Copula-based synthetic data augmentation for machine learning emulators

David Meyer[1,2], Thomas Nagler[3], Robin J. Hogan[4,1]

[1]Department of Meteorology, University of Reading, Reading, UK, [2]Department of Civil and Environmental Engineering, Imperial College London, London, UK, [3]Mathematical Institute, Leiden University, Leiden, The Netherlands, [4]European Centre for Medium-Range Weather Forecasts, Reading, UK

Correspondence to David Meyer (d.meyer@pgr.reading.ac.uk)

Abstract

Can we improve machine learning (ML) emulators with synthetic data? If data are scarce or expensive to source and a physical model is available, statistically generated data may be useful for augmenting training sets cheaply. Here we explore the use of copula-based models for generating synthetically augmented datasets in weather and climate by testing the method on a toy physical model of downwelling longwave radiation and corresponding neural network emulator. Results show that for copula-augmented datasets, predictions are improved by up to 62 % for the mean absolute error (from 1.17 to 0.44 W m^{-2}).

1. Introduction

The use of machine learning (ML) in weather and climate is becoming increasingly popular (Huntingford et al., 2019; Reichstein et al., 2019). ML approaches are being applied to an increasingly diverse range of problems such as improving the modelling of radiation (e.g. Cheruy et al., 1996; Chevallier et al., 1998, 2000; Krasnopolsky et al., 2005; Meyer et al., 2021; Ukkonen et al., 2020; Veerman et al., 2021), ocean (e.g. Bolton and Zanna, 2019; Krasnopolsky et al., 2005), chemistry (e.g. Nowack et al., 2018), convection (e.g. Krasnopolsky et al., 2013), the representation of sub-grid processes (e.g. Brenowitz and Bretherton, 2018; Gentine et al., 2018; O'Gorman and Dwyer, 2018; Rasp et al., 2018), and the post-processing of model outputs (e.g. Krasnopolsky and Lin, 2012; Rasp and Lerch, 2018).

When it comes to training ML models for weather and climate applications two main categories may be identified: one where input and output pairs are directly provided (e.g. where both come from observations), and a second where inputs are provided but corresponding outputs are generated through a *physical model* (e.g. parameterization schemes or even a whole weather and climate model). Although the former may be considered the most common training strategy in use today, when the underlying physics is well understood (e.g. radiative transfer) and numerical codes are available, the latter may be of particular interest for developing one-to-one *emulators* (i.e. statistical surrogates of their physical counterparts) which can be used to improve computational performance for a trade-off in accuracy (e.g. Chevallier et al., 1998; Meyer et al., 2021; Ukkonen et al., 2020; Veerman et al., 2021). Here, for clarity, we will only be focusing on the latter case and refer to them as emulators throughout the paper.

In ML, the best way to make a model more generalizable is to train it on more data (Goodfellow et al., 2016). However, depending on the specific field and application, input data may be scarce, representative of only a subset of situations and domains, or in the case of synthetically generated data, may require large computational resources and bespoke infrastructures, or specific domain knowledge. For example, generating atmospheric profiles using a general circulation model (GCM) may require in-depth knowledge of the GCM and large computational resources (e.g. NWP-SAF datasets used for training emulators in Meyer et al., 2021).

A possible solution to these issues may be found by augmenting the available input dataset with more samples. Although this may be a straightforward task for classification problems (e.g. by translating or adding noise to an image), this may not be the case for parameterizations of physical processes used in weather and climate models. In this context, it is common to work with high dimensional and strongly dependent data (e.g. between physical quantities such as air temperature, humidity, and pressure across grid points), and although this dependence may be well approximated by simple physical laws (e.g. the ideal gas law for conditions found in the Earth’s atmosphere), this is often not the case, making the generation of representative data across multiple dimensions challenging (e.g. the nonlinear relationship between cloud properties, humidity and temperature).

To serve a similar purpose to real data, synthetically generated data thus need to preserve the statistical properties of real data in terms of individual behaviour and (inter-)dependences. Several methods may be suitable for generating synthetic data generation such as copulas (e.g. Patki et al., 2016), variational autoencoders (e.g. Wan et al., 2017) and, more recently, generative adversarial networks (GANs; e.g. Xu and Veeramachaneni, 2018). Although the use of GANs for data generation is becoming increasingly popular amongst the core ML community, these require multiple models to be trained, leading to difficulties and computational burden (Tagasovska et al., 2019). Variational approaches, on the other hand, make strong distributional assumptions, potentially detrimental to generative models (Tagasovska et al., 2019). Compared to black-box deep learning models, the training of vine copulas is relatively easy and robust, while taking away a lot of guesswork in specifying hyperparameters and network architecture. Furthermore, copula models give a direct representation of statistical distributions, making them easier to interpret and tweak after training. As such, copula-based models have been shown to be effective for generating synthetic data, comparable to the real data in the context of privacy protection (Patki et al., 2016).

