Learning bias corrections for climate models using deep neural operators

Aniruddha Bora*[1], Khemraj Shukla*[1], Shixuan Zhang[2], Bryce Harrop[2], Ruby Leung[2] and George Em Karniadakis[1]

[1] Division of Applied Mathematics, Brown University,
170, Hope Street Providence, RI-02906
george_karniadakis@brown.edu
[2]Pacific Northwest National Laboratory
902 Battelle Blvd, Richland, WA 99354

Abstract

Numerical simulation for climate modeling resolving all important scales is a computationally taxing process. Therefore, to circumvent this issue a low resolution simulation is performed, which is subsequently corrected for bias using reanalyzed data (ERA5), known as nudging correction. The existing implementation for nudging correction uses a relaxation based method for the algebraic difference between low resolution and ERA5 data. In this study, we replace the bias correction process with a surrogate model based on the Deep Operator Network (DeepONet). DeepONet (Deep Operator Neural Network) learns the mapping from the state before nudging (a functional) to the nudging tendency (another functional). The nudging tendency is a very high dimensional data albeit having many low energy modes. Therefore, the DeepoNet is combined with a convolution based auto-encoder-decoder (AED) architecture in order to learn the nudging tendency in a lower dimensional latent space efficiently. The accuracy of the DeepONet model is tested against the nudging tendency obtained from the E3SMv2 (Energy Exascale Earth System Model) and shows good agreement. The overarching goal of this work is to deploy the DeepONet model in an online setting and replace the nudging module in the E3SM loop for better efficiency and accuracy.

Introduction

The atmospheric circulation is characterized by complex multiscale physical processes, hence posing a lot of challenges for numerical simulations. These simulated processes may exhibit inconsistencies and significant deviations from the true states and are known as "biases". They are developed due to limited spatial resolution in climate models and inaccurate physical parameterizations in comparison to the observed atmospheric states. The limited resolution simulations are less taxing on computational resources, but their biases limit their usefulness. This trade off results into larger uncertainties in climate predictions. In general, such biases are removed using post processing steps, such as model output statistics for weather forecasting (Glahn and Lowry 1972), ensemble bias correction for seasonal prediction (Arribas et al. 2011), (Stockdale et al. 1998), etc. In existing implementations of the online bias correction approach, the bias correction is based on a data driven approach with lower generalization capabilities (Watt-Meyer et al. 2021). In this work, we propose a neural operator based surrogate to learn the bias correction with a larger generalization capability. In particular, we utilize Deep Operator Network (DeepONet) (Lu et al. 2021) by augmenting the existing neural network architecture with autoencoders (Oommen et al. 2022).

Methodology

In this section, we will discuss the data generation, methodology to train DeepONet, and the architecture details of the proposed surrogate model.

Data generation

The time evolution of a physical quantity X_m in the Energy Exascale Earth System Model-E3SM (E3SM Project 2018) is given by (Sun et al. 2019) :

$$\frac{\partial X_m}{\partial t} = F(X_m), \tag{1}$$

where X_m is the state variable. The nudging tendency for the state variable is given by

$$\left(\frac{\partial X_m}{\partial t}\right)_{ndg} = \begin{cases} \frac{X_p(t_p)-X_m(t)}{\tau}, & t = t_i, \\ 0, & t \neq t_i, \end{cases} \tag{2}$$

where t_i is the time at i^{th} hour when the $ERA5$ data is available, which in our case is every 3 hours. The data used for training and testing the DeepONet are generated by performing EAMv2 simulations, which are nudged towards the reference data sets (i.e. 3-hourly ERA5 reanalysis data (Hersbach et al. 2020)). This time-dependent nudging tendency is determined by the model X_m and reference state X_p as well as the nudging relaxation time scale τ. This is shown in (2). Nudging tendencies are calculated at every 30 minutes following (2). In (2), X_p is obtained by linearly interpolating the 3-hourly ERA5 reanalysis data to the model times. In this study, the EAMv2 simulations are nudged only for U (zonal wind) and V (meridional wind) and we have used three data sets, which are (i) data before nudging, (ii) nudging tendency and (iii) ERA5 reanalyzed data-set. These data sets, in relation to each other, are expressed as,

$$\left(\frac{\partial X_m}{\partial t}\right)_{\mathbf{cor}} = \left(\frac{\partial X_m}{\partial t}\right)_{\mathbf{uncor}} + \left(\frac{\partial X_m}{\partial t}\right)_{ndg}, \tag{3}$$

[*] These authors contributed equally.

where $(\partial X_m/\partial t)_{uncor}$ and $(\partial X_m/\partial t)_{ndg}$ denote the before-nudged state and nudging tendency, respectively. The $(\partial X_m/\partial t)_{cor}$ represents the corrected stated of the state variable X_m, which should be close to ERA5. By construction, (3) tries to converge prediction from model towards true observations.

