JAMES | Journal of Advances in Modeling Earth Systems

RESEARCH ARTICLE
10.1029/2021MS002477

Machine Learning Emulation of Gravity Wave Drag in Numerical Weather Forecasting

Matthew Chantry[1], Sam Hatfield[2], Peter Dueben[2], Inna Polichtchouk[2], and Tim Palmer[1]

[1]Atmospheric, Oceanic and Planetary Physics, University of Oxford, Oxford, UK, [2]European Centre for Medium-Range Weather Forecasts, Reading, UK

Key Points:

- Nonorographic gravity wave drag parametrization can be accurately emulated with a neural network
- These emulators produce accurate and stable forecasts over long timescales
- Neural networks can reduce the cost of an increased complexity scheme

Correspondence to:

M. Chantry,
matthew.chantry@physics.ox.ac.uk

Citation:

Chantry, M., Hatfield, S., Dueben, P., Polichtchouk, I., & Palmer, T. (2021). Machine learning emulation of gravity wave drag in numerical weather forecasting. *Journal of Advances in Modeling Earth Systems*, *13*, e2021MS002477. https://doi.org/10.1029/2021MS002477

Received 25 JAN 2021
Accepted 14 JUN 2021

Abstract We assess the value of machine learning as an accelerator for the parameterization schemes of operational weather forecasting systems, specifically the parameterization of nonorographic gravity wave drag. Emulators of this scheme can be trained to produce stable and accurate results up to seasonal forecasting timescales. Generally, networks that are more complex produce emulators that are more accurate. By training on an increased complexity version of the existing parameterization scheme, we build emulators that produce more accurate forecasts. For medium range forecasting, we have found evidence that our emulators are more accurate than the version of the parametrization scheme that is used for operational predictions. Using the current operational CPU hardware, our emulators have a similar computational cost to the existing scheme, but are heavily limited by data movement. On GPU hardware, our emulators perform 10 times faster than the existing scheme on a CPU.

Plain Language Summary The ability of computers to construct models from data (machine learning) has had significant impacts on many areas of science. Here, we use this ability to construct a model of an element of a numerical weather forecasting system. This element captures one physical process in the model, a part of the model that describes the propagation of large-scale waves through the atmosphere, but the long-term aim would be to make many models each capturing a process. The goal is that the computer-generated model will perform the task more efficiently than the existing model. Testing is then carried out to ensure that our computer model performs as accurately as the existing model. This is a challenging step, as learning is carried out over short time periods (seconds), but forecasts need to be accurate over years. Our computer-generated models produce accurate forecasts on all tested timescales. On current computers, they are not faster, but will be if weather forecasting centers invest in computers with graphics processing units.

1. Introduction

Numerical weather prediction has a proud history of saving lives and protecting property in societies particularly vulnerable to extremes of weather. As these extremes become more extreme still under the influence of climate change, it is important that numerical weather prediction systems improve further. One way to enhance the skill of numerical weather prediction is to increase model resolution. Not only will higher resolution directly enhance the ability of models to simulate small-scale extreme events, but also enables the information in observations to be better assimilated at initial time, and will reduce the dependence of models on inaccurate parameterized processes, and in this way will reduce systematic errors (Palmer, 2020). However, increasing model resolution is computationally expensive: a doubling of resolution can increase computing costs by up to a factor of 16. Therefore, we must find ways of improving the computational efficiency of our models, so that valuable computing resources can be targeted where they will be most effective.

Many physical processes that are involved in the forecasting of weather or climate occur at spatial scales smaller than the numerical grid of the models. Therefore, while the physics might be understood, it is necessary to derive closure schemes, which capture the effect of these physical processes on the grid scales. Examples of physical processes are radiation, convection and gravity wave drag, the latter of which will be our focus here. Inherently, by the nature of being a closure scheme, these physical parameterization schemes are uncertain and imperfect. Together, these schemes typically account for a significant portion of both the computational burden and the number of lines of code in the code base. In the European Center for Medium-Range Weather Forecast's (ECMWF) Integrated Forecasting System (IFS) model, they contribute

to about a third of the overall computational cost of running the model. As more Earth System complexity is introduced into numerical weather prediction models, e.g., aerosols and atmospheric chemistry, these numbers will only increase.

The recent boom in both hardware and software developments around machine learning has caused many fields to examine the possible boons that machine learning can bring. In the field of weather and climate forecasting, researchers are examining the applicability of machine learning techniques to a spectrum of problems (Chantry et al., 2021), covering changes from the seismic to the incremental. Seismic changes include investigations into whether machine learning can replace the whole forecasting system, either by learning from observational data (Sønderby et al., 2020) or atmospheric reanalysis (Rasp et al., 2020). Early results are promising in the area of nowcasting (Sønderby et al., 2020), but still lag behind classical modeling for short and medium range forecasting (Weyn et al., 2019) with evidence (Rasp & Thuerey, 2021) and arguments (Palmer, 2020) that there is insufficient data when moving to high resolutions. For seasonal forecasting, machine learning techniques again show promising results, e.g., forecasting El Nino sea-surface temperatures (Dijkstra et al., 2019). At the incremental end of the spectrum lies approaches where machine learning aims to learn the behavior of the existing kernels of a weather forecasting model, with the aim of building machine-learning emulators that can accelerate the kernel (Brenowitz & Bretherton, 2018; Chevallier et al., 1998; Krasnopolsky, 1997; O'Gorman & Dwyer, 2018). Successful acceleration of these kernels could allow reductions in run-time or reinvestments of the computation savings into increased complexity kernels or increases in spatial resolution. We shall investigate this approach here. The appeal of this approach is the ability to leverage the existing physics knowledge in a small self-contained unit. This application of machine learning to weather and climate forecasting is closely related to the use of reduced numerical precision to accelerate weather forecasting (Hatfield et al., 2019; Váňa et al., 2017), whereby a slightly less accurate version of a kernel can be used undetected beneath the uncertainty and inaccuracy of the system. On current GPUs, neural networks (NN) can be easily run at reduced precision for both training and inference, leveraging increased linear algebra performance (NVIDIA, 2017). While existing atmospheric algorithms need to be reformulated to fit within the compact dynamic range of half-precision (Hatfield et al., 2019), NN naturally fits within this range due to data normalization.

Several groups have already tackled the wider problem of physical parameterization schemes. These efforts fall into two groups; one seeks to build new parametrization schemes by training on observations or higher-resolution models, e.g., Brenowitz and Bretherton (2019), who built an NN to learn all parameterized physics by coarse-graining higher-resolution aqua-planet simulations. The second aims to emulate an existing parametrization scheme in order to increase the performance. This approach started in the 1990s when both Krasnopolsky (1997) and Chevallier et al. (1998) used NNs to emulate and accelerate radiation schemes. Their approaches were broadly successful but did not lead to widespread adoption within operational weather or climate models. In the case of Chevallier et al. (1998), when the number of vertical layers increased beyond 60 layers, the emulators could not be trained to work at sufficient accuracy (Morcrette et al., 2008). More recently, Yuval and O'Gorman (2020) used random forests to emulate a convection scheme. Efforts have also been made to accelerate increased complexity (dubbed super) parameterization schemes (Rasp et al., 2018; Gentine et al., 2018). Veerman and Pincus (2021) emulate a radiation scheme and assess the computational cost relative to the existing scheme. Ukkonen et al. (2020) emulate the gas optics scheme within a radiation scheme, providing acceleration for this kernel. Gettelman et al. (2020) use a NN to learn an increased complexity microphysics model and reduce the cost down to that of their reference scheme. Of this recent body of work, most have involved some simplification steps relative to an operational forecast model. These include reduced horizontal or vertical grids and aqua-planet configurations. Here, we will assess our emulators when coupled to models running on the 25 km grid used for long-range forecasting at ECMWF, a significant step toward operational forecasting.

Gravity wave drag is a parameterized physical process that has not yet been considered for emulation. It is typically parameterized as two processes, orographic and nonorographic. In the IFS model, these are two separate schemes and it is the latter that we shall mostly focus on in this study. Orographic gravity wave drag is discussed in Section 5. Nonorographic gravity waves can be generated by dynamics such as fronts and convection and occur on a vast range of scales (Gardner et al., 1989; Ern et al., 2004) meaning that while some gravity waves are resolved by current forecast resolutions, others need to be parameterized. In the IFS model, this is achieved using the scheme from Warner and McIntyre (2001) and in particular

the variant described in Orr et al. (2010) (henceforth referred to as NOGWD). Like most current physical parameterization schemes, it operates on a vertical column of fluid and can be run in parallel across all the columns within a given grid. In this scheme, a fixed amount of momentum is launched upward from a given height at a range of launch angles in the horizontal direction and over a range of phase speeds. At each model level, the stability of these waves within the atmospheric profile is calculated, and depending on this stability, momentum is deposited at these layers. Accurate parameterization of nonorographic gravity wave drag is important for maintaining an accurate zonal-mean wind and temperature distribution (Garcia & Boville, 1994) and for capturing the phase and amplitude of the Quasi-Biennial Oscillation (QBO) (Dunkerton, 1997). We choose to emulate NOGWD, as it is a medium complexity scheme that has particularly important effects on seasonal timescales. The effects can be seen both in the stratosphere and troposphere (Polichtchouk, Shepherd, & Byrne, 2018; Polichtchouk, Shepherd, Hogan, & Bechtold, 2018). Recent work has shown the challenges in using offline-trained NN in free-running simulations (Brenowitz et al., 2020) and we will assess whether these problems also exist in our scenario. By sitting in the middle of the complexity spectrum, NOGWD has enough complexity to challenge machine-learning methods but it is nontrivial that an emulator will be faster than the current scheme. If we can demonstrate the efficiency gains in this parameterization scheme, we would postulate that many of the parameterization schemes can be efficiently emulated using NN.

Next, in Section 2, we discuss the data used for the machine learning, followed by our machine learning methodology (Section 3). In Section 4, we emulate the NOGWD scheme and test the performance both decoupled and coupled to the IFS model. In Section 5, we address several important questions in taking parameterization emulators into operational prediction systems. We also present new results and discussions on the key issues of conservation properties, computational cost, alternate network structures, generalizability and emulating other parameterization schemes.

2. Data

The specific goal of this project is to create emulators of the nonorographic gravity wave drag parameterization found in the IFS model. To generate the data, we modify cycle 45R1 of IFS (the operational cycle in 2018, ECMWF, 2018) to output all time-dependent variables either inputted to or outputted from the scheme in question, saved on the native cubic-octahedral grid. We use the horizontal grid TCo399, equivalent to ∼25 km grid-point spacing. In the vertical dimension, we use 91 model levels spanning the surface to 1 Pa. As outlined above, momentum is launched at a set number of launch angles in the horizontal plane, and for a range of phase speeds. Typically, four launch angles are used, describing momentum in the four cardinal directions. The wavespeed spectrum is discretized into 20 elements. Further explanation of these parameters can be found in Orr et al. (2010). When generating data sets, we increase these values to 16 and 100, respectively, which we refer to from hereon as the HC (high complexity) scheme. These changes increase the cost of the existing scheme by a factor of 20, but provide a higher resolution discretization of the equations governing NOGWD. Scinocca (2003) suggest that 35 elements is sufficient but differences can still be observed between 35 and 1,000 elements. We observe that the HC variant of the NOGWD scheme produces more accurate forecasts (depicted later in Figures 4 and 5). In preliminary testing, we found that our networks produce a lower mean-squared-error against the testing data set when both training and testing data came from the HC scheme, compared with when the standard complexity scheme provides the testing and training data. This is perhaps because the HC has smoother vertical tendency profiles. In our online testing, we will compare both the standard formulation (with 4 and 20) and the HC variant of the existing scheme later.

