https://doi.org/10.5194/npg-2020-38
Preprint. Discussion started: 25 September 2020

Nonlinear Processes in Geophysics
Discussions

Training a convolutional neural network to conserve mass in data assimilation

Yvonne Ruckstuhl[1], Tijana Janjić [1], and Stephan Rasp[2]

[1]Meteorological Institute Munich, Ludwig-Maximilians-Universität München, Germany
[2]ClimateAi, San Francisco, USA

Correspondence: Yvonne Ruckstuhl (yvonne.ruckstuhl@lmu.de)

Abstract. In previous work, it was shown that preservation of physical properties in the data assimilation framework can significantly reduce forecasting errors. Proposed data assimilation methods, such as the quadratic programming ensemble (QPEns) that can impose such constraints on the calculation of the analysis, are computationally more expensive, severely limiting their application to high dimensional prediction systems as found in earth sciences. In order to produce from a less computationally expensive, unconstrained analysis, a solution that is closer to the constrained analysis, we propose to use a convolutional neural network (CNN) trained on analyses produced by the QPEns. In this paper, we focus on conservation of mass and show in an idealized setup that the hybrid of a CNN and the ensemble Kalman filter is capable of reducing analysis and background errors to the same level as the QPEns. To obtain these positive results, it was in one case necessary to add a penalty term to the loss function of the CNN training process.

1 Introduction

The ensemble Kalman Filter (EnKF Evensen, 1994; Burgers et al., 1998; Evensen, 2009) and versions thereof are powerful data assimilation algorithms that can be applied to problems that need an estimate of a high dimensional model state, as in weather forecasting. An important condition for a successful application of the EnKF to a large system is the use of localisation. Any localisation method aims to diminish sampling errors caused by the computational limitation of ensemble size. By doing so, mass conservation as guaranteed by a numerical model is violated during the data assimilation (Janjić et al., 2014). It was shown in Janjić et al. (2014), Zeng and Janjić (2016),Zeng et al. (2017) and Ruckstuhl and Janjić (2018) that failing to conserve certain background properties like mass, energy and enstrophy can be highly detrimental to the estimation of the state. Janjić et al. (2014) proposed a new data assimilation algorithm, the Quadratic Programming Ensemble (QPEns), which replaces the analysis equations of the EnKF with an ensemble of minimization problems subject to physical constraints. Zeng et al. (2017) showed in an idealized setup with a two week forecast generated by a two dimensional shallow water model that error growth is significantly reduced if the enstrophy is constrained. Similarly Ruckstuhl and Janjić (2018) illustrated the benefit of constraining the total mass and positivity of precipitation on a simple test case for convective scale data assimilation. The obstacle that remains for applying the QPEns on large systems is the computational demand of the constrained minimization problems that have to be solved for each ensemble member at each assimilation cycle. In this work we propose to use an artificial neural network to correct the unconstrained solution, instead of solving the constrained minimization problems.

Artificial neural networks (NN), are powerful tools to approximate arbitrary non-linear functions (Nielsen, 2015). A NN learns to recognize patterns based on example, rather than being explicitly programmed. An important advantage is that no direct knowledge of the function is needed. Instead, a data set consisting of input-output pairs is used to train the NN to predict the output corresponding to a given input. Especially in the fields of image recognition and natural language processing, NNs are state-of-the-art and have become a standard tool (LeCun Yann et al., 2015). In numerical weather prediction NNs are not yet fully integrated, though interest is rising quickly (Reichstein et al., 2019). Recent review of use of NNs in meteorology can be found in McGovern et al. (2019). Explored applications include (but are not limited to) post processing of raw model output based on observations (McGovern et al., 2017; Rasp and Lerch, 2018), representing subgrid processes in weather and climate models using high resolution model simulations (Krasnopolsky et al., 2013; Rasp et al., 2018; Brenowitz and Bretherton, 2019), combining NN with a knowledge based model as a hybrid forecasting approach (Pathak et al., 2018b) and replacing the numerical weather prediction model all together (Dueben and Bauer, 2018; Pathak et al., 2018a; Weyn et al., 2020; Scher and Messori, 2019; Rasp et al., 2020).

Fully replacing data assimilation by a NN has been attempted by Cintra and de Campos Velho (2014) in the context of a simplified atmospheric general circulation model. They trained on a cycling data set produced by the Localized Ensemble Transform Kalman Filter (LETKF, Bishop et al., 2001; Hunt et al., 2007) and show that the trained NN performs nearly as good as the LETKF with significantly reduced computational effort. Other applications of NNs in context of data assimilation are for observational bias correction (Jin et al., 2019) and tuning of covariance localization (Moosavi et al., 2019). Similarly, in this paper we take an approach that combining the NN with a data assimilation algorithm will allow extracting the most information from sparse and noisy observations, as argued in Brajard et al. (2019). We aim to produce better results than standard data assimilation algorithms at minimal additional computational costs, by training on data produced by the QPEns.

We generate our training data by performing twin experiments with the one dimensional modified shallow water model (Würsch and Craig, 2014) which was designed to mimic important properties of convection. These aspects include an acute regime switch when convection is triggered (conditional instability) and a significant time lag between the onset of convection and its observation. The model is briefly introduced in section 2.1, followed by the settings of the twin experiments in section 2.2. Section 2.3 provides a report on the generation of the training data. Since both our input and output are full model states, the obvious choice is to train a convolutional neural network (CNN), as the convolution with kernels naturally acts as a form of localisation. The CNN architecture we use for this application is described in section 2.4. The results are presented in section 3, followed by the conclusion in section 4.

