https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Contribution of physical latent knowledge to the emulation of an atmospheric physics model: a study based on the LMDZ Atmospheric General Circulation Model

Ségolène Crossouard[1], Soulivanh Thao[1], Thomas Dubos[2], Masa Kageyama[1], Mathieu Vrac[1], and Yann Meurdesoif[1]

[1]Laboratoire des Sciences du Climat et de l'Environnement (LSCE-IPSL), CEA-CNRS-UVSQ-Université Paris-Saclay, 91190 Gif-sur-Yvette, France
[2]Laboratoire de Météorologie Dynamique (LMD-IPSL), Ecole Polytechnique, 91120 Palaiseau, France

Correspondence: Ségolène Crossouard (segolene.crossouard@lsce.ipsl.fr)

Abstract.

In an Atmospheric General Circulation Model (AGCM), the representation of subgrid-scale physical phenomena, also referred to as physical parameterizations, requires computational time which constrains model numerical efficiency. However, the development of emulators based on Machine Learning offers a promising alternative to traditional approaches. We have developed offline emulators of the physics parameterizations of an AGCM, ICOLMDZ, in an idealized aquaplanet configuration. The emulators reproduce the profiles of the tendencies of the state variables for each independent atmospheric column. In particular, we compare Dense Neural Network (DNN) and U-Net models. The U-Net provides better predictions in terms of mean and variance. For the DNN, while it consistently delivers good performances in predicting the mean tendencies, the variability is not well captured, posing challenges for our application. We then investigate why the DNN's predictions are poorer compared to those of the U-Net, in terms of physical processes. We find that turbulence is not well emulated by the DNN. Leveraging a priori knowledge of how turbulence is parameterized in the phyLMDZ model, we show that incorporating physical knowledge through latent variables as predictors into the learning process leads to a significant improvement of the variability emulated with the DNN model. This improvement brought by the addition of these new predictors is not limited to the DNN, as the U-Net has also shown enhanced results. This study hence emphasizes the importance of adding physical knowledge in Neural Network (NN) models to improve predictions and to ensure better interpretability. It opens perspectives on a deeper understanding of the emulator, as well as exploring the contribution of new physical predictors, aiming to make climate simulations and projections.

1 Introduction

Numerical climate models, developed since the 1950s, produce numerical simulations in order to predict the evolution of the state of the atmosphere (Balaji, 2021). These models are used for short-term weather forecasting but are also applied on longer time scales to simulate past and future climates. These models, also known as General Circulation Models (GCM), are crucial

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

for understanding climate mechanisms and projecting future climate change (Manabe and Wetherald, 1975). One of their core components is the atmospheric model, which includes two main distinct components, called dynamics and physics components. The dynamics component involves solving the Navier-Stokes-like equations on a three-dimensional grid, using a fluid dynamic solver to provide a numerical representation of atmospheric movements (Krasnopolsky et al., 2013). The physics component, which gathers the ensemble of physical parameterizations, is used to represent small-scale phenomena that affect atmospheric dynamics but cannot be explicitly resolved by the dynamical core because its grid is too coarse or because of dynamical assumptions such as the hydrostatic approximation (Balaji et al., 2022). Radiation, orographic waves, exchanges of energy and quantities of matter between the surface and the atmosphere, or cloud formation with vertical atmospheric movements known as convection and cloud-radiation interactions are examples of parameterized subgrid phenomena (Balaji et al., 2022). The dynamical core relies on well-established equations, whereas the parameterizations are based on heuristic/phenomenological/empirical approaches (Hourdin et al., 2017). Specifically, parameterizations are designed to represent the aforementioned phenomena, so that the general behavior of the atmosphere is reproduced at scales larger than the spatial resolution. The main purpose of a GCM is to iteratively calculate the evolution of state variables that control and describe meteorology and climate, starting from an initial condition and by updating the state of the climate system at each time step. This enables better understanding of past climates, analyses of the current climate and projections for the future climate by modifying boundary conditions and forcings, particularly in terms of greenhouse gas concentration. However, to generate these projections, a large number of simulations have to be carried out to cover the wide range of possible outcomes, sampling the variability of the factors influencing the climate system, including natural processes and human activities. A substantial number of simulations must also be computed due to the non-stationarity of the climate system, which needs to be considered when studying paleoclimates and extreme events. This large number of simulations requires a large computing time, in particular the physical component, which limits the numerical efficiency of the models. Moreover, as approximations, parameterizations of subgrid phenomena are still the source of large uncertainties in climate simulations (Sanderson et al., 2008; Balaji et al., 2022). Progress can be made in the representation of these sub-mesh phenomena, notably by developing a new generation of higher-resolution models (Bauer et al., 2021) or by using Machine Learning (ML) techniques (Schneider et al., 2017).

The burgeoning field of ML techniques opens new horizons, particularly with Deep Learning (DL) and its backbone, Neural Networks (NNs) (LeCun et al., 2015), not only to develop new parameterizations but also to produce accurate, robust and fast emulators of parts of a climate model. In particular, if they reliably reproduce physical processes, they would provide an efficient alternative to traditional process representation. These emulators could then replace one or more parameterizations that are computationally expensive and so, have the potential to enhance numerical efficiency, enabling the exploration of phenomena that are computationally expensive. There is currently a rich literature on the development of physical surrogate models since it is a research topic currently under active development. To provide a context for our study, we will emphasize four key aspects of the emulation of physical processes: the diversity of emulated parameterizations, the model configurations used, the ML algorithms developed and the addition of physical knowledge in the emulator.

Many studies focused on emulating specific existing parameterizations, usually the most expensive ones. These ML algorithms have been developed for ocean parameterizations (Guillaumin and Zanna, 2021) but also for atmospheric parameter-

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

izations, such as radiative transfer (Krasnopolsky et al., 2005; Pal et al., 2019; Roh and Song, 2020; Lagerquist et al., 2021; Belochitski and Krasnopolsky, 2021; Meyer et al., 2022), moist convection (Brenowitz and Bretherton, 2018; O'Gorman and Dwyer, 2018; Brenowitz et al., 2020), microphysics (Seifert and Rasp, 2020; Gettelman et al., 2021; Arnold et al., 2023), or even gravity wave drags (Chantry et al., 2021; Espinosa et al., 2022). On the other hand, some studies decided to tackle the emulation of all the parameterizations at once. It is the case of Gentine et al. (2018) and Rasp et al. (2018) who were the first to develop an emulator that learns physics tendencies of embedded Cloud-Resolving Models (CRMs). Their pioneering studies have shown promising results, as they developed an emulator based on Dense Neural Networks (DNNs), using an aquaplanet setup, with good capability to represent convection.

Like Gentine et al. (2018) and Rasp et al. (2018), some studies have opted to simplify the experiment of their model by using idealized aquaplanet configurations (Brenowitz and Bretherton, 2019; Chantry et al., 2021; Beucler et al., 2020; Yuval and O'Gorman, 2021; Behrens et al., 2022), while others have added orography or even continental surfaces directly (Han et al., 2020; Mooers et al., 2021; Wang et al., 2021; Bretherton et al., 2022; Han et al., 2023; Watt-Meyer et al., 2024; Hu et al., 2024), thereby increasing the complexity of the physical processes and making the emulation problem even more challenging.

Another aspect that distinguishes the various studies carried out is the choice of architecture, as a wide range of emulators have been developed using diverse ML designs. ML has been experiencing explosive growth in recent years, leading to increasingly sophisticated algorithms. Initially, DNNs of fully connected layers (Rasp et al., 2018; Pal et al., 2019) were used for emulation tasks, as well as other ML algorithms such as Random Forests (RFs) (O'Gorman and Dwyer, 2018; Limon and Jablonowski, 2023). Yuval et al. (2021) developed an emulator based on a DNN that learns from a high-resolution three-dimensional aquaplanet simulation, specifically capturing unresolved vertical processes that influence thermodynamic and moisture variables. They focused on comparing convection predictions made by the DNN and a RF developed by Yuval and O'Gorman (2020), for the same problem. The offline evaluation —where the emulator is not coupled to the rest of the model— demonstrated that DNN outperformed the RF algorithm while also using less memory. However the two models are comparable when the evaluation is done online. Furthermore, Convolutional Neural Networks (CNN) were introduced for emulation because of their ability to capture spatial and temporal relationships in data (Bolton and Zanna, 2019; Liu et al., 2020). Han et al. (2020) conducted a study to emulate convection and cloud parameterizations in a realistic configuration where the emulator was built from convolutional layers to which residual functions were added, forming a Residual Network (ResNet). In this architecture, the traditional connected layers were replaced by convolution layers. This study achieved good results in predicting temperature and moisture tendencies driven by moist physics processes. This ResNet model proved to be more accurate than a fully connected NN or a conventional CNN. More recently, studies focusing on emulating with advanced architectures have emerged such as U-Nets (Lagerquist et al., 2021; Hu et al., 2024), which have proved effective in handling more intricate multi-scale tasks by capturing spatial features at various resolutions. The results obtained by Heuer et al. (2024) suggest that the U-Net architecture seems to perform better and has a lower error than the other models tested to emulate the convective subgrid fluxes.

In general, emulators can be seen as black boxes due to their lack of interpretability (Reichstein et al., 2019). Incorporating physics-based knowledge into the emulator, either as input or in the architecture itself, can help to improve emulator perfor-

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

mance while reducing its black-box character. Some papers have explored techniques to add constraints to the emulator. For instance, Beucler et al. (2019) and Beucler et al. (2021) have incorporated physical knowledge into the loss function and have designed the architecture with conservation layers so that established conservation laws (mass, momentum and energy) are respected when emulating convective processes.

The present work investigates the potential of developing an emulator of the physical parameterizations of the ICOLMDZ Atmospheric General Circulation Model (AGCM) developed at IPSL (Hourdin et al., 2020; Dubos et al., 2015). The emulator could improve numerical performance, as currently almost half of the total computing time is given to the physical part of the model. We will evaluate the capabilities of several emulators aiming to reproduce all the physics of our standard model phyLMDZ while trying to explain their differences in performance. These emulators are based on DNNs and U-Nets. We chose to use DNNs after noticing that even relatively simple fully connected networks yielded promising results as reported in the literature. We also explored more complex architectures, particularly focusing on U-Nets which demonstrate robustness in capturing spatial variations thanks to their encoder-decoder design. In addition, Residual Blocks (ResBlocks) —skipping connections between block input and output to directly learn residual functions— are added to U-Net to facilitate model convergence. Using these models, our focus will be on how integrating physical information into the emulation process can contribute to building more efficient emulators. This work is conducted using an idealized aquaplanet configuration. All emulators are tested in an offline setup without coupling to the dynamics component.

The present paper is structured as follows. Section 2 describes the data used from the ICOLMDZ model, as well as the inputs and outputs of the emulators and the necessary preprocessing steps applied to it. Section 3 is dedicated to the methods used, including a description of the various NN architectures and the evaluation metrics. Section 4 provides an evaluation and a comparison of the first emulators of the physical parameterizations developed. Section 5 addresses the motivation for adding physical knowledge and focuses on the evaluation of new emulators that incorporate this physical knowledge as input. Section 6 presents a general discussion of the results. In Sect. 7, the results are summarized, and finally an outline of future work is given.

2 Data

2.1 ICOLMDZ simulation data

We use the IPSL climate model and more particularly its ICOLMDZ atmospheric component (see Fig. 1), which combines two main distinct components: DYNAMICO, the icosahedral dynamical core which solves the equations governing the atmospheric circulation (Dubos et al., 2015), and phyLMDZ, the atmospheric physics component which represents the other processes affecting the atmosphere (Hourdin et al., 2020). We use DYNAMICO at low resolution (200 km). It is composed of an unstructured grid of 16,002 cells and has a vertical resolution of 79 levels. It has the advantage of circumventing the polar singularity, unlike a regular longitude-latitude mesh, thus improving scalability. Like the dynamics part, phyLMDZ runs at a vertical resolution of 79 levels. However, in DYNAMICO, the calculations are performed on the three-dimensional grid, allowing horizontal exchanges with the surrounding grid points whereas phyLMDZ follows a single-column approach. In other

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

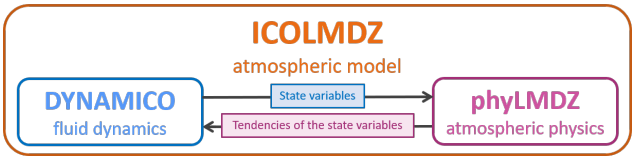

Figure 1. Scheme of the ICOLMDZ atmospheric model.

words, in phyLMDZ, calculations are done column by column without interaction between them, because subgrid processes involve transport in the vertical direction only.

In ICOLMDZ, DYNAMICO provides the state variables, such as temperature, humidity and winds, to phyLMDZ, which then calculates the tendencies associated with each state variable, for each physical phenomenon —or parameterization. We can define a tendency of a variable as the difference between two successive values of a given variable per unit time. For a state variable X at a time t, the total physical tendency $\left(\frac{\partial X}{\partial t}\right)_{phy}$ resulting from subgrid processes is the sum of the tendencies due to each individual process p, such as:

$$\left(\frac{\partial X}{\partial t}\right)_{phy} = \sum_{p} \left(\frac{\partial X}{\partial t}\right)_{p} \tag{1}$$

where p corresponds to the parameterizations. Then, phyLMDZ returns this tendency to DYNAMICO, which calculates the dynamical tendency $\left(\frac{\partial X}{\partial t}\right)_{dyn}$ and computes the evolution of the variable X for the next time step $t + \delta t$. Overall, we can write this integration of a given prognostic variable X, from one time step to the next, as:

$$X_{t+\delta t} = X_t + \left(\frac{\partial X}{\partial t}\right)_{dyn} \delta t + \left(\frac{\partial X}{\partial t}\right)_{phy} \delta t. \tag{2}$$

We decided to use ICOLMDZ in an idealized aquaplanet setup where the Sea Surface Temperature (SST) is prescribed and depends on latitude only (Medeiros et al., 2016). In this configuration, the diurnal cycle is activated, but the seasonal cycle is not. There is no topography or continent and we do not include the coupling to the land surface component of the IPSL model, ORCHIDEE (Krinner et al., 2005). We ran a one-year long simulation with variables output every 15 minutes, resulting in 96 time steps per day. This amounts to a total of 34,560 time steps for the entire year. From this aquaplanet simulation, we extracted the boundary conditions, the prognostic state variables and their tendencies. In the following sections of this study, the notation *d_X* will be adopted to denote the physical tendency of the variable *X*.

2.2 Input and output variables of emulators

Our goal is to develop a neural network to emulate all the physics of the phyLMDZ model. The input and output variables of the emulator were selected according to the phyLMDZ model design and are generated through the ICOLMDZ simulation. The input variables are key quantities computed by the dynamical core and provided to the phyLMDZ model, and also boundary conditions, while the output variables represent the subgrid physics tendencies calculated by phyLMDZ based on these input variables.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Table 1. List of input variables $\boldsymbol{x}$ for NNs.

