\sum_k X_k^2$ fluctuates around a constant mean value.

We now demonstrate how a learned inverse observation operator significantly improves forecast results by providing an effective initialization scheme for the non-convex optimization problem and by formulating a more benign objective function in physics space $\mathcal{P}$ instead of observation space $\mathcal{O}$.

LAYER	(T, X, C)
INPUT	(10, 10, 1)
CONV2D + BN + SILU	(10, 10, 128)
UPSAMPLE + CONV2D + BN + SILU	(10, 20, 64)
UPSAMPLE + CONV2D + BN + SILU	(10, 40, 32)
CONV2D + BN + SILU	(10, 40, 16)
CONV2D	(10, 40, 1)

Table 1. Fully-convolutional network for training the inverse observation operator for the Lorenz96 model. The table shows a layer with its respective output array dimensions time (T), space (X), and channel (C). The CONV2D layer applies periodic convolution in the space dimension and zero-padded convolution in the time dimension. The filter size for all convolutional layers is $(3, 3)$. BN denotes batch normalization. To upscale the grid by a factor of 2 in layers two and three, we use cubic interpolation. As a non-linearity we use the sigmoid-weighted linear unit (SILU), $\mathrm{silu}(x) = x/(1 + \exp(-x))$.

As an observation operator for the following experiments, we observe every 4th grid point. To approximate the inverse observation operator, we train a fully-convolutional network as described in Table 1. We train on a dataset of 32000 independent observation trajectories with batch size 8 for 500 epochs.

For data assimilation, we compare two initialization schemes. The baseline *averaging initialization* scheme initializes the optimizer with the observed grid points and uses the average over a data set of independent states as an estimate of the unobserved grid points. This is equivalent to a least-squares fit of the unobserved grid points. The *inverse initialization* scheme uses the learned inverse observation operator to create the initialization. To this end, we map a sequence of observations to a physical trajectory and use its first state for initialization as depicted in Figure 2. Figure 3 shows a qualitative comparison of these two initialization schemes, demonstrating that the learned inverse mapping leads to a much more accurate initialization. We found that first optimizing for an initial condition from a previous assimilation window, as in 4D-Var, does not improve baseline initialization. We compare optimizing in observation space (baseline) with the hybrid approach of first optimizing in physics space and then refining the results in observation space. For a fair comparison, both optimization methods are limited to 500 optimization steps with the hybrid method assigning 100 of these steps to optimization in physics space and the remaining 400 steps to refinement in observation space. The forecast results are shown in Figure 4. Inverse initialization improves forecasts for observation space optimization compared with average initialization. For the inaccurate averaging initialization, hybrid optimization significantly improves forecast quality compared with observation space optimization. This suggests that by first optimizing in physics space, we obtain an initialization for refinement in observation space that is located at a favorable basin of attraction. Adding inverse initialization to the hybrid optimization approach leads to a small additional improvement.

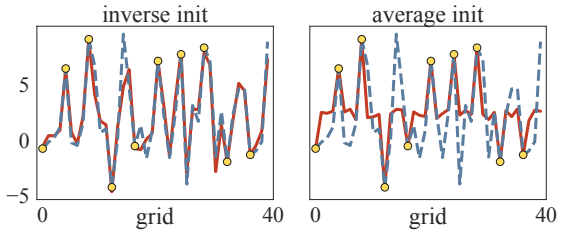

Figure 3. Comparison of initialization (solid red) with the ground truth initial state (dashed blue). The observed grid points are marked as yellow dots. The learned inverse observation mapping takes a trajectory of such subsampled states as input and generates the inverse initialization. Inverse initialization is much more accurate than averaging initialization.