2 Experiment setup

2.1 Model

The modified shallow water model (Würsch and Craig, 2014) consists of the following equations for the velocity u, rain r and water height level of the fluid h respectively:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{\partial(\phi + \gamma^2 r)}{\partial x} = \beta_u + D_u \frac{\partial^2 u}{\partial x^2}, \tag{1}$$

with

$$\phi = \begin{cases} \phi_c & if \quad h > h_c \\ gh & \text{else,} \end{cases} \tag{2}$$

$$\frac{\partial r}{\partial t} + u\frac{\partial r}{\partial x} = D_r \frac{\partial^2 r}{\partial x^2} - \alpha r - \begin{cases} \delta \frac{\partial u}{\partial x}, & h > h_r \quad \text{and} \quad \frac{\partial u}{\partial x} < 0 \\ 0, & \text{else,} \end{cases} \tag{3}$$

$$\frac{\partial h}{\partial t} + \frac{\partial (uh)}{\partial x} = D_h \frac{\partial^2 h}{\partial x^2}. \tag{4}$$

Here, h_c represents the level of free convection. When this threshold is reached the geopotential ϕ takes on a lower, constant value ϕ_c. The parameters D_u, D_r, D_h are the corresponding diffusion constants, $\gamma := \sqrt{gh_0}$ is the gravity wave speed for the absolute fluid layer h_0 ($h_0 < h_c$). The small stochastic Gaussian forcing β_u is added at random locations to the velocity at every model time step, in order to trigger perturbations that lead to convection. Parameters δ and α are the production and removal rate of rain respectively. When h reaches the rain threshold h_r ($h_r > h_c$), rain is 'produced' by adding rain water mass to the potential, leading to a decrease of the water level and of buoyancy. The model conserves mass, so the spatial integral over h is constant in time.

For the numerical implementation of the model, the one dimensional domain, representing 125 km is discretised with $n = 250$ points, yielding the state vector $\mathbf{x} = [\mathbf{u}^T \mathbf{h}^T \mathbf{r}^T]^T \in \mathbb{R}^{750}$. The time variable is discretised into time steps of 5 seconds. The Gaussian stochastic forcing β_u has a half width of 4 grid points and an amplitude of 0.002 m/s. This model was used for testing data assimilation methods in convective scale applications in Haslehner et al. (2016); Ruckstuhl and Janjić (2018).

2.2 Twin experiments

The nature run which mimics the true state of the atmosphere is a model simulation starting from an arbitrary initial state. The ensemble is chosen to be of small size with $N_{ens} = 10$, and, like the nature run, each member starts from an arbitrary initial state. Observations are assimilated every dT model time steps and are obtained by adding a Gaussian error to the wind

u and height h field of the nature run at the corresponding time with a standard deviation of $\sigma_u = 0.001$ m/s and $\sigma_h = 0.01$ m, and a lognormal error is added to the rain r field with parameters $\mu = -8$ and $\sigma = 1.5$. For all variables the observation error is roughly 10% of the maximum deviation from the variable mean. To mimic radar data, observations for all variables are available only on grid points where rain above a threshold of 0.005 dBZ is measured. A random selection, amounting to 10% of the remaining grid points, of additional wind observations are assimilated, which represents additional data available (for example obtained from aircraft).

To deal with undersampling, covariance localization using 5-th order polynomial function (Gaspari and Cohn, 1999) is applied with a localisation radius of four grid points. This corresponds to the localisation radius for which the EnKF yields minimum analysis RMSE values of the rain variable for an ensemble size of ten. An interior point method is used to solve the quadratic minimization problems of the QPEns. The constraints that are applied are mass conservation, i.e. $\mathbf{e}^T(\mathbf{h}^a - \mathbf{h}^f) = \mathbf{e}^T\delta\mathbf{h} = 0$, and positivity of precipitation, i.e. $\mathbf{r}^a = \delta\mathbf{r} + \mathbf{r}^f \geq 0$. Here, the superscript f denotes the background and a the analysis, and $\mathbf{e}$ is a vector of size $3n$ containing only values of one. For the EnKF negative values for rain are set to zero if they occur.

When the assimilation window dT is large enough, the accumulation of mass leads to divergence for the EnKF, that is, the analysis error is larger than that of an arbitrary model state. The QPEns converges for all dT, due to its ability to conserve mass. We therefore distinguish two cases, one where the EnKF converges ($dT = 60$, equivalent to 5 minutes real time), and one where the EnKF diverges ($dT = 120$, equivalent to 10 minutes real time).

2.3 Training data

We aim to produce initial conditions of the same quality as the ones produced by the QPEns by upgrading the initial conditions produced by the EnKF using a CNN. To that end, we generate QPEns cycling data $\{(\mathbf{Q}_t^f, \mathbf{Q}_t^a) : t = 1, 2, ..., T\}$, where $\mathbf{Q}$ stands for QPEns, the superscript f denotes the background and a the analysis. In parallel we create the data set $\{\mathbf{X}_t^a : t = 1, 2, ..., T\}$, where $\mathbf{X}_t^a$ is the unconstrained solution calculated from $\mathbf{Q}_t^f$. Note that by using the same $\mathbf{Q}_t^f$ as the constrained solution we train the CNN only to focus on differences in the minimization process and not on the possible differences in the background error covariances that could have accumulated during cycling. Later we validate this approach during cycling. Both data sets contain the entire ensemble of $N_{ens} = 10$ members, such that $(*)_t^{(*)} \in \mathbb{R}^{N_{ens} \times n \times 3}$, where the last dimension represents the 3 variables (u, h, r) and n is the number of grid points.

The output of our training set $\mathbf{Y}^{tr} \in \mathbb{R}^{N_{ens}T \times n \times 3}$ is simply a reshaped and normalized version of the data set $\{\mathbf{Q}_t^a : t = 1, 2, ..., T\}$. For the input of our training set $\mathbf{X}^{tr}$ we choose to use an index vector indicating the position of the radar observations $\{\mathbf{I}_t : t = 1, 2, ..., T\}$ in addition to the unconstrained solutions $\{\mathbf{X}_t^a : t = 1, 2, ..., T\}$, yielding $\mathbf{X}^{tr} \in \mathbb{R}^{N_{ens}T \times n \times 4}$, where the index vector $\mathbf{I}_t$ is copied N_{ens} times to obtain $\mathbf{I}_t^* \in \mathbb{R}^{N_{ens} \times n \times 3}$. For u and h the input and output data set is normalized by subtracting the climatological mean before dividing by the climatological standard deviation. For r, we do not subtract the climatological mean to maintain positivity.

A validation data set $\mathbf{X}^{valid}$ and $\mathbf{Y}^{valid}$ is created to monitor the training process exactly as the training data set but with a different random seed number. For both the training and validation data set we set $T = 4800$, which amounts to a total of $N_{ens}T = 48000$ training and validation samples respectively.