Input variable $\boldsymbol{x}$	Notation	Unit	Vertical level(s)
Zonal wind	u	$\mathrm{m\ s^{-1}}$	79
Meridional wind	v	$\mathrm{m\ s^{-1}}$	79
Temperature	T	K	79
Humidity	*qx1*	$\mathrm{kg\ kg^{-1}}$	79
Liquid water	*qx2*	$\mathrm{kg\ kg^{-1}}$	79
Ice water	*qx3*	$\mathrm{kg\ kg^{-1}}$	79
Wind curl	*rot*	$\mathrm{s^{-1}}$	79
Solar zenith angle	*sza*	deg	1
Surface pressure	*psol*	Pa	1
Sea surface temperature	*SST*	K	1
Size of vector $\mathbf{X}$			556

More precisely, the inputs —or predictors— of the neural networks include the vertical profiles of the six state variables with a length of 79, corresponding to vertical levels z, where $z \in \{1, \ldots, 79\}$, such as the profile of zonal wind u, meridional wind v, temperature T, humidity *qx1*, liquid water *qx2* and ice water *qx3*. We also added four other variables: the vertical profile of the wind curl *rot* used for non-orographic gravity wave drag, and three scalars corresponding to boundary conditions, which are the solar zenith angle *sza*, the surface pressure *psol* and the sea surface temperature *SST*. All of these inputs, listed in Table 1, are concatenated in a vector $\mathbf{X}$ of size 556, such as:

$$\mathbf{X} = [u, v, T, qx1, qx2, qx3, rot, sza, psol, SST]\,.$$

The outputs —also referred to as predictands or targets— consist of six vectors, each with a length of 79. These vectors represent profiles of the physical tendencies of the zonal wind *d_u*, meridional wind *d_v*, temperature *d_T*, humidity *d_qx1*, liquid water *d_qx2* and ice water *d_qx3*. All these variables, listed in Table 2, are stacked in an output vector $\mathbf{Y}$ which has a length of 474 and is:

$$\mathbf{Y} = [d\_u, d\_v, d\_T, d\_qx1, d\_qx2, d\_qx3]\,.$$

2.3 Preprocessing

Data preprocessing is a crucial step in any ML process. It involves a series of essential tasks, which will be described in more detail in the following (see Fig. 2). The first step of preprocessing involves creating three datasets from our data. A training set is used for training the NN model by allowing it to learn patterns and relationships in the data, and optimize its parameters; a validation set helps prevent overfitting by evaluating how the model performs on unseen data; and a test set, used after training

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Table 2. List of output variables ***y*** for NNs.

Output variable *y*	**Notation**	**Unit**	**Vertical levels**
Zonal wind tendency	*d_u*	$\mathrm{m\ s^{-2}}$	79
Meridional wind tendency	*d_v*	$\mathrm{m\ s^{-2}}$	79
Temperature tendency	*d_T*	$\mathrm{K\ s^{-1}}$	79
Humidity tendency	*d_qx1*	$\mathrm{kg\ kg^{-1}\ s^{-1}}$	79
Liquid water tendency	*d_qx2*	$\mathrm{kg\ kg^{-1}\ s^{-1}}$	79
Ice water tendency	*d_qx3*	$\mathrm{kg\ kg^{-1}\ s^{-1}}$	79
Size of vector **Y**			474

and validation, provides a final evaluation of the model performance. To create these three datasets, we have split the data by selecting a percentage of data in the time dimension. We can adopt the following data segmentation since we do not have a seasonal cycle in our aquaplanet simulation. The first 60 % correspond to the training dataset, the next 20 % are used to create the validation dataset, and finally, the test dataset contains the remaining 20 %. Thus, we obtain three datasets with 20,736 time steps for the training dataset and 6,912 time steps for the other two datasets.

Then, in order to ensure that all features have an equal influence in the training process and also to improve the performance and stability of the emulator during this process, we standardized the inputs and outputs so that all variables are on the same scale. To achieve this, let us consider a three-dimensional input variable $\boldsymbol{x}$ from the training dataset. We first calculated the mean $\mu_z\left(\boldsymbol{x}^{\text{train}}\right)$ at each vertical level such as:

$$\mu_z\left(\boldsymbol{x}^{\text{train}}\right) = \frac{1}{N_t N_i} \sum_{t=1}^{N_t} \sum_{i=1}^{N_i} \boldsymbol{x}_{t,i,z}^{\text{train}} \tag{3}$$

where the index t indicates the time step, i denotes the horizontal grid point and z represents the vertical level. The variable $\boldsymbol{x}_{t,i,z}^{\text{train}}$ corresponds to the value of $\boldsymbol{x}$ in the training dataset at time step t, horizontal grid point i and vertical level z. N_t and N_i are respectively the total numbers of time steps and horizontal grid points in the training dataset. The standard deviation $\sigma\left(\boldsymbol{x}^{\text{train}}\right)$ was also calculated on the training dataset on all time steps and grid points, across the entire vertical column, for each variable. The variance is not calculated for each level like $\mu_z\left(\boldsymbol{x}^{\text{train}}\right)$ to avoid giving disproportionate weight to levels with low variance. The standard deviation can be expressed by the following:

$$\sigma\left(\boldsymbol{x}^{\text{train}}\right) = \sqrt{\frac{1}{N_t N_i N_z} \sum_{t=1}^{N_t} \sum_{i=1}^{N_i} \sum_{z=1}^{N_z} \left(\boldsymbol{x}_{t,i,z}^{\text{train}} - \mu_z\left(\boldsymbol{x}^{\text{train}}\right)\right)^2} \tag{4}$$

here N_z is the total number of vertical levels. Once these parameters are calculated, we subtract the mean of the corresponding vertical level and divide by the standard deviation, for each variable at each vertical level in each dataset. This leads to training data with a mean of 0 and a standard deviation of 1. Therefore, the standardization of the input variable $\boldsymbol{x}$, regardless of the

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

dataset, can be written as:

$$\boldsymbol{x}'_{t,i,z} = \frac{\boldsymbol{x}_{t,i,z} - \mu_z(\boldsymbol{x}^{\text{train}})}{\sigma(\boldsymbol{x}^{\text{train}})}. \tag{5}$$

Regarding the three scalar predictors that have no dependency on the vertical, the following standardization is applied:

$$\boldsymbol{x}'_{t,i} = \frac{\boldsymbol{x}_{t,i} - \mu(\boldsymbol{x}^{\text{train}})}{\sigma(\boldsymbol{x}^{\text{train}})} \tag{6}$$

with parameters $\mu(\boldsymbol{x}^{\text{train}}) = \frac{1}{N_t N_i}\sum_{t=1}^{N_t}\sum_{i=1}^{N_i}\boldsymbol{x}_{t,i}^{\text{train}}$ and $\sigma(\boldsymbol{x}^{\text{train}}) = \sqrt{\frac{1}{N_t N_i}\sum_{t=1}^{N_t}\sum_{i=1}^{N_i}\left(\boldsymbol{x}_{t,i}^{\text{train}} - \mu(\boldsymbol{x}^{\text{train}})\right)^2}$. As we are in the case of multivariate emulation, it is necessary to normalize the outputs. Equation (5) is thus also applied to the six three-dimensional output variables $\boldsymbol{y}$ with the appropriate parameters calculated using the Eq. (3) and (4). Once these preprocessing steps have been completed, standardized vectors $\mathbf{X}'$ and $\mathbf{Y}'$ are used.

We then perform temporal and spatial sub-sampling of the three datasets to avoid spatio-temporal correlations and to reduce the cost of training. For this purpose, starting from the first time step for each dataset, we select one data point every ten time steps and thus obtain datasets with 2,074 time steps for the training dataset and 692 time steps for the validation and test datasets. Moreover, we choose to take one cell every twenty cells to create the final datasets. Finally, the training dataset contains data from 801 cells and the two others have 800 cells spread across the globe among the 16,002 cells. We took care to ensure that this spatial selection represented all the regions. Therefore, we have 1,661,274 samples (2,074 $\times$ 801 grid points) for the training dataset. We validate and test our models on two datasets with 553,600 samples (692 $\times$ 800 grid points). These numbers of samples are multiplied by the length of the input vector $\mathbf{X}'$ and the output vector $\mathbf{Y}'$, for all the three subsets, which correspond to variables aggregated over the vertical dimension. We have also retained the test dataset where only temporal sub-sampling has been applied, in order to study the results of the emulators across all grid cells of the aquaplanet.

3 Methods

3.1 Neural network architectures

An ML emulator estimates the output variables $\hat{\mathbf{Y}}'$ from predictors $\mathbf{X}'$ which have corresponding outputs $\mathbf{Y}'$ in the dataset, using a transfer function denoted $\hat{f}$, such as $\hat{\mathbf{Y}}' = \hat{f}(\mathbf{X}';\boldsymbol{\theta})$ where $\boldsymbol{\theta} = (\mathbf{w}, \mathbf{b})$ with $\mathbf{w}$ the weights and $\mathbf{b}$ the biases. Therefore, the goal of NNs during training is to minimize the error between prediction $\hat{\mathbf{Y}}'$ and reference data $\mathbf{Y}'$. In other words, the objective is to optimize the function $\mathcal{L}(\mathbf{Y}', \hat{\mathbf{Y}}')$ —referred to as the loss function in ML jargon— by updating the weights and biases of the model iteratively (LeCun et al., 2015). The challenge with a ML model is to find a balance between underfitting and overfitting. The first one occurs when the correlations in the training data are not captured by the predictive model because it is inappropriate. The second one arises when the predictive model fits the training data well, and fails to generalize to data that it has not yet seen during the training process. In other words, the model captures the correlations but also the noise produced by the training dataset (Goodfellow et al., 2016).

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

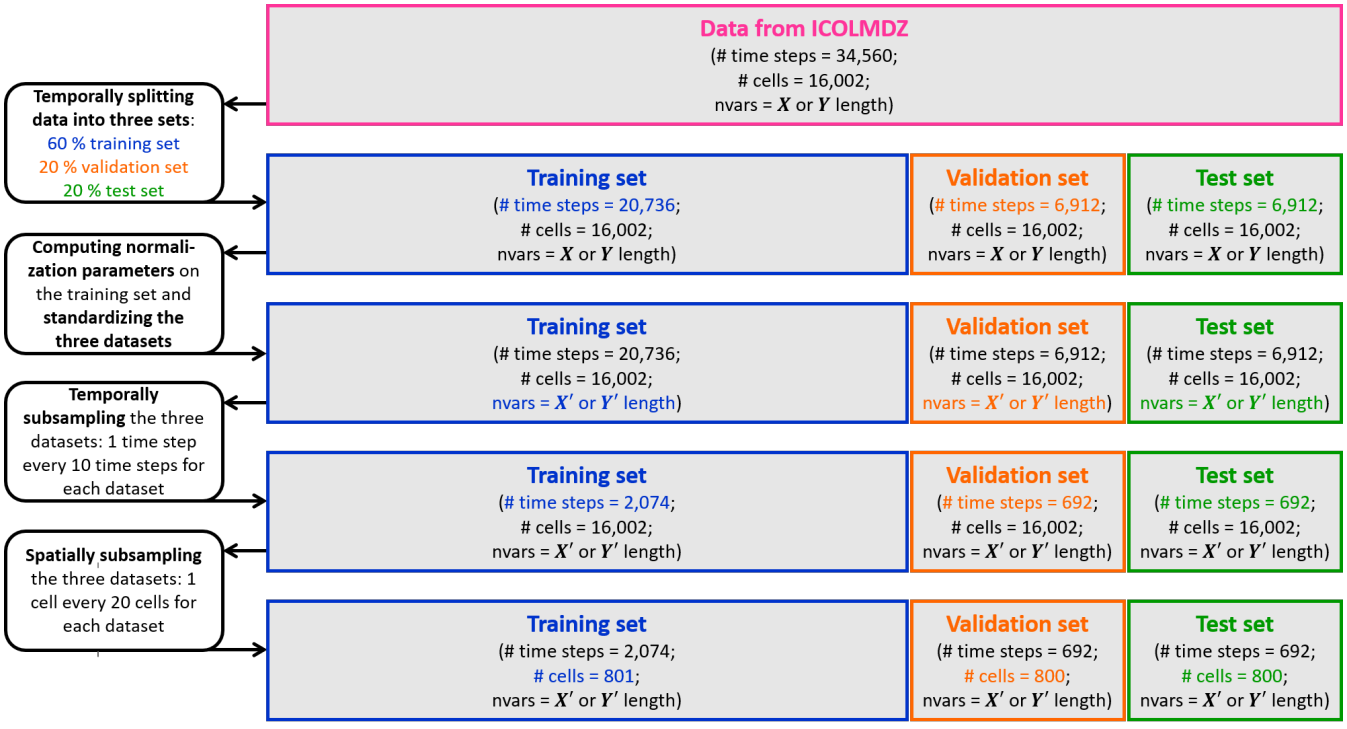

Figure 2. Diagram of the data preprocessing steps.

We want to develop an emulator encompassing all the physics of our standard model. In other words, its role is to provide predictions of the subgrid physical tendencies for each level of each atmospheric column at every time step. To this end, we have investigated and implemented several NN-based emulators using the Keras API (Chollet, 2015) and TensorFlow framework (Abadi et al., 2015). These include DNNs (Goodfellow et al., 2016), Convolutional Neural Networks (CNNs) (LeCun et al., 1990), U-Nets (Ronneberger et al., 2015), and U-Nets with ResBlocks (He et al., 2016). For these emulators, the hyperparameters were selected based on preliminary tests. Adam is the optimization algorithm chosen for gradient descent, which allows the weights to be updated iteratively and helps find the optimal values for these parameters. For each NN model, the input vector $\mathbf{X}'$ passes through a first input layer. The last layer corresponds to a dense layer with a size of the output vector $\mathbf{Y}'$. This last layer has no activation function due to the nature of the problem, which is a regression task. Below, we will provide descriptions of only a DNN model and a U-Net model with residual blocks, as these are the two models that will be studied throughout the rest of the article.

The first architecture we use is a DNN which consists of a succession of dense layers described in Table 3. It is composed by six connected hidden layers of 512 neurons each, with a full connectivity between adjacent layers. The activation function of the hidden layers is Rectified Linear Unit (ReLU) (Glorot et al., 2011; LeCun et al., 2015). It is trained with the learning rate scheduler of 10^{-3}, and the batch size is equal to 64. This neural network contains around 2 million parameters.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Table 3. Description of the DNN architecture used for the first learning —the initial training without laplacian— and the second learning —with laplacians as additional predictors.

Step	**Layer**	**Number of neurons**	**Number of parameters 1st learning**	**Number of parameters 2nd learning**
Input	Input	$\mathbf{X}'$ length	0	0
Hidden layers	Dense	512	285,184	527,872
	Dense	512	262,656	262,656
	Dense	512	262,656	262,656
	Dense	512	262,656	262,656
	Dense	512	262,656	262,656
	Dense	512	262,656	262,656
Output	Dense	$\mathbf{Y}'$ length	243,162	243,162
Total number of parameters			1,841,626	2,084,314

We have also developed a specific CNN architecture called a U-Net model (Ronneberger et al., 2015). This model has the advantage of extracting multi-scale features and capturing spatial dependencies between features through a series of convolution operations. It is known for its encoder-decoder structure, featuring a dual-path design: a down-sampling side for capturing contextual information by reducing spatial resolution, and an up-sampling side for achieving precise localization by increasing the resolution. For a given resolution, the encoding and decoding parts are linked with skip connections, and at the lowest resolution, these parts are connected via a bottleneck layer. Combined with this u-shaped architecture, ResBlocks have been included in this model whose aim is to predict the errors —or residuals— made by the model. Indeed, these blocks allow gradients to propagate more effectively during backpropagation, resulting in better convergence. Figure 3 gives a schematic description of the U-Net architecture built and Table 4 summarizes the layers. The temporal and spatial dimensions of data are merged before data is fed into this NN. The reverse operation if performed at the output. At the input, the 3 scalars are separated from the vectors in order to extend them across all vertical levels, and then concatenated with the other vector variables, so they can be integrated during the convolutional layers. This architecture requires 2D data, which is why the input data vector is reformatted into a 2D tensor. At the output, a 1D convolutional layer and a reshape layer are used to transform the data back into a vector format. The learning rate is set at 10^{-3}, and the batch size equals to 64. This model has around 13 million parameters.

