Achieving Energy Conservation in Neural Network Emulators for Climate Modeling

Anonymous Authors[1]

1. Motivation

The largest source of uncertainty in climate projections is the response of clouds to warming (12). The turbulent eddies generating clouds are typically only $\mathrm{O}\left(100\mathrm{m} - 10\mathrm{km}\right)$ - wide, meaning that climate models need to be run at spatial resolutions as fine as $\mathrm{O}\left(1\mathrm{km}\right)$ to prevent large biases. Unfortunately, computational resources currently limit climate models to spatial resolutions of $\mathrm{O}\left(100\mathrm{km}\right)$ when run for time periods relevant to societal decisions, e.g. 100 years (7). Therefore, climate models rely on semi-empirical models of cloud processes, referred to as *convective parametrizations* (14; 13). If designed by hand, convective parametrizations are unable to capture the complexity of cloud processes and cause well-known biases, including a lack of extreme precipitation events and unrealistic cloud structures (5; 4).

Recent advances in statistical learning offer the possibility of designing data-driven convective parametrizations by training algorithms on short-period but high-resolution climate simulations (6). The first attempts have successfully modeled the interaction between small-scale clouds and the large-scale climate, offering a pathway to improve the accuracy of climate predictions (2; 11; 9). However, machine learning-based climate models do not intrinsically conserve energy and mass, which is a major obstacle to their adoption by the physical science community for several reasons, e.g.:

1) Realistic simulations of climate change respond to relatively small $\mathrm{O}\left(1\mathrm{W\ m^{-2}}\right)$ radiative forcing from carbon dioxide. Inconsistencies of this magnitude can prevent this small forcing from being communicated down to the surface and the ocean where most of the biomass lives.

2) Artificial sources and sinks of mass and energy impact the weather and cloud formation on short timescales, resulting in large temperature and humidity drifts or biases for the long-term climate.

Current machine-learning convective parametrizations that conserve energy are based on decision trees (e.g. random forests), making them too computationally expensive for practical use (10). Since neural-network convective parametrizations can significantly reduce cloud biases in climate models while decreasing their overall computational cost (11), we ask: How can we enforce conservation laws in neural-network emulators of physical models?

After proposing two methods to enforce physical constraints in neural network models of physical systems in section 2, we apply them to emulate cloud processes in a climate model in section 3, before comparing their performances and how they improve climate predictions in section 4.

2. Theory

Consider a physical system represented by a function $f : \mathbb{R}^m \mapsto \mathbb{R}^p$ that maps an input $x \in \mathbb{R}^m$ to an output $y \in \mathbb{R}^p$:

$$y = f\left(x\right). \tag{1}$$

Many physical systems verify exact physical constraints, such as the conservation of energy or momentum. In this paper, we assume that these physical constraints $(\mathcal{C})$ can be written as an under-determined linear system of rank n:

$$(\mathcal{C}) \stackrel{\mathrm{def}}{=} \left\{ \boldsymbol{C} \begin{bmatrix} x \\ y \end{bmatrix} = 0 \right\}, \tag{2}$$

where $\boldsymbol{C} \in \mathbb{R}^n \times \mathbb{R}^{m+p}$ is a constraints matrix acting on the input and output of the system. The physical system has n constraints, and by construction: $n < p + m$. Our goal is to build a computationally-efficient emulator of the physical system f and its physical constraints $(\mathcal{C})$. For the sake of simplicity, we build this emulator using a feed-forward neural network (NN) trained on preexisting measurements of x and y, as shown in Figure 1. We measure the quality of (NN) using the mean-squared error, defined as:

$$\mathrm{MSE}\left(y, y_{\mathrm{NN}}\right) \stackrel{\mathrm{def}}{=} \left\| y - y_{\mathrm{NN}} \right\| \stackrel{\mathrm{def}}{=} \frac{1}{p} \sum_{k=1}^{p} \left(y_i - y_{\mathrm{NN},i}\right)^2, \tag{3}$$

where y_{NN} is the neural network's output and y the "truth". Our reference case, referred to as "unconstrained neural network" (NNU), optimizes (NN) using MSE as its loss function. To enforce the physical constraints $(\mathcal{C})$ in our neural network, we consider two options:

1. **Constraining the loss function** (NNL): In this setting, we penalize our neural network for violating