2.4 Convolutional neural network architecture

We choose to use a CNN with 4 convolutional hidden layers, consisting of 32 filters each with kernels of size 3 and the “selu" activation function

$$g(x) = \lambda \begin{cases} x, & \text{for } x \geq 0 \\ \alpha\left(e^x - 1\right), & \text{for } x < 0 \end{cases},$$

where $\lambda = 1.05070098$ and $\alpha = 1.67326324$. The output layer is a convolutional layer as well, where the number of filters is determined by the desired shape of the output of the CNN, which is a model state $(u,h,r) \in \mathbb{R}^{n \times 3}$. The output layer has therefore 3 filters and the kernel size is again 3. Note that the “localisation radius", that is, the maximum influence radius of a variable as assumed by the CNN is $(3-1)/2 * 5 = 5$, where 5 is number of layers and 3 the kernel size. We use a linear activation function for u and h and the “relu" activation function for r to ensure non-negativity of rain. We set the batch size to 96 and do 100 epochs. Unless stated otherwise, the loss function is defined as the root mean squared error (RMSE) over the grid points, averaged over the variables:

$$J\left(\mathbf{y}_j^p(\mathbf{w})\right) = \frac{1}{3}\sum_{v=1}^{3}\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_{j,i,v}^p - y_{j,i,v}\right)^2}, \; j = 1,\ldots,N_{ens}T$$

where $y_{j,i,v}^p$ and $y_{j,i,v}$ are the prediction and output for the v^{th} variable of the j^{th} sample at the i^{th} grid point respectively. The Adam algorithm is used to minimize $\frac{1}{N_{ens}T}\sum_{j=1}^{N_{ens}T} J\left(\mathbf{y}_j^p(\mathbf{w})\right)$ over the weights $\mathbf{w}$ of the CNN. The training is done with python library Keras (Chollet et al., 2015).

3 Results

We assign the name $dT5$ to the experiment corresponding to a cycling period of 5 minutes, and $dT10$ to the experiment corresponding to a cycling period of 10 minutes. Figure 1 shows the evolution of the loss function averaged over the samples for the training and validation data set for $dT5$ and $dT10$ respectively. Table 1 summarizes what the CNN has learned for each variable separately in the two cases. As the training data is normalized, we can conclude from the RMSE of the input data with respect to the output data (first row in Table 1 panels) that the mass constraint on h and the positivity constraints on r impacts the solution of the minimization problem for all variables with the same order of magnitude. Given our choice of loss function it is not surprising that the relative reduction of the gap between the input and output by the CNN is proportional to the size of the gap. By aiming to minimize the mean RMSE of all variables, the CNN reduced the violation of the mass constraint by about 20% for both experiments. However, for $dT5$ the reduction in the bias of the height field is 100%, while for $dT10$ it is a mere 30%.

https://doi.org/10.5194/npg-2020-38
Preprint. Discussion started: 25 September 2020

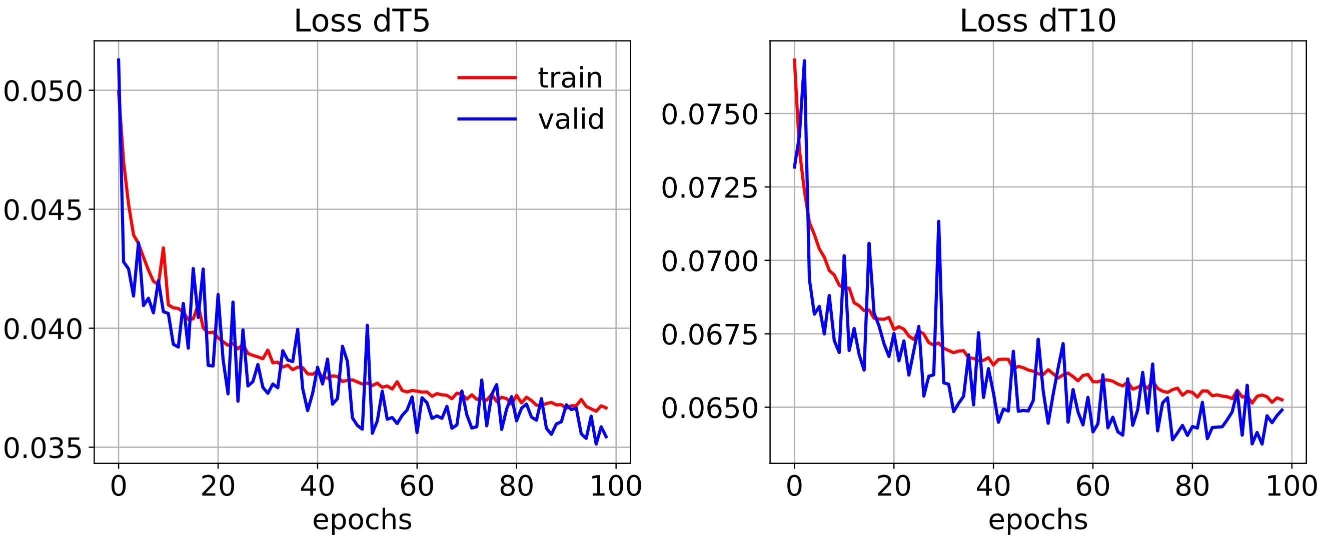

Figure 1. Value of the loss function J averaged over samples for the training (red) and validation (blue) data set as function of epochs for $dT5$ (left) and $dT10$ (right).

	Validation	loss	**u**	**h**	**r**	mass **h**	mass **r**	bias **h**
$dT5$	Input	4.9e-2	4.2e-2	4.6e-2	5.9e-2	1.8e-2	4.0e-3	1.4e-2
	Prediction	3.6e-2	4.1e-2	3.9e-2	2.7e-2	1.4e-2	2.9e-3	0.0
	Improvement (%)	27	2.8	15	55	22	28	100
$dT10$	Input	9.6e-2	7.9e-2	9.0e-2	1.2e-1	3.6e-2	8.0e-3	3.3e-2
	Prediction	6.5e-2	7.3e-2	6.9e-2	5.4e-2	2.9e-2	7.1e-3	2.3e-2
	Improvement (%)	32	7.8	24	55	20	11	30

Table 1. The loss function, the mean RMSE of the variables u,h,r, the absolute mass error divided by the number of grid points n for h and r, and the bias of h (columns) calculated for the input $\mathbf{X}^{valid}$ (top row) and the CNN prediction (middle row) with respect to the output $\mathbf{Y}^{valid}$ for the validation data sets. The last row shows the improvement of the prediction towards the output compared to the input in percentage. The top table corresponds to $dT5$, the bottom table to $dT10$.

