Regression forest approaches to gravity wave parameterization for climate projection

David S. Connelly[1], **Edwin P. Gerber**[1]

[1]Center for Atmosphere Ocean Science, Courant Institute, New York University

Key Points:

- Two kinds of regression forest emulate a gravity wave parameterization offline and online, with boosted forests outperforming random forests
- Relative to a neural network benchmark, the boosted forest exhibits similar online skill and ability to generalize to new climates
- Feature importance analysis informs a retraining procedure to improve online behavior of data-driven parameterizations

Corresponding author: David S. Connelly, dsconnelly@nyu.edu

Abstract

We train random and boosted forests, two machine learning architectures based on regression trees, to emulate a physics-based parameterization of atmospheric gravity wave momentum transport. We compare the forests to a neural network benchmark, evaluating both offline errors and online performance when coupled to an atmospheric model under the present day climate and in 800 and 1200 ppm CO_2 global warming scenarios. Offline, the boosted forest exhibits similar skill to the neural network, while the random forest scores significantly lower. Both forest models couple stably to the atmospheric model, and control climate integrations with the boosted forest exhibit lower biases than those with the neural network. Integrations with all three data-driven emulators successfully capture the Quasi-Biennial Oscillation (QBO) and sudden stratospheric warmings, key modes of stratospheric variability, with the boosted forest more accurate than the random forest in replicating their statistics across our range of carbon dioxide perturbations. The boosted forest and neural network capture the sign of the QBO period response to increased CO_2, though both struggle with the magnitude of this response under the more extreme 1200 ppm scenario. To investigate the connection between performance in the control climate and the ability to generalize, we use techniques from interpretable machine learning to understand how the data-driven methods use physical information. We leverage this understanding to develop a retraining procedure that improves the coupled performance of the boosted forest in the control climate and under the 800 ppm CO_2 scenario.

Plain Language Summary

Parameterizations are reduced-complexity models that estimate the effects of physical processes smaller than what can be resolved by the grid of a weather or climate model. While necessary for realistic simulations, they are a source of uncertainty in climate projections. Recently, machine learning has been used to augment or replace conventional parameterizations of atmospheric gravity waves, a type of motion by which disturbances near the Earth's surface can affect the wind higher up. We compare several machine learning approaches to the gravity wave parameterization problem. In particular, we test neural networks against random and boosted forests, which are built around flowchart-like models called regression trees. We find that boosted forests, though not widely used for climate model parameterization, are especially successful, scoring as well as or better than neural networks on various performance metrics. We then provide proof-of-concept of a novel method to retrain the boosted forest so that it uses its input data more in line with the physics of the system, and show that this technique improves the forest's behavior when used together with an atmospheric model.

1 Introduction

Momentum transport by atmospheric gravity waves is a key driver of several features of the large-scale circulation, such as the Quasi-Biennial Oscillation (QBO) in the tropical stratosphere (Fritts & Alexander, 2003), and influences others, including the tropospheric jet structure (Palmer et al., 1986). Waves transporting appreciable momentum can have horizontal and vertical wavelengths as small as 100 m. Because of the small scales at play, atmospheric models must rely on parameterizations of gravity wave momentum flux to represent these climate features (Anstey et al., 2022).

Gravity wave parameterizations (GWPs), however, must make simplifying assumptions about wave propagation to remain computationally feasible. The dynamics may be derived from the linearized equations of motion, for example, assuming planar waves that do not interact with each other. In addition, schemes are in general computationally constrained to operate on single atmospheric columns, so that they cannot model lateral propagation. Due to these limitations, GWPs are significant sources of model uncertainty. For example, parameterizations tuned to match present-day QBO statistics can produce different projections of QBO behavior in warmer climate simulations (Richter et al., 2022).

Recently, machine learning has been used to build new parameterizations of subgrid-scale phenomena including ocean eddy momentum forcings (Bolton & Zanna, 2019), radiative and microphysical tendencies in atmospheric moisture variables (Yuval & O'Gorman, 2020), and gravity wave momentum transport (Chantry et al., 2021; Espinosa et al., 2022). Such approaches attempt to learn a mapping from resolved-scale variables to subgrid quantities from data. Machine learning can be employed both to learn these relationships from increasingly available high-resolution data (Brenowitz & Bretherton, 2018; O'Gorman & Dwyer, 2018; Bolton & Zanna, 2019; Yuval & O'Gorman, 2020) and to accelerate existing parameterizations by replacing them with emulators (Chevallier et al., 1998; Chantry et al., 2021).

Building on work by Espinosa et al. (2022), we explore the use of machine learning models to emulate the behavior of an existing, physics-based reference parameterization, the Alexander and Dunkerton (1999) scheme (hereafter AD99) as it is implemented in the Model of an idealized Moist Atmosphere (MiMA), an intermediate-complexity climate model (Garfinkel et al., 2020). Emulation serves as an intermediate step towards training fully data-driven GWPs using a combination of observations of gravity waves and high-resolution simulations capable of resolving them.

In particular, emulation allows us to address challenges inherent to the design of data-driven GWPs — e.g. comparing machine learning architectures, achieving stable coupling with atmospheric models, and testing the ability of schemes to generalize — separately from issues of data availability and processing that arise when working with more realistic data sources. In the emulation problem, data is cheap to generate by running the reference parameterization. Moreover, AD99 produces a reference climate when coupled to MiMA, which can be perturbed by increasing the carbon dioxide concentration. As a result, there are straightforward performance metrics when coupling data-driven emulators and evaluating their performance in a warmer climate.

Espinosa et al. (2022) used neural networks to emulate AD99, finding that they could reproduce both the QBO and its AD99-predicted response to an increase in CO_2. The principal contributions of this work are the application of tree-based machine learning architectures to the same problem, the comparison of their offline and online performance with that of neural networks, and the use of offline feature importance analyses to develop a retraining procedure that improves online behavior. Section 2 introduces the parameterization to be emulated and characterizes the datasets used. Section 3 describes the machine learning architectures along with the interpretability techniques used to analyze them. Section 4 presents the performance of the emulators, both offline and in coupled integrations under a series of CO_2 perturbations. Section 5 analyzes the emulators' preferential use of particular input features to explain their offline and online behavior using so-called interpretable machine learning techniques. Section 6 reviews the main results and discusses future research directions.

2 Models and data

We begin with descriptions of the atmospheric model MiMA and the reference parameterization AD99. These are relevant to our work in that they are used to generate the training and offline test datasets, the main features of which are outlined in Section 2.2, and in that they are necessary for the coupled integration experiments summarized in Section 2.3.

2.1 MiMA and the AD99 parameterization

The Model of an idealized Moist Atmosphere (MiMA) is an atmospheric circulation model coupled with a purely thermodynamic, or slab, ocean model. The atmosphere component includes interactive moisture, full radiation with the Rapid Radiative Transfer Model scheme (Mlawer et al., 1997; Iacono et al., 2000), and Betts-Miller convection (Betts, 1986; Betts & Miller, 1986). The carbon dioxide concentration is set globally at 390 ppm, though we change this value in experiments presented later in this work. Ozone is distributed according to the monthly- and zonal-mean climatologies used in CMIP6 pre-industrial simulations (Checa-Garcia et al., 2018; Checa-Garcia, 2018). See Jucker and Gerber (2017) and Garfinkel et al. (2020) for a complete description of MiMA.

The AD99 gravity wave parameterization takes in resolved-scale variables from a single column and time step of an atmospheric model and returns the velocity tendencies from parameterized gravity waves at each vertical level. The scheme computes the forcing by propagating a collection of independent, monochromatic waves of varying phase speed from the tropopause. Each wave has an associated momentum flux determined by a parameterized source spectrum. That momentum flux spectrum, Gaussian in magnitude with width $35\,\mathrm{m\,s^{-1}}$, is positive for phase speeds greater than the source level velocity and negative elsewhere. The amplitude peak is at phase speed zero in the extratropics, where gravity waves are assumed to be mainly orographic, and at phase speed equal to the source level velocity in the tropics, where waves are assumed to be generated by convection. The source spectrum and launch level vary only with latitude, a key simplification of the scheme.

Waves are propagated upwards until they reach a level where one of several breaking criteria is met, where they deposit their momentum. Breaking occurs either when the density is sufficiently low to permit overturning or at a *critical level*, where the mean flow speed is very close to the wave's phase speed and the vertical group velocity is accordingly very small. Waves that reach the model top without breaking have their momentum flux evenly distributed over the highest three vertical levels. We refer the reader to Alexander and Dunkerton (1999) for a detailed discussion of the parameterization.