Architecture of Deep Operator Network

To learn a surrogate for prediction of the nudging tendency requires learning a function $\mathcal{G}$ from space $\mathbf{U}$ (before nudge data) to $\mathbf{V}$ (Nudging tendency), a function of spatio-temporal coordinates (x, y, t). DeepONet is a neural operator regression technique based on the universal approximation theorem (Chen and Chen 1995). The goal of DeepONet is to learn the mapping from an input space, $\mathbf{U}(\eta)$, to an output space $\mathbf{V}(\zeta)$.The conventional DeepONet architecture consists of two neural networks first, branch network (branch-net) and second trunk network (trunk-net). The branch-network and trunk-network take in as input samples $u(\eta) \in \mathbf{U}(\eta)$ and ζ, respectively. The output of the DeepONet can be represented as the operator $\mathcal{G} : \mathbf{U} \rightarrow \mathbf{V}$, such that $v(\zeta) = \mathcal{G}(u(\zeta))$, where $u(\zeta) \in \mathbf{U}$ and $v(\zeta) \in \mathbf{V}$. The nudging tendency has many low energy modes, less coherent structures, and are very high dimensional due to which they are very hard to approximate using neural networks. This is due to very low correlations between feature and label data. Oommen et. al (Oommen et al. 2022) has shown that these type of the problems can be learned successfully by using DeepONet in latent space discovered by encoders. In this study, we use two autoencoders to reduce the spatial dimension of $\mathbf{U}$ and $\mathbf{V}$ to represent the nudging tendency in a smooth latent manifold so that its easier to learn. The steps required to train the DeepONet are

i. The first step is to train two autoencoders, one for the DeepONet input, i.e., the before-nudged variables ($\psi_{bf}(z, \bar{x}, \bar{y}, t)$) and the other for the DeepONet output that is the nudging tendency ($\psi_{ndg}(z, \bar{x}, \bar{y}, t)$ variables. Both autoencoders are trained using the MSE loss as shown:

$$\mathcal{L}_{AE1} = ||\psi_{bf}(z, \bar{x}, \bar{y}, t) - \psi_{bf}(z, \bar{x}, \bar{y}, t; \theta_{AE1})||_2,$$
$$\mathcal{L}_{AE2} = ||\psi_{ndg}(z, \bar{x}, \bar{y}, t) - \psi_{ndg}(z, \bar{x}, \bar{y}, t; \theta_{AE2})||_2,$$

where AE1 and AE2 represent the autoencoders for the before-nudged and the nudging tendency variables. Also, θ_{AE1} and θ_{AE2} represent the corresponding trainable parameters for the respective autencoders.

ii. After training and validation of the encoder and decoder, we learn DeepONet in the latent space with features and labels defined by $\hat{\psi}_{bf}(z, \bar{x}, \bar{y}, t)$ and $\hat{\psi}_{ndg}(z \times ld, t)$, respectively. The output of DeepONet $\hat{\psi}_{ndg\_don}(z \times ld, t)$ is trained against the $\hat{\psi}_{ndg}(z \times ld, t)$ by minimizing the mean square error function expressed as

$$\mathcal{L} = ||\hat{\psi}_{ndg\_don}(z \times ld, t) - \hat{\psi}_{ndg}(z \times ld, t)||_2. \qquad (4)$$

iii. To recover the output of DeepONet in primitive space, we compose it with the decoder.