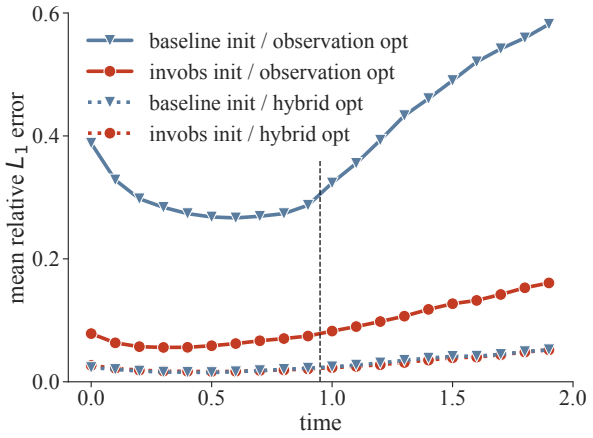

Figure 4. Forecast quality with a learned inverse observation operator for the Lorenz96 model. Quality measure is the L_1 forecast error relative to a random evolution of the system. Depicted is the mean error based on a sample of 100 trajectories. The vertical dashed line separates the assimilation window from the forecast window. Inverse initialization improves forecasts for observation space optimization compared with average initialization. For the inaccurate averaging initialization, hybrid optimization significantly improves forecast quality compared with observation space optimization. Adding inverse initialization to the hybrid optimization approach leads to a small additional improvement, which is significant with a p-value of $p < 10^{-4}$.

Figure 5 shows an example forecast of the system. The hybrid method initialized with the learned inverse mapping is able to capture the ground truth evolution of the system. In contrast, for the baseline method a visible approximation error remains throughout the system integration.

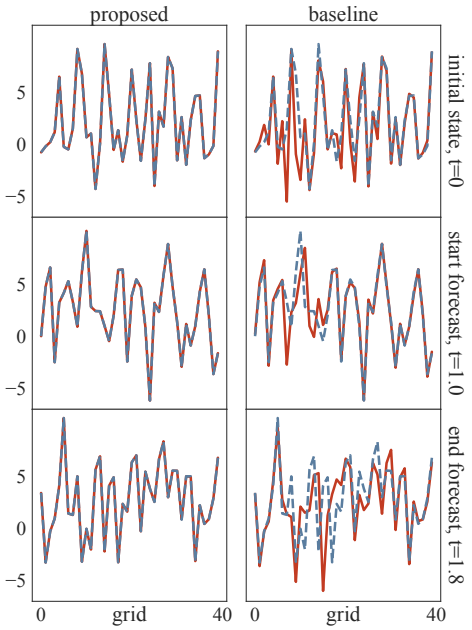

Figure 5. Forecast trajectory for the Lorenz96 model optimized from the initial conditions depicted in Figure 3. The forecast trajectory based on optimization with the inverse observation operator (inverse initialization, hybrid optimization) is qualitatively much closer to the ground truth evolution of the system than the baseline method (averaging initialization, observation space optimization).

LAYER	(T, X, Y, C)
INPUT	(10, 4, 4, 2)
CONV3D + BN + SILU	(10, 4, 4, 64)
UPSAMPLE + CONV3D + BN + SILU	(10, 8, 8, 32)
UPSAMPLE + CONV3D + BN + SILU	(10, 16, 16, 16)
UPSAMPLE + CONV3D + BN + SILU	(10, 32, 32, 8)
UPSAMPLE + CONV3D + BN + SILU	(10, 64, 64, 4)
CONV3D	(10, 64, 64, 2)

Table 2. Fully-convolutional network for training the inverse observation operator for Kolmogorov flow. The table shows a layer with its respective output array dimensions time (T), space (X and Y), and channel (C). The CONV3D layer applies periodic convolution in the two space dimensions and zero-padded convolution in the time dimension. The filter size for all convolutional layers is $(3, 3, 3)$. BN denotes batch normalization. We upsample the grid using bicubic interpolation by a factor of 2 and correspondingly halve the number of channels. As a non-linearity we use the sigmoid-weighted linear unit (SILU), $\mathrm{silu}(x) = x/(1 + \exp(-x))$.