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

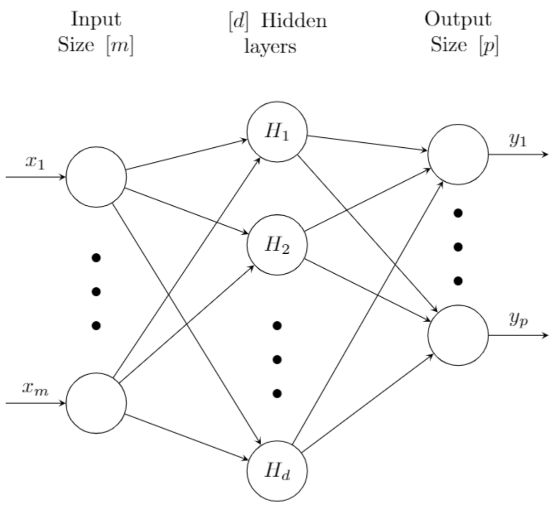

Figure 1. Standard feed-forward configuration (NN)

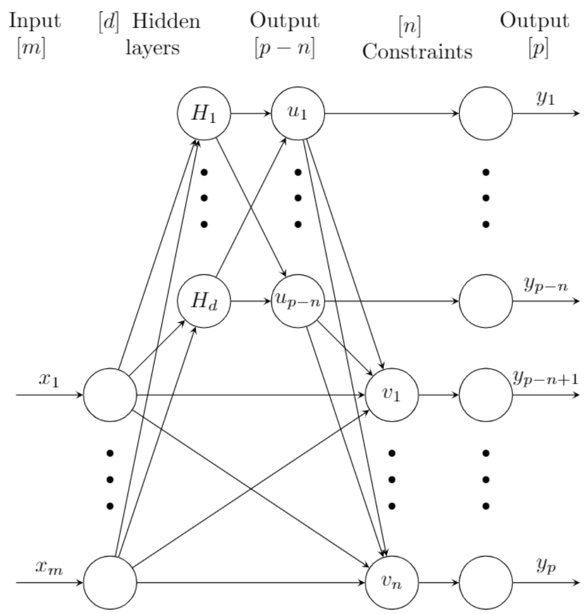

Figure 2. Architecture-constrained configuration (NNA)

physical constraints using a penalty $\mathcal{P}$, defined as the residual from the physical constraints:

$$\mathcal{P}\left(x, y_{\mathrm{NN}}\right) \stackrel{\text{def}}{=} \left\| \boldsymbol{C} \left[\begin{array}{c} x \\ y_{\mathrm{NN}} \end{array} \right] \right\|. \tag{4}$$

We apply this penalty by giving it a weight $\alpha \in [0,1]$ in the loss function $\mathcal{L}$, which is similar to a Lagrange multiplier:

$$\mathcal{L}(\alpha) = \alpha \mathcal{P}\left(x, y_{\mathrm{NN}}\right) + (1-\alpha)\,\mathrm{MSE}\left(y, y_{\mathrm{NN}}\right). \tag{5}$$

2. **Constraining the architecture** (NNA)**:** In this setting, we augment the simple network (NN) with n conservation layers to enforce the conservation laws $(\mathcal{C})$ to numerical precision (Figure 2), while keeping MSE as the loss function. The feed-forward network outputs an "unconstrained" vector $u \in \mathbb{R}^{p-n}$ whose size is only $(p-n)$, where n is the number of constraints. We then calculate the remaining component $v \in \mathbb{R}^n$ of the output vector y_{NN} using the n constraints. This defines n constraints layers $(\mathrm{CL}_{1..n})$ that ensure that the final output y_{NN} exactly respects the physical constraints $(\mathcal{C})$. A possible construction of $(\mathrm{CL}_{1..n})$ solves the system of equations $(\mathcal{C})$ from the bottom to the top row after writing it in row-echelon form via Gaussian elimination.

3. Application to Convective Parametrization for Climate Modeling

We now implement the three neural networks $(\mathrm{NNU}, \mathrm{NNL}, \mathrm{NNA})$ and compare their performances in the particular case of convective parametrization in the Super-Parametrized Community Atmosphere Model 3.0 (3; 8). We simulate an "ocean world" where the surface temperatures are fixed with a realistic equator-to-pole gradient (1). To facilitate the comparison, all networks have 5 hidden layers with 512 nodes each, and use leaky rectangular unit activation functions: $x \mapsto \max(0.3x, x)$ to better capture the system's non-linearity. We use the rmsprop optimizer (15) to train each network during 20 epochs, using 3 months of climate simulation with 30-minutes outputs as training data.

The goal of the neural network is to predict an output vector y of size 218 that represents the effect of cloud processes on climate, based on an input vector x of size 304 that represents the climate state. The 4 conservation laws can be written as a sparse matrix of size $4 \times (304 + 218)$ that acts on x and y to yield equation 2.

Each row of the conservation matrix $\boldsymbol{C}$ describes a different conservation law: The first row is the conservation of enthalpy, the second row is the conservation of mass, the third row is the conservation of terrestrial radiation and the last row is the conservation of solar radiation. In the architecture-constrained case, we output an unconstrained vector u of size $(218-4) = 214$, and calculate the 4 remaining components v of the output vector y by solving the system of equations $\boldsymbol{C} \left[\begin{array}{cc} x & y \end{array} \right]^T = 0$ from bottom to top.