The goal of this paper is to improve ML emulators by augmenting the physical model's inputs using copulas. We give a brief overview of methods in section 2.1 with specific implementation details in sections 2.2-2.6. Results are shown in section 3, with a focus on evaluating synthetically generated data in section 3.1 and ML predictions in section 3.2. We conclude with a discussion and prospects for future research in section 4.

2. Material and methods

2.1 Overview

The general method for *training* a ML emulator involves the use of paired *inputs* $X = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$ and *outputs* $Y = \{\boldsymbol{y}_1, \dots, \boldsymbol{y}_n\}$ corresponding to the best function approximation for a specific architecture and configuration. For *inference*, the trained ML emulator is then used to predict new outputs Y^* from inputs X^*. Outputs Y are generated through a physical model from X, and fed to the ML emulator for training (Figure 1 A). In this paper we introduce an additional step, that is, augmentation through copula-based synthetic data generation (Figure 1 B). The method is demonstrated with a toy model of downwelling radiation as the physical model (section 2.4) and a simple feedforward neural network (FNN) as the ML emulator (section 2.5). To evaluate the impact of copula-generated synthetic data on predictions we focus on predicting vertical profiles of longwave radiation from those of dry-bulb air temperature, atmospheric pressure, and cloud optical depth (other parameters affecting longwave radiative transfer, such as gas optical depth, are treated as constant in the simple model described in section 2.4). This task is chosen at it allows us to: (i) evaluate copula-based models for generating correlated multidimensional data (e.g. with dependence across several quantities and grid points), some of which (e.g. cloud optical depth) are highly non-Gaussian; (ii) develop a simple and fast toy physical model that may be representative of other physical parameterizations such as radiation, land surface, urban, cloud, or convection schemes; and (iii) develop a fast and simple ML emulator used to compute representative statistics. Here we define case A as the *baseline* and generate six different subcases for case B using (i) three levels of data *augmentation factors* (i.e. either 1x, 5x or 10x the number of profiles in the real dataset), and (ii) generated from three different copula types. In the following sections we give background information and specific implementation details about the general method used for setting up the source data (section 2.2), data generation (section 2.3), target generation (section 2.4), and estimation training (section 2.5) as shown in Figure 1.