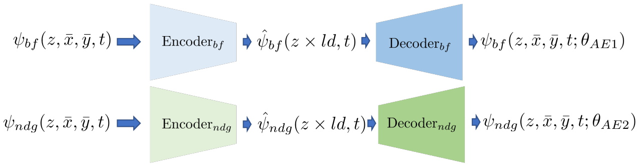

$\psi_{bf}(z, \bar{x}, \bar{y}, t)$
Encoder$_{bf}$
$\hat{\psi}_{bf}(z \times ld, t)$
Decoder$_{bf}$
$\psi_{bf}(z, \bar{x}, \bar{y}, t; \theta_{AE1})$
$\psi_{ndg}(z, \bar{x}, \bar{y}, t)$
Encoder$_{ndg}$
$\hat{\psi}_{ndg}(z \times ld, t)$
Decoder$_{ndg}$
$\psi_{ndg}(z, \bar{x}, \bar{y}, t; \theta_{AE2})$

Figure 1: A block diagram showing high level view of the autoencoder. Here $\psi_{bf}(z, \bar{x}, \bar{y}, t)$ and $\psi_{ndg}(z, \bar{x}, \bar{y}, t)$ are the input to the encoders $Encoder_{bf}$ (encoder for before nudged data) and $Encoder_{ndg}$ (encoder for nudging tendency) respectively. $\hat{\psi}_{bf}(z \times ld, t)$, $\hat{\psi}_{ndg}(z \times ld, t)$ are the latent dimension representation of the respective inputs and $\psi_{bf}(z, \bar{x}, \bar{y}, t; \theta_{AE1})$, $\psi_{ndg}(z, \bar{x}, \bar{y}, t; \theta_{AE2})$ are the reconstructed variable via the $Decoder_{bf}$ (decoder for before nudged data) and $Decoder_{ndg}$ (decoder for nudging tendency) respectively.

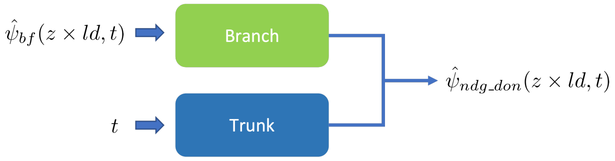

$\hat{\psi}_{bf}(z \times ld, t)$
Branch
t
Trunk
$\hat{\psi}_{ndg\_don}(z \times ld, t)$

Figure 2: A block diagram showing a high level view of the DeepONet. $\hat{\psi}_{bf}(z \times ld, t)$ (latent space representation of before nudged state) is the input for the branch network and t is the time which the input to the trunk network, which is the time for which we want to predict the nudging tendency $\hat{\psi}_{ndg\_don}(z \times ld, t)$.

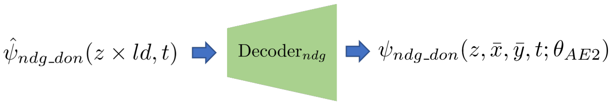

$\hat{\psi}_{ndg\_don}(z \times ld, t)$
Decoder$_{ndg}$
$\psi_{ndg\_don}(z, \bar{x}, \bar{y}, t; \theta_{AE2})$

Figure 3: A block diagram showing a high level view of the decoding process after DeepONet prediction. Once the prediction of the nudging tendency is obtained in the latent space, it is passed through the $Decoder_{ndg}$ to get the primitive space representation $\psi_{ndg\_don}(z, \bar{x}, \bar{y}, t; \theta_{AE2})$.

Algorithm 1: AE based DeepONet algorithm

```
Input: $\psi_{bf}, \psi_{ndg}, t$
Parameter: $\theta_{AE1}, \theta_{AE2}, \theta_{DeepONet}$
Output: $\psi_{ndg\_don}$
  $\psi_{bf} \rightarrow AE1 \rightarrow \hat{\psi}_{bf}$
  $\hat{\psi}_{bf}, t \rightarrow DeepONet \rightarrow \hat{\psi}_{ndg\_don}$
  $\hat{\psi}_{ndg\_don} \rightarrow AE2 \rightarrow \psi_{ndg\_don}$
  return $\psi_{ndg\_don}$
```

Results

To test our model, we train the DeepONet for a particular sub-region of the globe for the state variables U(zonal wind) and V (meridional wind). The latitude and longitude of the chosen sub region varies from 10N to 80N and 120W to 50W , respectively. This data is in a structured grid that has

Layer	Feature	Kernel	Activation
Conv2D	16	$(3,3)$	$ReLU$
maxpool		$(2,2)$	
Conv2D	8	$(3,3)$	$ReLU$
maxpool		$(2,2)$	
Conv2D	4	$(3,3)$	$ReLU$
maxpool		$(2,2)$	
Flatten			
Dense			

Table 1: Table representing architecture details of the encoder.