We evaluate the performances of $(\mathrm{NNU}, \mathrm{NNL}, \mathrm{NNA})$ on two different validation datasets:

(+0K) An "ocean world" similar to the training dataset.

(+4K) An "ocean world" where the surface temperature has

Validation	Metric	NNU	$\text{NNL}_{\alpha=0.01}$	NNA	MLR	$\text{NNL}_{\alpha=0.5}$
+0K	MSE	$156 \pm 1.0 \times 10^3$	$154 \pm 1.0 \times 10^3$	$169 \pm 1.0 \times 10^3$	$295 \pm 1.7.10^3$	$177 \pm 1.1 \times 10^3$
	$\mathcal{P}$	$458 \pm 5 \times 10^2$	$125 \pm 2 \times 10^2$	$7 \times 10^{-10} \pm 1 \times 10^{-9}$	$28 \pm 2 \times 10^1$	5.0 ± 5
+4K	MSE	$633 \pm 7 \times 10^3$	$471 \pm 5 \times 10^3$	$567 \pm 8 \times 10^3$	$747 \pm 1 \times 10^5$	$496 \pm 8 \times 10^3$
	$\mathcal{P}$	$3 \times 10^5 \pm 1 \times 10^6$	$2 \times 10^3 \pm 1 \times 10^4$	$2 \times 10^{-9} \pm 5 \times 10^{-9}$	$265 \pm 2 \times 10^3$	$470 \pm 2 \times 10^3$

Table 1. Mean-Squared Error and Physical Constraints Penalty $\mathcal{P}$ for different neural networks in units $\mathrm{W}^2\ \mathrm{m}^{-4}$ using the format (Mean $\pm$ Standard deviation).

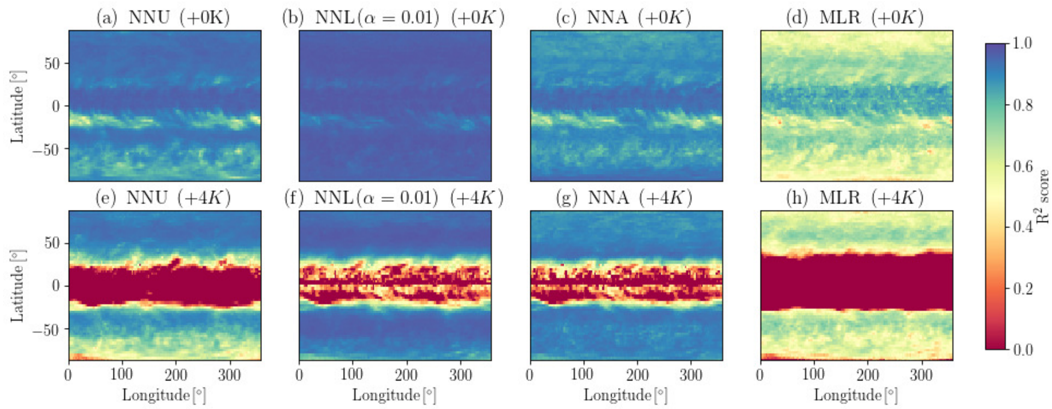

Figure 3. R^2 scores of different neural networks simulating the outgoing longwave radiation field over the entire planet for the (+0K) dataset (first row) and (+4K) dataset (second row).

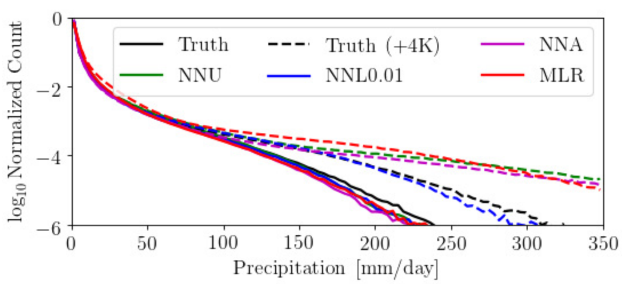

Figure 4. Histogram of precipitation for different neural networks over the (+0K, full lines) and the (+4K, dotted lines) validation datasets.

been uniformly warmed by 4K, a proxy for the effects of climate change.

4. Results

Table 1 compares the performance and the degree to which each neural network violates conservation laws, as measured by the mean-squared error and the penalty $\mathcal{P}$, respectively.