2.2 Training inputs and outputs

Machine learning parameterizations use data to learn a mapping from resolved scale variables to subgrid quantities — or, in this case, to learn the mapping encoded in AD99. To generate training data, we first run MiMA for 20 years, using AD99 as the gravity wave parameterization, to ensure the climate system has reached statistical equilibrium. We then integrate for a further 60 years, outputting data every six hours. At T42 resolution, this control run yields just under 12 million vertical profiles per year of model time. We sample 10 million profiles from the first 4 years of this 60-year run to use as training data, and another 10 million profiles from years 5-8 to use as an offline test set.

We explore the use of various resolved-scale variables as *input features*, the data passed as input to our machine learning emulators. These may include vertical profiles of wind u or v, temperature T, or buoyancy frequency N. We provide some models with a profile of the differences between winds at adjacent levels; this feature has units of $\mathrm{m\,s^{-1}}$ and, for brevity, we call it the *shear*. Every emulator takes in the latitude ϑ and surface pressure p_s of each training sample, except when we explicitly test the effects of withholding these features. We expect latitude to be an important feature because, in AD99, the peak of the gravity wave momentum flux source spectrum transitions from phase speed zero in the extratropics to phase speed moving with the tropopause flow in the tropics. The source level also varies with latitude to follow the tropopause.

The *targets*, the outputs associated with particular inputs that the machine learning model tries to predict, are the AD99 gravity wave accelerations at each of the 40 vertical levels in MiMA. Because AD99 handles zonal and meridional accelerations identically, our emulators are trained on both zonal and meridional data. When predicting zonal acceleration, they take u as an input feature, and likewise they take v when predicting meridional acceleration. Both the training and test datasets are evenly split between zonal and meridional samples.

The left panel of Figure 1 shows example zonal wind and temperature profiles, and the black curve in the right panel is the resulting gravity wave acceleration profile as parameterized by AD99. Note that the target accelerations vary by two to three orders of magnitude over the vertical column. We use the mean and variance of the accelerations at each vertical level, computed from the training samples, to standardize the targets to zero mean and unit variance. As a result, the emulators weight errors at each level equally, as

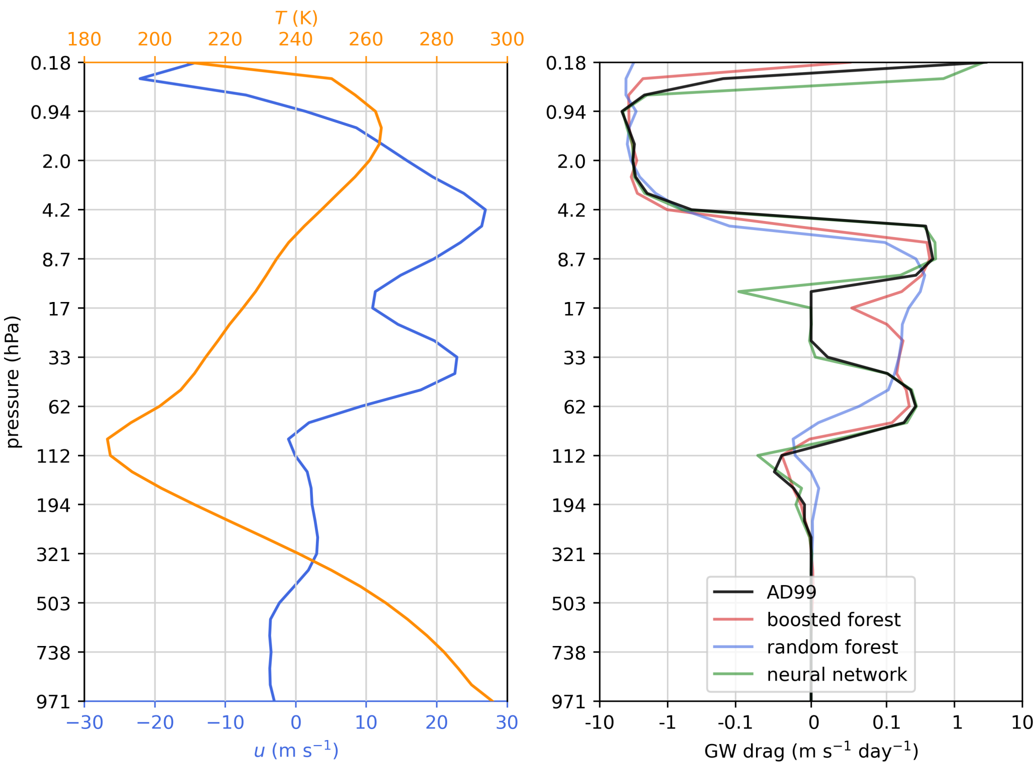

Figure 1. Example wind and temperature profiles (left) and the resulting parameterized accelerations predicted by AD99 and several emulators (right).

opposed to prioritizing performance near the model top at the expense of the lower levels. We rescale emulator predictions to their original units before using them to compute offline performance metrics like R^2 or passing them to MiMA in coupled runs.

2.3 Coupled integrations

After training and offline error analysis, we couple each trained emulator to MiMA, initialized with the final state of the spinup used to initialize the AD99 control run, and integrate for 60 years. The configuration is identical to that of the control run except that the emulators are used in place of AD99.

To assess the emulators' response to climate perturbations, we also run MiMA with the CO_2 concentration set at 800 ppm and at 1200 ppm. As with the 390 ppm CO_2 control runs, for each concentration value, we first integrate with AD99 for twenty years of spinup followed by sixty further years. We then couple each emulator (without retraining) and integrate for sixty years starting from the final state of the same spinup period.

For all coupled integrations, we retain only the last 56 years for analysis. Section 4 discusses the output of these integrations.

3 Machine learning architectures and interpretation

In this section, we first review tree- and forest-based machine learning architectures, distinguishing between random and boosted forests, and specify the neural network benchmark. Random forests have been used in atmospheric modeling before (O'Gorman & Dwyer, 2018); however, we believe this work is the first use in this context of boosting, which is well-known in the broader machine learning literature. Next, we summarize the interpretability metric we use to analyze the behavior of our emulators. Finally, we indicate the existing libraries we used and briefly describe new software written for this study.

3.1 Regression trees and forests

The way humans solve problems is often well-approximated by asking a series of yes-no questions about the available data and predicting accordingly. For example, if asked to guess the price of a house, one might first ask if it is located in New York, then whether it has more than two bedrooms, and so on, perhaps selecting later questions based on the answers to earlier ones.

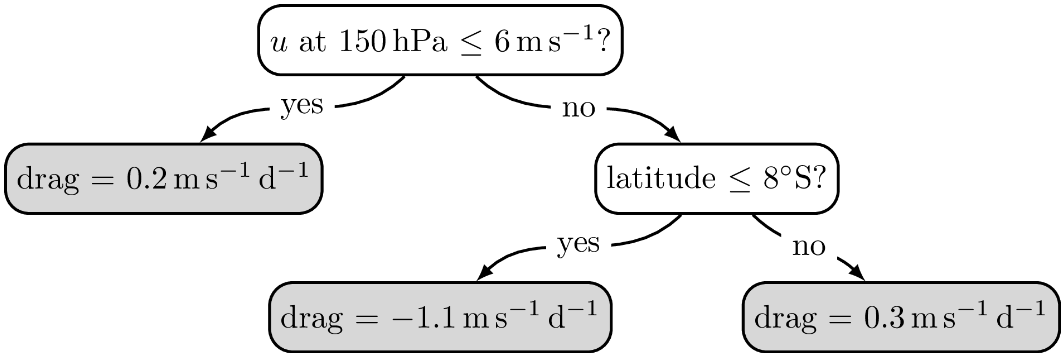

Figure 2. A simple regression tree. Leaf nodes are shaded. Note that the predictions of this example tree are scalars (e.g. accelerations at a particular level), while the trees used in this work yield vector-valued predictions (acceleration profiles).

Regression trees (Breiman et al., 1984) are machine learning models that attempt to make predictions in an analogous manner. Once trained, they make predictions by traversing a binary tree according to the given input features. The traversal repeatedly proceeds to one of the current node's two children based on whether a specified input feature exceeds a set threshold. The tree returns as its prediction the value stored at whichever leaf node the traversal terminates at. Figure 2 shows a simple schematic of a regression tree with features relevant to the gravity wave parameterization problem. See Text S1 for a detailed explanation of how regression trees are constructed from training data.