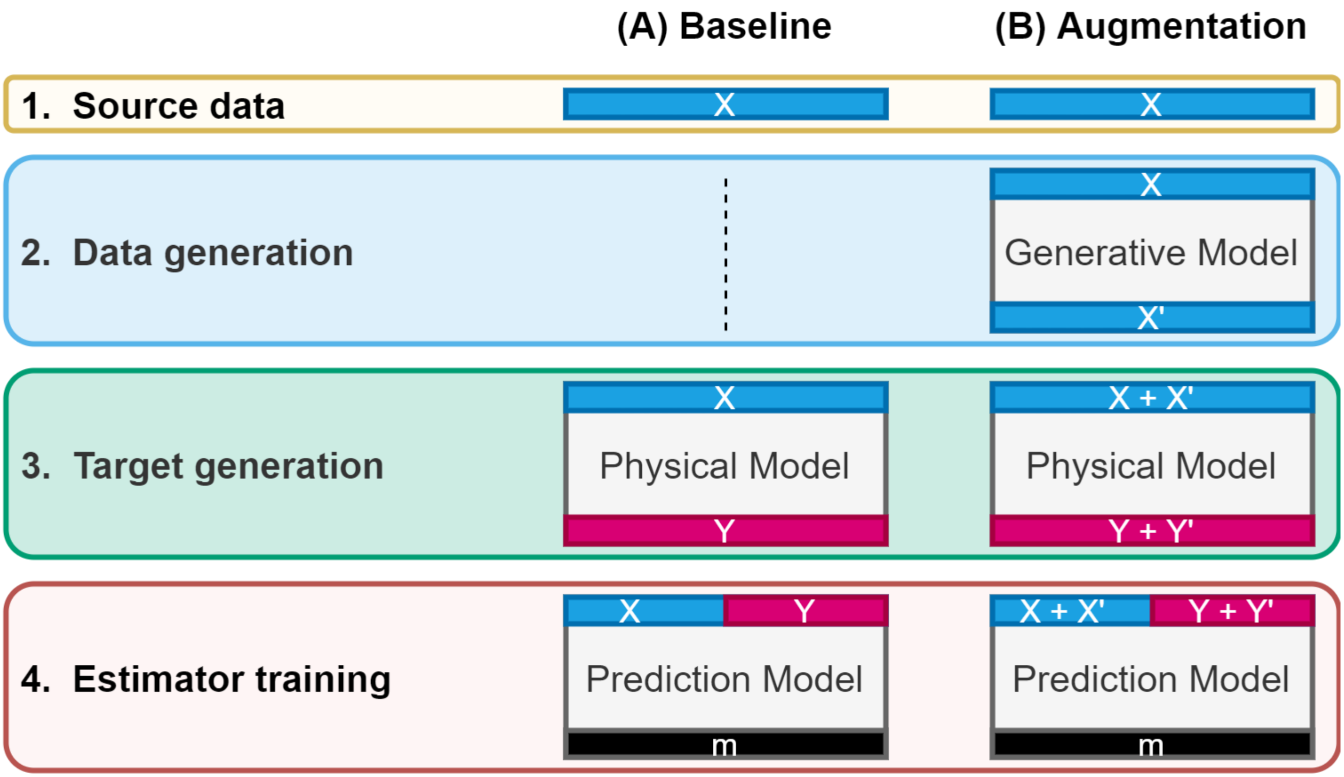

Figure 1. General strategies identified for training ML emulators. (A) inputs X are fed to the physical model to generate corresponding outputs Y; X and Y used to train the ML emulator. (B) a data generation model (here copula) is fitted to inputs X to generate synthetic inputs X'; inputs X and X' are fed to the physical model to generate corresponding outputs Y and Y'; both X and X', and Y and Y' are used to train the ML emulator. After training, the model (m; e.g. architecture and weights) is saved and used for inference on new data.

2.2 Source Data

Inputs are derived from the EUMETSAT Numerical Weather Prediction Satellite Application Facility (NWP-SAF; Eresmaa and McNally, 2014) dataset. This contains a representative collection of 25 000 atmospheric profiles previously used to evaluate the performance of radiation models (e.g. Hocking et al., 2021; Hogan and Matricardi, 2020). Profiles were derived from 137-vertical-level global operational short-range ECMWF forecasts, correlated in more than one dimension (between quantities and spatially across levels), and extending from top of the atmosphere (TOA; 0.01 hPa; level 1) to the surface (bottom of the atmosphere; BOA; level 137). Inputs X consist of profiles of dry-bulb air temperature (T in K; Figure 2a), atmospheric pressure (p in hPa; Figure 2b), and cloud layer optical depth (τ_c; Figure 2c) with τ_c derived from other quantities to simplify the development of models as described in section 2.4. T, p, and τ_c are then used as inputs to the physical model (section 2.4) to compute outputs Y containing profiles of downwelling longwave radiation ($L^{\downarrow}$ in W m^{-2}; Figure 2d). Prior to being used, source data are shuffled at random and split into three batches of 10 000 profiles (40 %) for *training* (X_{train}), 5 000 (20 %) for *validation* ($X_{\text{validation}}$), and 10 000 (40 %) for *testing* (X_{testing}). As both copula and ML emulators work on two-dimensional data, datasets are reshaped to a matrix with samples as rows and flattened profiles per quantity as columns. To compute plots and statistics, data are reconstructed to their original shape.