All neural networks perform better than the multiple-linear regression model (MLR), derived by replacing leaky rectangular units with the identity function and optimized independently. While the reference "unconstrained" network NNU performs well as measured by MSE, it does so by breaking conservation laws, resulting in a large penalty $\mathcal{P}$. Enforcing conservation laws via architecture constraints (NNA) works to satisfactory numerical precision on both validation datasets, resulting in a very small penalty $\mathcal{P}$. Giving equal weight to MSE and $\mathcal{P}$ in the loss function ($\text{NNL}_{\alpha=0.5}$) leads to mediocre performances in all areas. In contrast, introducing the penalty $\mathcal{P}$ in the loss function with a very small weight ($\alpha = 0.01$) leads to the best performance on the reference validation dataset $(+0K)$ and the best ability to generalize to unforeseen conditions $(+4K)$. The advantages of $\text{NNL}_{\alpha=0.01}$ are confirmed by the high R^2-score when predicting the outgoing longwave radiation (Figure 3), which can be used as a direct measure of radiative forcing in climate change scenarios.

Finally, while all networks reproduce the distribution of extreme precipitation events in the reference $(+0K)$ case, $\text{NNL}_{\alpha=0.01}$ is the only network that successfully generalizes its prediction skills to extreme precipitation events in a changed climate. Meanwhile, all other neural networks only perform marginally better than the MLR baseline. Overall, our results suggest that (1) constraining the network's architecture is a powerful way to ensure energy conservation over a wide range of climates and (2) introducing a very small information about physical constraints in the loss function can significantly improve the performance and generalization abilities of our neural network emulators.

References

[1] Joseph Allan Andersen and Zhiming Kuang. Moist static energy budget of MJO-like disturbances in the atmosphere of a zonally symmetric aquaplanet. *Journal of Climate*, 25(8):2782–2804, 2012.

[2] Noah Domino Brenowitz and Christopher Stephen Bretherton. Prognostic validation of a neural network unified physics parameterization. *researchgate.net*, 2018.

[3] William D. Collins, Cecilia M. Bitz, Maurice L. Blackmon, Gordon B. Bonan, Christopher S. Bretherton, James A. Carton, Ping Chang, Scott C. Doney, James J. Hack, Thomas B. Henderson, Jeffrey T. Kiehl, William G. Large, Daniel S. McKenna, Benjamin D. Santer, and Richard D. Smith. The Community Climate System Model version 3 (CCSM3). *Journal of Climate*, 19(11):2122–2143, jun 2006.

[4] C L Daleu, R S Plant, S J Woolnough, S Sessions, M J Herman, A Sobel, S Wang, D Kim, A Cheng, G Bellon, P Peyrille, F Ferry, P Siebesma, and L. Van Ulft. Intercomparison of methods of coupling between convection and large-scale circulation: 2. Comparison over nonuniform surface conditions. *Journal of Advances in Modeling Earth Systems*, 8(1):387–405, mar 2016.

[5] CL Daleu, RS Plant, and SJ Woolnough. Intercomparison of methods of coupling between convection and large-scale circulation: 1. Comparison over uniform surface conditions. *Journal of Advances*, 2015.

[6] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis. Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, jun 2018.

[7] IPCC. *Climate Change 2013 - The Physical Science Basis*. Cambridge, United KingdomNew York, NY, USA, cambridge edition, 2014.

[8] Marat Khairoutdinov, David Randall, and Charlotte DeMott. Simulations of the Atmospheric General Circulation Using a Cloud-Resolving Model as a Superparameterization of Physical Processes. *Journal of the Atmospheric Sciences*, 62(7):2136–2154, jul 2005.

[9] Vladimir M Krasnopolsky, Michael S Fox-Rabinovitz, and Alexei A Belochitski. 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. *Advances in Artificial Neural Systems*, 2013:1–13, 2013.

[10] Paul A O 'Gorman and John G Dwyer. Using machine learning to represent subgrid moist convection: potential for modeling of climate, climate change and extreme. *Journal of Advances in Modeling Earth Systems*, 2018.

[11] Stephan Rasp, Michael S Pritchard, and Pierre Gentine. Deep learning to represent sub-grid processes in climate models. *Proceedings of the National Academy of Sciences of the United States of America*, 115(39):9684–9689, sep 2018.

[12] Tapio Schneider, João Teixeira, Christopher S. Bretherton, Florent Brient, Kyle G. Pressel, Christoph Schär, and A. Pier Siebesma. Climate goals and computing the future of clouds. *Nature Climate Change*, 7(1):3–5, 2017.

[13] Steven C. Sherwood, Sandrine Bony, and Jean Louis Dufresne. Spread in model climate sensitivity traced to atmospheric convective mixing. *Nature*, 505(7481):37–42, jan 2014.

[14] Bjorn Stevens and Sandrine Bony. Water in the atmosphere. *Physics Today*, 66(6):29–34, jun 2013.

[15] Tijmen Tieleman, Geoffrey E. Hinton, Nitish Srivastava, and Kevin Swersky. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 4(2):26—-31, 2012.