5.2. Two-Dimensional Turbulence

Next, we study data assimilation for a two-dimensional turbulent fluid (Boffetta & Ecke, 2012). Machine learning in this setting requires modeling richer physics and poses a much more demanding computational problem. Furthermore, having in mind the application of data assimilation to numerical weather prediction, this class of models can be considered as the simplest approximation to modeling the flow of the atmosphere.

We consider the incompressible Navier-Stokes equation for a velocity field $\mathbf{u}$ and a pressure field p:

$$\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\nabla\mathbf{u} - \nu\nabla^2\mathbf{u} + \nabla p - \mathbf{F} = 0 \qquad (7)$$
$$\nabla \cdot \mathbf{u} = 0 \,,$$

where ν is the kinematic viscosity of the fluid. We choose the external forcing $\mathbf{F}$ to correspond to Kolmogorov flow (Chandler & Kerswell, 2013), with linear damping (Boffetta & Ecke, 2012) to ensure that the long-time behavior of the solution is statistically stationary:

$$\mathbf{F} = \sin(kx)\hat{\mathbf{x}} - \alpha\mathbf{u} \qquad (8)$$

For our experiments, we choose a domain $[0, 2\pi]^2$ with periodic boundary conditions, a wavenumber $k = 4$, a damping coefficient $\alpha = 0.1$, and a viscosity of $\nu = 10^{-2}$. We discretize the solution on a 64×64 grid and use standard numerical methods to solve the Navier-Stokes equation with a differentiable solver written in JAX (Kochkov et al., 2021). The Lyapunov time of the system is approximately 5.9 time units. Our flows are initialized with a random velocity field filtered with a spectral filter at a peak wavenumber 4, which is then integrated to a statistically stationary regime of the flow. We assimilate over trajectories of length $T = 10$ time steps, where the integration time between two such snapshots is $\Delta t \approx 0.18$, consisting of 25 internal solver integration steps. To save memory when computing gradients, we checkpoint the state from the forward pass only at each internal integration step rather than storing intermediate values (Griewank, 1994). This requires evaluating the forward pass twice, but reduces memory usage by two orders of magnitude.

We carry out data assimilation on the velocity field $\mathbf{u}$ of the flow. To analyze our forecasts, we use vorticity ω, the curl of the velocity field,

$$\omega := \left(\frac{\partial \mathbf{u}_y}{\partial x} - \frac{\partial \mathbf{u}_x}{\partial y} \right) \,. \qquad (9)$$

Vorticity describes the local direction of movement of the fluid. We visualize vorticity on a scale, which is cut off at $[-8, 8]$ with negative values (blue in Figures 6 and 9) denoting clockwise rotation and positive values (red in Figures 6 and 9) denoting counter-clockwise rotation.

We again use equidistant subsampling for the observation operator $\mathcal{H}$. In contrast to the Lorenz96 model the solution is smooth over the grid points, so we can use bicubic interpolation between observed grid points as the baseline initialization scheme. We now demonstrate the effect of

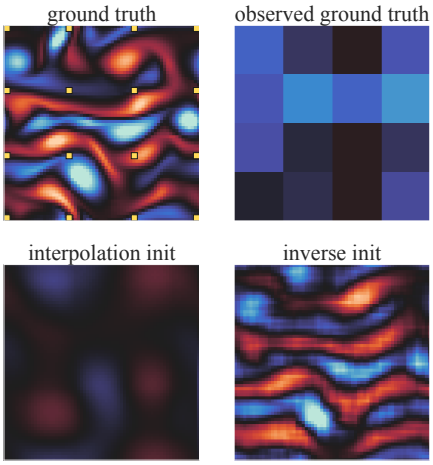

Figure 6. Comparison of initialization schemes with the ground truth initial state for Kolmogorov flow with a 16-subsampling observation operator (yellow dots). The learned inverse observation operator is trained on a trajectory of subsampled velocity fields. The observed ground truth vorticity exemplifies the amount of information of a single state of this trajectory. The trained model predicts a good approximation to the ground truth state only from a sequence of 4×4 points.