If their depth is unlimited, regression trees can achieve zero error on the training data. However, such trees typically generalize very poorly to unseen samples because they have learned the noise in the dataset. Instead, a lower-variance model can be constructed from an ensemble, or *forest*, of regression trees of fixed depth. In this study we consider two kinds of ensembles: random forests and boosted forests.

A random forest (Breiman, 2001) is a collection of regression trees, each of which is trained independently on a bootstrapped subsample of the training dataset. The prediction of the forest is simply the mean of the predictions of each constituent tree. Figure 3(a) shows this concept: individual ensemble members have high error, but their ensemble mean matches the target much more closely.

The term boosting (Schapire, 1990) encompasses a wide class of machine learning algorithms that train individual ensemble members sequentially and combine them into a more

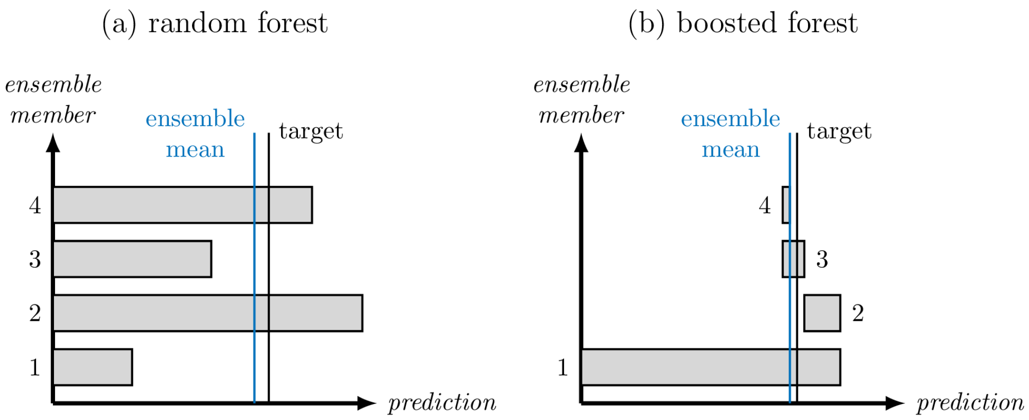

Figure 3. The types of regression forests considered in this work, shown conceptually. Each bar represents an individual tree in the ensemble, and the black "target" line is the true output associated with a particular sample. In (b), the bars for each tree are positioned relative to the sum of the preceding members. The blue line is the aggregate prediction of the ensemble — the mean in (a), and the sum in (b).

powerful model. In a boosted forest (Friedman, 2001), each tree is trained to reduce the residual errors accrued by its predecessors. Figure 3(b) illustrates how the trees in a boosted forest work to correct the under- or overshoot of the sum of the previous trees' predictions.

3.2 Forest hyperparameters

The training of both random and boosted forests is governed by several *hyperparameters*, tunable values chosen by the user. Most of these fall into one of two broad categories: those which set the size of the ensemble, and those which inject randomness into the training process. The former includes the number of trees in the forest and the maximum allowed depth of each tree. The latter includes the size of the subsampled dataset used to train each tree and the fraction of input features considered as potential splits at each node. Boosted forests also have a *learning rate*, a scalar less than unity multiplying the prediction of each constituent tree, which limits the rate at which the forest can "zero in" on the targets and helps prevent overfitting to the training data.

Table S3 summarizes the hyperparameter values used in this study. They were chosen using cross-validation: for each candidate hyperparameter set, a model is trained several times with different subsets of the training data held out, and the parameter set that minimizes the average out-of-set error is chosen. Hyperparameters were then kept constant for consistency across all experiments. We find that performance is robust to moderate changes in these hyperparameters.

3.3 WaveNet as a benchmark

We use the WaveNet neural network architecture, developed by Espinosa et al. (2022) to emulate AD99, as a benchmark against which to compare our forest emulators. The neural network features four shared fully-connected hidden layers followed by a branching structure with two independent fully-connected layers for each output pressure level. Our network has approximately 385,000 trainable parameters (the exact number depends on the input features chosen), roughly the minimum size found to be necessary by Espinosa et al. (2022) for successful coupled integrations. It is challenging to meaningfully compare parameter counts of neural networks with those of regression forests because their architectures differ so dramatically. The neural networks and regression forests we use have roughly comparable runtimes in coupled integrations, which is perhaps the more practically important metric of model complexity. Neither architecture was optimized for performance on our machine.

Our approach differs from that of Espinosa et al. (2022) in that we use mean-square error as our loss function (instead of the $\log \cosh$ loss) and we predict gravity wave drags at all 40 output levels (instead of predicting at the highest 33 levels and padding with zeros). We made these changes for the sake of simplicity and did not observe significant effects on network behavior. As with our forest emulators, we train one neural network to predict both zonal and meridional accelerations.

3.4 Feature importance and SHAP values

Because machine learning architectures can be fairly opaque, it is often difficult to assess whether a model has learned to use its input data in a physically plausible way. Moreover, even data-driven schemes that achieve low error on their training and test sets can behave unpredictably when coupled back to the atmospheric model. This suggests that parameterization design might be well-served by studying not just how highly a particular model scores, but also how it uses its data.

Feature importance metrics attempt to understand the behavior of a machine learning model by quantifying how individual features affect model predictions. The *SHAP (SHapley*

Additive exPlanation) value (Lundberg & Lee, 2017), an adaptation of game-theoretic Shapley values (Shapley, 1953), is a metric of feature importance defined for arbitrary machine learning models. Given any function φ and a sample $\boldsymbol{x}$, first consider

$$a_k(\boldsymbol{x}) = \mathop{\mathbb{E}}_{z_k}\Big[\varphi(\boldsymbol{z}) \;\Big|\; z_i = x_i \text{ for all } i \neq k\Big] \tag{1}$$

the average output of φ on inputs matching $\boldsymbol{x}$ except in the k^{th} component. The expectation in (1) is the interventional expectation (Pearl, 2000; Janzing et al., 2020) which, to avoid mistaking correlations between components for patterns in the behavior of φ, breaks the dependence of the k^{th} component on the others by averaging over the full distribution of z_k found in the training dataset. The SHAP value of feature k is then defined as $s_k(\boldsymbol{x}) \equiv \varphi(\boldsymbol{x}) - a_k(\boldsymbol{x})$, the change in φ that results when x_k is known exactly. Efficient algorithms exist for approximating $a_k(\boldsymbol{x})$, and by extension $s_k(\boldsymbol{x})$, for both regression trees (Lundberg et al., 2020) and neural networks (Shrikumar et al., 2017; Lundberg & Lee, 2017).

SHAP values are local, in the sense that they are calculated for each input sample. In Section 5 we compute dataset-averaged absolute values of the SHAP values for a global measure of feature importance. Moreover, the SHAP value as described is defined for scalar-valued functions, but the parameterizations we consider in this work have vector-valued outputs; we calculate SHAP values for each output channel separately and examine how features vary in their importance to predictions of gravity wave drag at different vertical levels.

3.5 Software implementation

The random and boosted forests we use are built using the decision tree interface in the Python library scikit-learn (Pedregosa et al., 2011), and our neural networks use PyTorch (Paszke et al., 2019). To couple our emulators with MiMA, we use Forpy (Rabel, 2020), which allows Python functions to be called from the Fortran numerical solver.

Support for multioutput regression, however, is limited in existing tree boosting libraries. For scalar problems, boosting admits an optimization known as gradient boosting, but that formulation involves Taylor expansions in the output space and so becomes impractical for multi-output target data. As such, we created Mubofo (*mu*ti-output *bo*osted *fo*rests), a tree boosting library extending Scikit-learn and based on the train-on-the-residuals perspective of boosting schematized in Figure 3 (Connelly, 2023). Mubofo also implements the vector-valued Gini importance calculations described in Text S2.

4 Offline and online evaluation

We first evaluate our emulators *offline*; that is, we examine their skill on the training and test datasets described in Section 2.2, uncoupled from the atmospheric model that generated that data. We then discuss the emulators' *online* performance — how the atmospheric model behaves when coupled to a data-driven emulator instead of to AD99, as outlined in Section 2.3. Because parameterizations are developed with the goal of enhancing atmospheric models, online performance is of greater importance than offline error. However, it is more difficult both to improve directly, because emulator training occurs offline, and to evaluate, because doing so requires computationally demanding integrations of the atmospheric model.