Once the models had been built, we trained them. Each emulator is trained for a certain number of epochs, with each epoch representing a complete pass of the training dataset through the algorithm. To evaluate the performance of emulators during training, the output tendencies from the NNs are compared with those from ICOLMDZ. The quantification of the difference between the NNs predictions and the reference data in the training dataset is done using a loss function $\mathcal{L}$. We have selected the Mean Squared Error (MSE), commonly used in regression problems, as the loss function. MSE is sensitive to outliers since

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

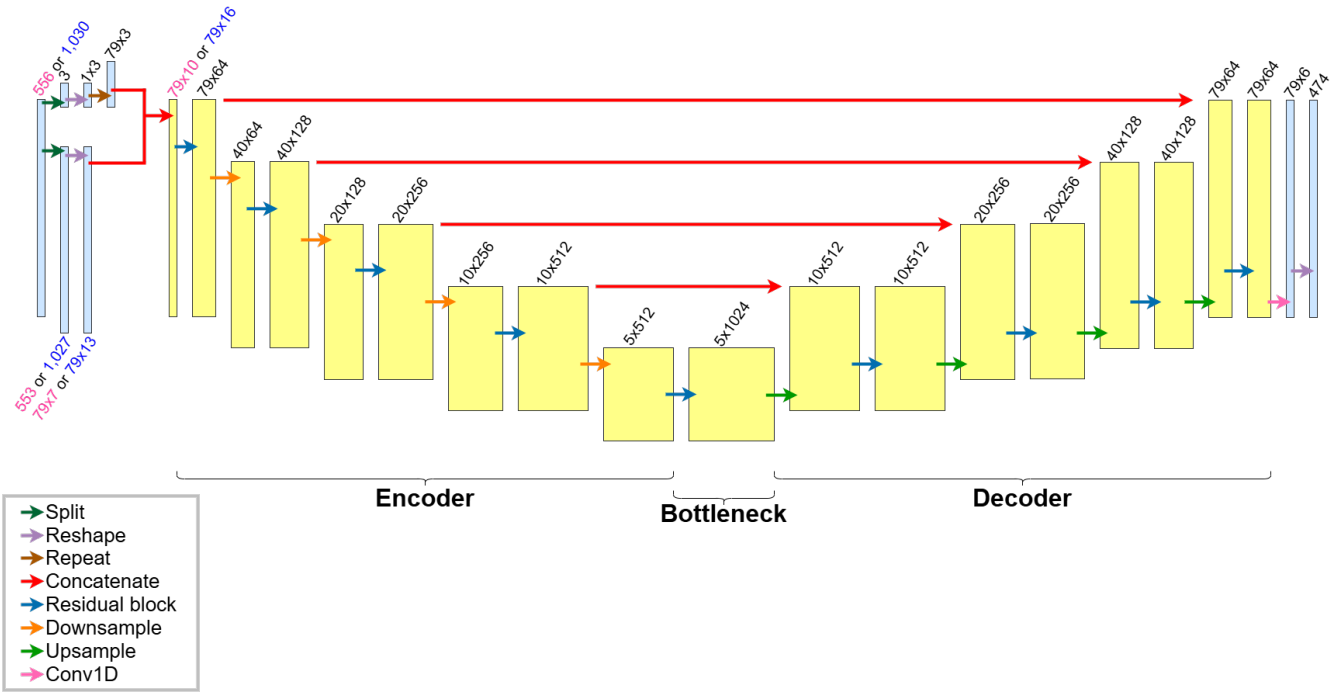

Figure 3. Diagram of the U-Net architecture. The blue and yellow boxes schematize the multi-channel features. The blue boxes correspond to the data formatting since the U-Net architecture —represented by the yellow boxes— works with 2D data whose dimensions correspond to the vertical levels times the number of variables. On the top (or sometimes the bottom) of each box is written the shape of the corresponding vector or tensor. The dimensions written in pink or blue at the start of the architecture are used to distinguish the size of data between the 1st learning (without laplacian) and the 2nd (with laplacians), respectively. The coloured arrows represent the different operations performed.

large errors are penalized much more than small one. It is defined as follows:

$$\mathrm{MSE}\left(\mathbf{Y}', \hat{\mathbf{Y}}'\right) = \frac{1}{N}\sum_{j=1}^{N} \|\mathbf{Y}'_j - \hat{\mathbf{Y}}'_j\|^2 \tag{7}$$

where $\|\|_2$ denotes the Euclidean norm, N is the total number of samples in the dataset, $\mathbf{Y}'_j$ the target and $\hat{\mathbf{Y}}'_j$ the prediction for the j^{th} value on the dataset. In our case, the loss function is computed directly on the standardized data, given that we are working in a multivariate setting. MSE is calculated on the training dataset but also on the validation dataset. Indeed, early stopping criterion has been used to prevent overfitting since this option monitors model performance on a validation dataset during the training process and stops when performance no longer improves. We set a patience of 20 epochs. In this article, the prediction performance of the DNN and the U-Net described above will be compared.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Table 4. Description of the U-Net architecture for the first training and the second. The pink and blue colours are used to distinguish differences in data shape between the 1st learning (without laplacian) and the 2nd (with laplacians), respectively. A residual block is composed by the succession of the following layers: Conv1D, BatchNormalization, ReLU, Conv1D, BatchNormalization, Add —to sum the input with the transformed output— and ReLU. A downsample layer corresponds to a MaxPooling1D. An upsample layer encompasses a Conv1DTranspose layer and a Concatenate layer which corresponds to the skip connection.

Step	**Layer**	**Number of features**	**Number of filters (or channels)**	**Number of parameters 1st learning**	**Number of parameters 2nd learning**
Input	Input	$\mathbf{X}'$ length	-	0	0
	Split	553 and 3	-	0	-
		1,027 and 3	-	-	0
	Lambda (reshape)	79	7	0	-
		79	13	-	0
		1	3	0	0
	Lambda (repeat)	79	3	0	0
	Concatenate	79	10	0	-
		79	16	-	0
Encoder	Residual block	79	64	15,808	17,344
	Downsample 1	40	64	0	0
	Residual block	40	128	83,840	83,840
	Downsample 2	20	128	0	0
	Residual block	20	256	233,216	233,216
	Downsample 3	10	256	0	0
	Residual block	10	512	1,318,400	1,318,400
	Downsample 4	5	512	0	0
Bottleneck	Residual block	5	1024	5,258,240	5,258,240
Decoder	Upsample 1	10	512	1,573,376	1,573,376
	Residual block	10	512	2,891,264	2,891,264
	Upsample 2	20	256	393,472	393,472
	Residual block	20	256	724,736	724,736
	Upsample 3	40	128	98,432	98,432
	Residual block	40	128	182,144	182,144
	Upsample 4	80	64	24,640	24,640
	Residual block	79	64	46,016	46,016
Output	Conv1D	79	6	1,158	1,158
	Lambda (reshape)	$\mathbf{Y}'$ length	-	0	0
Total number of parameters				12,844,742	12,846,278

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

3.2 Evaluation metrics

After the learning process is completed and parameters have been set, the final evaluation is done on the test dataset. We evaluate the offline performance of our emulator with different statistical indicators for each physical tendency that has been previously denormalized. Firstly, to perform a one-dimensional analysis of the results, we synthesized temporal and spatial information by computing statistics on all time steps and grid points. This gives an initial overview of the results of an emulator despite the spatio-temporal complexity of the problem. We calculate the global mean and the global variance of the six physical tendencies separately for each vertical level, over all time steps and all grid points combined. These metrics are applied to both the reference tendencies $\mathbf{Y}$ from the ICOLMDZ simulation and the predicted tendencies $\hat{\mathbf{Y}}$, in order to compare them. If we consider a reference physical tendency noted $\boldsymbol{y}$ from the test subset, then its global mean $\mu_z\left(\boldsymbol{y}^{\text{test}}\right)$ at vertical level z is expressed as:

$$\mu_z\left(\boldsymbol{y}^{\text{test}}\right) = \frac{1}{N_t N_i} \sum_{t=1}^{N_t} \sum_{i=1}^{N_i} \boldsymbol{y}_{t,i,z}^{\text{test}} \tag{8}$$

where N_t and N_i are respectively the total numbers of time steps and grid points in the test dataset. The term $\boldsymbol{y}_{t,i,z}^{\text{test}}$ include data for all time steps, grid points and vertical levels. The global variance $\sigma_z^2\left(\boldsymbol{y}^{\text{test}}\right)$ of this physical tendency $\boldsymbol{y}$ is given as follows:

$$\sigma_z^2\left(\boldsymbol{y}^{\text{test}}\right) = \frac{1}{N_t N_i} \sum_{t=1}^{N_t} \sum_{i=1}^{N_i} \left(\boldsymbol{y}_{t,i,z}^{\text{test}} - \mu_z\left(\boldsymbol{y}^{\text{test}}\right)\right)^2. \tag{9}$$

These two metrics are calculated for each vertical level, allowing us to examine the global vertical profiles of the mean and the variance of the tendency $\boldsymbol{y}$. Equations (8) and (9) are also applied to each predicted physical tendency $\hat{\boldsymbol{y}}$ to compare their results with those of $\boldsymbol{y}$, providing a first overall assessment of the emulator studied.

Moreover, we used two metrics for each tendency separately over all time steps, cells and vertical levels to assess the general performance of an emulator. Firstly, the coefficient of determination, denoted R^2, is defined as follows:

$$\mathrm{R}^2\left(\hat{\boldsymbol{y}}^{\text{test}}\right) = 1 - \frac{\mathrm{MSE}\left(\boldsymbol{y}^{\text{test}}, \hat{\boldsymbol{y}}^{\text{test}}\right)}{\mathrm{MSE}\left(\boldsymbol{y}^{\text{test}}, \mu(\boldsymbol{y}^{\text{test}})\right)} \tag{10}$$

$$= 1 - \frac{\sum_{t=1}^{N_t} \sum_{i=1}^{N_i} \sum_{z=1}^{N_z} \left(\boldsymbol{y}_{t,i,z}^{\text{test}} - \hat{\boldsymbol{y}}_{t,i,z}^{\text{test}}\right)^2}{\sum_{t=1}^{N_t} \sum_{i=1}^{N_i} \sum_{z=1}^{N_z} \left(\boldsymbol{y}_{t,i,z}^{\text{test}} - \mu\left(\boldsymbol{y}^{\text{test}}\right)\right)^2} \tag{11}$$

where N_z corresponds to the 79 vertical levels and $\mu\left(\boldsymbol{y}^{\text{test}}\right)$ is the average of $\boldsymbol{y}_{t,i,z}^{\text{test}}$. We have also calculated the global Root Mean Square Error (RMSE) which is:

$$\mathrm{RMSE}\left(\boldsymbol{y}^{\text{test}}, \hat{\boldsymbol{y}}^{\text{test}}\right) = \sqrt{\mathrm{MSE}\left(\boldsymbol{y}^{\text{test}}, \hat{\boldsymbol{y}}^{\text{test}}\right)} \tag{12}$$

$$= \sqrt{\frac{1}{N_t N_i N_z} \sum_{t=1}^{N_t} \sum_{i=1}^{N_i} \sum_{z=1}^{N_z} \left(\boldsymbol{y}_{t,i,z}^{\text{test}} - \hat{\boldsymbol{y}}_{t,i,z}^{\text{test}}\right)^2}. \tag{13}$$

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

We will sometimes compute the RMSE on specific subsets of the test dataset to exhibit performances that vary for instance across latitudes or vertical levels.

4 Initial training performance

The emulation of the six physical tendencies is performed simultaneously. However, we will first focus on the results of the zonal wind tendency, noted as *d_u*, and then we will briefly analyze the other five tendencies, for which figures are provided in the Supplementary Material.

4.1 Results

First, we examine the global vertical profiles of the mean (see Fig. 4a) and variance (see Fig. 4b) of the zonal wind tendency, to analyze the results obtained from our emulators with the test subset which contains the 16,002 cells. These profiles were created to bring both spatial and temporal information. Indeed, they were produced by averaging data over the whole globe and over all time steps that define the test subset. For the DNN, the predicted global mean profile of the zonal wind tendency shows a good overall agreement with the reference global mean between levels 25 and 79, even though the predicted curve does not perfectly match the reference data. Indeed, the curve of the predicted global mean oscillates around the reference global mean. Below level 25, the global mean predicted using the emulator deviates significantly from the reference data. Moreover, the global variance is underestimated in the predictions, with a variance ranges from 0 to 2.7×10^{-7} $\mathrm{m^2\ s^{-4}}$ —equivalent to a standard deviation ranges from 0 to 5.2×10^{-4} $\mathrm{m\ s^{-2}}$—, compared to that of the reference data, whose variance is between 0 and 4.5×10^{-7} $\mathrm{m^2\ s^{-4}}$ —i.e. a standard deviation ranges from 0 to 6.7×10^{-4} $\mathrm{m\ s^{-2}}$. This is particularly notable at lower vertical levels between levels 0 and 10 where there is higher variability in the reference data, posing a challenge for the emulator. As a result, while the emulator can predict the zonal wind tendency reasonably well on average, it struggles to capture extreme values. For the U-Net, the predicted mean global profile is smoother and aligns closely with the reference mean global profile except for the very first levels. The predicted variance ranges from 0 to 3.5×10^{-7} $\mathrm{m^2\ s^{-4}}$. It is similar to the reference variance for vertical levels above 30 but, like the DNN, the predicted global variance profile is underestimated between levels 0 and 25. However, on the other hand, when comparing the results obtained from both the DNN and the U-Net, we can highlight that the U-Net model produces better predictions than the DNN, whether in terms of global mean or global variance. These results lead us to explore the factors contributing to make the U-Net more effective and accurate than the DNN, as well as the reasons behind the underestimation of the variance from the DNN.