Next, we are interested in how the CNNs perform when applied within the data assimilation cycling. In Figure 2, we compare the performance of the EnKF, QPEns and the hybrid of CNN and EnKF, where CNN is applied as correction to the initial conditions computed by the EnKF. To avoid having to train a CNN for the spin-up phase where the increments are larger, we start the data assimilation for the EnKF and the CNN from the initial conditions produced by the QPEns at the 20^{th} cycle. The RMSEs shown in Figure 2 are calculated through time against nature values for both background and analysis.

https://doi.org/10.5194/npg-2020-38
Preprint. Discussion started: 25 September 2020

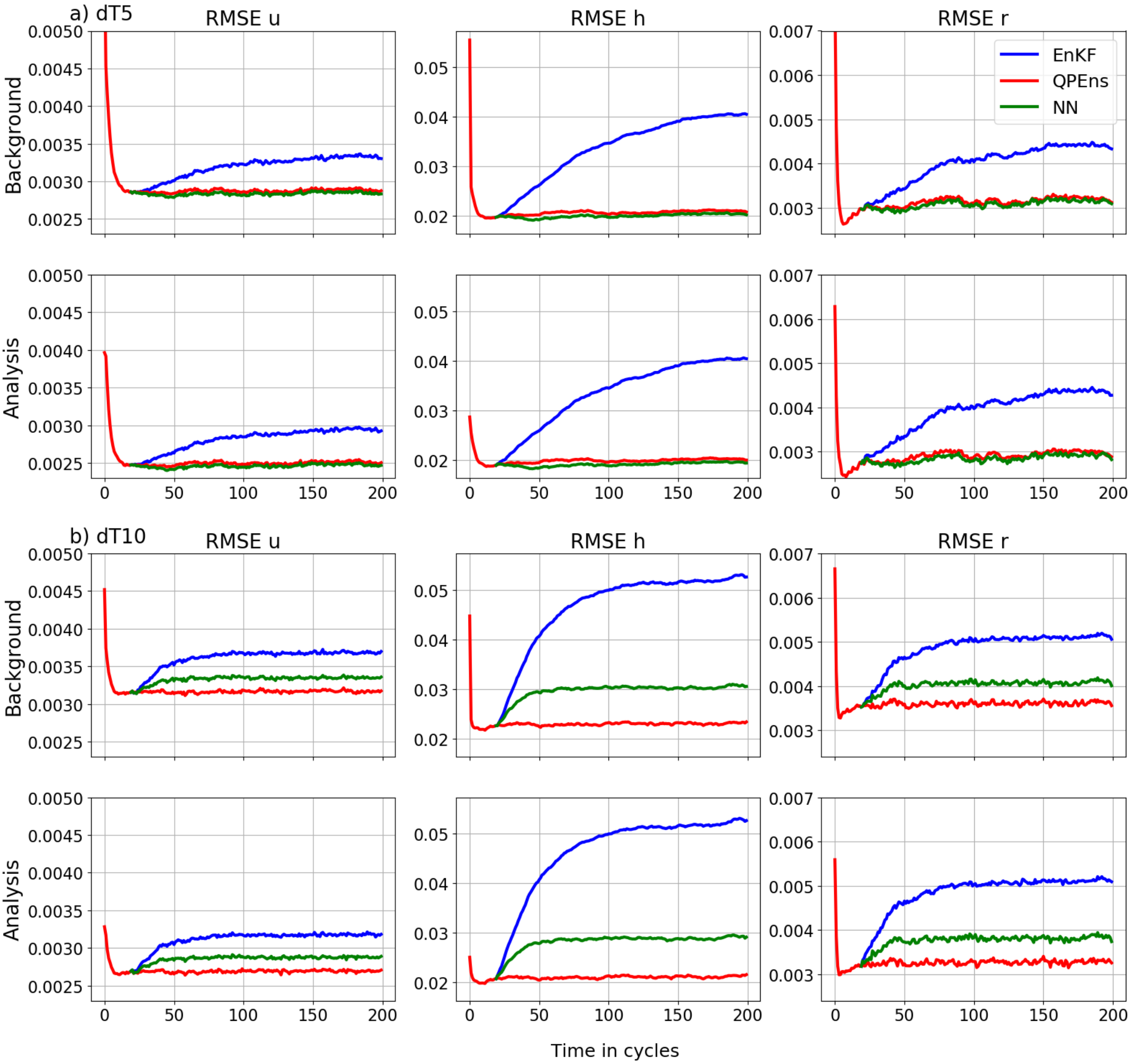

Figure 2. RMSE of the ensemble mean of the variables (columns) for the background (top rows) and analysis (bottom rows) as function of assimilation cycles for the EnKF (blue), the QPEns (red) and the CNN (green). The panels in a) corresponds to $dT5$ and in b) to $dT10$.