Layer	Feature	Kernel	Activation
Dense			$ReLU$
Reshape			
Transpose Conv2D	4	$(3,3)$	$ReLU$
Transpose Conv2D	8	$(3,3)$	$ReLU$
Transpose Conv2D	16	$(3,3)$	$ReLU$
Conv2D	4	$(3,3)$	
Reshape			
Dense			

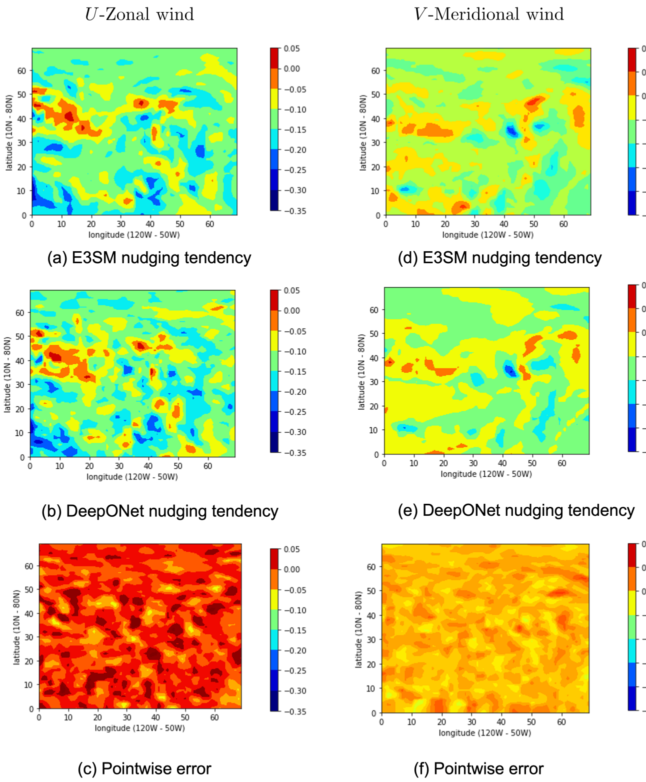
Table 2: Table representing architecture details of the decoder.

Branch-net	
Layer	Activation
Conv2D (feature=32,K=(3,3))	$ReLu$
Conv2D (feature=16,K=(3,3))	$ReLu$
Flatten	
Linear(ld)	tanh
Linear(32)	tanh
Linear(32)	tanh
Linear(32)	tanh
Linear($M \times ld$)	
Trunk-net	
Linear(32)	tanh
Linear(32)	tanh
Linear(32)	tanh
Linear(32)	tanh
Linear($M \times ld$)	

Table 3: Table representing the DeepONet architecture.

been interpolated from the original unstructured grid data. Although there is a difference among the points closer to the north pole as compared to the tropics, neural networks in general can capture the trend based on the data its being trained on, and hence takes into account this discrepancy between the points. The choice of the sub region is due to the larger variability of the atmospheric state U and V and is dominated by historical cyclones and storms. The DeepONet model is trained for the time span of October, which covers the famous hurricane Sandy in 2012. Due to the unavailability of different initial conditions, the train and test split was done along the dimension of the vertical atmospheric layers. 40 layers were randomly selected for the training and 16 layers were selected for the testing scenarios respectively. The results shown in this paper are for the testing cases. The DeepONet based model accurately predicts the nudging tendencies with a correlation higher than 0.7 with a confidence of 95%. A correct prediction of the storm track of Sandy shows the robustness of the DeepONet model. Figure 4, shows a sample snapshot and the pointwise error for the test results from the DeepONet for the state variables U (zonal-wind) and V (meridional wind).

In Figure 5, top/bottom left and top/bottom right sub-figures represent E3SM and DeepONet predicted nudging tendencies for U(zonal wind) and V (meridional wind), respectively.



(a) E3SM nudging tendency

(d) E3SM nudging tendency

(b) DeepONet nudging tendency

(e) DeepONet nudging tendency

(c) Pointwise error

(f) Pointwise error

Figure 4: Figure showing nudging tendency predicted by DeepONet for U (zonal wind) and V (meridional wind) and compared against the corresponding E3SM nudging tendency in the test domain. The Left and right column shows the E3SM nudging tendency, the DeepONet predicted nudging tendency, and the pointwise error for U and V respectively.