We compute some general statistical indicators to evaluate the global performance of the emulators in reproducing the zonal wind tendency *d_u*. Table 5 and Fig. 5 show the results. By comparing the global RMSE of the DNN and the U-Net, we observe that the DNN has a RMSE of 1.48×10^{-4} $\mathrm{m\ s^{-2}}$ and that of the U-Net is equal to 1.23×10^{-4} $\mathrm{m\ s^{-2}}$. Therefore, with a global RMSE approximately 16.9 $\%$ lower than that of the DNN, the U-Net architecture outperforms the DNN. This statement can also be made based on the results of the coefficient of determination. Indeed, this metric is equal to 0.63 for the U-Net, whereas

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

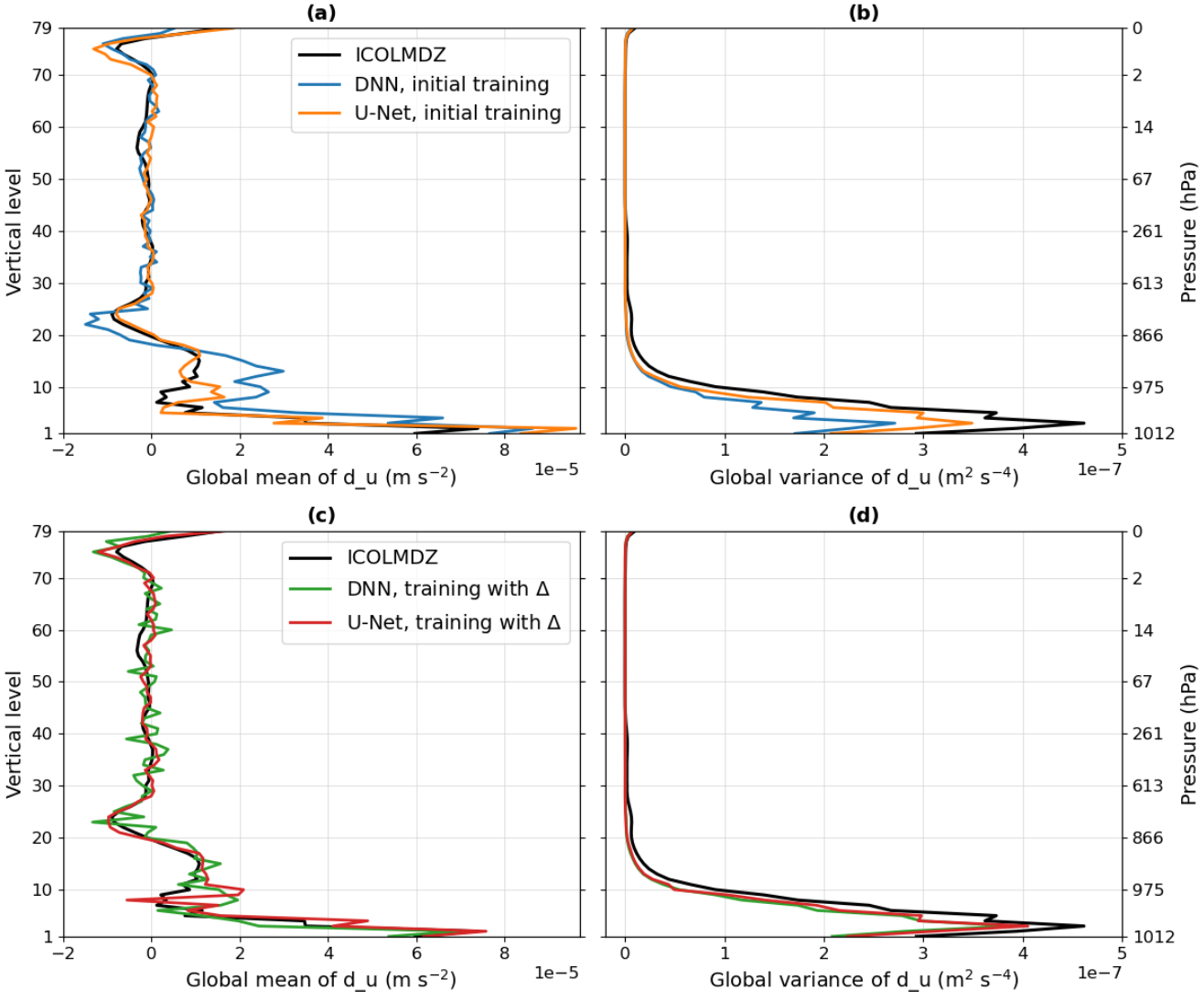

Figure 4. Global vertical profiles of the zonal wind tendency *d_u* of the mean (left) and the variance (right) for the test subset. Figures **(a)** and **(b)** show the results of the initial training, while figures **(c)** and **(d)** correspond to the second training where physical knowledge is added. In these figures, the black curves represent the reference data from the ICOLMDZ simulation. The blue and green curves correspond to predictions made by DNNs while the orange and red curves represent the results obtained using U-Nets.

it equals 0.46 for the DNN. This confirms that the DNN captures less variance than the U-Net model. Therefore, it is obvious that the U-Net architecture presents better skills at predicting the zonal wind tendency.

We have calculated the zonal RMSE of the zonal wind tendency for the DNN and the U-Net in order to study this metric's dependence on altitude and latitude (see Fig. 6a and 6b). The first notable observation, regardless of the emulator, is the North-

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Table 5. Results of the general metrics for the six tendencies over all time steps, cells and vertical levels, for the test subset, to evaluate the four NN-models studied.

Tendency	NN-model	RMSE $(y^{\text{test}}, \hat{y}^{\text{test}})$	R^2 $(\hat{y}^{\text{test}})$
d_u	DNN, initial training	1.48×10^{-4} m s^{-2}	0.46
	U-Net, initial training	1.23×10^{-4} m s^{-2}	0.63
	DNN, training with Δ	1.36×10^{-4} m s^{-2}	0.54
	U-Net, training with Δ	$\mathbf{1.16 \times 10^{-4}}$ **m s^{-2}**	**0.67**
d_v	DNN, initial training	1.35×10^{-4} m s^{-2}	0.48
	U-Net, initial training	1.18×10^{-4} m s^{-2}	0.60
	DNN, training with Δ	1.28×10^{-4} m s^{-2}	0.53
	U-Net, training with Δ	$\mathbf{1.13 \times 10^{-4}}$ **m s^{-2}**	**0.64**
d_t	DNN, initial training	3.37×10^{-5} K s^{-1}	0.74
	U-Net, initial training	3.07×10^{-5} K s^{-1}	0.78
	DNN, training with Δ	3.22×10^{-5} K s^{-1}	0.76
	U-Net, training with Δ	$\mathbf{2.97 \times 10^{-5}}$ **K s^{-1}**	**0.79**
d_qx1	DNN, initial training	2.50×10^{-8} kg kg^{-1} s^{-1}	0.34
	U-Net, initial training	2.05×10^{-8} kg kg^{-1} s^{-1}	0.56
	DNN, training with Δ	2.18×10^{-8} kg kg^{-1} s^{-1}	0.50
	U-Net, training with Δ	$\mathbf{1.90 \times 10^{-8}}$ **kg kg^{-1} s^{-1}**	**0.62**
d_qx2	DNN, initial training	5.12×10^{-9} kg kg^{-1} s^{-1}	0.39
	U-Net, initial training	4.69×10^{-9} kg kg^{-1} s^{-1}	0.49
	DNN, training with Δ	4.91×10^{-9} kg kg^{-1} s^{-1}	0.44
	U-Net, training with Δ	$\mathbf{4.57 \times 10^{-9}}$ **kg kg^{-1} s^{-1}**	**0.51**
d_qx3	DNN, initial training	7.50×10^{-10} kg kg^{-1} s^{-1}	0.61
	U-Net, initial training	7.17×10^{-10} kg kg^{-1} s^{-1}	0.65
	DNN, training with Δ	6.78×10^{-10} kg kg^{-1} s^{-1}	0.68
	U-Net, training with Δ	$\mathbf{6.76 \times 10^{-10}}$ **kg kg^{-1} s^{-1}**	**0.69**

South symmetry which is expected in our context as we study an aquaplanet with a constant equinox without seasonal cycle. Then, the results highlight the fact that both emulators exhibit a significantly higher zonal RMSE of the zonal wind tendency —equal to 5.0×10^{-4} m s^{-2}— in the first 10 vertical levels. These results align with the global vertical profiles of the variance analyzed previously, since it is at these vertical levels that the variability of the zonal wind tendency is most pronounced. Moreover, we see that emulators perform better in some regions, particularly near the equator and at high latitudes —above 60° S and 60° N —compared to the mid-latitudes— between 30° S and 60° S and between 30° N and 60° N. However, the

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

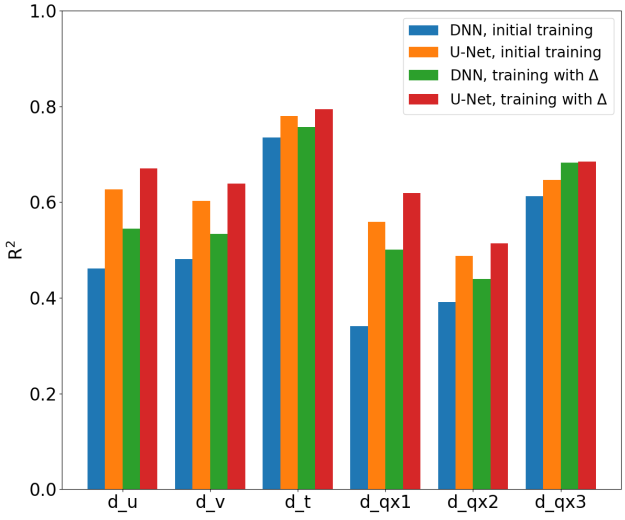

Figure 5. Coefficient of determination R^2 of the test subset for the six tendencies with each NN.

U-Net architecture performs better, as the pattern of high zonal RMSE is less widespread, especially at high latitudes around 60° S and 60° N. For each emulator, we observe quite similar patterns for the levels below 40, with a zonal RMSE around 1.5×10^{-4} m s^{-2}, mainly pronounced from latitudes 30° S and 30° N to the poles. The metric studied is weak above vertical level 40 at all latitudes. Indeed, neither emulator seems to be better at these vertical levels, particularly with regard to the representation of variance and this result can be seen in the zonal RMSE. This poses a need to further investigate and examine the results of the emulation by latitude bands, particularly between latitudes 30° and 60° in each hemisphere, in the first vertical levels.

4.2 Study by latitude bands

To better understand the results of the emulators, we examine them study by latitude belts, as the zonal wind tendency is the result of various physical processes that do not operate in the same way across regions. For this spatial study, we defined three latitude intervals for which global mean and variance profiles of the zonal wind tendency are calculated (Fig. 7). The first one covers the tropics, i.e. for data with a latitude, noted ϕ, in the range $[25°\ \mathrm{S},\ 25°\ \mathrm{N}]$ (see Fig. 7a and 7b). The second interval corresponds to the subtropical regions which are defined by the union of two latitude intervals such that $[45°\ \mathrm{S},\ 25°\ \mathrm{S}[\cup]25°\ \mathrm{N},\ 45°\ \mathrm{N}]$ (see Fig. 7c and 7d). Finally, the high latitudes are represented by the third interval which is $[90°\ \mathrm{S},\ 45°\ \mathrm{S}[\cup]45°\ \mathrm{N},\ 90°\ \mathrm{N}]$ (see Fig. 7e and 7f). As shown in Fig. 7a, 7c and 7e, mean profiles are very different from one latitude band to another. These figures illustrate that, regardless of the network used, the emulators provide a good representation of the global mean of the zonal wind tendency across latitude bands. The predicted profiles do not correspond exactly to the profiles of the reference data. Nevertheless, they tend to follow the overall behavior of the global means of the zonal wind tendency. In the tropics, both emulators struggle more in the first vertical levels as well as in the upper layers

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

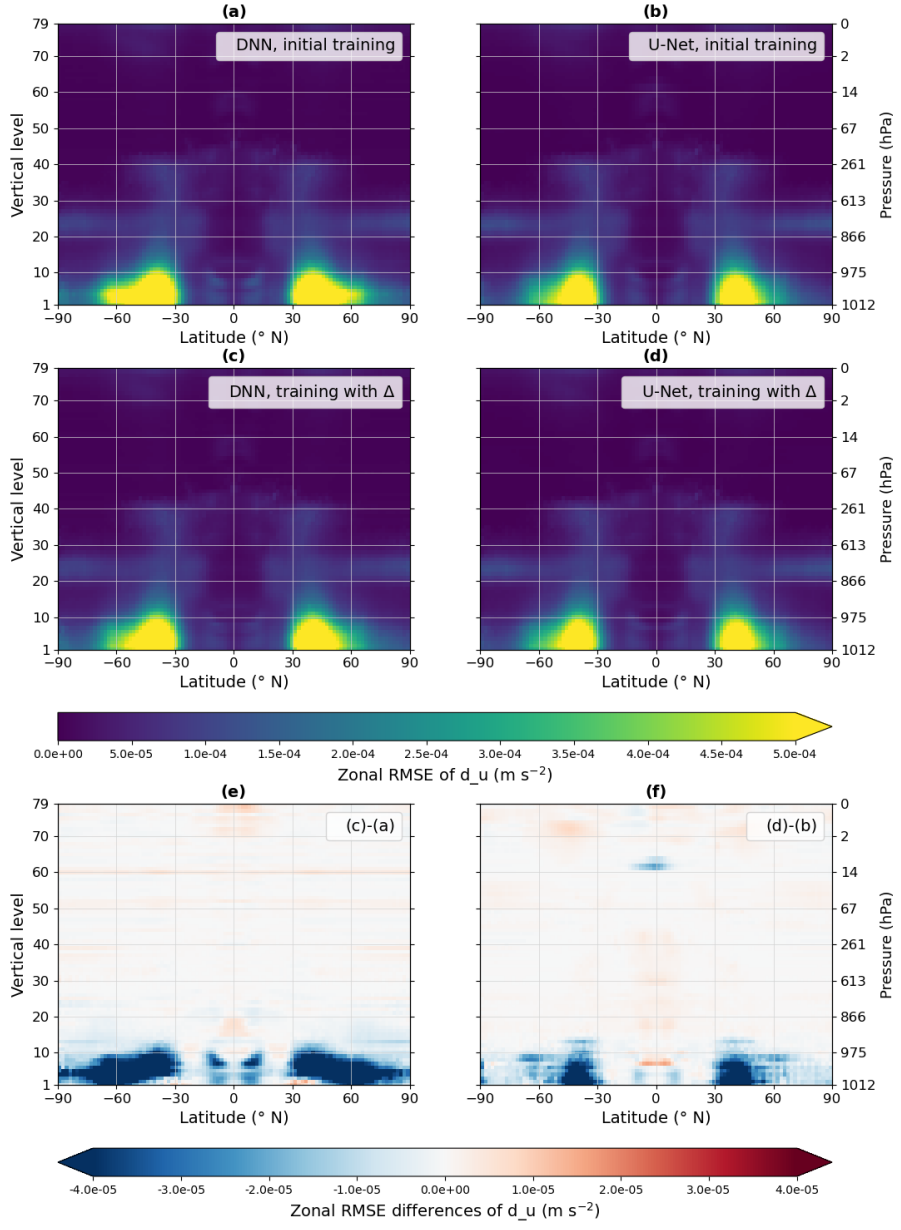

Figure 6. Cross-sections of the zonal RMSE of the zonal wind tendency *d_u* for the test subset. Figures **(a)** and **(b)** show the results from the DNN and the U-Net respectively, from the initial approach in both cases, whereas figures **(c)** and **(d)** are the results of the DNN and the U-Net, with laplacians. Figures **(e)** and **(f)** are respectively the zonal RMSE differences between DNN with laplacians **(c)** and DNN without them **(a)**, and U-Net with laplacians **(d)** and U-Net without them **(b)**.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

of the atmosphere as shown in Fig. 7a. Regarding the other two latitude bands, the emulators face some difficulties in the lower vertical levels (see Fig. 7c and 7e). As shown in Fig. 7d, the vast majority of the zonal wind tendency variability comes from extratropical regions, especially subtropical areas, where the variance is particularly high in the first 10 levels. Indeed, the maximum variance is equal to 9.4×10^{-7} $\mathrm{m^2\,s^{-4}}$ —equivalent to a maximum standard deviation of 9.7×10^{-4} $\mathrm{m\,s^{-2}}$— whereas in the tropics this maximum is around 1.6×10^{-8} $\mathrm{m^2\,s^{-4}}$ —which corresponds to a maximum standard deviation of 1.3×10^{-4} $\mathrm{m\,s^{-2}}$— (see Fig. 7b). The high latitudes also show considerable variability, with a maximum variance of 6.4×10^{-7} $\mathrm{m^2\,s^{-4}}$ —i.e. a maximum standard deviation of 8.0×10^{-4} $\mathrm{m\,s^{-2}}$— (see Fig. 7f). Above level 20, the variance for the three latitude bands appears to approach 0. Whatever the study area, the variances from the two emulators do not reach the variance of the reference data, but the U-Net model performs better than DNN. This raises the question of the origins of this underestimation of the variance. We know that physical phenomena, such as convection and turbulence, manifest differently depending on the latitudes. Therefore, this leads us to question and investigate the underlying physical mechanisms responsible for the significant variance, which is further discussed in Sect. 5.