4.1 Offline R^2 scores

The left panel of Figure 1 shows sample tropical zonal wind and temperature profiles passed by MiMA to the gravity wave scheme, and the right panel shows the gravity wave acceleration profiles as parameterized by AD99, one of each kind of regression forest emulator, and a neural network emulator. The AD99 profile exhibits local maxima just below the two maxima in the input wind profile, reflecting the deposition of momentum just below critical levels. The boosted forest and neural network emulators reproduce the shape of this profile, including the two maxima, although the neural network appears somewhat closer to the AD99 profile overall. The random forest, on the other hand, seems to smooth out many of the features of the target profile; this is unsurprising, since random forests have an intrinsic tendency to average.

These anecdotal impressions of performance are borne out by a more global assessment of error relative to AD99. Figure 4 shows the three emulators' coefficients of determination R^2, the fraction of target variance accounted for by emulator output, as a function of vertical level (on the left) and latitude (on the right). The R^2 score is unity for an emulator that explains the data exactly, zero for constant predictions of the target mean, and arbitrarily negative as emulator performance degrades. Dashed lines indicate performance on the training data, solid lines on the test data unseen during training. All three emulators perform slightly better on the training data than the held-out test data, as is expected, but no emulator exhibits the large train-test gap characteristic of overfitting. Figure 4 shows

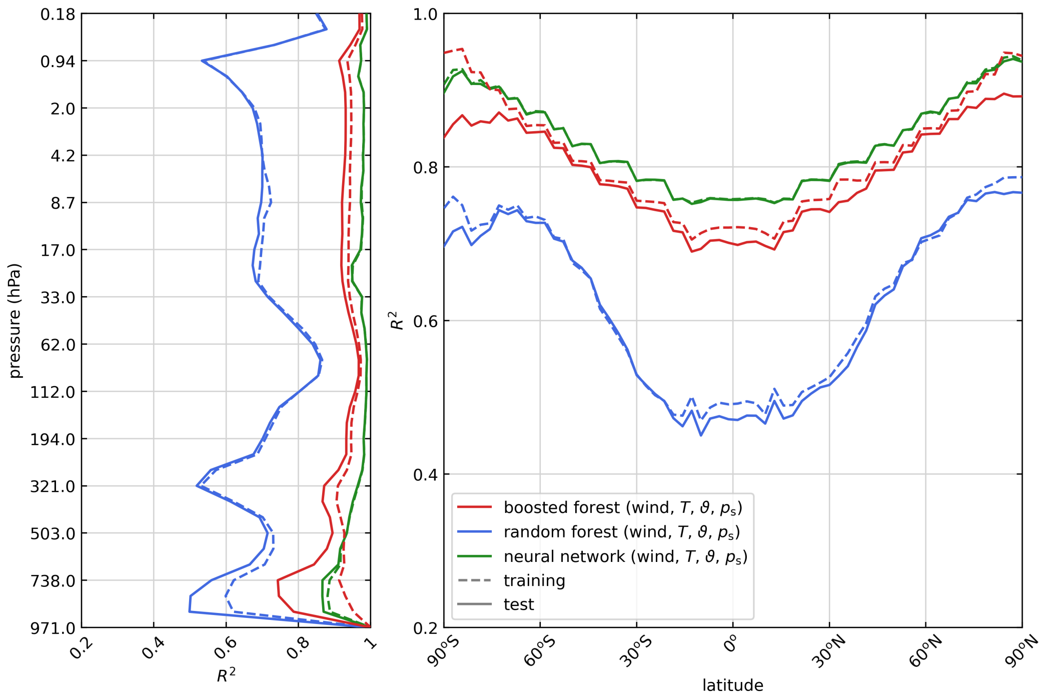

Figure 4. Offline R^2 scores of three data-driven emulators on the training and test datasets. Parentheses indicate the input features used by each model.

that the neural network has the best offline performance, closely followed by the boosted forest, with the random forest much worse than either.

Performance tends to be better aloft than near the surface. In AD99, layers below 321 hPa are sometimes below the tropopause-following source level and sometimes above it, depending on latitude, and emulator predictions are worse at these lower levels. Emulator error is also larger in the tropics, likely because the peak of the AD99 source spectrum switches there to following the mean flow at the source level, making the emulation task more complex. This degraded performance is not an artifact of sampling, because training samples were sampled weighted by grid box area, so that the tropics are well-represented in the training data.

Following Espinosa et al. (2022), we train boosted forests on different combinations of input features to assess the utility of various physical variables. All forests used the hyperparameter choices described in Table S3. Figure 5 shows the R^2 scores for these models. We observe that changing the input features does not result in significant performance increases or decreases, except that the forest that does not see latitude ϑ performs very

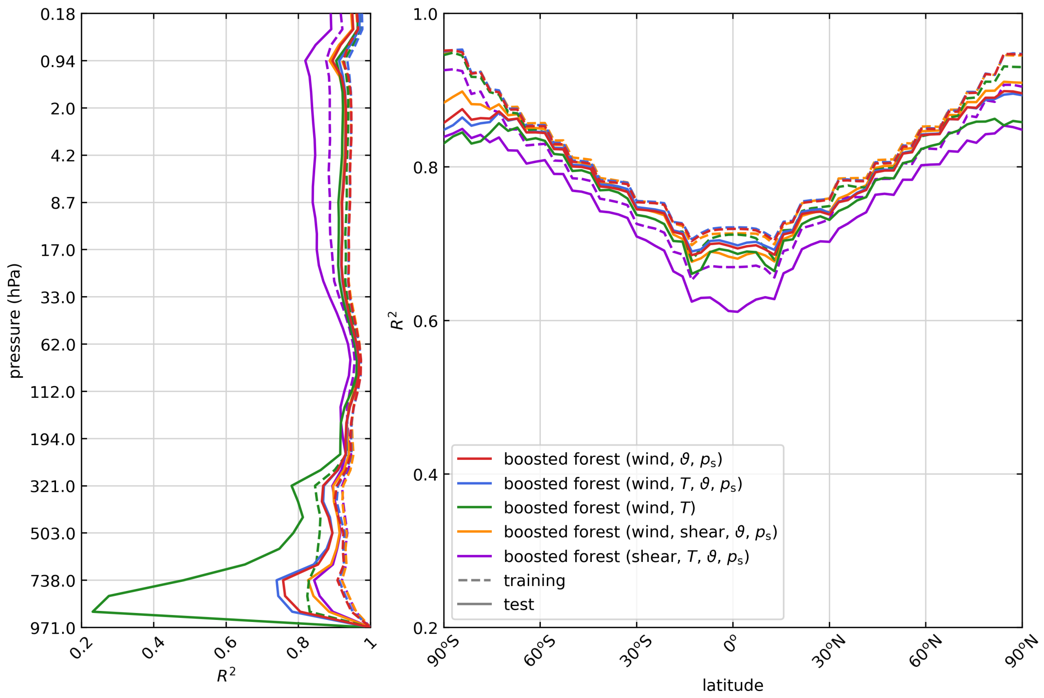

Figure 5. As in Figure 4, but for boosted forests using different combinations of input features.

poorly at vertical levels which can be below the source level. Otherwise, the differences between particular boosted forests are smaller than those between boosted forests and the other architectures considered in this work. For comparisons between architectures, we use the boosted forest trained on wind, temperature, latitude, and surface pressure, as it outperforms the others by a slight margin in the tropics, where the GWP-driven QBO occurs. This is also the same feature set used by AD99.

We performed a similar set of experiments with random forests. Again, their performance was largely unaffected by changing the input features, and all the random forests remained substantially worse than the boosted forests, as Figure 4 suggests. For this reason, the bulk of the analysis in the remainder of this work is focused on boosted forest emulators, with comparisons to random forests only when appropriate.