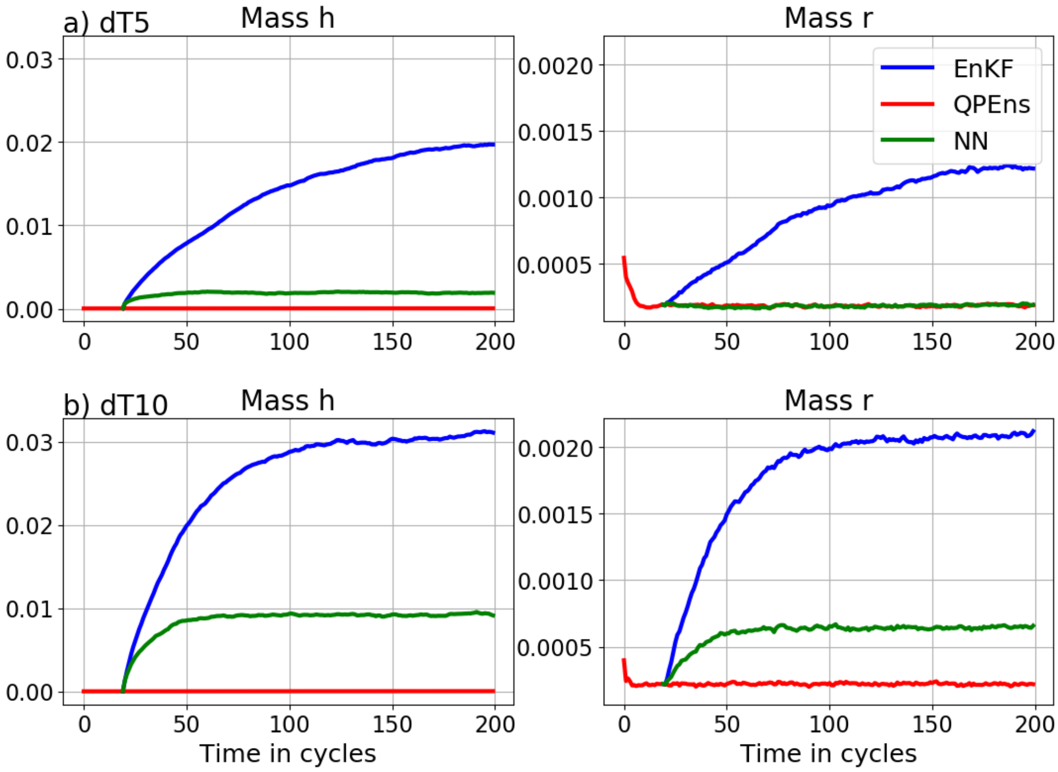

Figure 3. Absolute mass error of **h** (left) and **r** (right) for the analysis ensemble mean for the EnKF (blue), QPEns (red) and CNN (green). The plots in a) correspond to $dT5$ and in b) to $dT10$.

With respect to RMSEs, for $dT5$ the CNN performs as well as the QPEns, despite having learned during training only 27% of the difference between the EnKF and QPEns analysis in terms of the loss function. For $dT10$ the CNN does perform significantly better than the EnKF, but clearly remains inferior to the QPEns. Given that in terms of the RMSE over the grid points, the CNN for $dT10$ is slightly better than the one for $dT5$, we hypothesize that the key to good performance of the CNN applied within the data assimilation cycling lies with preventing the accumulation of mass in h. When mass accumulates in clear regions, that is regions where for the nature run holds $h < h_c$, it has a snowball effect not only on h itself but also on r, see Figure 3. After all, clouds, and later rain, are produced whenever $h > h_c$. For $dT5$ the CNN does not score much better than for $dT10$ in terms of absolute mass error. However it was able to effectively remove all bias in h (with a residual of $\mathcal{O}(10^{-5})$), in contrast to the CNN for $dT10$. In addition, when distinguishing between clear and cloudy regions, the absolute mass error for the CNN corresponding to $dT5$ is reduced by 80% in the clear regions, as opposed to a mere 30% for the CNN corresponding to $dT10$ (not shown).

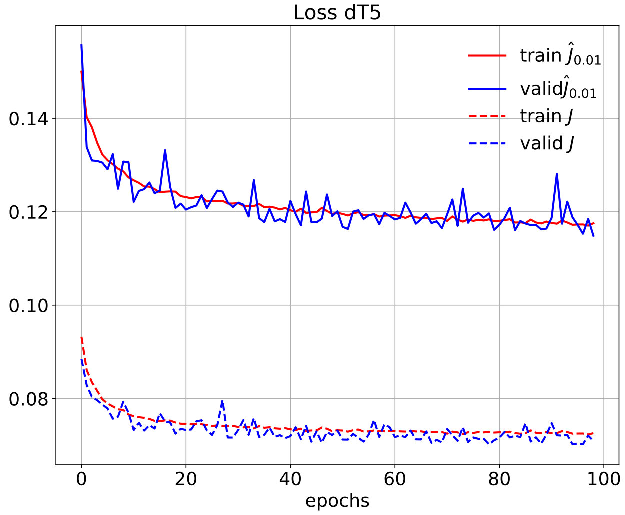

Figure 4. Value of the loss function $\hat{J}$ (solid) and J (dashed) averaged over samples for the training (red) and validation (blue) data set as function of epochs for $dT10_\gamma$.

To support this claim we trained an additional CNN with the training set corresponding to $dT = 10$, with a penalty term for the mass of h in the loss function:

$$\hat{J}_\gamma\left(\mathbf{y}_j^p(\mathbf{w})\right) = J\left(\mathbf{y}_j^p(\mathbf{w})\right) + \gamma\left|\sum_{i=1}^{n} y_{j,i,2}^p - \sum_{i=1}^{n} y_{j,i,2}\right|$$

where the parameter γ is tunable. The larger γ, the better the mass of h is conserved at the expense of the RMSE. We found a good trade-off for $\gamma = 0.01$. We refer to this experiment as $dT10_\gamma$. The training process is illustrated in Figure 4. The mass conservation term comprises about 40% of the total loss function $\hat{J}$. Both terms of the loss function are decreasing at approximately the same rate throughout the entire training process. Comparing Table 1 with Table 2 we conclude that by adding the penalty term for the mass violation in the loss function, 6% was lost in terms of loss function J, but 32% was gained in the conservation of mass. In clear regions, the mass violation reduction even went from 30% for $dT10$ to 85% for $dT10_\gamma$. Table 3 confirms that the CNN is especially active in clear regions. Indeed, the correlation coefficient between h and the increments in all variables is significant and negative, indicating that the prediction increments are larger in clear regions than in cloudy regions.

validation	loss	**u**	**h**	**r**	mass **h**	mass **r**	bias **h**
Input	9.6e-2	7.9e-2	8.9e-2	1.2e-1	3.6e-2	7.8e-3	3.3e-2
Prediction	7.1e-2	7.6e-2	8.4e-2	5.3e-2	1.8e-2	5.5e-3	0
Improvement (%)	26	4.1	6.6	54	52	29	100

Table 2. As table 1, but for $dT10_\gamma$.