Here, IFS is run for 30-day integrations, restarting every 30 days, in total covering a 3-year period. This was chosen to give a full seasonal cycle for each of training, testing and validation data. Data is saved every 20 h, in this way sampling the daily cycle. At our chosen resolution, there are 654,400 columns within our grid. Without further reductions, each 30-day period produces 23,558,400 input/output columns for training. We subsample every tenth column, changing the starting index for different 30-day periods (e.g., grid-columns 1, 11, 21... in the first period and 2, 12, 22... in the second period). In this fashion, every grid-point features in the training data set. Our training data set comprises 12 30-day periods, covering 2015, totaling 28,270,080 training pairs. Our validation and test data sets comprise three 30-day periods spread across 2016 and 2017, respectively, (starting 2016-02-25, 2016-06-24, 2016-12-21 and 2017-02-19, 2017-07-19, 2017-11-16,

respectively). Our training data set is significantly larger than the data set that is typically used in emulation training, e.g., Brenowitz et al. (2020) used 1,797,120 columns in training, although notably smaller than the 140 million examples used in Rasp et al. (2018). We tested reducing the data volume by training only on the first 10% of each month, but training for 10-times the number of epochs (number of complete passes through the data during training). Training with this data gave a significant degradation in our testing error (~40% larger), but did not show any obvious signs of over-fitting. For training, using a large data set but iterating for fewer epochs lead to better results on the validation and testing data sets. The size or precise period selected for the validation and testing data set did not change the results. The increased training data set provides increased generalization ability on unseen data.

To mirror the existing parameterization scheme, we include only the variables used to predict the velocity tendencies in the existing model. These are the horizontal velocity (u and v) and temperature profiles on the vertical model levels. In addition, there are three descriptors of the model levels at each grid-point: the pressure, half-level pressure and geopotential. The outputs of the existing scheme are horizontal velocity tendencies, showing the impact of unresolved gravity wave drag on the velocity field. A temperature tendency can then be derived from the velocity fields and NOGWD tendencies using a simple formula. $T_t = -\frac{1}{c_p}\left(uu_t + vv_t\right)$, where $_t$ denotes the NOGWD tendency and c_p is the dry air calorific capacity at constant pressure. In total, this produces a data set with $6 \times 91 = 546$ inputs and $2 \times 91 = 182$ outputs.

Reductions can be made to both input and output vector sizes by applying simple domain knowledge. First, examination of the existing scheme shows that the scheme is only active in the upper 63 layers of the atmosphere and only uses information in the top 63 layers of the input profiles. Second, pressure and half-level pressure can each be described on model levels as time-independent functions of surface pressure. Theoretically, geopotential can be reconstructed using the surface geopotential, pressure, temperature and humidity, the last of which is not inputted to the scheme. In practice, we find our models produce accurate results using only surface pressure and surface geopotential. Combining these ideas, we produce input and output vectors of size $3 \times 63 + 2 = 191$ and $2 \times 63 = 126$, respectively. Even after data reduction, the size of training data totals in excess of 30Gb, larger than our CPU memory. Therefore, we re-write the data into the TFRecord format (Abadi et al., 2015) to allow easy and efficient streaming of data from disk to GPU. Data are publicly available in 30-day chunks in the HDF5 file format for portability. (https://storage.ecmwf.europeanweather.cloud/NOGWD_L91_PUBLIC/README.html).

Data normalization is a typical step in any machine learning workflow. Here, we normalize both the input and output data from our data set. We examine two normalization approaches. The first, here dubbed "MEANSD", calculates elemental means and standard deviations for each feature (i.e., a variable at a given model level) and normalizes both inputs and outputs by these values. The second, dubbed "TMEANVMAX", normalizes temperature (plus pressure and geopotential) with the above method, but for each of the velocity inputs and tendency outputs, the entire column is divided by the largest standard deviation from the column and no mean is subtracted. With both methods, our loss function will be the L2 loss (mean squared error). With the first normalization, our models will seek to optimize each output feature equally, irrespective of the typical magnitude of the velocity tendency. With the second approach, the model will be encouraged to learn features in proportion to their absolute size. The latter method might seem a more natural choice given that the outputs will contribute to the next velocity field values. However, if we examine the structure of the velocity profiles in Figure 1, we see that the largest tendencies occur at the very top of the model atmosphere. The top of the atmosphere is weakly constrained by data assimilation and velocity fields in the top 10 layers of the 91 level IFS model that are strongly damped with a sponge layer to prevent wave reflection from the rigid upper boundary. Therefore, allowing the model to focus on these layers might not be constructive for an accurate forecast when coupled back to the full atmospheric model. We will therefore try both approaches and later test both in coupled simulations.

3. Machine Learning Methodology

In this work, we focus on NN as our tools for emulation. For those unfamiliar with NNs, we recommend Brenowitz and Bretherton (2018) who describe NN in the context of emulation. Currently, both NN and random forests are popular methodologies for parameterization scheme emulation or creation. Random forests can conserve physical quantities (e.g., energy conservation in O'Gorman & Dwyer, 2018), but recent work

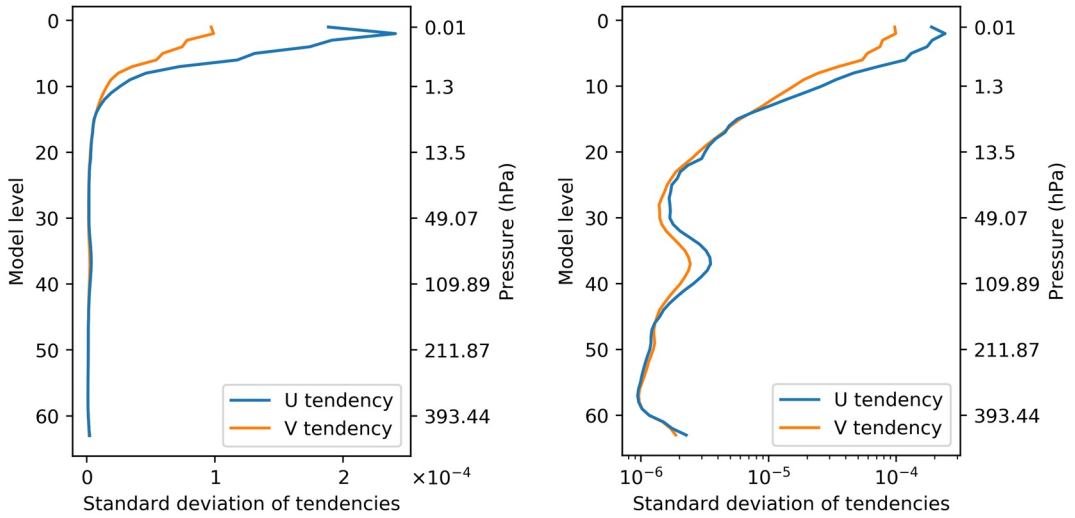

Figure 1. Standard deviation of the u and v wind tendencies from the high complexity NOGWD scheme (using the first 30 days of the data set), plotted on linear (left) and semilogarithmic (right) axes as a function of model level, plotting on the model levels where the scheme is activated. Velocity tendencies are largest at the top of the atmosphere (~0.01 hPa), with a second peak around the 30th model level (~70 hPa).

has shown how to also achieve this with NN (Beucler et al., 2019). Neural networks promise greater theoretical performance as random forests are limited by memory performance. However, random forest methods have recently been shown in a cloud parameterization scheme to be more stable in long-term simulations coupled to atmospheric models (Brenowitz et al., 2020). With this in mind, we will carefully assess whether our NN models run stably when coupled to the IFS model for seasonal timescales to further investigate this issue. We use Tensorflow (Abadi et al., 2015) to build and train our networks.

Predominantly, we will focus on fully connected NN in our search space. These networks are the most general purpose, with no explicit encoding of the vertical structure of our data set, which would occur with a convolutional-based network. However, we will present results for more sophisticated network architectures in Section 5.

Within the fully connected NN framework we wish to explore many of the remaining hyperparameters. We constrain our networks to have constant width (for each layer) but explore the space of layer width (between 1 and 1,000 neurons) and number of hidden layers (between 1 and 10 layers). For the activation function, we test ReLU, tanh, leaky ReLU and Swish (Ramachandran et al., 2017) activation functions, and consistently find that tanh and Swish produce the lowest training losses. After preliminary exploration, we settle on using the Adam optimizer with a learning rate of 10^{-4} and a batch size of 256. We train for 50 epochs, checkpointing after each epoch to select the best model on the validation data set.

4. Results

Given our desire to accelerate an existing parameterization scheme, the speed of any machine-learning emulator built is as important as the accuracy. Therefore, we present our offline results as a function of the degrees of freedom, DOF, (number of trainable parameters) in each network. For a fully connected NN, this is a good proxy for the number of floating point operations (or FLOPs) required to calculate a single inference step on a column of data. The degrees of freedom in a fully connected NN are dominated by the neuron weights, each of which is used once in a fused multiply-add operation as part of a matrix-vector multiplication. FLOPs are only one way of measuring the computational cost and are the appropriate measure when a calculation is operation-bound rather than memory-bound. However, as most parts of weather and climate models are not able to operate anywhere close to the peak performance of the hardware and as the ability to leverage peak performance will depend on pattern and data requirement of kernels, the measure of FLOPs may be misleading when estimating computing time. We will later return to this issue during online testing.

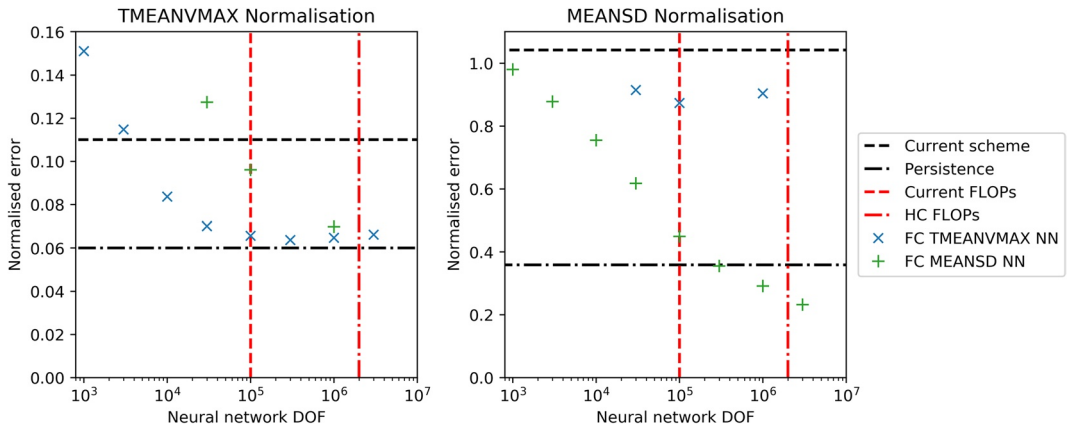

Figure 2. Offline root-mean-squared error of the best performing network for a range of prescribed degrees of freedoms. Networks are trained and tested on data using two normalization methods, dubbed MEANSD and TMEANVMAX. For each normalization plot, we plot models that were trained using both normalization methods (not all models are plotted on each plot). To contextualize the offline results, we plot two baselines. First, the error of the current scheme relative to the high complexity (HC) scheme (which was used to generate the training data). Second, the error when persisting the HC scheme tendencies for the following time step. A vertical red line denotes the approximate floating-point operations cost of the current scheme.