4.2 Climatological biases

Figure 6 shows the biases in zonal mean u and T relative to the AD99 control run of the coupled runs described in Section 2.3. The boosted forest shows the least bias overall. The random forest and neural network produce large biases mainly in the tropical stratosphere,

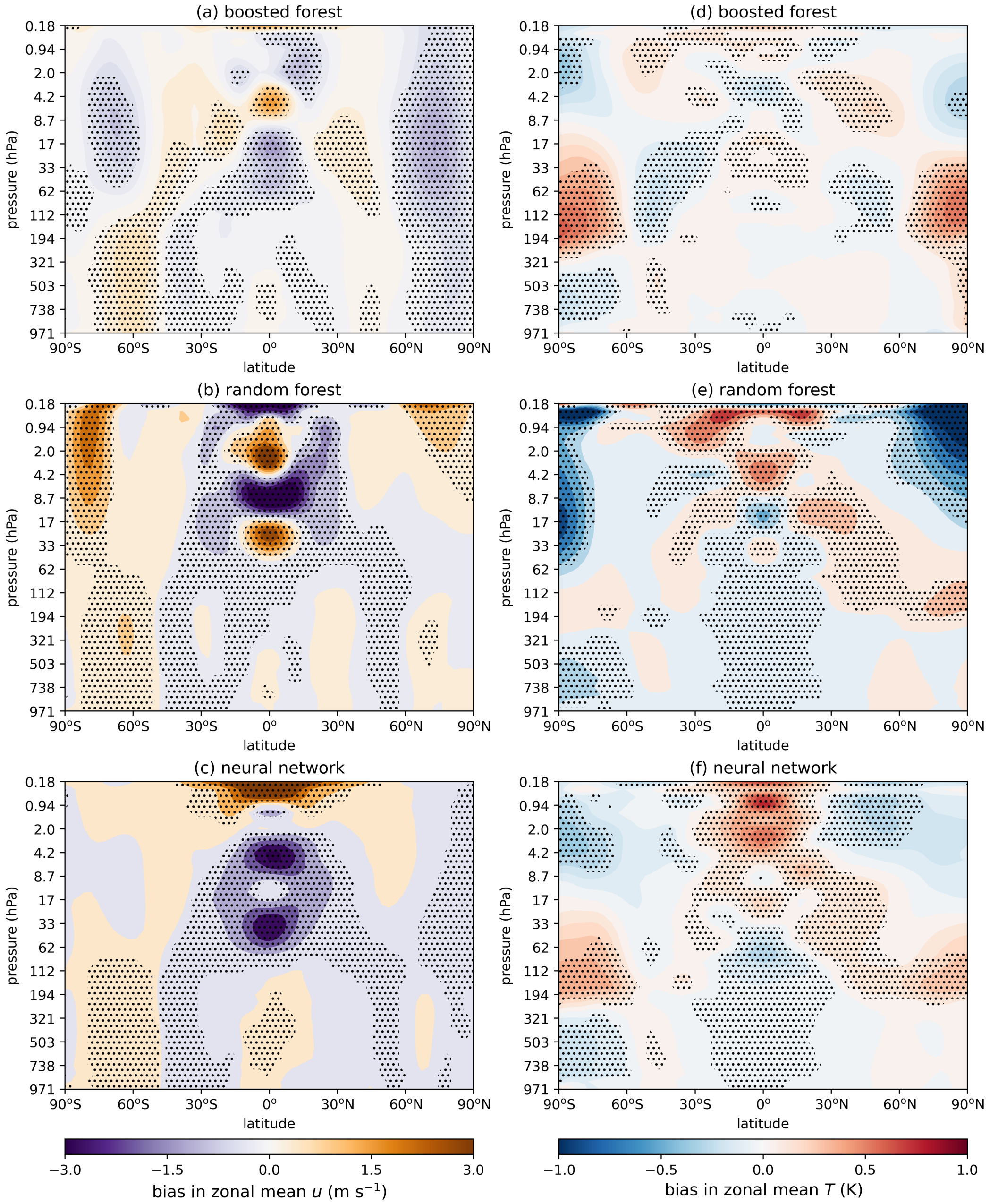

Figure 6. Biases with respect to AD99 integrations in zonal mean u (a-c, left column) and T (d-f, right column) from coupled integrations of the three emulators shown in Figure (4). Stippling indicates regions where bias is significant at the 95% level.

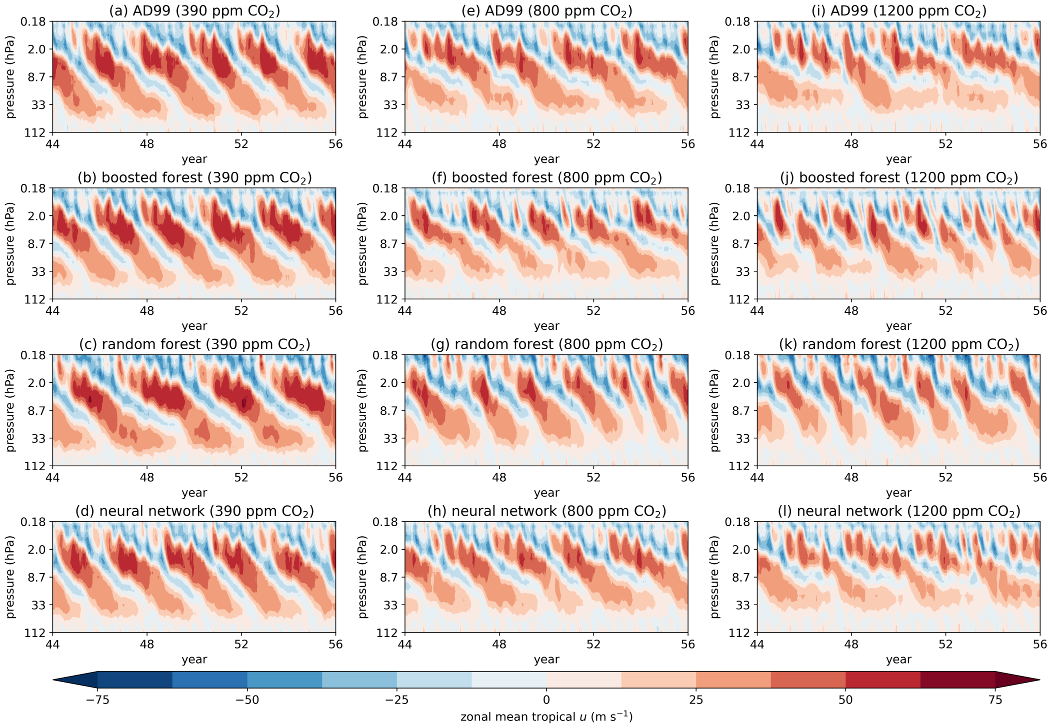

Figure 7. QBOs from the final twelve years of integrations of MiMA coupled to AD99 (top row) and the three data-driven emulators from Figure 4 (next three rows). The control climate integrations are on the left, the 800 ppm CO_2 runs in the middle, and the 1200 ppm CO_2 runs on the right.

where variability from the QBO dominates, and to a lesser extent near the poles. In particular, the tropospheric jet structure is emulated well and free of systematic bias. Remarkably, the random forest, which exhibited much poorer offline performance in Figure 4, runs stably online without significantly altering the zonal mean climate in the troposphere.

The zonal-mean biases were similar in the coupled runs with increased atmospheric carbon dioxide. The remainder of this section will focus on climate phenomena that are particularly dependent on the particular gravity wave parameterization: the QBO and the occurrence of sudden stratospheric warmings (SSWs).

4.3 The Quasi-Biennial Oscillation

Atmospheric models generally need parameterized gravity wave drag to represent the QBO in the tropical stratosphere (Anstey et al., 2022), and so the emulators' skill at re-

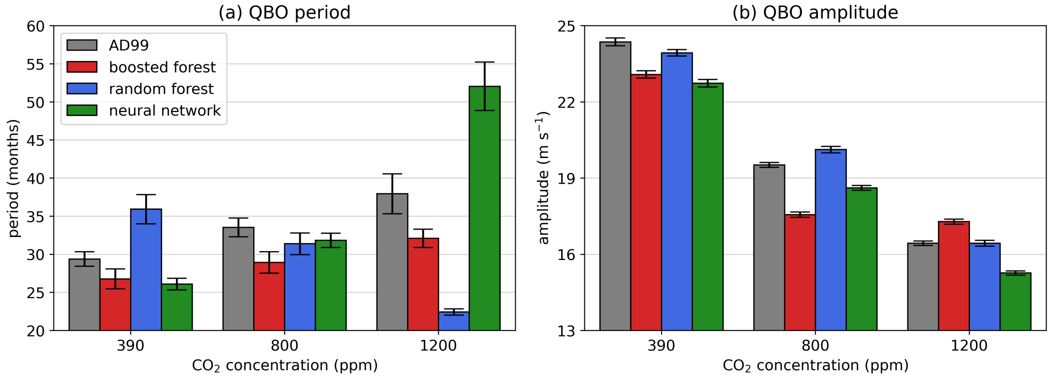

Figure 8. Periods (left) and amplitudes (right) of QBOs driven by AD99 and three emulators in integrations with three values of CO_2 concentration. Periods are determined by the dominant Fourier mode at 10 hPa, with uncertainty given by the half-width of the spectral peak around that mode. Amplitudes are the standard deviation at 10 hPa, with uncertainty determined from the 95% confidence intervals as calculated by bootstrapping.

producing the statistics of the QBO simulated by AD99 is a key metric of their online performance. For simplicity, we will refer to any simulated oscillation in the tropically- and zonally-averaged zonal wind as a QBO, even when the period is no longer "quasi-biennial". Figure 7 shows the QBO time series from MiMA integrations coupled to AD99 and the three emulator architectures. In all three carbon scenarios, AD99 and each of the three emulators produce a QBO. Especially in the high-carbon integrations, though, these oscillations vary both qualitatively and quantitatively. The period and amplitude response of the QBO for each parameterization is shown in Figure 8.