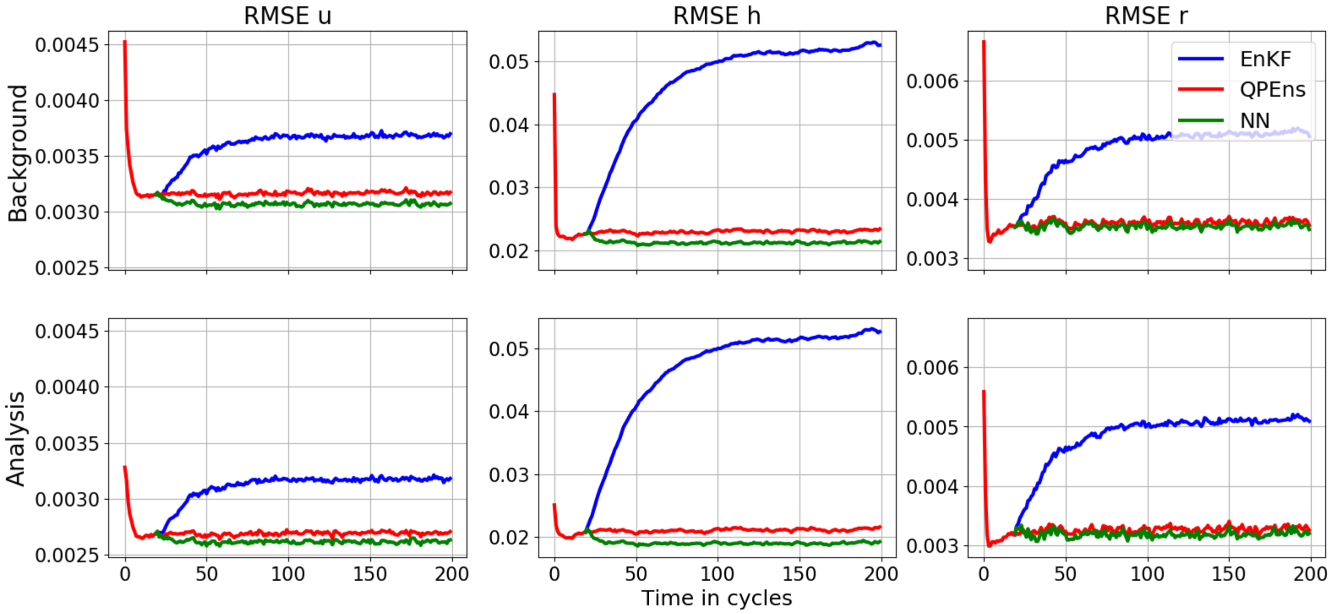

Figure 5. As Figure 2 for $dT10_\gamma$.

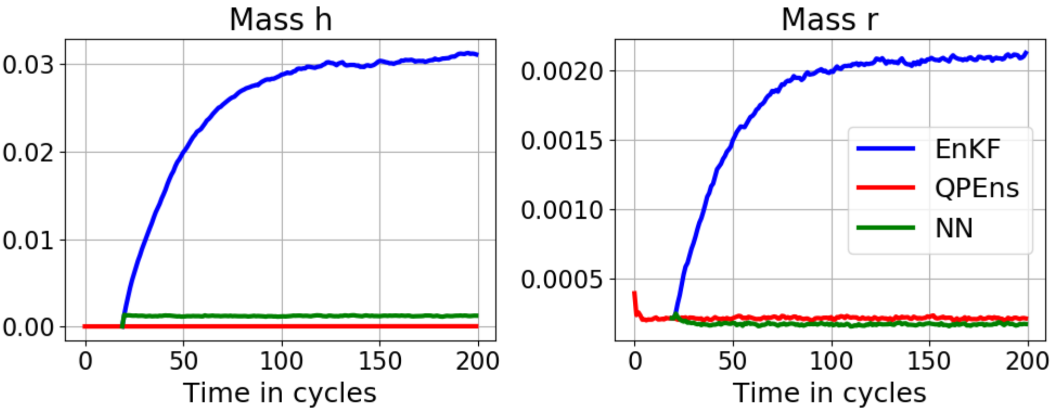

Figure 6. As Figure 3 for $dT10_\gamma$.

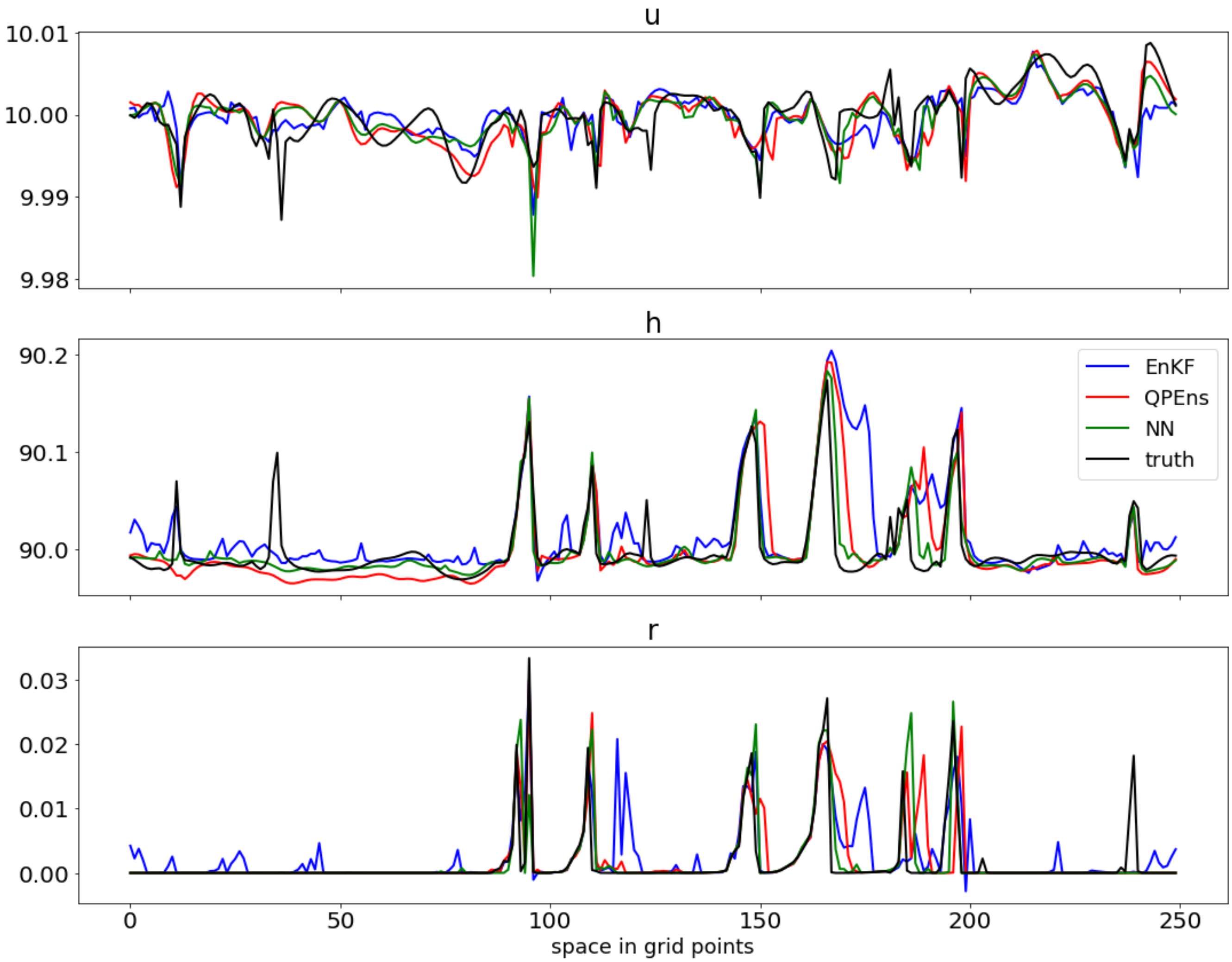

Figure 7. Ensemble mean snapshot for $dT10_{\gamma}$.