4.1. Offline Learning

In Figure 2, we plot the performance of our best models for a given number of DOF on the test data set for the two normalization schemes. To give context to the error and cost of these schemes, we plot the error on the test data set of using either the existing scheme or persisting the HC scheme tendencies for the following time step. In the operational version of IFS, persistence is used every second hour for the NOGWD scheme to reduce costs. Persistence on longer timescales would lead to a dramatically worse model. Relative to the HC scheme, we are easily able to construct NN that produce lower error than the existing scheme. Beating persistence is a more challenging task that is only achieved for our most expensive MEANSD normalization networks. For the largest TMEANVMAX networks, our error no longer decreases and indeed shows a slight increase for the largest networks that we train. Potentially, the addition of regularization (e.g., on the weights) could help with training large networks. To understand the impact of normalization, we test a subset of networks trained with MEANSD by evaluating their outputs with the TMEANVMAX normalization. For large networks, we see similarly good scores as those trained with TMEANVMAX. On assessing TMEANVMAX networks using the MEANSD normalization, we see much poorer performance, even for large network sizes, suggesting that the smaller magnitude features have not been learnt as well. Coupled testing is required to explore whether these features have an impact on forecast quality. This offline testing phase helps identify the best network architectures for a given number of DOF. For both normalization methods, there is no clear pattern relating the depth, width and the DOF. In Table 1, we present our optimal networks which will be used in the coupled testing. For a given DOF choice, we use our best network with the Tanh activation function, as this has comparable performance with the Swish function but this activation function already exists with the Fortran standard. These networks consist of 4–6 hidden layers, but test data set losses vary little between 3 and 10 layers. This suggests that the precise network architecture is unimportant for these fully connected fixed-width networks in our problem.

Table 1
Table of Model Architectures Used in the Coupled Simulations

Normalisation	DOF	Hidden layers	Hidden width	Nonlinearity
TMEANVMAX	30,000	51	6	Tanh
MEANSD	30,000	55	5	Tanh
TMEANVMAX	100,000	136	4	Tanh
MEANSD	100,000	122	5	Tanh
TMEANVMAX	1,000,000	461	5	Tanh
MEANSD	1,000,000	416	6	Tanh

4.2. Coupling Neural Networks Within IFS

Small offline errors are not a necessary or sufficient condition to find good performance when using the networks online with simulations with the IFS. Theoretically, seemingly small biases could accumulate as large errors and trigger instabilities even if the mean errors are small. To

understand the impact of our schemes on the wider forecast, we couple our NNs back into the IFS code replacing the existing NOGWD scheme. This requires interfacing NNs, typically written in Python, with Fortran, and is a problem faced by many machine-learning researchers in this field. In our work, we solve this problem by writing a Fortran module to load the weights saved and reproduce the NN architecture using matrix-matrix multiplication algorithms found in the BLAS library (Blackford et al., 2002). This works well for simple network architectures but would be more difficult to realize for the complex network structures that can be built in libraries such as Tensorflow. Recently, Ott et al. (2020) have produced an open-source package, which accomplishes a similar task.

4.3. Medium Range Forecasting

To demonstrate the performance of our networks in online mode, that is, coupled to the IFS, we complete a 120 h forecast at ∼25 km (TCo399) resolution using a variety of model configurations. Our truth here is chosen to be a simulation using the high-complexity variant of the existing NOGWD scheme. We choose the HC NOGWD scheme as truth instead of reanalysis because the aim is to emulate the HC scheme with an NN and not the atmospheric reanalysis. Against this, we plot the forecast error using the existing NOGWD scheme, using no activated NOGWD scheme at all and using six NN configurations covering both normalization methods and a range of network sizes. For each simulation, we calculate the horizontally averaged RMSE relative to a forecast using the HC scheme and plot the error as a function of pressure after 120 h in Figure 3 (comparable results were found for other lead times). For the TMEANVMAX normalization, we find similar performance for each network, irrespective of network complexity, with each of the schemes producing a comparable forecast to the existing variant of the scheme. For the MEANSD normalization, the higher complexity networks with 10^5 and 10^6 DOF produce the closest forecast to the truth, marginally more accurate than the existing version of the scheme. However, the small MEANSD network produces a notably degraded forecast. This suggests that if one is trying to produce the cheapest possible forecast, then the TMEANVMAX normalization allows a network to learn the key features cheaply. However, to produce a best possible forecast there is value in a more equal weighting of the features, as given by the MEANSD normalization.

We now carry out 10-day forecasts at the TCo399 resolution starting each day within July and December 2019. In Figures 4 and 5, we plot the results averaged across the start dates. We measure the difference in RMSE for pairs of forecast models, each assessed against the ECMWF operational analysis for the temperature (Figure 4) and wind fields (Figure 5). The top four panels of each plot show the improvement/degradation (blue/red contours) in forecast quality when using the HC variant of the existing NOGWD scheme versus the existing scheme. Hatching denotes statistically significant differences (Geer, 2016). The other eight panels show equivalent results when using the 100k MEANSD or 100k TMEANVMAX NNs. For both fields, we see the very similar patterns of change in forecast quality for both the HC and NN schemes, demonstrating the accuracy to which this HC scheme has been learnt. Especially for the temperature field, both the HC and NN schemes provide notable improvements in forecast quality, particularly over the poles where the error is reduced by more than 0.2°C for long lead-times. For the winds, there is a balance between improvement with some degradation in the tropics around 1 hPa. Overall, the NN has a positive impact on the forecast quality. In these tests, there is no clear winner between the MEANSD and TMEANVMAX normalization approaches.

4.4. Year-Long Simulations

In order to fully measure the accuracy and stability of our NNs, we now assess their performance in long-range forecasting. When the NOGWD scheme was first introduced, it was found to have a positive impact on the ability of the IFS model to capture the quasi-biennial oscillation (QBO) when compared with the previous Rayleigh friction model Orr et al. (2010). Crucially, the Rayleigh friction model fails to capture the descent phase and produces an overly regular oscillation period. Additionally, the latitude-pressure distribution of zonal winds and temperatures in the middle-atmosphere were improved by the introduction of the NOGWD scheme Orr et al. (2010). For computational cost reasons, we carry out these simulations with TL159 (∼125 km horizontal resolution), but do not retrain for this different resolution. To examine the

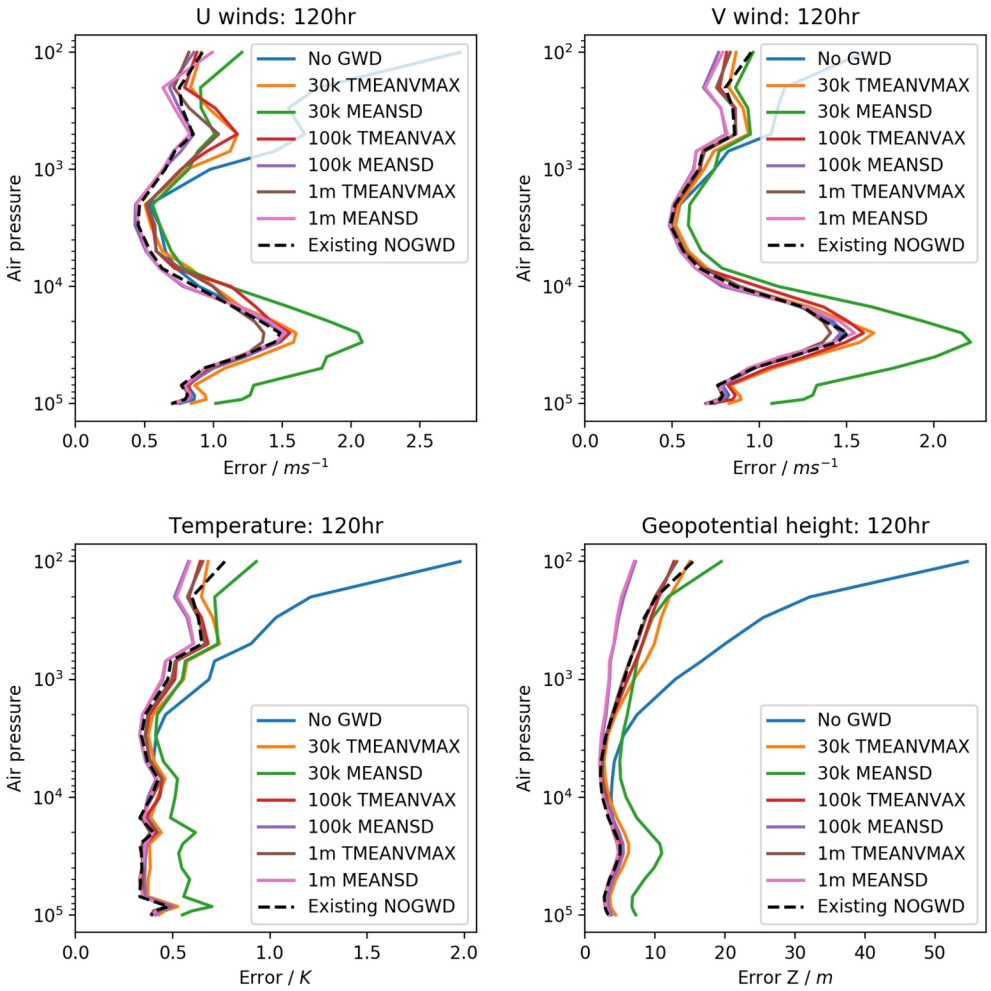

Figure 3. Horizontal-average forecast errors in simulations after 120 h at TCo399 (~25 km) resolution. Truth is taken to be a simulation of integrated forecasting system using the high complexity version of the NOGWD scheme. Against this, we measure the existing NOGWD scheme, no NOGWD scheme and six variants of our neural networks (NNs) covering a range of complexities for both normalization methods. Except for the 30,000 (30k) degree of freedom MEANSD NN, each of the NNs have comparable errors to the original scheme, with several of the MEANSD schemes outperforming the original.

behavior of our networks on long timescales, we carry out six year-long simulations starting on the first of November in the years 2009–2014. We forecast this period with the high-complexity and normal versions of the existing NOGWD scheme, the Rayleigh friction scheme and four of the configurations described in the previous section, omitting the cheapest networks. In Figure 6, we plot the time-pressure behavior of the equatorial zonal wind, calculated as the horizontal average between latitudes −5 and 5. Discontinuities indicate where a new simulation is started and these exists for each experiment. Figure 7 shows differences in jet structure from the HC truth run. Each of the NNs tested significantly outperforms the old Rayleigh friction scheme and shows clear descent of the jet altitude. We find that the MEANSD normalized networks do a better job at capturing the strong descents observed in the two variants of the original schemes in 2013 and 2015. Interestingly, we find only a small improvement for the 1 million (1m) DOF models over the 100,000 (100k) DOF models. For the 100k MEANSD normalization, we observe a small amount of noise at the top of the atmosphere, but this appears to have no noticeable impact on the jet structure.