4.3 Results on the other five tendencies

After focusing on the zonal wind tendency, we now discuss the ability of the emulators to reproduce the other five tendencies: the meridional wind, temperature, humidity, liquid water and ice water tendencies. The zonal wind tendency and these five tendencies are emulated simultaneously. Firstly, the results of the global metrics for these five tendencies (see Table 5 and Fig. 5) show systematically better scores with the use of the U-Net model compared to the DNN. The global vertical profiles (Fig. S5a and S5b for *d_v*, Fig. S7a and S7b for *d_t*, Fig. S9a and S9b for *d_qx1*, S11a and S11b for *d_qx2*, Fig. S13a and S13b for *d_qx3*) and the cross-sections of the zonal RMSE (Fig. S6a and S6b for *d_v*, Fig. S8a and S8b for *d_t*, Fig. S10a and S10b for *d_qx1*, S12a and S12b for *d_qx2*, Fig. S14a and S14b for *d_qx3*) of these tendencies are presented in the Supplementary Material.

The comparison of the global vertical profiles of the mean of these five tendencies and the zonal wind tendency (see Fig. 4a, S5a, S7a, S9a, S11a and S13a) highlights the fact that tendencies do not have the same vertical structures. This is due to different physical processes which are active at different altitudes. We can provide a physical interpretation of these mean global vertical profiles of tendencies. Overall the meridional wind tendency is close to 0 along the vertical as all physical processes balance and do not have preferential direction. At the bottom of the atmosphere, the temperature tendency is positive mainly due to turbulence, thermals and large scale condensation, and to a lesser extent, to convection and longwave radiation. Then, this tendency decreases with altitude until the tropopause, around level 40. For the first vertical levels, the humidity tendency is positive due to various physical processes such as turbulence, thermals, evaporation and large scale condensation for instance. In the lower atmosphere, the liquid water tendency is positive due to large scale condensation processes. However, the humidity and liquid water tendencies decrease with altitude. Between vertical levels 25 and 35, these tendencies are negative because the water precipitates or solidifies. This explains why, for these same vertical levels, the solid water tendency increases. Then, this tendency becomes negative with the sedimentation process. Above the tropopause, the three water tendencies are close to 0 due to the temperature and pressure conditions.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

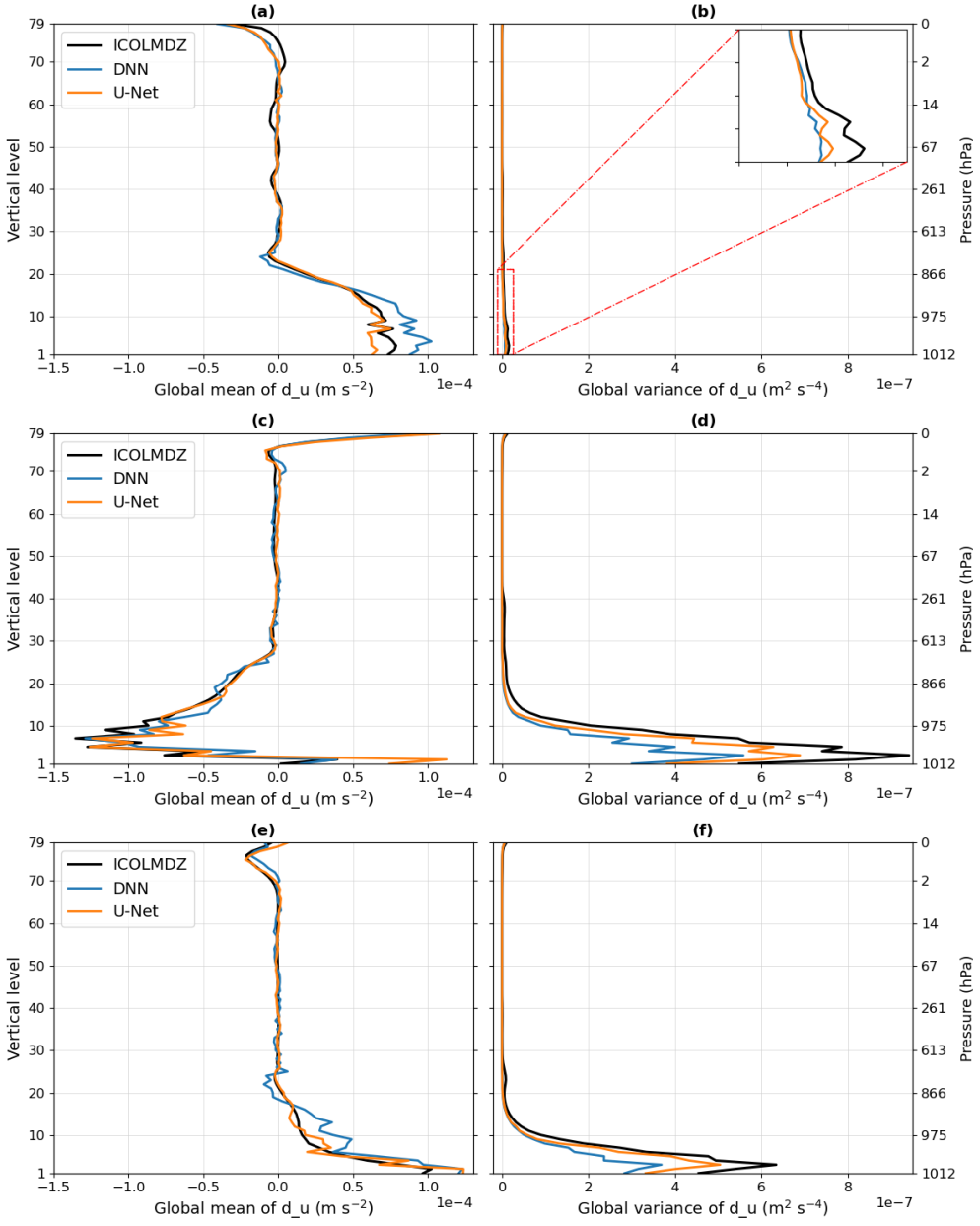

Figure 7. Global vertical profiles of the zonal wind tendency *d_u* of the mean (left) and the variance (right) for the test subset, obtained with the initial approach. Figures **(a)** and **(b)** correspond to the tropics ($\phi \in [25^\circ\ \mathrm{S};\ 25^\circ\ \mathrm{N}]$), figures **(c)** and **(d)** represent the subtropical regions ($\phi \in [45^\circ\ \mathrm{S},\ 25^\circ\ \mathrm{S}[\cup]25^\circ\ \mathrm{N},\ 45^\circ\ \mathrm{N}]$) and figures **(e)** and **(f)** are the high latitudes ($\phi \in [90^\circ\ \mathrm{S},\ 45^\circ\ \mathrm{S}[\cup]45^\circ\ \mathrm{N};\ 90^\circ\ \mathrm{N}]$). The data from the ICOLMDZ simulation are represented by the black curves. The blue curves come from the DNN and the orange curves represent the U-Net, for the initial training.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Nevertheless, the overall conclusion is the same for all the tendencies: where the variability of the tendency studied is the highest (see Fig. 4b, S5b, S7b, S9b, S11b and S13b), usually at the bottom of the atmosphere, emulators struggle to represent the global mean correctly. It appears that the U-Net architecture has a better ability to capture the reference tendency than the DNN. The global mean curves are sometimes smoother for the U-Net than the DNN. Moreover, the representation of the global variance is always underestimated by both emulators, especially where the variance is the strongest, whatever the tendency studied. However, this behavior is even more pronounced with the DNN and the U-Net demonstrates better performances in capturing the variability of the tendency studied. Some specific features of each tendency can be discussed. From the surface to level 20, the global mean of the reference meridional wind tendency presents some oscillations around 0. For the same levels, the DNN tends to underestimate this global mean, whereas in the upper of the atmosphere, both emulators overestimate the global mean and the variance is not captured at all. Above level 20 for the meridional wind tendency and level 40 for the humidity, liquid water tracer and ice water tracer tendencies, the global means of the reference data appear to remain close to 0. The global means predicted by the two emulators, for each tendency, are quite unstable and oscillate around the reference data. Indeed, while the global mean of the DNN fluctuates between negative and positive values around the reference mean global, the U-Net is quite close to the reference data, but overestimates it. This highlights the challenges faced by the emulator. The global variance of the ice water tracer tendency is quite well represented with both emulators, despite a slight underestimation compared with the reference curve. For the temperature tendency, the DNN underestimates the global mean, while the U-Net, despite its overestimation at high altitudes, appears to be slightly closer to the global mean especially at the bottom of the atmosphere. The performances of the DNN and the U-Net architecture in terms of reproducing variability are quite similar and reveal to be quite accurate. However, the U-Net seems to be slightly better in general, except at the variance peak around the 70^{th} vertical level.

Finally, the cross-sections of the zonal RMSE of the five tendencies show similar results to the cross-sections of the zonal RMSE of the zonal wind tendency discussed previously (see Fig. 6a-b, S6a-b, S8a-b, S10a-b, S12a-b and S14a-b). For a chosen tendency, even if the region where the highest zonal RMSE is the same for both emulators, we see that the RMSE values are more pronounced with the DNN, as the region is much larger than for the U-Net. However, the location of this pattern depends on the tendency. For the meridional wind tendency, it is at the same latitudes and vertical levels as the zonal wind tendency, but, we observe an additional pattern at the top of the atmosphere between 30° and 60° in each hemisphere. For the temperature, humidity and liquid water tendencies, we see that the zonal RMSE is high for latitudes between 55° S and 55° N, and for vertical levels below 30. Concerning the solid water tendency, there is a pattern with a high zonal RMSE defined between levels 20 and 40 for latitudes from 30° to the pole in each hemisphere.

Therefore, this brings us to consider two key points: the potential factors that may be responsible for the underestimation of the variance with both emulators, and also, what might explain the better performance of the U-Net compared to the DNN.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

5 Refined approach: integrating physical knowledge

In this section, we analyze the physical phenomena that could be the cause of lower performance with the DNN compared to the U-Net and we provide new predictors in addition to those already used as input in our emulators in an attempt to improve the results of our DNN.

5.1 Breakdown of tendencies into physical processes

To understand the underestimation of the variance and its physical origin, we break down the zonal wind tendency into physical phenomena included in its computation within phyLMDZ. Indeed, as mentioned above, in this model each tendency is the sum of various tendencies due to different physical processes (see Eq. 1). We decided to carry out this study on three grid cells —noted A, B and C— located in different regions of the globe to examine the contribution of each phenomenon. We define a first point located in the high latitudes ($\phi_A = 78.5°$ N; $\lambda_A = 7.86°$ E), another in the subtropics ($\phi_B = 33.2°$ N; $\lambda_B = 70.2°$ E) and a last one near the equator ($\phi_C = 0.413°$ N; $\lambda_C = 48.6°$ E).

To provide a comparison between the three cells, we investigate their vertical profiles from a randomly chosen time step as shown in Fig. 8 for B and in the Supplementary Material for A (Fig. S1) and C (Fig. S3). The vertical profiles are those of the total zonal wind tendency and also those of the physical processes that contribute to and influence this total tendency. These physical processes include turbulence *d_u_turb*, convection *d_u_conv*, thermals *d_u_therm*, gravity wave drags associated to fronts *d_u_gwd_front* and those associated to precipitations *d_u_gwd_precip*. Firstly, we can note that the scales of the zonal wind tendency signals differ among the three points (see Fig. S1a, 8a and S3a). This variation in scale highlights differences in intensity with latitude: the zonal wind tendency is highest in the subtropics (10^{-3} m s^{-2}), lower in the high latitudes (10^{-4} m s^{-2}), and lowest near the equator (10^{-5} m s^{-2}). However, by decomposing the vertical profile of the zonal wind tendency into vertical profiles of the various physical processes involved, we observe that these processes do not have the same influence depending on the point and so, on the latitude studied. Turbulence movements (see Fig. S1a, 8a and S3a) are present whatever the point studied near the surface but also at higher altitudes, between vertical levels 20 and 40, for points located at high latitudes and in the subtropics. Turbulence is the physical process whose intensity is greatest for these two points. At high latitudes as in the subtropics, convection (see Fig. S1c, 8c and S3c) does not play a role, unlike at the equator where it is the physical process that makes the major contribution, particularly between vertical levels 20 and 40. The contribution of thermals (see Fig. S1d, 8d and S3d) is located in the vertical levels below the 30th in high latitudes and especially at the equator. However, for the subtropics they have no impact on the total tendency. Whatever the point studied, gravity wave drags linked to the fronts (see Fig. S1e, 8e and S3e) are only present at the top of the atmosphere. This is also the case for gravity wave drags from precipitations (see Fig. S1f, 8f and S3f) whose signal is much stronger at the point near the equator than at the other two points. Therefore, this approach shows that turbulence is the predominant physical process in subtropical regions (Fig. 8) and at high-latitude regions (Fig. S1), where the emulator capabilities are limited concerning the correct reproduction of the variability.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

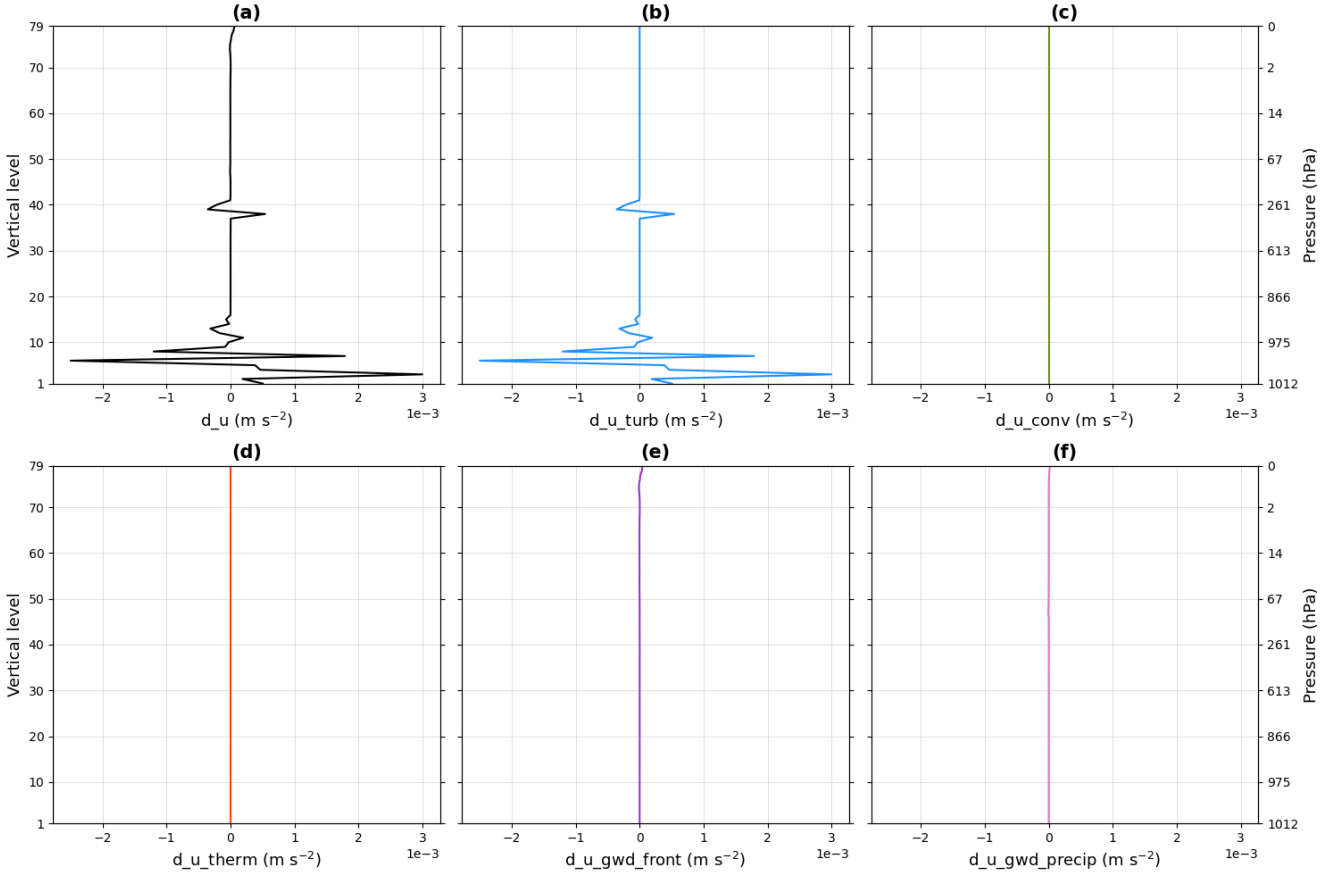

Figure 8. Vertical profile of the zonal wind tendency *d_u* **(a)** of the point B located in the subtropics for one time step, and vertical profiles of the physical processes involved in the decomposition of this tendency: turbulence **(b)**, convection **(c)**, thermals **(d)**, gravity wave drags associated to fronts **(e)** and gravity wave drags associated to precipitations **(f)**.