		$\mathbf{Y}-\mathbf{X}$			$dT10$: $\mathbf{Y}^p-\mathbf{X}$			$dT10_\gamma$: $\mathbf{Y}^p-\mathbf{X}$		
		u	**h**	**r**	**u**	**h**	**r**	**u**	**h**	**r**
X	**u**	-0.1	0.0	0.0	-0.2	0.0	0.0	-0.1	0.0	0.0
	h	0.0	-0.1	-0.1	0.1	-0.2	-0.2	-0.3	-0.4	-0.2
	r	0.0	-0.1	-0.3	0.1	-0.2	-0.4	-0.2	-0.3	-0.3
$\frac{d\mathbf{X}}{dx}$	**u**	-0.1	0.0	0.0	-0.2	0.1	0.0	-0.1	0.1	0.0
	h	-0.2	-0.1	-0.2	-0.4	-0.2	-0.2	-0.5	-0.1	-0.2
	r	-0.1	-0.1	-0.3	-0.2	-0.2	-0.3	-0.2	-0.1	-0.4

Table 3. Correlation coefficient for increments of the output (left) and the prediction for $dT10$ (middle) and $dT10_\gamma$ (right) with the input (top) and the gradient of the input (bottom).

Figure 5, 6 and 7 show the data assimilation results for $dT10_\gamma$. It is striking that the CNN performs slightly better than the QPEns. When comparing the input $\mathbf{X}$, output $\mathbf{Y}$ and the CNN prediction $\mathbf{Y}^p$ to the nature run, we found that for the clear regions $\mathbf{Y}^p$ is slightly closer to the nature run in terms of RMSE than the QPEns and significantly closer than the EnKF (not shown). We speculate that this is because the QPEns generally lacks mass in regions where there are no clouds in both the nature run and the QPEns estimate. The EnKF on the other hand, overestimates the mass in these regions. As a result, the true value of h lies between the QPEns and EnKF estimates. In these regions it is therefore favourable that the CNN can not completely close the gap between the input and output data, as it leads to a better fit to the nature run. We also performed an experiment where only the clear regions of h are updated by the CNN and the other variables and cloudy regions of h remain equal to the EnKF solution, and similar results were obtained as in Figure 5 and 6. We therefore conclude that the success of this approach lies in the ability of the CNN to correct for errors of h in clear regions.

4 Conclusion

Geoscience phenomena have several aspects that are different from standard data science applications, for example governing physical laws, noisy, non-uniform in space and time observations from many different sources, as well as rare interesting events. This makes use of NN challenging in particular for convective scale applications, although attempts have been made for predicting rain, hail or tornadoes (McGovern et al., 2019). The approach taken in this study, combines noisy and sparse observations with a dynamical model using a data assimilation algorithm, but also uses a NN in order to improve on conservation of physical laws. In previous work it was shown in idealized setups that conserving physical properties like mass in the data assimilation framework using the QPEns can significantly improve the estimate of the nature run. Here we show that it is possible to obtain similar positive results by training a CNN to conserve mass in a weak sense. By training on the unconstrained (EnKF)/constrained (QPEns) input/output pair, the CNN was already able to reduce the mass violation significantly. However, we found that adding a penalty term for mass violation in the loss function was necessary in one of the two test cases to produce data assimilation results that are as good as those corresponding to the QPEns.

These encouraging results prompt the question of the feasibility of this approach applied to fully complex numerical weather prediction systems. The challenge here lies in the generation of the training data. First the effectiveness of conserving different quantities has to be verified in a non-idealized numerical weather prediction framework, where the quantities to be conserved are not always known. A second consideration is the computational costs. Advances are made in this regard (Janjic et al., under review), but effort and collaboration with optimization experts is still required to allow the generation of a reasonably large training data set.

Author contributions. All of the authors contributed to research behind this paper, as well as to writing of the manuscript.

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

Acknowledgements. "The research leading to these results has been done within the subproject B6 of the Transregional Collaborative Research Center SFB / TRR 165 "Waves to Weather" (www.wavestoweather.de) funded by the German Research Foundation (DFG). Tijana Janjić is thankful to DFG for funding her research through Heisenberg Programm JA 1077/4-1.

References

Bishop, C. H., Etherton, B. J., and Majumdar, S.: Adaptive sampling with the ensemble transform Kalman filter. Part I: Theoretical aspects., Mon. Wea. Rev., 129, 420–436, 2001.

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, Geoscientific Model Development Discussions, 2019, 1–21, https://doi.org/10.5194/gmd-2019-136, https://gmd.copernicus.org/preprints/gmd-2019-136/, 2019.

Brenowitz, N. D. and Bretherton, C. S.: Spatially Extended Tests of a Neural Network Parametrization Trained by Coarse-Graining, Journal of Advances in Modeling Earth Systems, 11, 2728–2744, https://doi.org/10.1029/2019MS001711, https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2019MS001711, 2019.

Burgers, G., van Leeuwen, P. J., and Evensen, G.: Analysis Scheme in the Ensemble Kalman Filter., Mon. Wea. Rev, 126, 1719–1724, 1998.

Chollet, F. et al.: Keras, https://keras.io, 2015.

Cintra, R. S. C. and de Campos Velho, H. F.: Data Assimilation by Artificial Neural Networks for an Atmospheric General Circulation Model: Conventional Observation, CoRR, abs/1407.4360, http://arxiv.org/abs/1407.4360, 2014.

Dueben, P. and Bauer, P.: Challenges and design choices for global weather and climate models based on machine learning, Geosci. Model Dev., 11, 3999–4009, https://doi.org/10.5194/gmd-11-3999-2018, 2018.

Evensen, G.: Sequential data assimilation with a nonliear quasi-gepstrophic model using Monte Carlo methods to forecast error statistics., Journal of Geophysical Research, 99, 10 143–10 162, 1994.