From the same experiments in Figures 8 and 9, we plot the June-July-August (JJA) zonal-mean zonal wind and temperature distribution, averaged over all six start-dates. This period was chosen to capture winter in the southern hemisphere, comparable results were seen for the December-January-February period for the

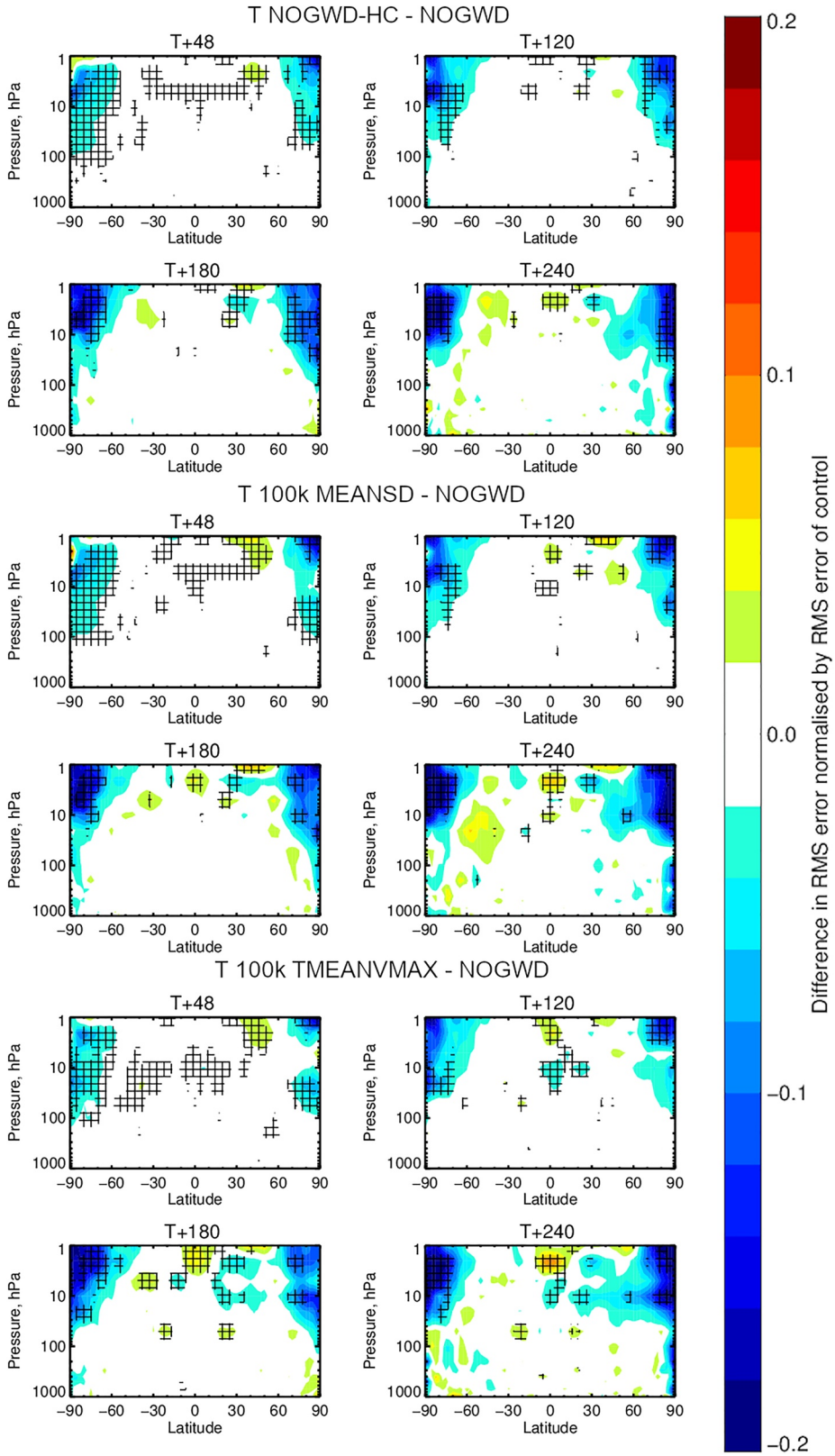
T NOGWD-HC - NOGWD
T+48
T+120
T+180
T+240
T 100k MEANSD - NOGWD
T+48
T+120
T+180
T+240
T 100k TMEANVMAX - NOGWD
T+48
T+120
T+180
T+240
Pressure, hPa
Latitude
Difference in RMS error normalised by RMS error of control
0.2
0.1
0.0
−0.1
−0.2

Figure 4.

northern-hemisphere winter period (not plotted). For the zonal winds, we see some improvement in the jet structure for the more expensive NNs. For the temperature field, we see a slightly better performance for the networks normalized using the TMEANVMAX scheme, in contrast to the results for the QBO where the MEANSD schemes produce better jet structure.

In these yearly integrations, we find some evidence that NNs trained with the MEANSD normalization approach outperform the TMEANVMAX method. Reexamining Figure 1, we see a local maximum in tendency activity around the 35th model level (~70 hPa). In the MEANSD approach, each level has equal contribution to the loss function, encouraging our networks to capture these variations. In the TMEANVMAX approach, these mid-level tendencies are an order of magnitude smaller than those at the top of the atmosphere and so have little contribution to our loss function. As such, the network struggles to reproduce the descent as faithfully.

5. Discussion

5.1. Conservation Properties

The existing NOGWD scheme conserves momentum. A fixed amount of momentum flux is launched upwards within a column, with momentum deposited at each layer depending on wave stability. All remaining momentum that reaches the uppermost layer is deposited there to ensure conservation. In our first iterations of NNs, we utilized the same approach and trained our networks to predict all other output layers and used momentum conservation to deduce the uppermost tendency. This approach is similar to the work of Beucler et al. (2019). In our formulation, our training loss does not explicitly include the uppermost layer, whereas Beucler et al. (2019) use conservation properties to deduce the uppermost layer which is then included in the loss calculation. However, during our long-range forecasting experiments we found that this momentum conservation approach occasionally destabilized our simulations. Due to the very strong difference in mass per volume of air, a correction of momentum at the top of the model, that is caused by a relatively small change in momentum close to the ground, can have a significant detrimental effect on the local momentum budget at the model top which can trigger model instabilities. For our final networks, we abandon exact momentum conservation and instead directly predict the top layer tendencies along with all other layers using our networks. With this approach, we are able to produce stable and accurate parameterization schemes. Further work could be undertaken to follow Beucler et al. (2019) and use conservation of momentum before the loss is calculated to test if this stabilizes the forecasts.

5.2. Performance Analysis

Given our motivation for building these networks, that is, improving the overall performance of the IFS model, an assessment of the performance is a crucial albeit complex step. As previously highlighted, the current NOGWD scheme uses approximately 100,000 FLOPs to calculate the NOGWD tendencies for a single column. We test our performance in 10-day forecasts at TCo399 resolution, using 66 processors each using six threads. The current NOGWD scheme is run at the IFS standard of double precision, whereas the NNs are run at the trained precision, single-precision. In this setup, 1.6 million columns are evaluated per thread during the simulation. In our test, IFS setup 300 s or 1% of the total simulation time are spent on the NOGWD calculations. By comparison, when using one of our 100k DOF models in the equivalent setup, the NOGWD scheme takes 360 s per thread, slightly slower than the current version. For a 30k DOF model, this is reduced to 250 s, slightly cheaper than the current scheme but an insignificant performance increase. To better understand the performance bottlenecks of both schemes, we also run each scheme decoupled from the IFS model. Simulating the same number of columns (1.6 million) in decoupled testing takes 18 s for the existing NOGWD scheme, 20 s for a 100k DOF model or 10 s for a 30k DOF model. This shows that both the original and NN scheme are

Figure 4. Change in the zonally averaged RMS error against the analyzed state of the atmosphere relative to the existing NOGWD scheme for the temperature field. Top: the change when using the high-complexity (HC) variant of the existing scheme after 48, 120, 180 & 240 h. Middle: equivalent changes when using our 100k MEANSD NN. Bottom: equivalent changes when using our 100k TMEANVMAX NN. Results for each are averaged over 62 forecasts corresponding to initializing the model at midnight every day in July and December 2019. Hatching denotes statistically significant differences. Significant improvements are seen for HC and neural networks (NN) models over the existing scheme. In particular, the strong similarities between the patterns for the HC scheme and NN indicating the NN has accurately emulated the HC scheme.

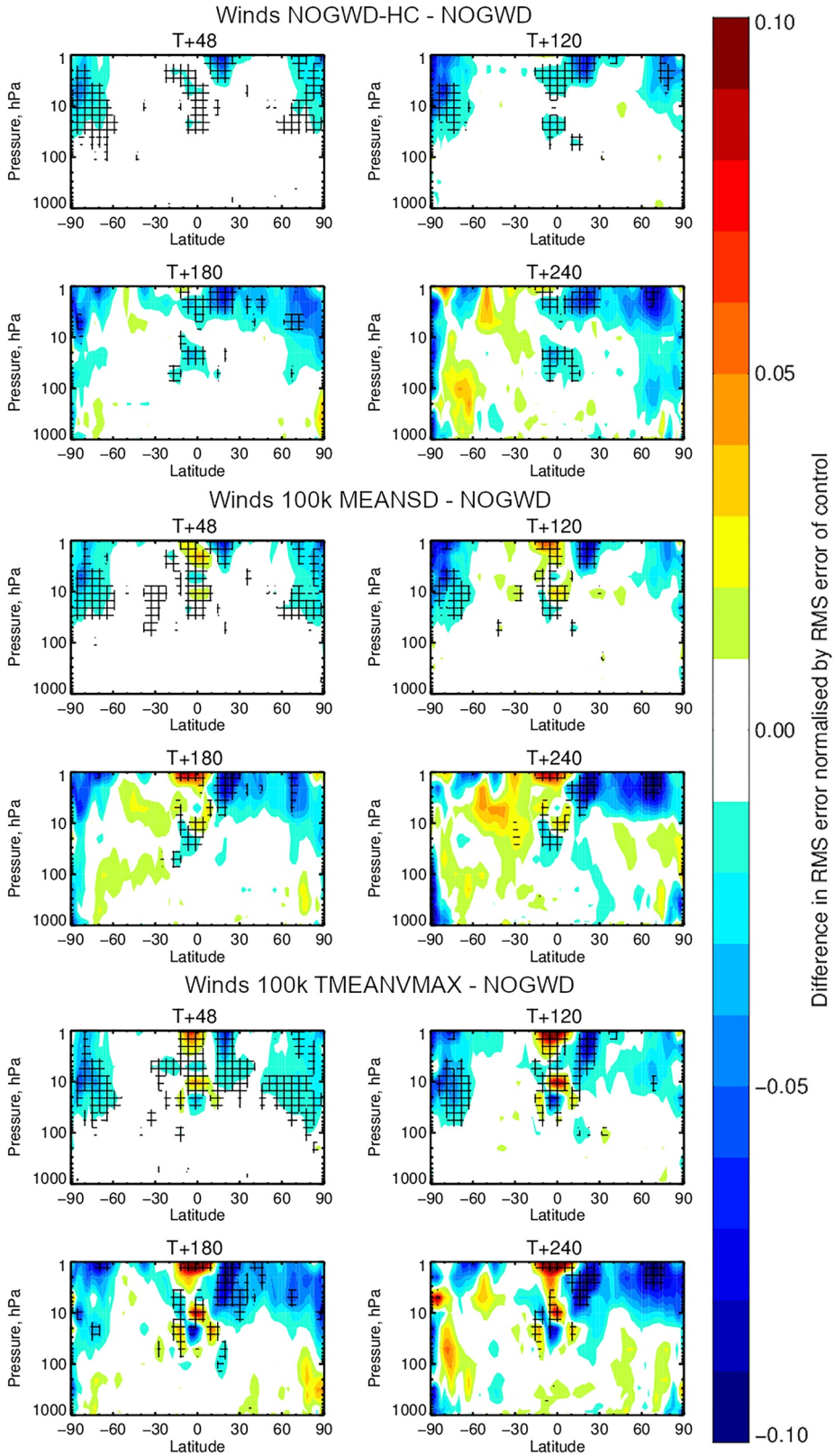

Figure 5. Equivalent to Figure 4 but for the horizontal wind components. While more red (reduced accuracy) is observed than for the temperature field, both the high-complexity (HC) and neural network (NN) solutions show more improvements and crucially show similar patterns, indicating the NN has accurately emulated the HC scheme. Both NN solutions introduce some degradation in the tropics around 1 hPa.