The period of the QBO driven by AD99 increases monotonically with CO_2 concentration. The sign of this response is captured by the boosted forest and neural network emulators, though the boosted forest periods are biased low and the neural network significantly overshoots the AD99 period in the 1200 ppm case. (Note, however, that the QBO in Figure 7i is not nearly as divergent from the AD99 oscillation as this calculation might suggest.) The random forest, by contrast, drives a QBO with a significantly longer period in the control 390 ppm CO_2 scenario, and the period decreases with increasing carbon dioxide. This failure to replicate the behavior of AD99 is perhaps unsurprising, given the random forest's relatively poor offline performance (Figure 4).

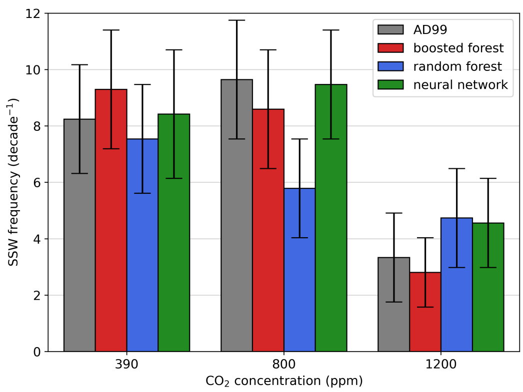

Figure 9. Sudden stratospheric warming frequencies from the MiMA runs described in Section 2.3. SSWs are identified using the criterion of Butler et al. (2017), based on sign changes in the zonal mean zonal wind at 60°N and 10 hPa. The uncertainties are the 95% confidence intervals, as calculated by bootstrapping winters and counting SSWs in the subsampled populations. This approach assumes that SSWs in different winters are independent.

The amplitude of the reference QBO decreases in response to increased CO_2. All three emulators show at least broadly similar behavior (Figure 8(b)), though the boosted forest significantly overestimates the response in the 800 ppm CO_2 case, and the neural network amplitudes are biased low. It is of note that the random forest, heretofore the worst of the three emulators, is the most able to reproduce the correct QBO amplitudes across all three integrations.

4.4 Sudden stratospheric warmings

Sudden stratospheric warmings (SSWs) are abrupt increases in the temperature of the wintertime polar stratosphere accompanied by a reversal in the polar vortex. Their occurrence is governed in part by gravity wave propagation — either directly through wave-driven momentum flux (Song et al., 2020) or indirectly through moderation by the QBO (Butler et al., 2017) — and their frequency thus provides an additional statistic with which to assess the emulators' online performance. Figure 9 shows the SSW frequencies for AD99 and the three emulators at three CO_2 concentrations.

In the 390 ppm CO_2 integrations, the random forest appears to best reproduce the SSW frequency, but the uncertainties are large and the confidence intervals for all three emulators overlap considerably with that of AD99. When the CO_2 concentration is raised to 800 ppm, AD99 produces SSWs more frequently. This response is not well captured by any emulator: none of them demonstrates a large response relative to the uncertainties. However, in the more extreme 1200 ppm CO_2 runs, all three emulators capture the large decrease in SSW frequency exhibited by AD99.

5 Feature importance analysis

The offline and online results presented in Section 4 show that the neural network slightly outperforms the boosted forest offline, performs comparably online in the control climate, and is somewhat better at reproducing the AD99 response to the 800 ppm CO_2 scenario. The final goal of this study is to use the interpretability tools laid out in Section 3.4 to calibrate the online behavior of regression forest models towards more desirable outcomes.

5.1 Computed SHAP values

To further analyze the behavior of our emulators, we calculate SHAP values (Section 3.4), a measure of feature importance, the relative importance a model ascribes to various input features in making its predictions. Figure 10 shows dataset-averaged SHAP values for the three emulators in Figure 4 and for the AD99 parameterization itself. Each panel shows the importance of input features from all levels (wind and temperature importances are summed per level) to predictions at a single level. Note that SHAP values have the same units as the parameterization outputs, which in this case are standard deviations of the gravity wave acceleration at the prediction level.

The SHAP profiles for AD99, the boosted forest, and the neural network show preferential use of input information from at and just below the prediction level. This pattern matches our physical intuition, given that AD99 simulates strictly upward propagation of waves, and indicates that the boosted forest and neural network have learned to make predictions according to the "physics" encoded in AD99. (The importance maxima at levels immediately above the prediction level do not contradict this characterization: AD99 calculates drags at level interfaces before interpolating to full levels, so that information from one level higher than the prediction level is used.)

The random forest profiles, however, are much more muted. This suggests that the random forest does not identify the set of features on which the drag at each level depends, reflecting a failure to learn the physical structure of the problem and at least partially

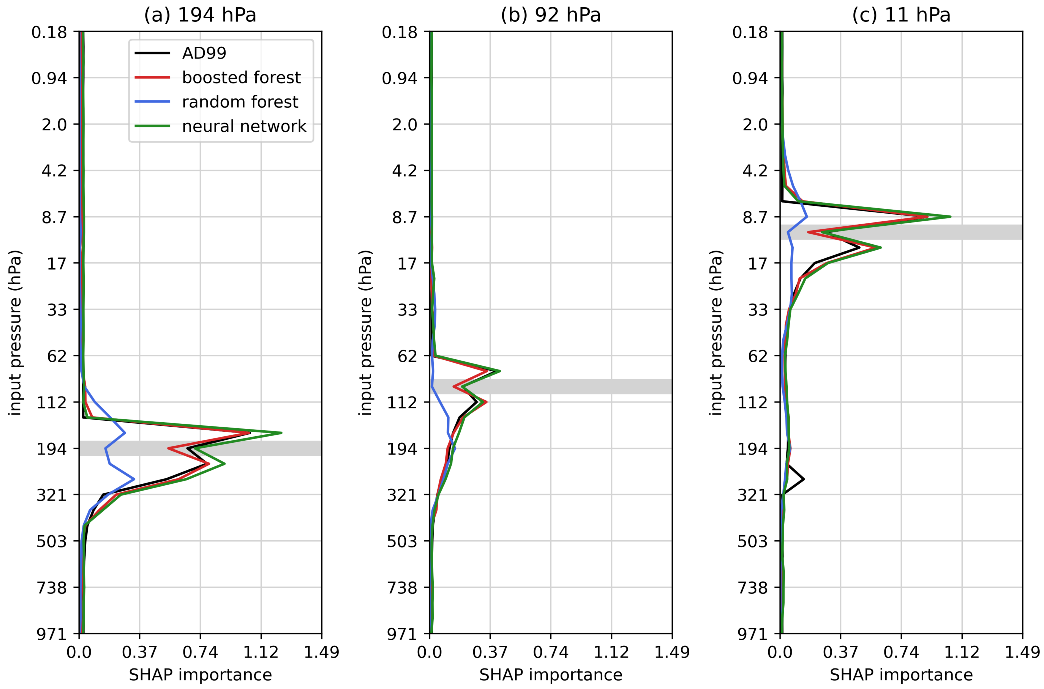

Figure 10. Test dataset-averaged absolute SHAP values for predictions at several vertical levels for AD99 and the three emulators in Figure 4. Each input pressure level includes the importances of both wind and T features. The prediction level is highlighted in gray.

explaining the poor R^2 scores in Figure 4. The emulators with high R^2 scores, the boosted forest and neural network, use the input features almost identically to AD99 — as opposed to, say, achieving good performance by relying on spurious correlations between features.

Of additional interest is the peak in the AD99 importance profile near 200 hPa in Figure 10(c), showing that even for predictions aloft, AD99 makes use of layers near the tropopause. Indeed, in AD99 the tropical phase speed spectrum is set by the source level winds, and many waves are immediately filtered at the tropopause. All three emulators appear to under-emphasize input information near the source level.

For the random and boosted forests, we computed *Gini importances*, an additional metric of feature importance defined only for regression tree-based architectures, and found them to be in good qualitative agreement with the SHAP values shown here. See Text S2 and Figure S4 for details. We therefore believe the conclusions about the models considered here to be reasonably robust to feature importance method.