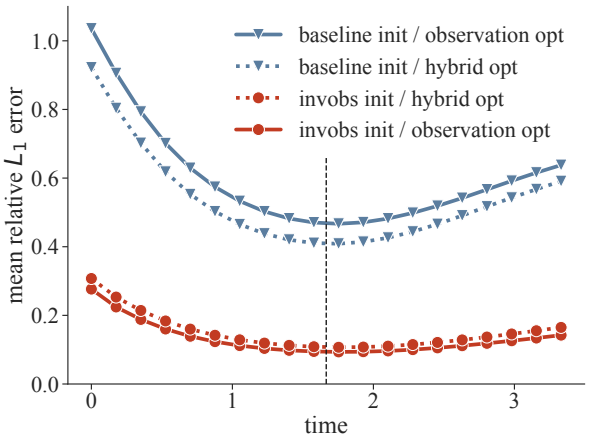

Figure 7. Data assimilation results for Kolmogorov flow using a learned inverse observation operator. Quality measure is the L_1 forecast error relative to a random evolution of the system. Depicted is the mean error based on a sample of 100 trajectories. Trajectories are obtained by evolving the initial states returned by corresponding optimization methods. Using the hybrid method for optimization improves assimilation quality with inaccurate interpolation initialization. The inverse initialization scheme significantly improves forecasts for observation space optimization. Adding inverse initialization to hybrid optimization does not improve performance. The difference between observation space optimization vs. hybrid optimization for both initialization schemes is significant with a p-value of $p < 10^{-8}$.

a learned inverse observation operator, when this operator observes every 16th grid point. For training, we employ a fully-convolutional network as shown in Table 2. We train on a dataset of 32000 independent observation trajectories with batch size 8 for 500 epochs.

For data assimilation, we compare bicubic interpolation as the baseline *interpolation initialization* with *inverse initialization* derived from the learned inverse observation operator, as depicted in Figure 2. Figure 6 compares these initialization methods with the ground truth initial state. As with the Lorenz96 model analyzed in Section 5.1, we also compare optimizing in observation space with the hybrid approach of first optimizing in physics space and subsequently refining this solution in optimization space. We limit both optimization methods to 1000 steps, with the hybrid approach using 100 of these steps to optimize in physics space and the remaining 900 to refine in observation space. Figure 7 shows the results, with three implications analogous to experiments on the Lorenz96 model. First, hybrid optimization improves assimilation quality even when using the inaccurate interpolation initialization. This implies that by first optimizing in physics space, we arrive at a favorable basin of attraction for optimization in observation space. Secondly, using inverse initialization for optimizing in observation space significantly improves forecasts. Finally, adding inverse initialization to hybrid optimization does not improve performance. This is presumably because the initialized state is already a good approximation to the optimization target, so there is no added advantage in optimizing in physics space. In contrast, it is more sensible to directly refine this state by optimizing in observation space. This effect becomes evident when analyzing how each of these optimization methods decreases the objective function in observation space during optimization, as depicted in Figure 8. For the less accurate interpolation initialization, a more favorable basin of attraction can be reached for some samples by first optimizing in physics space. However, since inverting the observation space trajectory only approximates the true trajectory, fitting against this target precludes progress after an initial phase of optimization steps. Hence, with a fixed budget of optimization steps there is a trade-off between finding a favorable basin of attraction by optimizing in physics space and finding a higher-accuracy solution by optimizing in observation space. Figure 9 qualitatively compares vorticity forecasts for the baseline method (interpolation initialization, observation space optimization) with our method based on the inverse observation operator for initialization and hybrid space optimization. The dominant features of the flow are visibly better predicted by the proposed method. Note that the structure of initial states differs from that of the following trajectories. Certain perturbations vanish quickly during the evolution of the system and are therefore not optimized to vanish in the initial state.