Based on a full-year simulation, we studied the mean vertical profiles of these three points for the total zonal wind tendency but also for the physical processes. Figure 9 shows the results for B, and those for A and C are presented in the Supplementary Material (Fig. S2 and S4). They include a minimum/maximum envelope that contains all the vertical profiles for the entire year, as well as a hatched area representing the standard deviation around the mean profile. The mean vertical profile of the total zonal wind tendency illustrated in Fig. 9a highlights that the average annual value is zero. Thus, all physical processes balance over a year. This result is also found for A and C. Whatever the cell studied, several points can be noted. Firstly, the majority of the total zonal wind tendency variability is located between the 1st and the 20th vertical level. This variability is higher for B than A, and higher for A than C. From the bottom of the atmosphere to vertical level 38, 45 or 65 —respectively for the point A, B or C— phenomena relating to turbulence, convection and thermals are present and contribute to the total zonal wind tendency. On the other hand, gravity wave drags occur at the top of the atmosphere with a lower amplitude than for the other

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

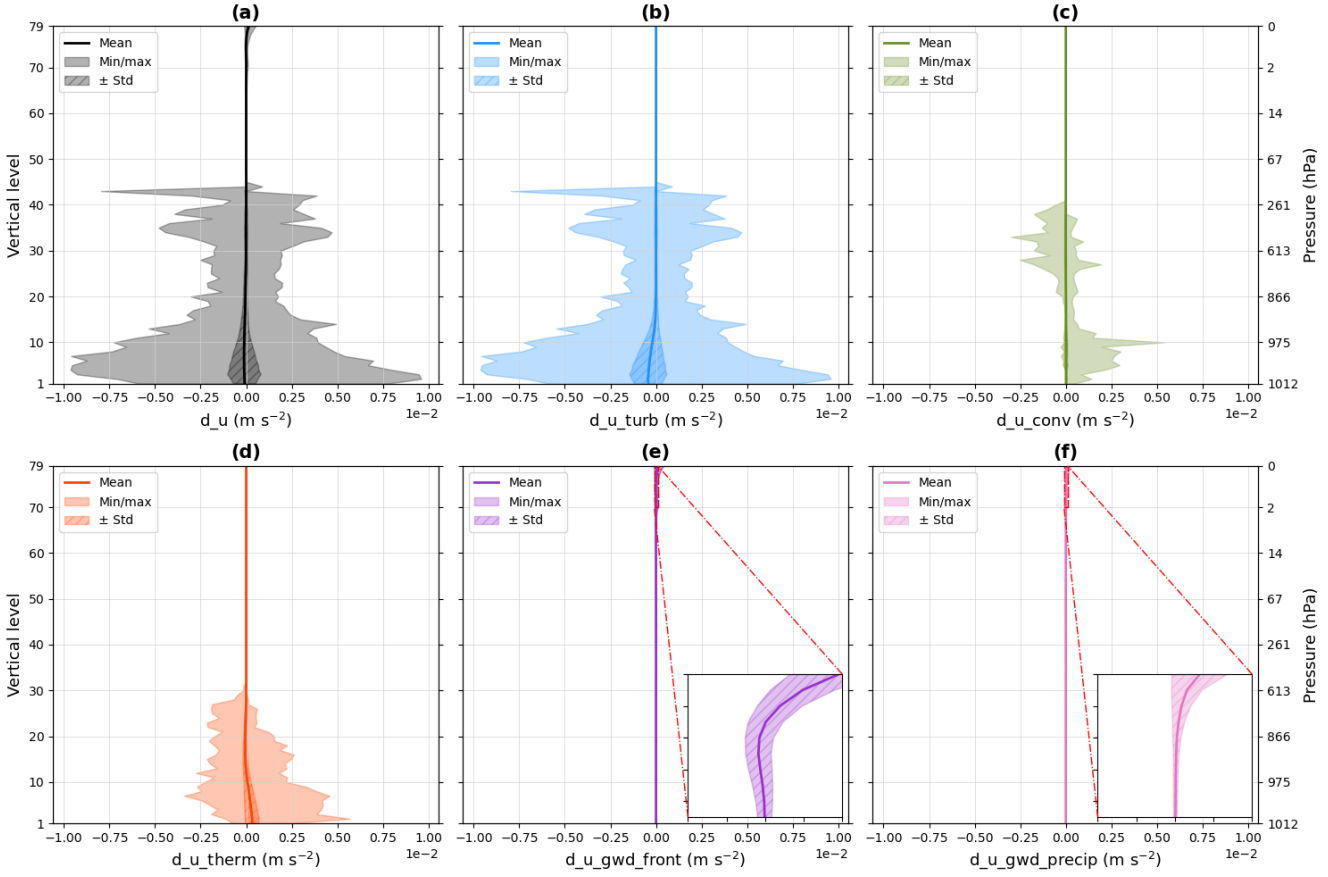

Figure 9. Mean vertical profile of the zonal wind tendency d_u **(a)** of the point B, located in the subtropics, with minimum/maximum envelope and standard deviation bounds (hatched areas). The data used are from a full-year simulation. The same types of curves as the zonal wind tendency can be generated for the physical processes involved in the decomposition of this tendency: turbulence **(b)**, convection **(c)**, thermals **(d)**, gravity wave drags associated to fronts **(e)** and gravity wave drags associated to precipitations **(f)**.

physical processes. As the total tendency is the sum of the other tendencies, it is clear that turbulence is the main contributor to which other processes are added. In conclusion, this breakdown of the zonal wind tendency into physical processes emphasizes the fact that the vast majority of signal variability comes from subtropical regions and is linked to turbulence.

Knowing how turbulence is parameterized in phyLMDZ, it is possible to introduce physical knowledge into the learning process in an attempt to improve the performance of the emulator in terms of its representation of turbulence. In phyLMDZ, the zonal wind tendency due to turbulence is of the form:

$$m_z \frac{\partial u}{\partial t} = -\frac{\partial}{\partial z} m_z \phi_u \tag{14}$$

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

with m_z a mass-weighting coefficient depending on the altitude z and ϕ_u the vertical turbulent flux of the zonal wind which is given by:

$$\phi_u = -K_z \frac{\partial u}{\partial z} \tag{15}$$

where K_z is the turbulent mixing coefficient which depends on the altitude z. We introduce the laplacian of the zonal wind, noted Δu_z, which can be written as follows:

$$\Delta u_z = \frac{\partial^2 u}{\partial z^2}. \tag{16}$$

Turbulence can then be expressed in terms of this laplacian, up to a factor (considering that we neglect K_z). With this in mind, we add this vertical laplacian of the zonal wind into the emulator as a predictor. We refer to this laplacian as a hidden variable, often known as a latent variable, because it is obtained directly from the zonal wind. We consider the zonal wind u_z at the vertical point z. Using the discrete approximation with the three-point finite difference method, its laplacian Δu_z can be calculated as follows:

$$\Delta u_z = \frac{u_{z+1} - 2u_z + u_{z-1}}{(\delta z)^2} \tag{17}$$

where u_{z+1} and u_{z-1} correspond to the values of u at the vertical points $z+1$ and $z-1$, respectively. In the denominator, $\delta z = 1$ because the step between vertical levels is constant. By applying Dirichlet boundary condition, we assume that for $z = 1$ and $z = 79$, we have $\Delta u_1 = 0$ and $\Delta u_{79} = 0$ respectively. The same equations (Eq. 15, 16 and 17) can be applied to the other five state variables *v*, *T*, *qx1*, *qx2* and *qx3*, to obtain their vertical laplacians, respectively noted Δv, ΔT, $\Delta qx1$, $\Delta qx2$ and $\Delta qx3$. Then, we develop new emulators in which these laplacians were added as new predictors (see Table 6) in order to improve the emulation of the turbulence. In the rest of the article, these emulators are referred to as laplacian-aware emulators. In this way, the new input stacked vector $\mathbf{X}$ has a length of 1,030 and looks like:

$$\mathbf{X} = \big[u, \Delta u, v, \Delta v, T, \Delta T, qx1, \Delta qx1, qx2, \\ \Delta qx2, qx3, \Delta qx3, rot, sza, psol, SST\big].$$

The same pre-processing steps described in Sect. 2.3 are applied to this new input stacked vector $\mathbf{X}$. The complexity of this new learning problem has been increased compared to the original problem due to the fact that we add six new predictors, each of which extends over the 79 vertical levels of the atmospheric column.

5.2 Training with physical knowledge

In order to assess the impact of adding new predictors, we first studied the global vertical profiles of the mean and variance (Fig. 4c and 4d) to compare them with the results obtained without the addition of these latent variables (Fig. 4a and 4b). As we can see in Fig. 4a and 4c, by comparing the same model architecture without and with the addition of laplacians, the global means of the predicted zonal wind tendency are quite similar in the vertical levels above 25. When we add the new predictors,

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Table 6. List of input variables $\boldsymbol{x}$ for NNs with the addition of the laplacians of the six state variables.

Input variable $\boldsymbol{x}$	Notation	Unit	Vertical level(s)
Zonal wind	u	$\mathrm{m\ s^{-1}}$	79
Laplacian of the zonal wind	Δu	$\mathrm{m\ s^{-1}}$ per squared vertical level	79
Meridional wind	v	$\mathrm{m\ s^{-1}}$	79
Laplacian of the meridional wind	Δv	$\mathrm{m\ s^{-1}}$ per squared vertical level	79
Temperature	T	K	79
Laplacian of the temperature	ΔT	K per squared vertical level	79
Humidity	*qx1*	$\mathrm{kg\ kg^{-1}}$	79
Laplacian of the humidity	$\Delta qx1$	$\mathrm{kg\ kg^{-1}}$ per squared vertical level	79
Liquid water	*qx2*	$\mathrm{kg\ kg^{-1}}$	79
Laplacian of the liquid water	$\Delta qx2$	$\mathrm{kg\ kg^{-1}}$ per squared vertical level	79
Ice water	*qx3*	$\mathrm{kg\ kg^{-1}}$	79
Laplacian of the ice water	$\Delta qx3$	$\mathrm{kg\ kg^{-1}}$ per squared vertical level	79
Wind curl	*rot*	$\mathrm{s^{-1}}$	79
Solar zenith angle	*sza*	deg	1
Surface pressure	*psol*	Pa	1
Sea surface temperature	*SST*	K	1
Size of vector **X**			1,030

the results are better in the first vertical levels, for both architectures. Moreover, we note that the predicted mean global profile is smoother for the U-Net than the DNN. Figures 4b and 4d highlight the fact that the initial learning revealed the difficulty of the DNN architecture in correctly representing the variability, contrary to the U-Net architecture, in the low vertical levels. Despite the fact that the representation of the variance is not perfect, it drastically improves with the new emulations for levels below 10. No improvement in the variance between levels 10 and 30 can be seen.

The global RMSE results for the zonal wind tendency obtained with the laplacian-aware emulators shown in Table 5 reveal that, as for the emulators without laplacian (see Sect. 4.1), the U-Net architecture achieves a RMSE that is 14.7 % lower compared to the DNN. The U-Net models outperform the DNN architectures in both cases, with or without laplacian. Moreover, for a selected type of NN, the one in which laplacians have been added as predictors has a lower RMSE score. Indeed, the NN with laplacians has a global RMSE that is 8.1 % or 5.7 % —respectively for DNNs and U-Nets— better than that of the NN without laplacian. Thus, the addition of laplacians improves the prediction of the zonal wind tendency. When comparing the values of the coefficient of determination (see Fig. 5), we underline that despite the laplacian, the DNN scores are lower than those of the U-Net without the laplacian. Indeed, we see that the U-Net with laplacians ($R^2 = 0.67$) outperforms the others, followed by the U-Net without laplacian ($R^2 = 0.63$), then the DNN with laplacians ($R^2 = 0.54$) and the last one is the DNN without laplacian ($R^2 = 0.46$). Therefore, we improve the prediction of the zonal wind tendency by adding the laplacian of

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

the zonal wind in both NN models. However, despite the addition of this new predictor, the DNN model does not achieve the performance of the U-Net in terms of predicting the zonal wind tendency.

Similarly to the emulators following the initial approach without laplacian, the zonal RMSE of the zonal wind tendency is calculated for the laplacian-aware emulators. The results are presented in Fig. 6c and 6d. As mentioned above for this metric in the context of the emulator without laplacian, the patterns in the first vertical levels are quite similar between the DNN and the U-Net, although the peak of the metric spreads at higher latitudes for the DNN. When comparing emulators where the laplacians have been added to emulators of the direct approach without them, we note that the new zonal RMSE is quite similar. The zonal RMSE differences of the zonal wind tendency between the original emulators and the new ones are shown in Fig. 6e and 6f. In either hemisphere, for vertical level below 10, the visible patterns between the latitudes 0° and 15°, but also between 30° and 90°, suggest that the laplacian-aware emulators are better in these regions. This improvement is even more significant for the DNN as the pattern is much more prominent than for the U-Net. For the other vertical levels, the gain from adding the laplacians is significantly smaller. In some regions, it seems that models without laplacian may even demonstrate better capabilities at higher altitudes, but this improvement is negligible compared to the enhancement observed at the lower vertical levels. Above the 10^{th} vertical level, the zonal RMSE differences is mostly close to 0 meaning that the two models compared are similar: no improvement or degradation in the results is observed with the new emulators.