5.2 SHAP-informed retraining

Although Figure 10 indicates that the boosted forest and neural network use wind and temperature information in a physically plausible way, it does not explain the differences in online behavior observed in Figures 7 and 8. The left panel of Figure 11 shows the SHAP importance of latitude to all prediction levels for AD99 and the three emulators. AD99 has a large maximum in latitude SHAP value near the tropopause. Strikingly, the neural network values match this maximum quite closely, while the boosted forest considerably under-emphasizes latitude in this region of the atmosphere. (Further aloft, all three emulators have latitude SHAP values less than those of AD99.)

This result suggests that underuse of latitude might be a key factor differentiating the boosted forest from the neural network. Moreover, one might wonder if a boosted forest forced to pay closer attention to latitude might have better online behavior. In particular, the distribution of input latitudes is necessarily fixed, while the distribution of input flow variables is subject to shift under climate perturbations.

To test this, we trained a boosted forest with identical hyperparameters to the one considered thus far; however, we added the latitude of each training sample as an additional target. The idea is that, since latitude is now both an input and an output, the input

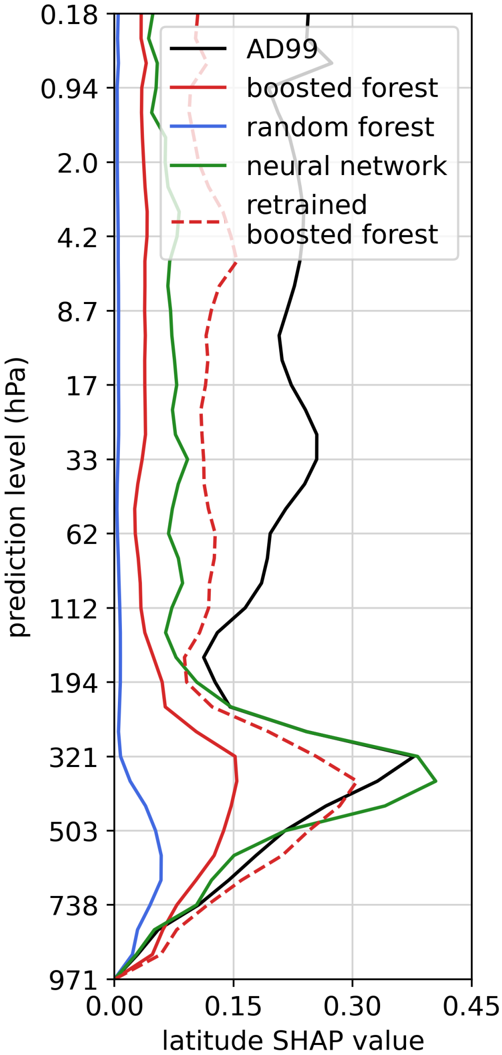

Figure 11. Test dataset-averaged absolute SHAP importance of latitude to parameterization outputs at each vertical level for AD99, the three emulators from Figure 4 (solid lines), and the boosted forest retrained as described in Section 5.2.

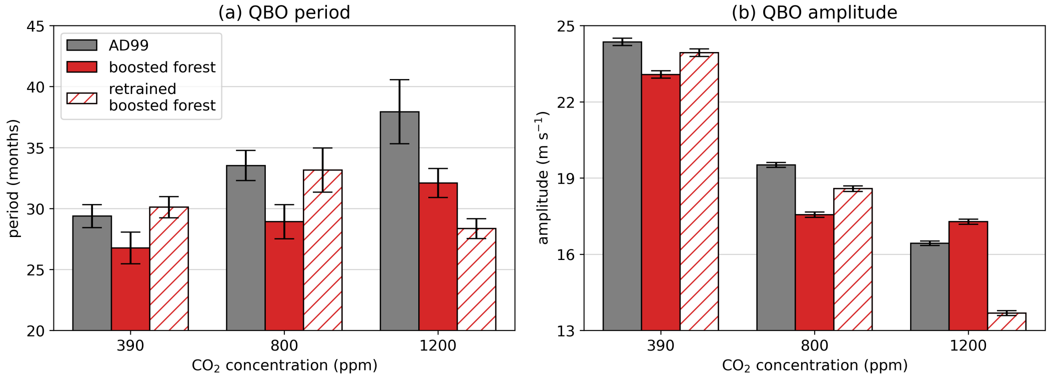

Figure 12. As in Figure 8, but including the boosted forest trained as described in Section 5.2.

latitude should be more useful in partitioning the training output vectors into low-impurity subsets (see Text S1). Latitude should therefore be selected at more important nodes in the constituent trees. We can multiply the latitude values added as targets by a constant (introducing one new hyperparameter) to control the strength of this effect. At prediction time, we simply discard the predicted latitude and retain only the predicted drags.

The boosted forest trained in this manner achieves R^2 scores (not shown) essentially indistinguishable from the boosted forest in Figure 4. Nor was the use of level-specific information (as in Figure 10) significantly different from the original boosted forest. However, the dashed line in Figure 11 demonstrates that this training procedure produces a forest for which latitude is two to three times as important throughout the vertical extent of the atmosphere. More significantly, Figure 12 shows that when coupled to MiMA, the retrained boosted forest drives a QBO with period and amplitude closer to those of AD99 then the original forest in both the 390 ppm and 800 ppm CO_2 scenarios. The QBO statistics in the 1200 ppm integration remain poor, suggesting that this extreme carbon perturbation is simply beyond the ability of this boosted forest architecture to generalize.

6 Discussion

In this study, we used boosted and random forests to emulate a gravity wave parameterization, performed offline and online evaluations of emulator performance, and investigated whether emulator use of input features respected known physical properties of the data. To our knowledge, this work represents the first use of boosted forests in the design of parameterizations for climate models. Boosted forests are not uncommon in the wider machine learning literature; for example, they are often used in winning submissions to competitions (Bojer & Meldgaard, 2021). But while random forests have been employed in several parameterization studies (Belochitski et al., 2011; O'Gorman & Dwyer, 2018; Yuval & O'Gorman, 2020), boosted forests have not, perhaps because of the absence of native support in boosting libraries for the multioutput problems that abound in climate modeling.

We have shown that boosted forests significantly outperform random forests according to almost every metric, even with a relatively simple implementation of multioutput boosting that lacks much of the flexibility of the optimized boosting libraries used for scalar problems. Boosted forests may therefore be a valuable and yet-underused tool as climate models continue to move towards incorporating data-driven parameterizations, especially since they were as skillful, or nearly so, as neural networks. In particular, even though a boosted forest makes inferences based exclusively on outputs in the training data, ours was able to generalize to out-of-sample conditions equally as well as the neural network benchmark: both schemes performed well under moderately enhanced CO_2, but struggled under our most extreme scenario. Out-of-sample generalization is often challenging for data-driven methods, but recent work by Sun et al. (submitted) suggests that transfer learning may be a solution if one can provide a small amount of high-quality data from the new regime — for example, data from a high-resolution simulation under increased CO_2.

We further interrogated our data-driven schemes with methods from interpretable machine learning to quantify how they used input features to make predictions. While the Gini importance is a natural interpretability metric for regression forests (Text S2), we found that it provided nearly the same information (Figure S4) as SHapley Additive exPlanation (SHAP) analysis (Lundberg & Lee, 2017), a method-agnostic approach that can be used on any ML scheme and even on the original physics-based parameterization. The skillful boosted forest and neural network emulators exhibited SHAP values nearly matching those of AD99, much more closely than did the under-performing random forest (Figure 10). For the machine learning architectures considered in this work, then, emulation skill appears to go hand in hand with capacity to learn elements of the spatial structure of the problem. This observation suggests that these kinds of data-driven models may be able to infer similar structures from more realistic data sources, for which the true SHAP values will of course be unavailable.

Moreover, the analysis in Section 5.2 demonstrates the utility of SHAP analysis for understanding and improving online behavior beyond more common error metrics like R^2 scores. The offline errors in Figure 4 showed the boosted forest performing worse than the neural network, and the QBO statistics in Figure 8 confirmed the forest to be less successful online under certain scenarios. Only the SHAP analysis culminating in Figure 11, though, provided actionable information, informing us that the boosted forest was not sufficiently taking into account latitudinal variation at prediction time. This informed a training procedure to improve the online behavior. Our approach remains somewhat *ad hoc*: the decision to focus on the input latitude was made by eye, and there may be superior ways to constrain forests to emphasize given input features. Nonetheless, we believe this result to be a useful preliminary towards calibrating the online behavior of data-driven parameterizations that may lack the explicit parameters used to tune physics-based schemes.