Summary

In this work, we developed a DeepONet based surrogate model, which successfully learns the bias correction in climate models. We first trained two auto-encoders for U

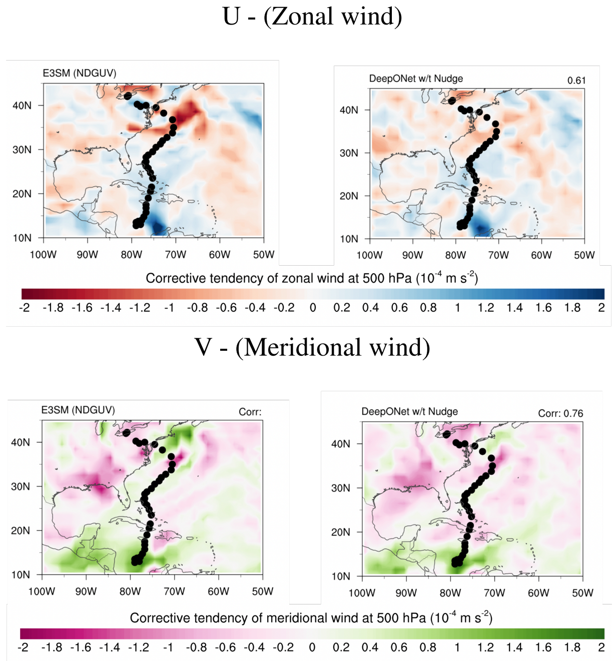


Figure 5: Figure showing the track of hurricane sandy in 2012. The left column sub-figures are from the E3SM nudged UV simulations and the right column sub-figures are the DeepONet simulation trained with nudging tendency for U(zonal wind) and V(meridional wind) respectively.

and V providing their latent representations. The evolution of latent representation of U and V are learned by DeepONet. Latent representation of features and labels helps DeepONet to learn the system dynamics fast and efficiently. The accuracy of DeepONet modes is established by testing it to predict the path of hurricane Sandy. The ongoing efforts include integration of the trained DeepONet model in the E3SM, so that it can be used to correct the biases in an online mode for more skillful simulations.

Acknowledgments

The authors would like to thank DARPA for funding this work through the project AI-assisted Climate Tipping-point Modeling (ACTM), Agreement No. HR00112290029. K.S. and A.B would also like to acknowledge Vivek Oommen (Brown University). K.S. and A.B would also like to acknowledge the use of the CCV's Oscar HPC cluster at Brown University.

References

Arribas, A.; Glover, M.; Maidens, A.; Peterson, K.; Gordon, M.; MacLachlan, C.; Graham, R.; Fereday, D.; Camp, J.; Scaife, A.; et al. 2011. The GloSea4 ensemble prediction system for seasonal forecasting. *Monthly Weather Review*, 139(6): 1891–1910.

Chen, T.; and Chen, H. 1995. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4): 911–917.

E3SM Project, D. 2018. Energy Exascale Earth System Model v1.0. [Computer Software] https://doi.org/10.11578/E3SM/dc.20180418.36.

Glahn, H. R.; and Lowry, D. A. 1972. The use of model output statistics (MOS) in objective weather forecasting. *Journal of Applied Meteorology and Climatology*, 11(8): 1203–1211.

Hersbach, H.; Bell, B.; Berrisford, P.; Hirahara, S.; Horányi, A.; Muñoz-Sabater, J.; Nicolas, J.; Peubey, C.; Radu, R.; Schepers, D.; et al. 2020. The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730): 1999–2049.

Lu, L.; Jin, P.; Pang, G.; Zhang, Z.; and Karniadakis, G. E. 2021. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3): 218–229.

Oommen, V.; Shukla, K.; Goswami, S.; Dingreville, R.; and Karniadakis, G. E. 2022. Learning two-phase microstructure evolution using neural operators and autoencoder architectures. *npj Computational Materials*, 8(1): 190.

Stockdale, T. N.; Anderson, D. L.; Alves, J. O. S.; and Balmaseda, M. A. 1998. Global seasonal rainfall forecasts using a coupled ocean–atmosphere model. *Nature*, 392(6674): 370–373.

Sun, J.; Zhang, K.; Wan, H.; Ma, P.-L.; Tang, Q.; and Zhang, S. 2019. Impact of nudging strategy on the climate representativeness and hindcast skill of constrained EAMv1 simulations. *Journal of Advances in Modeling Earth Systems*, 11(12): 3911–3933.

Watt-Meyer, O.; Brenowitz, N. D.; Clark, S. K.; Henn, B.; Kwa, A.; McGibbon, J.; Perkins, W. A.; and Bretherton, C. S. 2021. Correcting weather and climate models by machine learning nudged historical simulations. *Geophysical Research Letters*, 48(15): e2021GL092555.