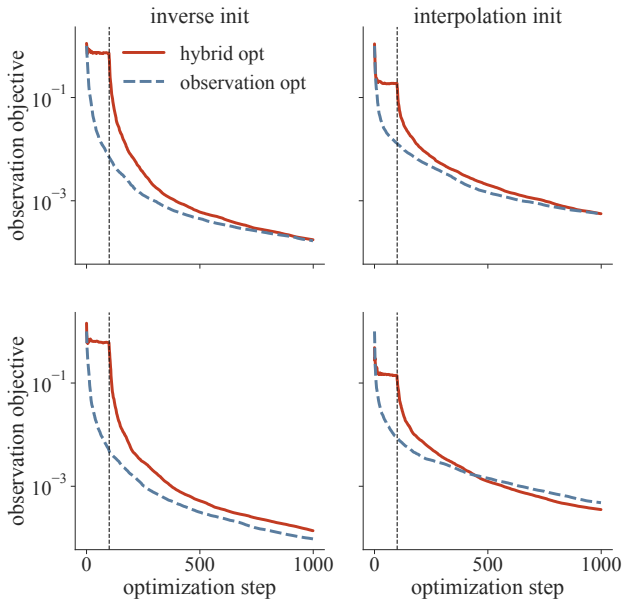

Figure 8. Observation space data assimilation objective (2) during optimization. Values are relative to the initial value on a log-scale for two different samples (rows). Depicted are observation space optimization (dashed blue) and hybrid optimization (solid red). For both methods, we evaluate the *same* observation space objective function along the optimization path. The vertical dashed line signifies the change from physics space to observation space in the hybrid method. For inaccurate interpolation initialization, a favorable basin of attraction can be reached for some samples by first optimizing in physics space. Inverse initialization provides such a good initial condition that there is no added advantage of first optimizing in physics space.

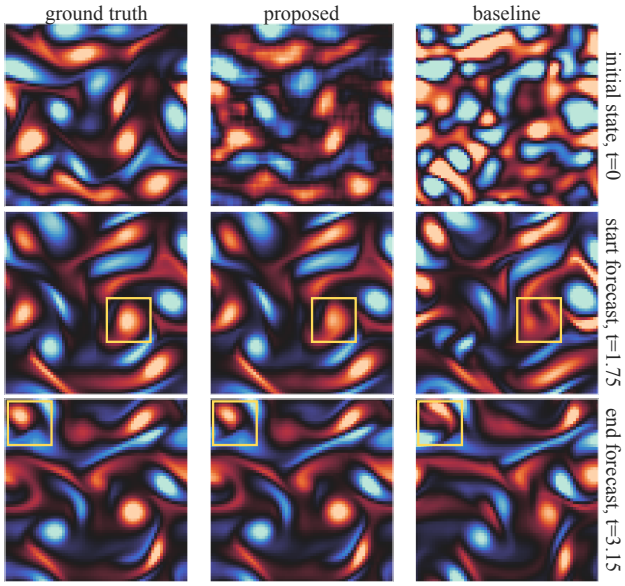

Figure 9. Vorticity snapshots of a forecast trajectory for Kolmogorov flow. The proposed method (inverse initialization, hybrid optimization) more accurately captures ground truth vorticity features (yellow squares) compared with the baseline (interpolation initialization, observation space optimization). Depicted are the initial state and snapshots from the start and end of the forecast window. Note that certain perturbations vanish quickly during the evolution of the system and are therefore not optimized to vanish in the initial state.

	LORENZ96		KOLMOGOROV	
	OBS	HYBRID	OBS	HYBRID
BASELINE	1	0.08	1	0.88
INVERSE	0.25	**0.07**	**0.20**	0.23

Table 3. Mean L_1 forecast error of the first forecast state. All values are relative to the baseline method for the respective model. The table compares both initialization schemes (baseline, inverse) and optimization methods (observation space, hybrid) for the Lorenz96 model and Kolmogorov flow. The best optimization setting is emphasized in bold face. Using the learned inverse observation operator improves optimizability for both models.