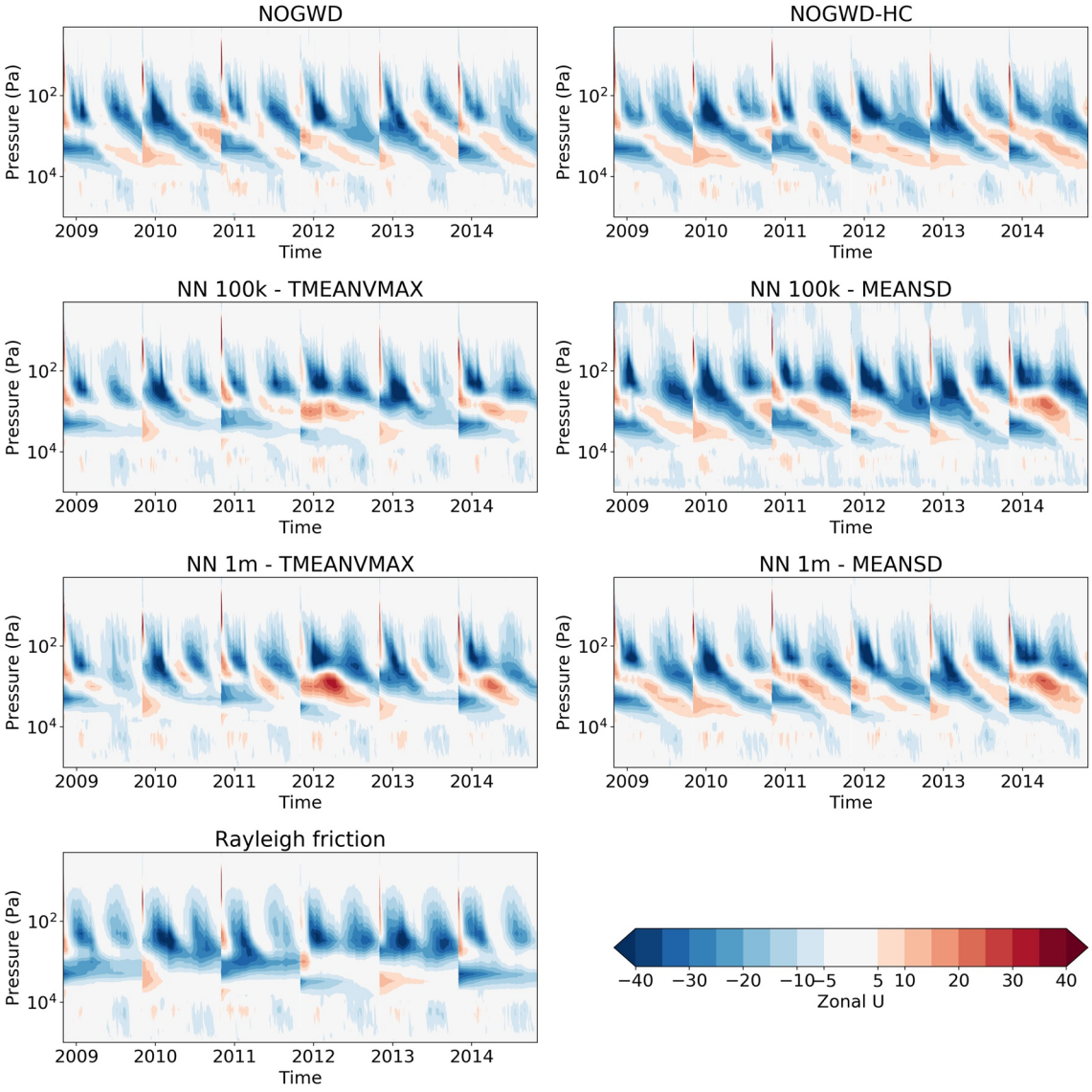

Figure 6. Average zonal-mean zonal jet between latitudes −5 to 5 for six consecutive yearlong integrations at TL159 resolution (~125 km). Plots show the ability of the integrated forecasting sysytem model to capture the Quasi-Biennial Oscillation (QBO) when coupled to different NOGWD schemes: NOGWD, the current scheme; NOGWD HC, the increased complexity variant; four neural network (NN) schemes covering different complexities and normalization approaches; Rayleigh friction, the precursor scheme to the current NOGWD scheme. Both variations of the current scheme produce similar dynamics, which are faithfully reproduced by the MEANSD NNs. The TMEANVAX networks outperform Rayleigh friction but fail to adequately reproduce the descent phase of the QBO.

heavily limited by data movement, so approaches that can overcome this problem can have dramatic performance increases. One such vision is a heterogeneous hardware cluster where some of the parameterization schemes are calculated on GPUs with the output being sent back to the CPU for the remaining simulations. We have shown here that this could be achieved by training NN emulators, meaning that the existing code does not have to be converted to CUDA or other GPU-appropriate languages. To understand the possible benefits of this architecture, we test our NNs on a NVidia V100 GPU. Even when including the time for transferring the data to and from the GPU, a 100k DOF NN takes 4 s to simulate 1.6 million columns. Even this time is dominated by overhead costs, as a 1m DOF NN also takes 4 s. Therefore, while the CPU performance when coupled to the IFS is not currently faster than the existing scheme, there is scope for dramatic performance gains on heterogeneous hardware, particularly if more parameterization schemes can be accurately emulated

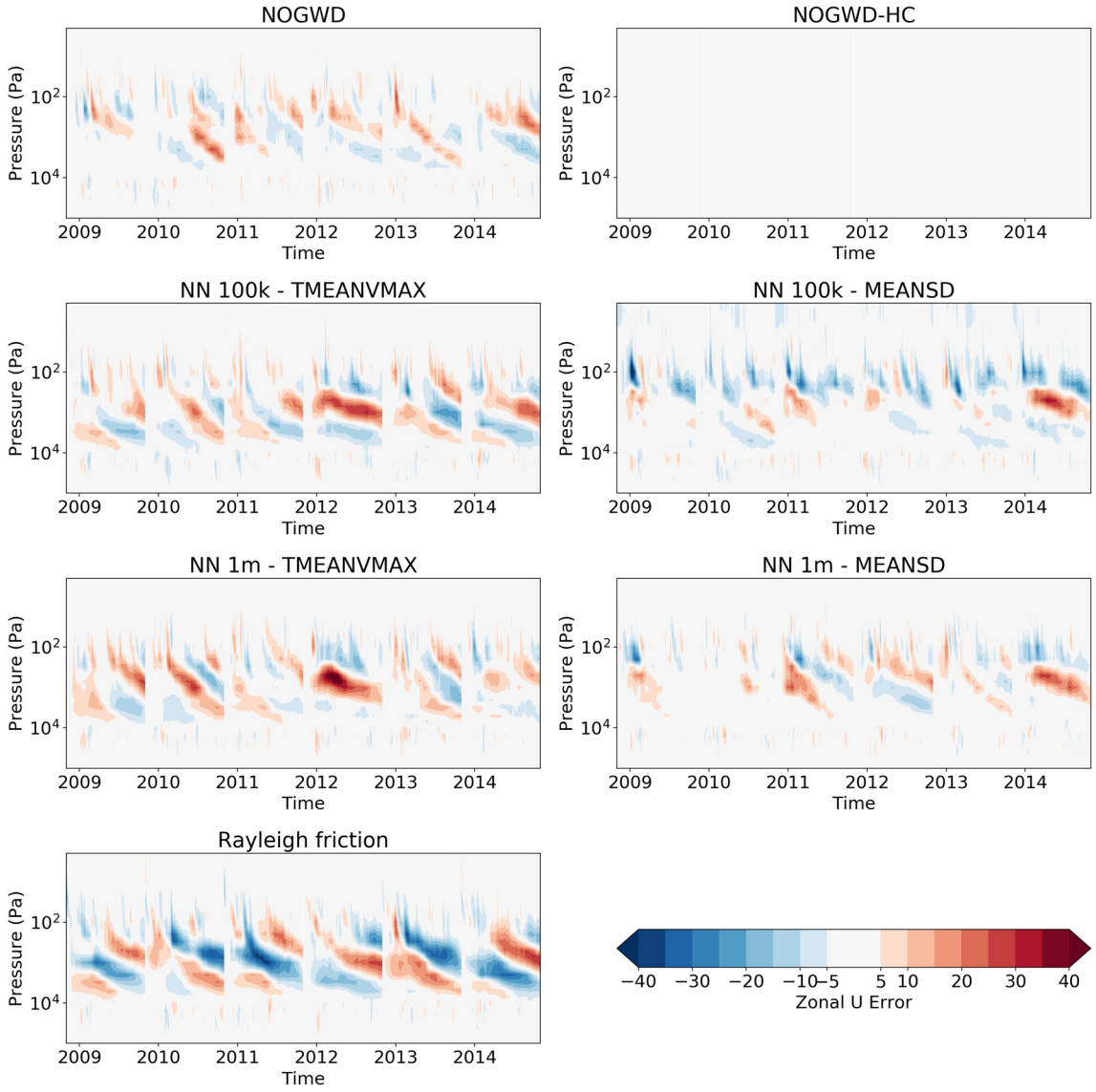

Figure 7. Difference in the zonal jets between the simulations plotted in Figure 6. NOGWD high-complexity is treated as the reference scheme and hence has no error. Both 100,000 (100k) and 1 million (1m) degree of freedom MEANSD schemes have comparable errors to the existing NOGWD scheme. Errors for the TMEANVMAX scheme are larger.

by NNs ECMWF's scalability program (Bauer et al., 2020) that is currently adapting existing parametrization schemes to be GPU portable, enabling a GPU comparison in the future.