5.3 Results for other tendencies

We examine the emulation of the other five tendencies with the laplacian-aware emulators. As indicated in the Table 6, the laplacians of the six state variables have been added as predictors, motivated by the aim of improving the emulation of turbulence. Overall, for all the tendencies, we can draw the same conclusions as for the zonal wind tendency. Indeed, the global RMSE of the Table 5 and the coefficient of determination R^2 shown in Fig. 5 systematically indicate that laplacians help to obtain better predictions. Figure 5 highlights that, even if laplacians were added to the DNN model, it remains less efficient than the U-Net without these latent variables. However, this is not true for the ice water tracer tendency, noted *d_qx3*, for which the DNN architecture with laplacians is better than the U-Net without these predictors (see Fig. 5). This is probably due to the fact the turbulence does not directly influence this tendency. Indeed, the breakdown of the ice water tracer tendency (not shown here) reveals that turbulence is not involved in its computation within phyLMDZ. We will discuss this in more detail later in Sect. 6. Figure 5 also shows that, overall, some tendencies are better predicted than others, such as the temperature tendency for instance. In the Supplementary Material, the new global vertical profiles of these tendencies (Fig. S5c and S5d for *d_v*, Fig. S7c and S7d for *d_t*, Fig. S9c and S9d for *d_qx1*, Fig. S11c and S11d for *d_qx2*, Fig. S13c and S13d for *d_qx3*) are shown and can be compared with the profiles described previously in Sect. 4.3. It is also the case for the cross-sections of the zonal RMSE for each tendency (Fig. S6c and S6d for *d_v*, Fig. S8c and S8d for *d_t*, Fig. S10c and S10d for *d_qx1*, Fig. S12c and S12d for *d_qx2*, Fig. S14c and S14d for *d_qx3*). The results for the cross-sections of the zonal RMSE differences of the five tendencies (Fig. S6e and S6f for *d_v*, Fig. S8e and S8f for *d_t*, Fig. S10e and S10f for *d_qx1*, Fig. S12e and S12f for *d_qx2*, Fig. S14e and S14f for *d_qx3*) are also shown in the Supplementary Material.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

The global mean profiles of the tendencies are closely approaching those of the ICOLMDZ data with these new learning, in both cases, although they are not perfectly identical to the reference data curves (see Fig. S5c, S7c, S9c, S11c and S13c). Moreover, for the vertical levels where the global mean is close to zero, the profiles still oscillate around the reference profile. As with the zonal wind tendency, the global variance profiles obtained with these new emulators remain underestimated, but they offer a better representation of the variability, especially in vertical levels where it is highest (see Fig. S5d, S7d, S9d, S11d and S13d). This is particularly noteworthy for the DNN model. For almost all tendencies, the new prediction performance of the DNN is comparable to that of the U-Net. Nevertheless, we can identify some specific features depending on the tendency. For instance, the behavior of the meridional wind tendency at the top of the atmospheric column has still not been captured by the emulators, in both cases. For the temperature tendency, the addition of laplacians does not seem to have any impact on the global variance for the U-Net: the DNN performs slightly better than the U-Net for certain vertical levels, thanks to the contribution of the laplacians. Concerning the results for the liquid water tracer tendency, the addition of the laplacians does not seem to have benefited the global mean. The curves are not smooth and still oscillate significantly around the reference global mean with large amplitudes. For the first levels, the profile from the DNN is significantly more underestimated compared to the first emulator without laplacian. However, the global variance highlights the improvement made by the DNN, particularly in the vertical levels with higher variability. Thus, for the liquid water tracer tendency, the improvement in global variance with the DNN seems to be at the cost of the global mean. For the ice water tracer tendency, the comparison shows that there has been no significant improvement. The curve of the DNN with laplacians even oscillates with larger amplitudes. However, the global mean curves from the predictions generally align with the behavior of the global reference mean curve. At the variance peak, we observe a slight improvement in its representation by both models.

With the laplacian-aware emulators, we see the same patterns as with the initial approach, where the zonal RMSE is high, as shown in the cross-sections (see Fig. S6c-d, S8c-d, S10c-d, S12c-d and S14c-d). Nevertheless, these patterns seem less defined and spread for certain regions compared to initial learning, meaning that the addition of the laplacians has improved the prediction of these tendencies. This result is emphasized with the zonal RMSE differences (see Fig. S6e-f, S8e-f, S10e-f, S12e-f and S14e-f), for all the tendencies. Indeed, with both NN architectures, the addition of physical knowledge into the learning process improves predictions whatever the tendency. Therefore, the same conclusions can be drawn as for the zonal wind tendency.

6 Discussion

Several hypotheses can be made about the better efficiency of the U-Net architecture compared to the DNN model. Firstly, the U-Net's design favors the coherence and the dependence on the vertical structure. In theory, convolutional filters have the capacity to compute the laplacian and to use it for each vertical level. Moreover, the implementation of ResBlocks in the U-Net architecture helps to achieve better convergence by avoiding the vanishing gradient problem (Veit et al., 2016).

The addition of the laplacians has proven to be beneficial for all tendencies, particularly in terms of variance. However, this enhancement varies in magnitude depending on the tendency studied. This reflects the influence of turbulence on the tendency.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Indeed, even if turbulence appears in the breakdown of a tendency into physical processes, it is not specifically the main process, as in the case of the zonal wind tendency, for some regions. The physical processes contributing to the meridional wind tendency are the same as those for the zonal wind tendency. Most of the variability in the first vertical levels comes from turbulence, and the variability at the top of the atmosphere is due, in particular, to gravity wave drags from fronts. Concerning the temperature tendency, other physical processes are involved —for instance longwave radiation, evaporation and large-scale condensation— but turbulence is still present, especially at pressure levels below 261 hPa (corresponding to vertical levels below 40), without being the main process. This is also the case for the humidity tendency, as thermals and large-scale condensation, for instance, are two main physical processes involved in its calculation, in addition to turbulence which is present at pressure levels below 261 hPa. Thus, adding the laplacian to the predictors helps to improve the representation of turbulence for these tendencies. On the other hand, turbulence does not appear in the breakdown of the liquid water tracer and solid water tracer tendencies into physical processes. However, these two tendencies are strongly linked to the humidity tendency which depends on turbulence. This explains why the addition of laplacians does not result in a significant improvement of the variability for these two tendencies. As a results, the question of adding new variables that bring physical knowledge in the emulators to improve prediction of the other tendencies arises and should be considered.

7 Conclusions

In this paper, we have explored the potential of emulating all the complex physical parameterizations of the ICOLMDZ atmospheric model, over the entire atmospheric column, in other words over the 79 vertical levels, and all grid points. To do so, we have developed several emulators based on different NN architectures and evaluated them in offline configuration. These emulators take the vertical profiles of state variables and variables relating to boundary conditions as input to reproduce the tendencies of state variables for each atmospheric column. The state variables are the zonal and meridional winds, the temperature, and the three water tracers. We have shown that a U-Net, where some ResBlock are added, is capable of reproducing the physical tendencies, while a DNN struggles to achieve as good results. After studying in more detail the physical processes involved in the tendencies studied, we demonstrate the effectiveness of adding some physical knowledge, in particular related to turbulence, into the learning process as predictors. Indeed, the addition of latent variables such as the laplacians of the state variables to the input of our emulators has improved the quality of the predictions in terms of variability. For instance, for the zonal wind tendency, this is particularly the case for the first ten vertical levels in the subtropics where the variability is higher. Moreover, this improvement is particularly significant for the DNN since, with the addition of these new predictors, this NN achieves global results comparable to the U-Net architecture. Nevertheless, even if this addition of new predictors primarily impacted the DNN, it should be noted that adding laplacians also improves the representation of the variance of the U-Net model, but to a lesser degree; its initial predictions (without laplacian) being better than those of the DNN. Therefore, our study demonstrates the importance of integrating relevant physical predictors in NN models —here through the laplacian—, thereby reducing the black-box effect that exists with the use of DL models and improving interpretability.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

The present work is a foundational first step towards a more ambitious project: emulating the physical parameterizations to make climate simulations and projections. To achieve this, several steps and major challenges remain.

First, we need to quantify the performance of each model in order to see if the contribution of ML accelerate the representation of physical parameterizations compared to phyLMDZ. For this, we estimate the time to obtain the six physical tendencies for one physics time step, on all cells across the aquaplanet and at all vertical levels, i.e. for one atmospheric column, with both the phyLMDZ traditional parameterizations and the NNs. On the one hand, at the end of the simulation with ICOLMDZ, the dynamics part provides the total time spent in physics which amounts to approximately 4,800 seconds for the whole year of simulation. Therefore, it takes an average of 0.140 second to obtain tendencies for a single physics time step. This diagnosis is obtained on Central Processing Units (CPUs) with a setup of 5 nodes and 164 physical cores used. On the other hand, the inference time estimation with our NNs is evaluated based on the *predict()* function from Keras/TensorFlow for a single physics time step. We estimate an inference time of around 0.045 second by time step with the DNN model, and we have a similar order of magnitude with the U-Net model, both models run on a single Graphics Processing Unit (GPU) NVIDIA V100. Therefore, in these configurations and for the inference of a single time step, the emulators offer a computational time gain of a factor of about three with respect to phyLMDZ. However, it remains to be seen whether this gain would remain when the emulators are coupled with DYNAMICO, the dynamics part of the model.

A challenge is the generalization of data that has not been seen before. Indeed, emulators must be developed in such a way that they have the ability to cope with unseen climate conditions and generalize to different climates in a context of climate change. There have been recent developments (Clark et al., 2022; Beucler et al., 2024), to test the performance of the emulators on multi-climates to assess their ability to extrapolate and to obtain reliable and accurate results on scenarios that are not known by these emulators. For instance, Han et al. (2023) have created an emulator based on Convolutional Residual Neural Network to reproduce temperature and moisture tendencies, and cloud liquid and ice water from moist physics processes, with a realistic configuration. This study reveals the success to generalize, in an offline setup, to a warmer climate thanks to the use of convective memory and a specific NN architecture.

After conducting the offline evaluation, the next step is to couple the emulator with DYNAMICO, the dynamics part of the model. This would allow for assessing the value of the learning process and for studying if the model's stability is maintained. The coupling between the climate model —written in Fortran and using CPUs— and the emulator —developed in Python and using GPUs— can be done in several ways thanks to software packages and interfaces (Curcic, 2019; Ott et al., 2020; Partee et al., 2021; Zhang et al., 2024). This online step is crucial to provide a better assessment of the value of the learning process, while also enabling us to investigate both the accuracy and the stability of online runs with our emulator. Indeed, one key challenge currently under active study is achieving stable simulations when integrating the emulator with the rest of the model. New studies have recently been published in which the emulation of parameterizations has been carried out on a realistic configuration and the emulator developed has been coupled to the atmospheric model (Han et al., 2023; Watt-Meyer et al., 2024). With their NNs tested for different configurations to reproduce subgrid processes —specifically turbulence, convection and radiation—, Lin et al. (2024) highlighted potential correlations between the emulators' offline and online performance. They also showed that online stability does not necessarily work conjointly with online performance. Hu et al.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

(2024) have demonstrated that it is possible to couple their U-Net emulator, developed to emulate all the physical variables required when using real geography, with their dynamical core, thus obtaining a stable hybrid model. This was made achievable especially thanks to the incorporation of cloud microphysics constraints and also through the addition of large-scale forcings and convective memory.

Finally, with the goal of conducting climate simulations and projections, these different steps must be repeated on a configuration where orography or even continental surfaces are added, in other words on a realistic configuration. Given the increasing complexity of the physical processes that need to be represented with this new setup, the question of adding new variables in the predictors arises and needs to be studied.

Code and data availability. The Python scripts and data used in this article to train the neural networks, make predictions on the test dataset, and generate the figures are archived in an online repository at https://doi.org/10.14768/0205d9a0-53cb-497b-a7cd-2251182c8fb7 (Crossouard et al., 2025).

Author contributions. All authors conceptualized and defined the main scientific question. SC ran the ICOLMDZ simulations with the help of MK and YM. SC built the datasets with the contribution of TD and YM. SC designed the NNs with the help of MV and ST. SC developed, tested and analyzed the performance of the NNs with ST. All authors contributed to the validation, the analysis and the discussion of the results. SC designed the figures and wrote the article with comments from all co-authors.

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

Acknowledgements. This project received funding from the FOCUS EJN (FOCUS Etablissement Jumeaux Numériques) of the CEA (Commissariat à l'Energie Atomique et aux Energies Alternatives). This work was granted access to the HPC and AI resources on the supercomputer Jean Zay of IDRIS (Institut du Développement et des Ressources en Informatique Scientifique) under the allocations A0130100826 and A0150102212 attributed by GENCI (Grand Equipement National de Calcul Intensif). This work also benefited from state aid managed by the National Research Agency under France 2030 bearing the reference ANR-22EXTR-0011 (TRACCS-PC10-LOCALISING project). The authors acknowledge Arnaud Caubel for his help with the technical aspects. The authors acknowledge the computing and data center ESPRI (Ensemble de Services Pour la Recherche à l'IPSL) for their help in accessing the data.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, https://www.tensorflow.org/, 2015.

Arnold, C., Sharma, S., Weigel, T., and Greenberg, D.: Efficient and Stable Coupling of the SuperdropNet Deep Learning-based Cloud Microphysics (v0.1.0) to the ICON Climate and Weather Model (v2.6.5), https://doi.org/10.5194/egusphere-2023-2047, 2023.

Balaji, V.: Climbing down Charney's ladder: machine learning and the post-Dennard era of computational climate science, Phil. Trans. R. Soc. A., 379, 20200 085, https://doi.org/10.1098/rsta.2020.0085, 2021.

Balaji, V., Couvreux, F., Deshayes, J., Gautrais, J., Hourdin, F., and Rio, C.: Are general circulation models obsolete?, Proc. Natl. Acad. Sci. U.S.A., 119, e2202075 119, https://doi.org/10.1073/pnas.2202075119, 2022.

Bauer, P., Stevens, B., and Hazeleger, W.: A digital twin of Earth for the green transition, Nat. Clim. Chang., 11, 80–83, https://doi.org/10.1038/s41558-021-00986-y, 2021.

Behrens, G., Beucler, T., Gentine, P., Iglesias-Suarez, F., Pritchard, M., and Eyring, V.: Non-Linear Dimensionality Reduction with a Variational Encoder Decoder to Understand Convective Processes in Climate Models, J Adv Model Earth Syst, 14, e2022MS003 130, https://doi.org/10.1029/2022MS003130, arXiv:2204.08708 [physics], 2022.

Belochitski, A. and Krasnopolsky, V.: Robustness of neural network emulations of radiative transfer parameterizations in a state-of-the-art general circulation model, Geosci. Model Dev., 14, 7425–7437, https://doi.org/10.5194/gmd-14-7425-2021, 2021.

Beucler, T., Rasp, S., Pritchard, M., and Gentine, P.: Achieving Conservation of Energy in Neural Network Emulators for Climate Modeling, http://arxiv.org/abs/1906.06622, arXiv:1906.06622 [physics], 2019.

Beucler, T., Pritchard, M., Gentine, P., and Rasp, S.: Towards Physically-consistent, Data-driven Models of Convection, http://arxiv.org/abs/2002.08525, arXiv:2002.08525 [physics], 2020.

Beucler, T., Pritchard, M., Rasp, S., Ott, J., Baldi, P., and Gentine, P.: Enforcing Analytic Constraints in Neural-Networks Emulating Physical Systems, Phys. Rev. Lett., 126, 098 302, https://doi.org/10.1103/PhysRevLett.126.098302, arXiv:1909.00912 [physics], 2021.

Beucler, T., Gentine, P., Yuval, J., Gupta, A., Peng, L., Lin, J., Yu, S., Rasp, S., Ahmed, F., O'Gorman, P. A., Neelin, J. D., Lutsko, N. J., and Pritchard, M.: Climate-invariant machine learning, Sci. Adv., 10, eadj7250, https://doi.org/10.1126/sciadv.adj7250, 2024.