5.3. Result Summary

To summarize the results, we compare the relative performance of each optimization setting for the first forecast state, as depicted in Table 3. By using the learned inverse observation operator, the forecast error can be significantly reduced for both models. The relative merit of exploiting this operator for initialization and transformation of the objective function depends on the properties of the physical model. For the Lorenz96 model, hybrid optimization in addition to inverse initialization notably improves performance. For Kolmogorov flow, the learned inverse mapping already provides an extremely good initialization and hence optimizing in physics space does not further reduce the forecast error.

6. Conclusion

Data assimilation is the perfect problem class to explore the combination of physical modeling and machine learning since applications naturally involve rich physics and vast amounts of data. We demonstrate in this paper that a traditional variational data assimilation pipeline is improved by using a learned inverse observation operator. Exploiting this operator, we transform the 4D-Var optimization problem and show significantly enhanced forecast quality on two canonical chaotic models, the Lorenz96 model and a two-dimensional turbulent fluid flow. More broadly, our work shows that the core functionality of modern machine learning frameworks – support for automatic differentiation, hardware accelerators and deep learning – can advance research for data assimilation and other physics constrained optimization problems.

Acknowledgement

We thank Stephan Rasp, Casper Sønderby, and Jason Hickey for fruitful discussions on the topic. This work was conducted during an internship of T.F. at Google Research and was supported by the Munich Center for Machine Learning.

References

Ackmann, J., Düben, P. D., Palmer, T. N., and Smolarkiewicz, P. K. Machine-learned preconditioners for linear solvers in geophysical fluid flows. *arXiv 2010.02866*, 2020.

Andersson, E., Fisher, M., Hólm, E. V., Isaksen, L., Radnóti, G., and Trémolet, Y. Will the 4D-Var approach be defeated by nonlinearity? *ECMWF Technical Memoranda*, (479), 2005.

Bannister, R. N. A review of forecast error covariance statistics in atmospheric variational data assimilation. ii: Modelling the forecast error covariance statistics. *Quarterly Journal of the Royal Meteorological Society*, 134 (637):1971–1996, 2008.

Bannister, R. N. A review of operational methods of variational and ensemble-variational data assimilation. *Quarterly Journal of the Royal Meteorological Society*, 143 (703):607–633, 2017.

Bar-Sinai, Y., Hoyer, S., Hickey, J., and Brenner, M. P. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, 2019.

Bauer, P., Thorpe, A., and Brunet, G. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, 2015.

Beatson, A., Ash, J., Roeder, G., Xue, T., and Adams, R. P. Learning composable energy surrogates for PDE order reduction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Bocquet, M., Pires, C. A., and Wu, L. Beyond gaussian statistical modeling in geophysical data assimilation. *Monthly Weather Review*, 138(8):2997 – 3023, 2010.

Boffetta, G. and Ecke, R. E. Two-dimensional turbulence. *Annu. Rev. Fluid Mech.*, 44(1):427–451, 2012.

Bonavita, M. and Laloyaux, P. Machine learning for model error inference and correction. *Journal of Advances in Modeling Earth Systems*, 12(12):e2020MS002232, 2020.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Brajard, J., Carrassi, A., Bocquet, M., and Bertino, L. Combining data assimilation and machine learning to infer unresolved scale parametrisation. *arXiv 2009.04318*, 2020a.

Brajard, J., Carrassi, A., Bocquet, M., and Bertino, L. Combining data assimilation and machine learning to emulate a dynamical model from sparse and noisy observations: A case study with the lorenz 96 model. *Journal of Computational Science*, 44:101171, 2020b.

Chandler, G. J. and Kerswell, R. R. Invariant recurrent solutions embedded in a turbulent two-dimensional kolmogorov flow. *Journal of Fluid Mechanics*, 722:554–595, 2013.

ECMWF. *Data Assimilation*. Number 2 in IFS Documentation CY46R1. ECMWF, 2019.

Farchi, A., Laloyaux, P., Bonavita, M., and Bocquet, M. Using machine learning to correct model error in data assimilation and forecast applications. *arXiv 2010.12605*, 2020.