5.3. Reduced Numerical Precision

Recent developments of GPU and Tensor Processing Unit (TPU) hardware have given users access to low numerical precision floating point numbers with significantly improved performance for NNs. Outside of machine learning, elements of both the dynamics (Hatfield et al., 2019) and parameterized physics (Saffin et al., 2020) can be calculated at half-precision with no impact on forecast quality. To test the applicability of reduced precision networks for our emulation, we utilize Tensorflow's mixed-precision capability which stores variables at 32 bits (single-precision) but uses 16 bits (half-precision) for intermediate calculations. Through both our training and offline testing phases, we find no notable degradation in the accuracy of our networks. Currently, most CPU architecture does not support calculations at half-precision, so online

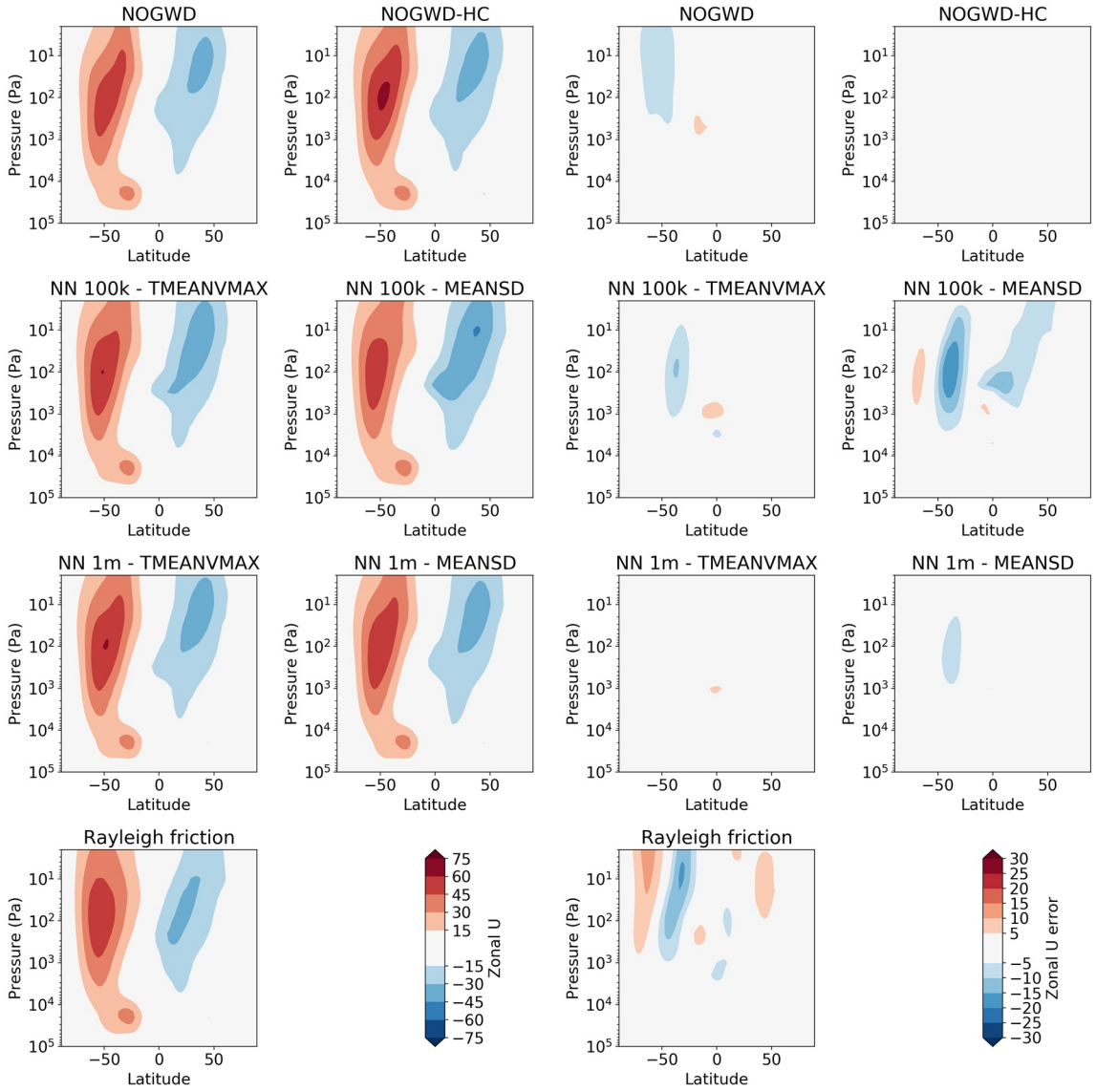

Figure 8. June-July-August average zonal jet as a function of latitude and pressure level for variants of the NOGWD scheme. Plots on the left show the jet with differences to the NOGWD high-complexity scheme depicted on the right. Here, the more expensive schemes slightly outperform their cheaper counterparts.

testing cannot currently be easily carried out. Recent work in emulation of convection parameterization used emulated half-precision with good network accuracy (Yuval et al., 2020). It is future work to couple our emulators to the IFS model in such a way that GPUs and reduced numerical precision can be leveraged.

5.4. Alternate Network Structures

In this study, we considered fully connected fixed-width NNs, one of the simplest network designs available. In this architecture, each node in the first layer has access to all input features, with no knowledge of the vertical structure of the atmosphere. Given that many calculations in physics are local in space, e.g., calculation of a vertical derivative, this suggests that improved performance could be found by a more appropriate choice of network architecture.

One choice is using convolutional layers, and a fixed local (in the vertical direction) operator where the same weights are used for each vertical layer in the atmosphere. Given that our model levels are unequally spaced in both distance from the Earth (from O[m] to O[km] distance per layer) and pressure values, this

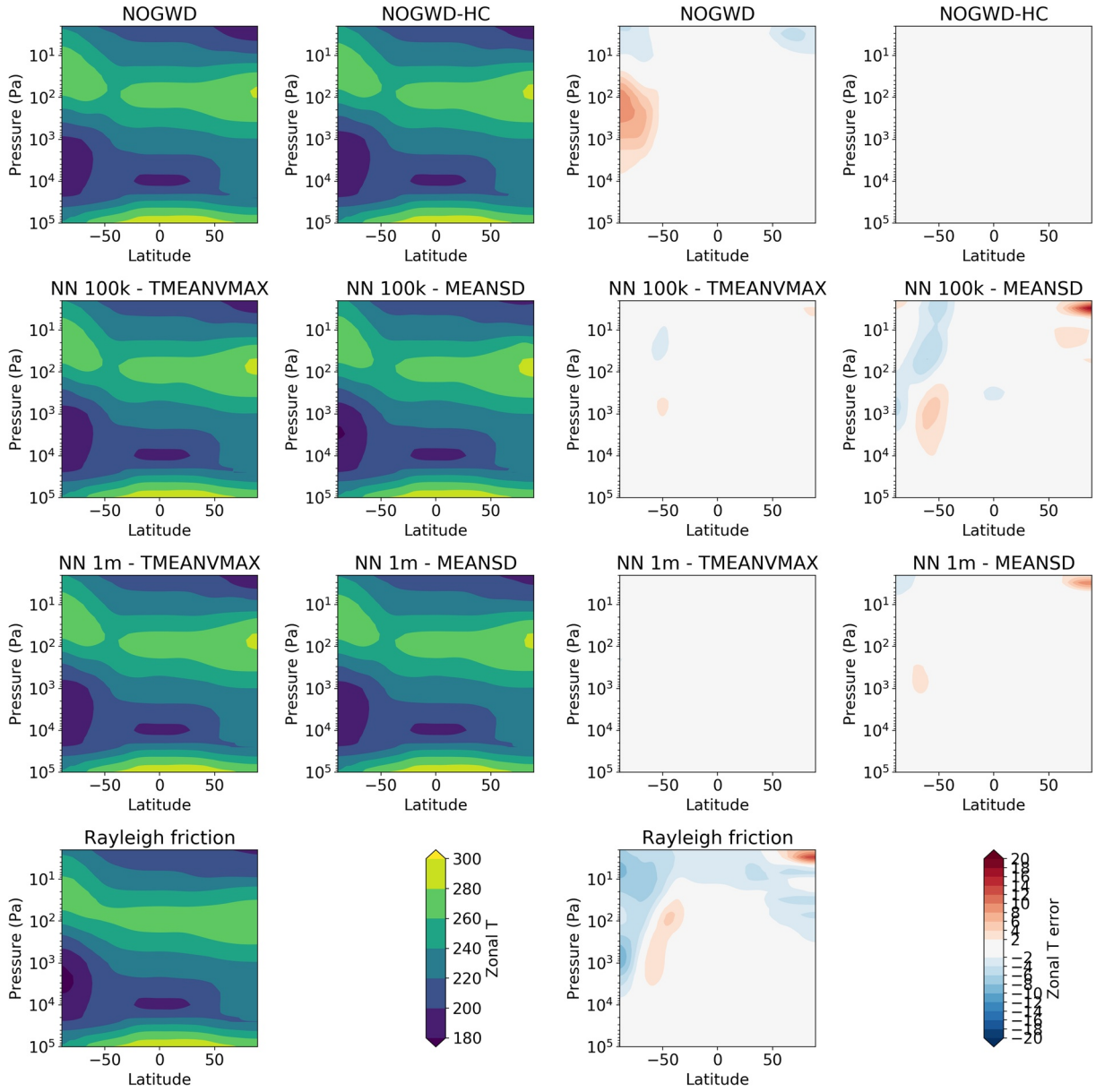

Figure 9. June-July-August average temperature profiles. Left plots show the temperature profiles and right plots show the deviations from the NOGWD high-complexity result. Each scheme produces very similar temperature profile to our reference simulation. The MEANSD models have a small warm bias over the Arctic.

appears to be a poor choice for our application. When testing with fully convolutional networks, we find that offline errors are dramatically higher than those for our fully connected networks, irrespective of the number of filters or convolutional layers used.

An alternate choice is to use locally connected layers. These have the same stencil shape as convolution layers, but the weights are trained individually for each vertical layer. It is therefore possible to encode a finite difference vertical derivative on our vertical grid within this architecture. Despite this, our testing finds that models comprising entirely locally connected layers still fail to match the performance of our fully connected networks, with testing errors more than 2 times larger than our best 100k DOF fully connected networks.

Examining the algorithm of the existing NOGWD scheme, we can understand why purely local approaches struggle to achieve good predictive performance. In the existing scheme, while most of the calculations are

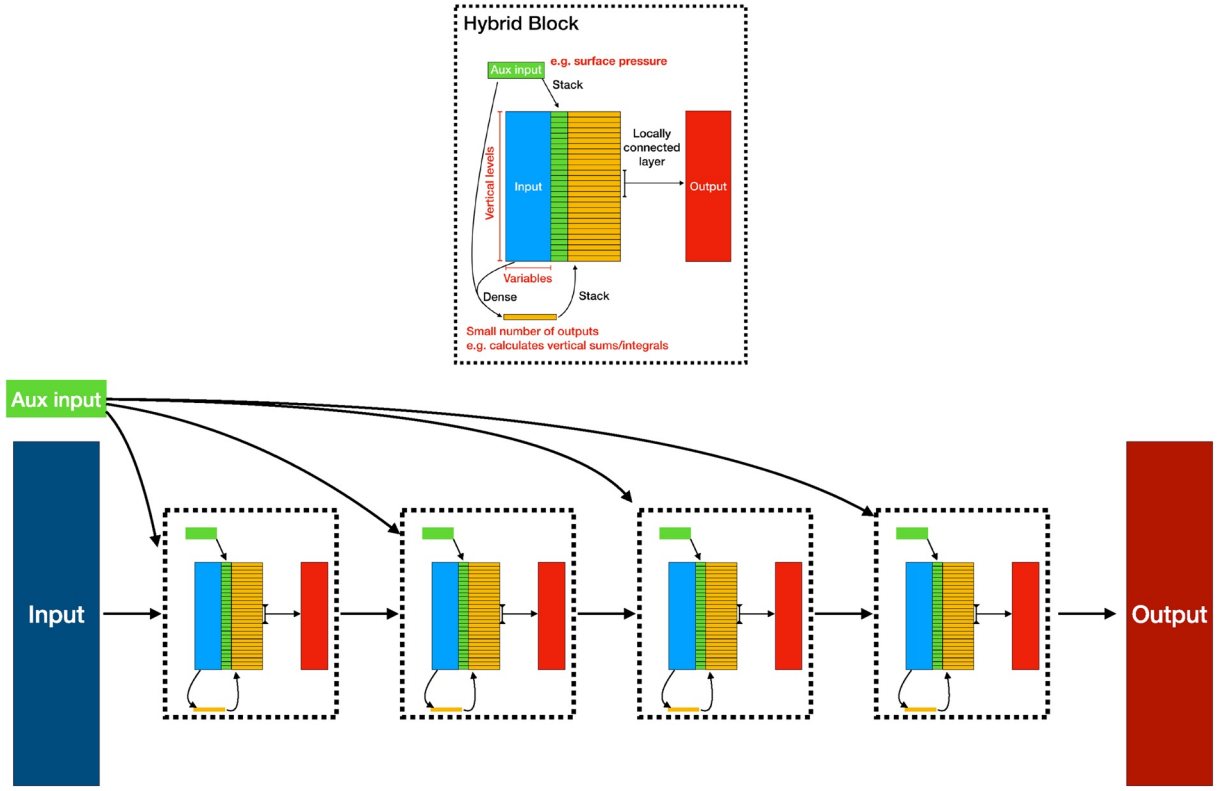

Figure 10. Schematic of our hybrid convolution deep network. Top: our hybrid block. Bottom: An example model layout consisting of four blocks. The Input block to the hybrid networks are the profiles of winds and temperature. The Aux input refers to the surface pressure and geopotential, which are injected to each vertical layer of each block. Arrows indicate layers, e.g., a dense (fully connected) layer taking the inputs and aux inputs. Stack indicates the repeating of an object to enable concatenation with data that has vertical structure (e.g., wind profiles.).