Evensen, G.: Data Assimilation: The Ensemble Kalman Filter, Springer, 2009.

Gaspari, G. and Cohn, S. E.: Construction of correlation functions in two and three dimensions, Quart. J. Roy. Meteor. Soc., 125, 723–757, 1999.

Haslehner, M., Janjic, T., and Craig, G. C.: Testing particle filters on simple convective-scale models. Part 2: A modified shallow-water model, Quarterly Journal of the Royal Meteorological Society, 142, 1628–1646, https://doi.org/10.1002/qj.2757, http://dx.doi.org/10.1002/qj.2757, 2016.

Hunt, B. R., Kostelich, E. J., and Szunyogh, I.: Efficient Data Assimilation for Spatiotemporal Chaos: A local Ensemble Transform Kalman filter., Physica D, 230, 112–126, 2007.

Janjić, T., McLaughlin, D., Cohn, S. E., and Verlaan, M.: Conservation of mass and preservation of positivity with ensemble-type Kalman filter algorithms, Mon. Wea. Rev., 142, 755–773, 2014.

Janjic, T., Ruckstuhl, Y., and Toint, P. L.: A data assimilation algorithm for predicting rain, Quarterly Journal of the Royal Meteorological Society, under review.

Jin, J., Lin, H. X., Segers, A., Xie, Y., and Heemink, A.: Machine learning for observation bias correction with application to dust storm data assimilation, Atmospheric Chemistry and Physics, 19, 10 009–10 026, https://doi.org/10.5194/acp-19-10009-2019, https://acp.copernicus.org/articles/19/10009/2019/, 2019.

Krasnopolsky, V. M., Fox-Rabinovitz, M. S., and Belochitski, A. A.: Using ensemble of neural networks to learn stochastic convection parameterizations for climate and numerical weather prediction models from data simulated by a cloud resolving model, Adv Artif Neural Syst, 2013, http://dx.doi.org/10.1155/2013/485913, 2013.

LeCun Yann, Bengio Yoshua, and Hinton Geoffrey: Deep learning, Nature, 521, 436, https://doi.org/10.1038/nature1453910.1038/nature14539, 2015.

McGovern, A., Elmore, K. L., Gagne, David John, I., Haupt, S. E., Karstens, C. D., Lagerquist, R., Smith, T., and Williams, J. K.: Using Artificial Intelligence to Improve Real-Time Decision-Making for High-Impact Weather, Bulletin of the American Meteorological Society, 98, 2073–2090, https://doi.org/10.1175/BAMS-D-16-0123.1, https://doi.org/10.1175/BAMS-D-16-0123.1, 2017.

McGovern, A., Lagerquist, R., John Gagne, David, I., Jergensen, G. E., Elmore, K. L., Homeyer, C. R., and Smith, T.: Making the Black Box More Transparent: Understanding the Physical Implications of Machine Learning, Bulletin of the American Meteorological Society, 100, 2175–2199, https://doi.org/10.1175/BAMS-D-18-0195.1, https://doi.org/10.1175/BAMS-D-18-0195.1, 2019.

Moosavi, A., Attia, A., and Sandu, A.: Tuning Covariance Localization Using Machine Learning, in: Computational Science ? ICCS 2019. ICCS 2019. Lecture Notes in Computer Science, edited by et al. (eds), R. J., vol. 11539, Springer, Cham., https://doi.org/https://doi.org/10.1007/978-3-030-22747-0_16, 2019.

Nielsen, M. A.: Neural networks and deep learning, determination press, 2015.

Pathak, J., Hunt, B., Girvan, M., Lu, Z., and Ott, E.: Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach, Phys. Rev. Lett., 120, 024 102, https://doi.org/10.1103/PhysRevLett.120.024102, https://link.aps.org/doi/10.1103/PhysRevLett.120.024102, 2018a.

Pathak, J., Wikner, A., Fussell, R., Chandra, S., Hunt, B. R., Girvan, M., and Ott, E.: Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model, Chaos: An Interdisciplinary Journal of Nonlinear Science, 28, 041 101, https://doi.org/10.1063/1.5028373, 2018b.

Rasp, S. and Lerch, S.: Neural networks for post-processing ensemble weather forecasts, CoRR, abs/1805.09091, 2018.

Rasp, S., Pritchard, M. S., and Gentine, P.: Deep learning to represent subgrid processes in climate models, Proceedings of the National Academy of Sciences, 115, 9684–9689, https://doi.org/10.1073/pnas.1810286115, 2018.

Rasp, S., Dueben, P. D., Scher, S., Weyn, J. A., Mouatadid, S., and Thuerey, N.: WeatherBench: A benchmark dataset for data-driven weather forecasting, 2020.

Reichstein, M., Camps-Valls, G., and Stevens, B.: Deep learning and process understanding for data-driven Earth system science, Nature, 566, 195–204, https://doi.org/10.1038/s41586-019-0912-1, 2019.

Ruckstuhl, Y. and Janjić, T.: Parameter and state estimation with ensemble Kalman filter based algorithms for convective-scale applications, Quart .J. Roy. Meteorol. Soc, 144, 826–841, https://doi.org/10.1002/qj.3257, https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3257, 2018.

Scher, S. and Messori, G.: Weather and climate forecasting with neural networks: using general circulation models (GCMs) with different complexity as a study ground, Geosci. Model Dev., 12, 2797–2809, https://doi.org/10.5194/gmd-12-2797-2019, 2019.

Weyn, J. A., Durran, D. R., and Caruana, R.: Improving data-driven global weather prediction using deep convolutional neural networks on a cubed sphere, 2020.

Würsch, M. and Craig, G. C.: A simple dynamical model of cumulus convection for data assimilation research, Meteorologische Zeitschrift, 23, 483–490, 2014.

Zeng, Y. and Janjić, T.: Study of conservation laws with the Local Ensemble Transform Kalman Filter, Quarterly Journal of the Royal Meteorological Society, 142, 2359–2372, https://doi.org/10.1002/qj.2829, http://dx.doi.org/10.1002/qj.2829, 2016.

Zeng, Y., Janjić, T., Ruckstuhl, Y., and Verlaan, M.: Ensemble-type Kalman filter algorithm conserving mass, total energy and enstrophy, Quarterly Journal of the Royal Meteorological Society, 143, 2902–2914, https://doi.org/10.1002/qj.3142, 2017.