Geer, A. J. Learning earth system models from observations: machine learning or data assimilation? *ECMWF Technical Memoranda*, (863), 2020.

Griewank, A. Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation. *Optim. Methods Softw.*, 1(1), 1994.

Gustafsson, N., Janjić, T., Schraff, C., Leuenberger, D., Weissmann, M., Reich, H., Brousseau, P., Montmerle, T., Wattrelot, E., Bučánek, A., Mile, M., Hamdi, R., Lindskog, M., Barkmeijer, J., Dahlbom, M., Macpherson, B., Ballard, S., Inverarity, G., Carley, J., Alexander, C., Dowell, D., Liu, S., Ikuta, Y., and Fujita, T. Survey of data assimilation methods for convective-scale numerical weather prediction at operational centres. *Quarterly Journal of the Royal Meteorological Society*, 144(713): 1218–1256, 2018.

Ham, Y.-G., Kim, J.-H., and Luo, J.-J. Deep learning for multi-year enso forecasts. *Nature*, 573(7775):568–572, 2019.

Hoyer, S., Sohl-Dickstein, J., and Greydanus, S. Neural reparameterization improves structural optimization. *arXiv 1909.04240*, 2019.

Jiang, C. M., Esmaeilzadeh, S., Azizzadenesheli, K., Kashinath, K., Mustafa, M., Tchelepi, H. A., Marcus, P., Prabhat, and Anandkumar, A. Meshfreeflownet: A physics-constrained deep continuous space-time super-resolution framework. In *Proceedings of the International*

Conference for High Performance Computing, Networking, Storage and Analysis, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., and Hoyer, S. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21), 2021.

Kudyshev, Z. A., Kildishev, A. V., Shalaev, V. M., and Boltasseva, A. Machine learning–assisted global optimization of photonic devices. *Nanophotonics*, 10(1): 371–383, 2021.

Lorenz, E. N. Predictability: a problem partly solved. In *Seminar on Predictability*, pp. 1–18. ECMWF, 1995.

Mack, J., Arcucci, R., Molina-Solana, M., and Guo, Y.-K. Attention-based convolutional autoencoders for 3D-variational data assimilation. *Computer Methods in Applied Mechanics and Engineering*, 372:113291, 2020.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision – ECCV 2020*, 2020.

Nocedal, J. and Wright, S. J. *Numerical Optimization*. Springer, 2nd edition, 2006.

Rasp, S. and Thuerey, N. Data-driven medium-range weather prediction with a ResNet pretrained on climate simulations: A new model for weatherbench, arxiv 2008.08626, 2020.

Rasp, S., Dueben, P. D., Scher, S., Weyn, J. A., Mouatadid, S., and Thuerey, N. Weatherbench: A benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002203, 2020.

Sønderby, C. K., Espeholt, L., Heek, J., Dehghani, M., Oliver, A., Salimans, T., Agrawal, S., Hickey, J., and Kalchbrenner, N. Metnet: A neural weather model for precipitation forecasting. *arXiv 2003.12140*, 2020.

Tabeart, J. M., Dance, S. L., Lawless, A. S., Nichols, N. K., and Waller, J. A. Improving the condition number of estimated covariance matrices. *Tellus A: Dynamic Meteorology and Oceanography*, 72(1):1–19, 2020.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Um, K., Brand, R., Fei, Y., Holl, P., and Thuerey, N. Solver-in-the-loop: Learning from differentiable physics to interact with iterative PDE-solvers. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Watt-Meyer, O., Brenowitz, N. D., Clark, S. K., Henn, B., Kwa, A., McGibbon, J. J., Perkins, W. A., and Bretherton, C. S. Correcting weather and climate models by machine learning nudged historical simulations. *Earth and Space Science Open Archive*, 2021.

Zhuang, J., Kochkov, D., Bar-Sinai, Y., Brenner, M. P., and Hoyer, S. Learned discretizations for passive scalar advection in a 2-D turbulent flow. *arXiv 2004.05477*, 2020.