local in vertical space, there are several points where nonlocal calculations take place, for example when velocities relative to that of the launch level wind velocities are calculated. With a purely local method, information can only propagate through the network based upon the filter width at each layer, so with a width of five, a minimum of 12 layers would be required for information to be able to propagate from the lowest to highest vertical layer.

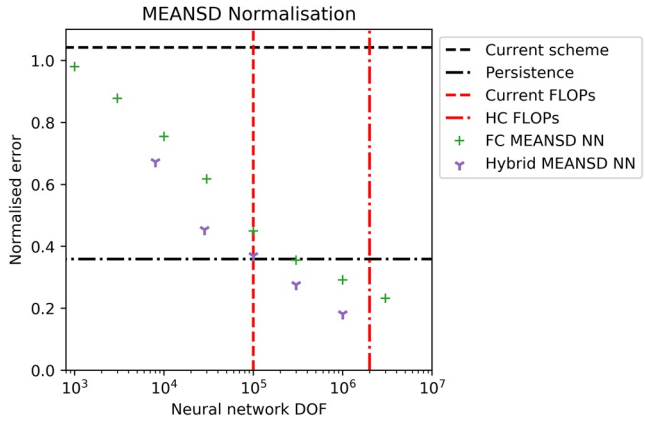

Figure 11. Offline results on the test data set using the MEANSD normalization method. In addition to the results presented in Figure 2, we plot our best performing hybrid networks as outlined in Figure 10. Consistently, the hybrid network outperforms the fully connected networks, with approximately three-times reduction in the required degree of freedom for an equivalent test error.

With the above in mind, we design a hybrid approach, utilizing both fully connected and locally connected layers, a schematic of which is plotted in Figure 10. Each block of this network comprise a small number of dense connections, which can combine both the inputs to that layer and the auxiliary inputs, which in this context are surface pressure and surface geopotential. The output of these dense layers are then stacked vertically, alongside auxiliary inputs. All of these features are then passed through a set of locally connected filters. This design seeks to predominantly leverage the local nature of the physical parameterization scheme while also enabling information to propagate throughout the domain within a single block. In Figure 11, we replot the results for the fully connected networks using MEANSD normalization and add the results of our hybrid networks. To explore the hyper-parameters associated with our hybrid networks (e.g., the number of blocks or number of dense outputs), we use the HyperOpt tool (Bergstra et al., 2013). For all DOF values, our hybrid networks significantly outperform their fully connected equivalents, producing comparable results to a fully connected network with three-times the DOF. There is certainly room for further improvements in searching

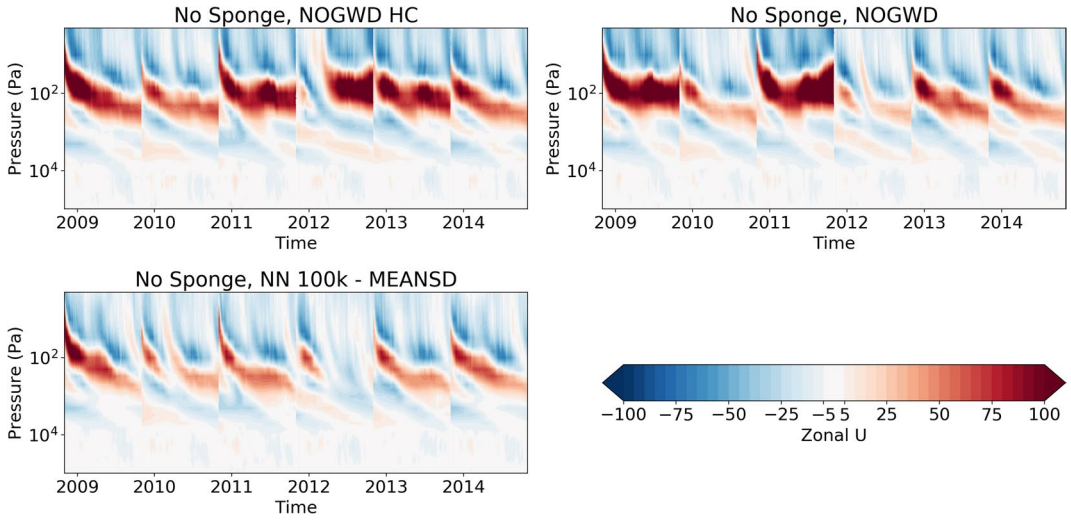

Figure 12. Average zonal-mean zonal jet between latitudes −5 to 5 for six consecutive yearlong integrations at TL159 resolution (~125 km). Experiments match equivalent names from Figure 6 except that the numerical sponge at the top of the atmosphere has been removed. Wind speeds in the stratosphere are significantly increased, yet the Quasi-Biennial Oscillation phase and amplitude are captured by the NN.

this and other network architecture spaces. Due to the increased network architecture complexity, we have not yet implemented these networks within our Fortran approach. This network architecture could prove to be a useful approach across the spectrum of physical parameterization schemes.

5.5. Generalizability to Model Resolution and Version Changes

If a body of work is undertaken to produce emulators of physical parameterization schemes within an operational modeling system, it is important to understand possible sensitivities to changes in either the model resolution or model version. If any changes of these hyperparameters necessitated the learning of new NNs, and hence generation of new training data sets, then this could quickly become a very large infrastructure project to maintain. Vertical resolution is an intrinsic part of each parameterization scheme, and so changes to vertical resolution will require generation of new schemes. However, for horizontal resolution, in the context of the NOGWD scheme, we found no sensitivity to changing the spatial resolution between 25 and 125 km. Our NNs are able to generalize across the subtle changes in atmospheric profiles seen across these resolutions. We tested this both offline and online, e.g., our results for long-range forecasting are simulated at a lower spatial resolution than the training data set. For model cycle upgrades, we tested performance across three consecutive cycles of IFS (45r1, 46r1 & 47r1) and found no noticeable degradation in performance. In Figure 12, we show the evolution of the equatorial zonal-mean zonal jet with the sponge layer removed. Despite no training on a data set with the sponge removed, our MEANSD NN produces a stable atmospheric structure. This is despite a significant increase in stratospheric wind speeds (note the change in contour levels compared with Figure 6). The phase and amplitude of the QBO is comparable with our simulation with the HC NOGWD scheme with no sponge layer. It is more evident that our NNs show reasonable robustness to model changes. However, we cannot rule out degradation in scenarios where very significant model changes are made.

5.6. Orographic Gravity Wave Drag

Armed with the successes of our emulation of NOGWD, we carried out an equivalent task to learn the orographic gravity wave drag (OGWD) scheme within IFS. We used the same methodology for data generation, same normalization methods and same network architecture choices. Orographic gravity wave drag shares the same inputs as its nonorographic counterpart, with the addition of four parameters describing the unresolved orography. The existing OGWD scheme is active only in locations with significant unresolved orography and

when the atmospheric profile is susceptible to the waves generated by that orography. The result of this is a scheme that is inactive for most of the globe and is often inactive even in locations with significant unresolved orography.

Unfortunately, our attempts to learn this OGWD scheme with fully connected NNs were unsuccessful. Increasing the degrees of freedom did not reduce the testing error, unlike our results for the NOGWD scheme. Our NNs struggled to derive this nuanced, yet computationally cheap (~20,000 FLOPs per column), interaction between the orography and atmospheric profile. The most common result of training was that the networks predicted zero velocity tendency for almost all input columns. In online testing, our model performed similarly well to a forecast without an OGWD scheme. In attempting to overcome these problems, we tested many approaches. Learning a single grid-point, that is, generating a network, which could predict the tendencies for only a single point in space, was possible but is not a scalable solution. The existing OGWD scheme is only utilized when subgrid orographic variability exceeds a threshold, we followed this example and trained only on locations where this threshold was met. Additionally, we tested methods to change the balance of training data to exclude a proportion of our training data where the scheme was inactive, but this often resulted in a dramatic increase in false positives (OGWD activity where the current scheme is inactive).

From this experience, we summarize that training emulators of physical parameterization schemes is a nontrivial task, not guaranteed to succeed even with large amounts of available data. Our outlook is that most parameterization schemes share more in common with the NOGWD scheme and therefore will be possible to learn. However, this remains to be tested.

6. Conclusions

In this work, we successfully emulate the nonorographic gravity wave drag scheme from the operational IFS forecasting model. Despite training offline, we are able to produce an emulator, which can run stably when coupled to the IFS for seasonal timescales. The most exacting test of our emulators, reproducing the descent of the QBO, was successfully achieved by our NNs. We have tested two different normalization approaches that, interestingly, showed similar performance across our tests. Broadly, our networks have similar computational cost to the existing scheme when coupled to the IFS on a CPU-based architecture. However, both the existing scheme and our NNs are heavily limited by data movement when run within the IFS. The major advantage of our networks is the ease of portability to heterogeneous architectures, where NN emulators of parameterization schemes could be offloaded to GPUs. This approach will have the greatest benefit if many parameterization schemes can be emulated with NNs. However, as discussed above for the orographic gravity wave drag, other schemes might prove more challenging to emulate as accurately as the NOGWD scheme. Beyond forecasting, there are other values of building parameterization emulation. The 4D-var data assimilation approach uses a tangent-linear and adjoint to the forecasting model, constructed of tangent-linear and adjoint versions of each kernel. This is a challenging process typically derived by hand. However, with an accurate NN emulator, the tangent linear and adjoint versions can be trivially derived due to the simple nature of a NNs algorithm. In our sister-paper Hatfield et al. (2021), we will test how these tangent linear and adjoint versions perform within a data-assimilation framework. Further exploration of the performance of these emulators would be required before one could be considered for operational forecasting. This would include high-resolution testing (9 km), seasonal forecast tests, ensemble forecasts and using the emulator within the nonlinear integrations within 4D-var.