Bolton, T. and Zanna, L.: Applications of Deep Learning to Ocean Data Inference and Subgrid Parameterization, J Adv Model Earth Syst, 11, 376–399, https://doi.org/10.1029/2018MS001472, 2019.

Brenowitz, N. D. and Bretherton, C. S.: Prognostic Validation of a Neural Network Unified Physics Parameterization, Geophysical Research Letters, 45, 6289–6298, https://doi.org/10.1029/2018GL078510, 2018.

Brenowitz, N. D. and Bretherton, C. S.: Spatially Extended Tests of a Neural Network Parametrization Trained by Coarse-Graining, J Adv Model Earth Syst, 11, 2728–2744, https://doi.org/10.1029/2019MS001711, 2019.

Brenowitz, N. D., Beucler, T., Pritchard, M., and Bretherton, C. S.: Interpreting and Stabilizing Machine-learning Parametrizations of Convection, Journal of the Atmospheric Sciences, 77, 4357–4375, https://doi.org/10.1175/JAS-D-20-0082.1, arXiv:2003.06549 [physics], 2020.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Bretherton, C. S., Henn, B., Kwa, A., Brenowitz, N. D., Watt-Meyer, O., McGibbon, J., Perkins, W. A., Clark, S. K., and Harris, L.: Correcting Coarse-Grid Weather and Climate Models by Machine Learning From Global Storm-Resolving Simulations, J Adv Model Earth Syst, 14, e2021MS002 794, https://doi.org/10.1029/2021MS002794, 2022.

Chantry, M., Hatfield, S., Dueben, P., Polichtchouk, I., and Palmer, T.: Machine Learning Emulation of Gravity Wave Drag in Numerical Weather Forecasting, J Adv Model Earth Syst, 13, e2021MS002 477, https://doi.org/10.1029/2021MS002477, 2021.

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

Clark, S. K., Brenowitz, N. D., Henn, B., Kwa, A., McGibbon, J., Perkins, W. A., Watt-Meyer, O., Bretherton, C. S., and Harris, L. M.: Correcting a 200 km Resolution Climate Model in Multiple Climates by Machine Learning From 25 km Resolution Simulations, J Adv Model Earth Syst, 14, e2022MS003 219, https://doi.org/10.1029/2022MS003219, 2022.

Crossouard, S., Thao, S., Dubos, T., Kageyama, M., Vrac, M., and Meurdesoif, Y.: Supplementary material for the article entitled "Contribution of physical latent knowledge to the emulation of an atmospheric physics model: a study based on the LMDZ Atmospheric General Circulation Model", https://doi.org/10.14768/0205d9a0-53cb-497b-a7cd-2251182c8fb7, 2025.

Curcic, M.: A parallel Fortran framework for neural networks and deep learning, http://arxiv.org/abs/1902.06714, arXiv:1902.06714 [cs], 2019.

Dubos, T., Dubey, S., Tort, M., Mittal, R., Meurdesoif, Y., and Hourdin, F.: DYNAMICO-1.0, an icosahedral hydrostatic dynamical core designed for consistency and versatility, Geosci. Model Dev., 8, 3131–3150, https://doi.org/10.5194/gmd-8-3131-2015, 2015.

Espinosa, Z. I., Sheshadri, A., Cain, G. R., Gerber, E. P., and DallaSanta, K. J.: Machine Learning Gravity Wave Parameterization Generalizes to Capture the QBO and Response to Increased CO_2, Geophysical Research Letters, 49, e2022GL098 174, https://doi.org/10.1029/2022GL098174, 2022.

Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., and Yacalis, G.: Could Machine Learning Break the Convection Parameterization Deadlock?, Geophysical Research Letters, 45, 5742–5751, https://doi.org/10.1029/2018GL078202, 2018.

Gettelman, A., Gagne, D. J., Chen, C., Christensen, M. W., Lebo, Z. J., Morrison, H., and Gantos, G.: Machine Learning the Warm Rain Process, J Adv Model Earth Syst, 13, e2020MS002 268, https://doi.org/10.1029/2020MS002268, 2021.

Glorot, X., Bordes, A., and Bengio, Y.: Deep Sparse Rectifier Neural Networks, In Proc. of the 14th International Conference on Artifical Intelligence and Statistics (AISTATS), 15, 315–323, 2011.

Goodfellow, I., Bengio, Y., and Courville, A.: Deep Learning, MIT Press, Cambridge, MA, 2016.

Guillaumin, A. P. and Zanna, L.: Stochastic-Deep Learning Parameterization of Ocean Momentum Forcing, J Adv Model Earth Syst, 13, e2021MS002 534, https://doi.org/10.1029/2021MS002534, 2021.

Han, Y., Zhang, G. J., Huang, X., and Wang, Y.: A Moist Physics Parameterization Based on Deep Learning, J Adv Model Earth Syst, 12, e2020MS002 076, https://doi.org/10.1029/2020MS002076, 2020.

Han, Y., Zhang, G. J., and Wang, Y.: An Ensemble of Neural Networks for Moist Physics Processes, Its Generalizability and Stable Integration, J Adv Model Earth Syst, 15, e2022MS003 508, https://doi.org/10.1029/2022MS003508, 2023.

He, K., Zhang, X., Ren, S., and Sun, J.: Deep Residual Learning for Image Recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, IEEE, Las Vegas, NV, USA, ISBN 978-1-4673-8851-1, https://doi.org/10.1109/CVPR.2016.90, 2016.

Heuer, H., Schwabe, M., Gentine, P., Giorgetta, M. A., and Eyring, V.: Interpretable multiscale Machine Learning-Based Parameterizations of Convection for ICON, J Adv Model Earth Syst, 16, e2024MS004 398, https://doi.org/10.1029/2024MS004398, arXiv:2311.03251 [physics], 2024.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Hourdin, F., Mauritsen, T., Gettelman, A., Golaz, J.-C., Balaji, V., Duan, Q., Folini, D., Ji, D., Klocke, D., Qian, Y., Rauser, F., Rio, C., Tomassini, L., Watanabe, M., and Williamson, D.: The Art and Science of Climate Model Tuning, Bulletin of the American Meteorological Society, 98, 589–602, https://doi.org/10.1175/BAMS-D-15-00135.1, 2017.

Hourdin, F., Rio, C., Grandpeix, J., Madeleine, J., Cheruy, F., Rochetin, N., Jam, A., Musat, I., Idelkadi, A., Fairhead, L., Foujols, M., Mellul, L., Traore, A., Dufresne, J., Boucher, O., Lefebvre, M., Millour, E., Vignon, E., Jouhaud, J., Diallo, F. B., Lott, F., Gastineau, G., Caubel, A., Meurdesoif, Y., and Ghattas, J.: LMDZ6A: The Atmospheric Component of the IPSL Climate Model With Improved and Better Tuned Physics, J Adv Model Earth Syst, 12, e2019MS001 892, https://doi.org/10.1029/2019MS001892, 2020.

Hu, Z., Subramaniam, A., Kuang, Z., Lin, J., Hannah, W. M., Brenowitz, N. D., Romero, J., and Pritchard, M. S.: Stable Machine-Learning Parameterization of Subgrid Processes with Real Geography and Full-physics Emulation, Journal of Advances in Modeling Earth Systems, 2024.

Krasnopolsky, V. M., Fox-Rabinovitz, M. S., and Chalikov, D. V.: New Approach to Calculation of Atmospheric Model Physics: Accurate and Fast Neural Network Emulation of Longwave Radiation in a Climate Model, Monthly Weather Review, 133, 1370–1383, https://doi.org/10.1175/MWR2923.1, 2005.

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, Advances in Artificial Neural Systems, 2013, 1–13, https://doi.org/10.1155/2013/485913, 2013.

Krinner, G., Viovy, N., De Noblet-Ducoudré, N., Ogée, J., Polcher, J., Friedlingstein, P., Ciais, P., Sitch, S., and Prentice, I. C.: A dynamic global vegetation model for studies of the coupled atmosphere-biosphere system, Global Biogeochemical Cycles, 19, 2003GB002 199, https://doi.org/10.1029/2003GB002199, 2005.

Lagerquist, R., Turner, D., Ebert-Uphoff, I., Stewart, J., and Hagerty, V.: Using deep learning to emulate and accelerate a radiative-transfer model, Journal of Atmospheric and Oceanic Technology, https://doi.org/10.1175/JTECH-D-21-0007.1, 2021.

LeCun, Y., Boser, B. E., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. E., and Jackel, L. D.: Handwritten Digit Recognition with a Back-Propagation Network, In Proc. Advances in Neural Information Processing Systems, pp. 396–404, 1990.

LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning, Nature, 521, 436–444, https://doi.org/10.1038/nature14539, 2015.

Limon, G. C. and Jablonowski, C.: Probing the Skill of Random Forest Emulators for Physical Parameterizations Via a Hierarchy of Simple CAM6 Configurations, J Adv Model Earth Syst, 15, e2022MS003 395, https://doi.org/10.1029/2022MS003395, 2023.

Lin, J., Yu, S., Peng, L., Beucler, T., Wong-Toi, E., Hu, Z., Gentine, P., Geleta, M., and Pritchard, M. S.: Sampling Hybrid Climate Simulation at Scale to Reliably Improve Machine Learning Parameterization, https://doi.org/10.22541/essoar.172072688.86581349/v1, 2024.

Liu, Y., Caballero, R., and Monteiro, J. M.: RadNet 1.0: exploring deep learning architectures for longwave radiative transfer, Geosci. Model Dev., 13, 4399–4412, https://doi.org/10.5194/gmd-13-4399-2020, 2020.

Manabe, S. and Wetherald, R. T.: The Effects of Doubling the CO2 Concentration on the climate of a General Circulation Model, Journal of the Atmospheric Sciences, https://journals.ametsoc.org/view/journals/atsc/32/1/1520-0469_1975_032_0003_teodtc_2_0_co_2.xml, section: Journal of the Atmospheric Sciences, 1975.

Medeiros, B., Williamson, D. L., and Olson, J. G.: Reference aquaplanet climate in the C ommunity A tmosphere M odel, V ersion 5, J Adv Model Earth Syst, 8, 406–424, https://doi.org/10.1002/2015MS000593, 2016.

Meyer, D., Hogan, R. J., Dueben, P. D., and Mason, S. L.: Machine Learning Emulation of 3D Cloud Radiative Effects, J Adv Model Earth Syst, 14, e2021MS002 550, https://doi.org/10.1029/2021MS002550, 2022.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Mooers, G., Pritchard, M., Beucler, T., Ott, J., Yacalis, G., Baldi, P., and Gentine, P.: Assessing the Potential of Deep Learning for Emulating Cloud Superparameterization in Climate Models With Real-Geography Boundary Conditions, J Adv Model Earth Syst, 13, e2020MS002 385, https://doi.org/10.1029/2020MS002385, 2021.

O'Gorman, P. A. and Dwyer, J. G.: Using Machine Learning to Parameterize Moist Convection: Potential for Modeling of Climate, Climate Change, and Extreme Events, J Adv Model Earth Syst, 10, 2548–2563, https://doi.org/10.1029/2018MS001351, 2018.

Ott, J., Pritchard, M., Best, N., Linstead, E., Curcic, M., and Baldi, P.: A Fortran-Keras Deep Learning Bridge for Scientific Computing, http://arxiv.org/abs/2004.10652, arXiv:2004.10652 [cs], 2020.

Pal, A., Mahajan, S., and Norman, M. R.: Using Deep Neural Networks as Cost-Effective Surrogate Models for Super-Parameterized E3SM Radiative Transfer, Geophysical Research Letters, 46, 6069–6079, https://doi.org/10.1029/2018GL081646, 2019.

Partee, S., Ellis, M., Rigazzi, A., Bachman, S., Marques, G., Shao, A., and Robbins, B.: Using Machine Learning at Scale in HPC Simulations with SmartSim: An Application to Ocean Climate Modeling, http://arxiv.org/abs/2104.09355, arXiv:2104.09355 [cs], 2021.

Rasp, S., Pritchard, M. S., and Gentine, P.: Deep learning to represent subgrid processes in climate models, Proc. Natl. Acad. Sci. U.S.A., 115, 9684–9689, https://doi.org/10.1073/pnas.1810286115, 2018.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., and Prabhat: 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.

Roh, S. and Song, H.: Evaluation of Neural Network Emulations for Radiation Parameterization in Cloud Resolving Model, Geophysical Research Letters, 47, e2020GL089 444, https://doi.org/10.1029/2020GL089444, 2020.

Ronneberger, O., Fischer, P., and Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation, http://arxiv.org/abs/1505.04597, arXiv:1505.04597 [cs], 2015.

Sanderson, B. M., Piani, C., Ingram, W. J., Stone, D. A., and Allen, M. R.: Towards constraining climate sensitivity by linear analysis of feedback patterns in thousands of perturbed-physics GCM simulations, Clim Dyn, 30, 175–190, https://doi.org/10.1007/s00382-007-0280-7, 2008.

Schneider, T., Lan, S., Stuart, A., and Teixeira, J.: Earth System Modeling 2.0: A Blueprint for Models That Learn From Observations and Targeted High-Resolution Simulations, Geophysical Research Letters, 44, https://doi.org/10.1002/2017GL076101, 2017.

Seifert, A. and Rasp, S.: Potential and Limitations of Machine Learning for Modeling Warm-Rain Cloud Microphysical Processes, J Adv Model Earth Syst, 12, e2020MS002 301, https://doi.org/10.1029/2020MS002301, 2020.

Veit, A., Wilber, M., and Belongie, S.: Residual Networks Behave Like Ensembles of Relatively Shallow Networks, https://doi.org/10.48550/arXiv.1605.06431, arXiv:1605.06431 [cs], 2016.

Wang, X., Han, Y., Xue, W., Yang, G., and Zhang, G. J.: Stable climate simulations using a realistic GCM with neural network parameterizations for atmospheric moist physics and radiation processes, https://doi.org/10.5194/gmd-2021-299, 2021.

Watt-Meyer, O., Brenowitz, N. D., Clark, S. K., Henn, B., Kwa, A., McGibbon, J., Perkins, W. A., Harris, L., and Bretherton, C. S.: Neural Network Parameterization of Subgrid-Scale Physics From a Realistic Geography Global Storm-Resolving Simulation, J Adv Model Earth Syst, 16, e2023MS003 668, https://doi.org/10.1029/2023MS003668, 2024.

Yuval, J. and O'Gorman, P. A.: Neural-network parameterization of subgrid momentum transport in the atmosphere, https://doi.org/10.1002/essoar.10507557.1, 2021.

Yuval, J. and O'Gorman, P. A.: Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions, Nat Commun, 11, 3295, https://doi.org/10.1038/s41467-020-17142-3, 2020.

https://doi.org/10.5194/egusphere-2025-1418
Preprint. Discussion started: 26 May 2025

Yuval, J., O'Gorman, P. A., and Hill, C. N.: Use of Neural Networks for Stable, Accurate and Physically Consistent Parameterization of Subgrid Atmospheric Processes With Good Performance at Reduced Precision, Geophysical Research Letters, 48, e2020GL091 363, https://doi.org/10.1029/2020GL091363, 2021.

Zhang, T., Morcrette, C., Zhang, M., Lin, W., Xie, S., Liu, Y., Van Weverberg, K., and Rodrigues, J.: A Fortran-Python Interface for Integrating Machine Learning Parameterization into Earth System Models, https://doi.org/10.5194/gmd-2024-79, 2024.