Finally, from a practical standpoint, multioutput boosted forest libraries like the one implemented here will need to be made more efficient and self-contained if they are to provide a competitive alternative to neural networks in climate research. We found anecdotally that boosted forests constructed with only a few deep trees followed by many much shallower ones could perform nearly as well as, and considerably faster than, the constant-depth forests used here, though more investigation is required to fully explore the interplay between forest size and skill.

7 Data Availability Statement

The code used to run the experiments in this work is available, with documentation, at `https://github.com/dsconnelly/willow`. The random and boosted forests were trained using Mubofo (Connelly, 2023), a Python package maintained by author D. S. Connelly and available through Python Package Index (PyPI) at `https://pypi.org/project/mubofo`. The source code may be found at `https://github.com/dsconnelly/mubofo` and is distributed under the BSD-3-Clause license.

Mubofo is built around scikit-learn (Pedregosa et al., 2011), and the neural network was trained using PyTorch (Paszke et al., 2019). The SHAP values were computed with the shap Python package (Lundberg & Lee, 2017) available at `https://github.com/shap/shap`. The idealized atmospheric model MiMA is described in Jucker and Gerber (2017) and Garfinkel et al. (2020) and is available at `https://github.com/mjucker/MiMA`. The coupling interface

Forpy (Rabel, 2020) is available at `https://github.com/ylikx/forpy`. The authors are not involved with the maintenance of any of these software packages.

Acknowledgments

This work was supported by Schmidt Futures, a philanthropic initiative founded by Eric and Wendy Schmidt, as part of the Virtual Earth System Research Institute (VESRI), the U.S. National Science Foundation through award OAC-2004572, and the U.S.-Israel Binational Science Foundation through award 2020316.

References

Alexander, M. J., & Dunkerton, T. J. (1999). A Spectral Parameterization of Mean-Flow Forcing due to Breaking Gravity Waves. *Journal of the Atmospheric Sciences*, *56*(24), 4167-4182.

Anstey, J. A., Butchart, N., Hamilton, K., & Osprey, S. M. (2022). The sparc quasi-biennial oscillation initiative. *Quarterly Journal of the Royal Meteorological Society*, *148*(744), 1455-1458.

Belochitski, A., Binev, P., DeVore, R., Fox-Rabinovitz, M., Krasnopolsky, V., & Lamby, P. (2011). Tree approximation of the long wave radiation parameterization in the NCAR CAM global climate model. *Journal of Computation and Applied Mathematics*, *236*, 447-460.

Betts, A. K. (1986). A new convective adjustment scheme. Part I: Observational and theoretical basis. *Quarterly Journal of the Royal Meteorological Society*, *112*(473), 677-691.

Betts, A. K., & Miller, M. J. (1986). A new convective adjustment scheme. Part II: Single column tests using GATE wave, BOMEX, ATEX and arctic air-mass data sets. *Quarterly Journal of the Royal Meteorological Society*, *112*(473), 693-709.

Bojer, C. S., & Meldgaard, J. P. (2021). Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, *37*, 587-603.

Bolton, T., & Zanna, L. (2019). Applications of Deep Learning to Ocean Data Inference and Subgrid Parameterization. *Journal of Advances in Modeling Earth Systems*, *11*, 379-399.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5-32.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and Regression Trees.* Chapman and Hall/CRC.

Brenowitz, N. D., & Bretherton, C. S. (2018). Prognostic Validation of a Neural Network Unifed Physics Parameterization. *Geophysical Research Letters*, *45*, 6289-6298.

Butler, A. H., Sjoberg, J. P., Seidel, D. J., & Rosenlof, K. H. (2017). A sudden stratospheric warming compendium. *Earth System Science Data*, *9*, 63-76.

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*.

Checa-Garcia, R. (2018). *CMIP6 Ozone forcing dataset: supporting information.* Zenodo. doi: 10.5281/zenodo.1135127

Checa-Garcia, R., Hegglin, M. I., Kinnison, D., Plummer, D. A., & Shine, K. P. (2018). Historical Tropospheric and Stratospheric Ozone Radiative Forcing Using the CMIP6 Database. *Geophysical Research Letters*, *45*(7), 3264-3273.

Chevallier, F., Chéruy, F., Scott, N. A., & Chédin, A. (1998). A Neural Network Approach for a Fast and Accurate Computation of a Longwave Radiation Budget. *Journal of Applied Meteorology*, *37*, 1385-1397.

Connelly, D. S. (2023). *Mubofo.* Github repository. Retrieved from `https://github.com/dsconnelly/mubofo`

Espinosa, Z. I., Sheshadri, A., Cain, G. R., Gerber, E. P., & DallaSanta, K. J. (2022). Machine Learning Gravity Wave Parameterization Generalizes to Capture the QBO and Response to Increased CO2. *Geophysical Research Letters*, *49*(8).

Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, *29*(5), 1181-1232.

Fritts, D. C., & Alexander, M. J. (2003). Gravity wave dynamics and effects in the middle atmosphere. *Reviews of Geophysics*, *41*(1).

Garfinkel, C. I., White, I., Gerber, E. P., Jucker, M., & Erez, M. (2020). The Building Blocks of Northern Hemisphere Wintertime Stationary Waves. *Journal of Climate*, *33*(13), 5611-5633.

Iacono, M. J., Mlawer, E. J., Clough, S. A., & Morcrette, J.-J. (2000). Impact of an improved longwave radiation model, RRTM, on the energy budget and thermodynamic properties of the NCAR community climate model, CCM3. *Journal of Geophysical Research*, *105*(D11), 14873-14890.

Janzing, D., Minorics, L., & Bloebaum, P. (2020). Feature relevance quantification in explainable AI: A causal problem. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics* (Vol. 108, p. 2907-2916). PMLR.

Jucker, M., & Gerber, E. P. (2017). Untangling the Annual Cycle of the Tropical Tropopause Layer with an Idealized Moist Model. *Journal of Climate*, *30*(18), 7339-7358.

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., ... Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, *2*(1), 56–67.

Lundberg, S. M., & Lee, S.-I. (2017). A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc.

Mlawer, E. J., Taubman, S. J., Brown, P. D., Iacono, M. J., & Clough, S. A. (1997). Radiative transfer for inhomogeneous atmospheres: RRTM, a validated correlated-k model for the longwave. *Journal of Geophysical Research*, *102*(D14), 16663-16682.

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*, 2548-2563.

Palmer, T. N., Shutts, G. J., & Swinbank, R. (1986). Alleviation of a systematic westerly bias in general circulation and numerical weather prediction models through an orographic gravity wave drag parametrization. *Quarterly Journal of the Royal Meteorological Society*, *112*, 1001-1039.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc.

Pearl, J. (2000). *Causality.* Cambridge University Press.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825-2830.

Rabel, E. (2020). *Forpy.* Github repository. Retrieved from `https://github.com/ylikx/forpy`

Richter, J. H., Butchart, N., Kawatani, Y., Bushell, A. C., Holt, L., Serva, F., ... Yukimoto, S. (2022). Response of the Quasi-Biennial Oscillation to a warming climate in global climate models. *Quarterly Journal of the Royal Meteorological Society*, *148*(744), 1490-1518.

Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning*, *5*, 197-227.

Shapley, L. S. (1953). A value for n-person games. In H. W. Kuhn & A. W. Tucker (Eds.), *Contributions to the Theory of Games* (p. 307-317).

Shrikumar, A., Greenside, P., & Kundaje, A. (2017). Learning Important Features Through Propagating Activation Differences. In *Proceedings of the 34th International Conference on Machine Learning* (pp. 3145–3153). PMLR.

Song, I.-S., Lee, C., Chun, H.-Y., Kim, J.-H., Jee, G., Song, B.-G., & Bacmeister, J. T. (2020). Propagation of gravity waves and its effects on pseudomomentum flux in a sudden stratospheric warming event. *Atmospheric Chemistry and Physics*, *20*, 7617-7644.

Sun, Y. Q., Pahlavan, H. A., Chattopadhyay, A., Hassanzadeh, P., Lubis, S. W., Alexander, M. J., … Guan, Y. (submitted). Data Imbalance, Uncertainty Quantification, and Generalization via Transfer Learning in Data-driven Parameterizations: Lessons from the Emulation of Gravity Wave Momentum Transport in WACCM. *Journal of Advances in Modeling the Earth System*.

Yuval, J., & O'Gorman, P. (2020). Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature Communications*, *11*.