Acknowledgments
M. Chantry and T. Palmer were supported by grant DCR00481 from the Office of Naval Research Global. ECMWF provided computing resources resources, including the European Weather Cloud, which is an ECMWF and EUMETSAT project. Thanks to Ioan Hadade for his technical support. P. Dueben gratefully acknowledges funding from the Royal Society for his University Research Fellowship as well as the AI4Copernicus and MAELSTROM projects that are funded under Horizon 2020 and the European High-Performance Computing Joint Undertaking (JU; grant agreement No 101016798 and 955513). The JU receives support from the European Union's Horizon 2020 research and innovation program and United Kingdom, Germany, Italy, Luxembourg, Switzerland, Norway. S. Hatfield and P. Dueben have also received funding from the ESIWACE Horizon 2020 project (grant agreement No 823988). T. Palmer also received funding from an ERC Advanced Grant, ITHACA 741112.

Data Availability Statement

The data used to build the machine learning emulators has been archived and is freely accessible at https://doi.org/10.5281/zenodo.4740758.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., et al. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems*. Retrieved from https://www.tensorflow.org/

Bauer, P., Quintino, T., Wedi, N., Bonanni, A., Chrust, M., Deconinck, W., et al. (2020). *The ecmwf scalability program: Progress and plans*. European Centre for Medium Range Weather Forecasts.

Bergstra, J., Yamins, D., & Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International conference on machine learning* (pp. 115–123).

Beucler, T., Pritchard, M., Rasp, S., Ott, J., Baldi, P., & Gentine, P. (2019). *Enforcing analytic constraints in neural-networks emulating physical systems*. arXiv preprint arXiv:1909.00912.

Blackford, L. S., Petitet, A., Pozo, R., Remington, K., Whaley, R. C., Demmel, J., et al. (2002). An updated set of basic linear algebra subprograms (blas). *ACM Transactions on Mathematical Software*, *28*(2), 135–151. https://doi.org/10.1145/567806.567807

Brenowitz, N. D., & Bretherton, C. S. (2018). Prognostic validation of a neural network unified physics parameterization. *Geophysical Research Letters*, *45*(12), 6289–6298. https://doi.org/10.1029/2018gl078510

Brenowitz, N. D., & Bretherton, C. S. (2019). Spatially extended tests of a neural network parametrization trained by coarse-graining. *Journal of Advances in Modelling Earth Systems*, *11*(8), 2728–2744. https://doi.org/10.1029/2019ms001711

Brenowitz, N. D., Henn, B., McGibbon, J., Clark, S. K., Kwa, A., Perkins, W. A., et al. (2020). *Machine learning climate model dynamics: Offline versus online performance*. arXiv preprint arXiv:2011.03081.

Chantry, M., Christensen, H., Düben, P., & Palmer, T. (2021). Opportunities and challenges for machine learning in weather and climate modeling: Hard, medium and soft AI. *Philosophical Transactions of the Royal Society of London - A*, *379*, 20200083. https://doi.org/10.1098/rsta.2020.0083

Chevallier, F., Chéruy, F., Scott, N., & Chédin, A. (1998). A neural network approach for a fast and accurate computation of a longwave radiative budget. *Journal of Applied Meteorology*, *37*(11), 1385–1397. https://doi.org/10.1175/1520-0450(1998)037<1385:annafa>2.0.co;2

Dijkstra, H., Petersik, P., Hernandez-Garcia, E., & Lopez, C. (2019). The application of machine learning techniques to improve el nino prediction skill. *Frontiers in Physics*, *7*, 153. https://doi.org/10.3389/fphy.2019.00153

Dunkerton, T. J. (1997). The role of gravity waves in the quasi-biennial oscillation. *Journal of Geophysical Research*, *102*(D22), 26053–26076. https://doi.org/10.1029/96jd02999

ECMWF. (2018). *Ifs documentation (cy45r1)*. Retrieved from https://www.ecmwf.int/en/publications/ifs-documentation

Ern, M., Preusse, P., Alexander, M. J., & Warner, C. D. (2004). Absolute values of gravity wave momentum flux derived from satellite data. *Journal of Geophysical Research*, *109*(D20). https://doi.org/10.1029/2004jd004752

Garcia, R. R., & Boville, B. A. (1994). "Downward control" of the mean meridional circulation and temperature distribution of the polar winter stratosphere. *Journal of the Atmospheric Sciences*, *51*(15), 2238–2245. https://doi.org/10.1175/1520-0469(1994)051<2238:cotmmc>2.0.co;2

Gardner, C. S., Miller, M. S., & Liu, C.-H. (1989). Rayleigh Lidar observations of gravity wave activity in the upper stratosphere at Urbana, Illinois. *Journal of the Atmospheric Sciences*, *46*(12), 1838–1854. https://doi.org/10.1175/1520-0469(1989)046<1838:rloogw>2.0.co;2

Geer, A. J. (2016). Significance of changes in medium-range forecast scores. *Tellus A: Dynamic Meteorology and Oceanography*, *68*(1), 30229. https://doi.org/10.3402/tellusa.v68.30229

Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., & Yacalis, G. (2018). Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*, *45*(11), 5742–5751. https://doi.org/10.1029/2018gl078202

Gettelman, A., Gagne, D. J., Chen, C.-C., Christensen, M., Lebo, Z., Morrison, H., & Gantos, G. (2020). Machine learning the warm rain process. *Journal of Advances in Modeling Earth Systems*, e2020MS002268. https://doi.org/10.1029/2020MS002268

Hatfield, S., Chantry, M., Düben, P., & Palmer, T. (2019). Accelerating high-resolution weather models with deep-learning hardware. In *Proceedings of the platform for advanced scientific computing conference* (pp. 1–11). https://doi.org/10.1145/3324989.3325711

Hatfield, S., Düben, P., Lopez, P., Geer, A., Chantry, M., & Palmer, T. (2021). *Neural networks as the building blocks for tangent-linear and adjoint models*. arXiv preprint arXiv.

Krasnopolsky, V. (1997). A neural network forward model for direct assimilation of ssm/i brightness temperatures into atmospheric models. *Research activities in atmospheric and oceanic modeling*.

Morcrette, J.-J., Mozdzynski, G., & Leutbecher, M. (2008). A reduced radiation grid for the ecmwf integrated forecasting system. *Monthly Weather Review*, *136*(12), 4760–4772. https://doi.org/10.1175/2008mwr2590.1

NVIDIA. (2017). *NVIDIA tesla V100 GPU architecture (tech. Rep.)*. Retrieved from http://www.nvidia.com/content/gated-pdfs/Volta-Architecture-Whitepaper-v1.1.pdf

O'Gorman, P. A., & Dwyer, J. G. (2018). Using machine learning to parameterize moist convection: Potential for modeling of climate, climate change, and extreme events. *Journal of Advances in Modeling Earth Systems*, *10*(10), 2548–2563.

Orr, A., Bechtold, P., Scinocca, J., Ern, M., & Janiskova, M. (2010). Improved middle atmosphere climate and forecasts in the ecmwf model through a nonorographic gravity wave drag parameterization. *Journal of Climate*, *23*(22), 5905–5926. https://doi.org/10.1175/2010jcli3490.1

Ott, J., Pritchard, M., Best, N., Linstead, E., Curcic, M., & Baldi, P. (2020). *A fortran-keras deep learning bridge for scientific computing*. arXiv preprint arXiv:2004.10652.

Palmer, T. (2020). *A vision for numerical weather prediction in 2030*. arXiv preprint arXiv:2007.04830.

Polichtchouk, I., Shepherd, T. G., & Byrne, N. J. (2018). Impact of parametrized nonorographic gravity wave drag on stratosphere-troposphere coupling in the northern and southern hemispheres. *Geophysical Research Letters*, *45*(16), 8612–8618. https://doi.org/10.1029/2018gl078981

Polichtchouk, I., Shepherd, T. G., Hogan, R., & Bechtold, P. (2018). Sensitivity of the brewer–dobson circulation and polar vortex variability to parameterized nonorographic gravity wave drag in a high-resolution atmospheric model. *Journal of the Atmospheric Sciences*, *75*(5), 1525–1543. https://doi.org/10.1175/jas-d-17-0304.1

Ramachandran, P., Zoph, B., & Le, Q. V. (2017). *Swish: A self-gated activation function*. arXiv preprint arXiv:1710.05941, 7.

Rasp, S., Düben, P. D., Scher, S., Weyn, J. A., Mouatadid, S., & Thuerey, N. (2020). *Weatherbench: A benchmark dataset for data-driven weather forecasting*. arXiv preprint arXiv:2002.00469.

Rasp, S., Pritchard, M. S., & Gentine, P. (2018). Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences*, *115*(39), 9684–9689. https://doi.org/10.1073/pnas.1810286115

Rasp, S., & Thuerey, N. (2021). Data-driven medium-range weather prediction with a resnet pretrained on climate simulations: A new model for weatherbench. *Journal of Advances in Modeling Earth Systems*, e2020MS002405. https://doi.org/10.1029/2020MS002405

Saffin, L., Hatfield, S., Düben, P., & Palmer, T. (2020). Reduced-precision parametrization: Lessons from an intermediate-complexity atmospheric model. *Quarterly Journal of the Royal Meteorological Society*, *146*(729), 1590–1607. https://doi.org/10.1002/qj.3754

Scinocca, J. F. (2003). An accurate spectral nonorographic gravity wave drag parameterization for general circulation models. *Journal of the Atmospheric Sciences*, *60*(4), 667–682. https://doi.org/10.1175/1520-0469(2003)060<0667:aasngw>2.0.co;2

Sønderby, C. K., Espeholt, L., Heek, J., Dehghani, M., Oliver, A., Salimans, T., & Kalchbrenner, N. (2020). *Metnet: A neural weather model for precipitation forecasting*. arXiv preprint arXiv:2003.12140.

Ukkonen, P., Pincus, R., Hogan, R. J., Pagh Nielsen, K., & Kaas, E. (2020). Accelerating radiation computations for dynamical models with targeted machine learning and code optimization. *Journal of Advances in Modeling Earth Systems*, *12*(12), e2020MS002226. https://doi.org/10.1029/2020ms002226

Váňa, F., Düben, P., Lang, S., Palmer, T., Leutbecher, M., Salmond, D., & Carver, G. (2017). Single precision in weather forecasting models: An evaluation with the ifs. *Monthly Weather Review*, *145*(2), 495–502. https://doi.org/10.1175/MWR-D-16-0228.1

Veerman, M., & Pincus, R. (2021). Predicting atmospheric optical properties for radiative transfer computations using neural networks. *Philosophical Transactions of the Royal Society of London - A*, *379*. https://doi.org/10.1098/rsta.2020.0095

Warner, C., & McIntyre, M. (2001). An ultrasimple spectral parameterization for nonorographic gravity waves. *Journal of the Atmospheric Sciences*, *58*(14), 1837–1857. https://doi.org/10.1175/1520-0469(2001)058<1837:auspfn>2.0.co;2

Weyn, J. A., Durran, D. R., & Caruana, R. (2019). Can machines learn to predict weather? using deep learning to predict gridded 500-hpa geopotential height from historical weather data. *Journal of Advances in Modeling Earth Systems*, *11*(8), 2680–2693. https://doi.org/10.1029/2019ms001705

Yuval, J., & O'Gorman, P. A. (2020). Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature Communications*, *11*(1), 1–10. https://doi.org/10.1038/s41467-020-17142-3

Yuval, J., O'Gorman, P. A., & Hill, C. N. (2020). 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*, e2020GL091363. https://doi.org/10.1029/2020GL091363