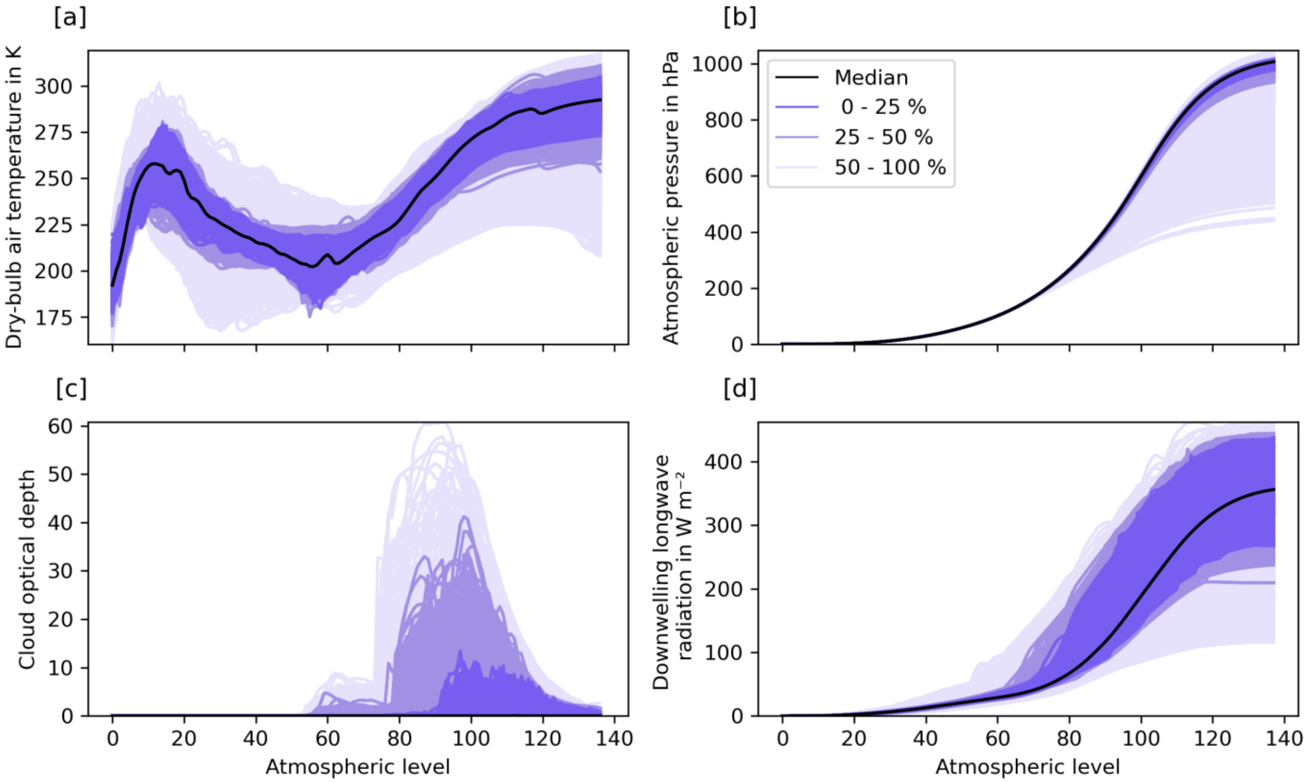

Figure 2. Profiles of (a) dry-bulb air temperature, (b) atmospheric pressure, (c) cloud layer optical depth from the NWP-SAF dataset (25 000 profiles; Eresmaa and McNally, 2014) and corresponding profiles of longwave radiation computed using the toy physical model described in section 2.4. Profiles are ordered using band depth statistics (López-Pintado and Romo, 2009) and shown for their most central (median) profile and grouped for the central 0–25 %, 25 – 50 % and 50 – 100 %.

Table 1. Profiles of input and output quantities used in this study. Input quantities are dry-bulb air temperature T, atmospheric temperature p and cloud layer optical depth τ_c. T and p are taken directly from the NWP-SAF dataset (Eresmaa and McNally, 2014). τ_c is derived from other quantities as described in section 2.4. The output quantity downwelling longwave radiation $L^{\downarrow}$ is computed using the physical model described in section 2.4. Atmospheric model levels are 137 for full levels (FL) and 138 for half levels (HL).

Symbol	Name	Unit	Dimension
(a) Inputs			
T	Dry-bulb air temperature	K	FL
p	Atmospheric pressure	Pa	FL
τ_c	Cloud optical depth	1	FL
(b) Output			
$L^{\downarrow}$	Downwelling longwave radiation	W m^{-2}	HL

2.3 Data generation

Data generation is used to generate additional input samples (here atmospheric profiles) to be fed to the physical (section 2.4) and ML (section 2.5) emulator. Optimally, synthetically generated data should resemble the observed data as closely as possible with respect to (i) the individual behaviour of variables (e.g. the dry-bulb air temperature at a specific level), and (ii) the dependence across variables and dimensions (e.g. the dry-bulb air temperature across two levels). Copulas are statistical models that allow these two aims to be disentangled (Trivedi and Zimmer, 2006; Joe, 2014) and to generate new samples that are statistically similar to the original data in terms of their individual behaviour and dependence.

2.3.1 Background on copula models

Suppose we want to generate synthetic data from a probabilistic model for n variables $Z_1, \dots, Z_n$. To achieve the first aim, we need to find appropriate *marginal cumulative distributions* $F, \dots, F_n$. A simple approach is to approximate them by the corresponding empirical distribution functions. To achieve the second aim, however, we need to build a model for the *joint distribution function* $F(z_1, \dots, z_n)$. The key result, Sklar's theorem (Sklar, 1959), states that any joint distribution function can be written as

$$F(z_1, \dots, z_n) = C\big(F_1(z_1), \dots, F_n(z_n)\big). \tag{1}$$

The function C is called copula and encodes the dependence between variables.

Copulas are distribution functions themselves. More precisely, if all variables are continuous, C is the joint distribution of the variables $U_1 = F_1(Z_1), \dots, U_n = F_n(Z_n)$. This fact facilitates estimation and simulation from the model. To estimate the copula function C, we (i) estimate marginal distributions $\widehat{F}_1, \dots, \widehat{F}_n$, (ii) construct *pseudo-observations* $\widehat{U}_1 = \widehat{F}_1(Z_1), \dots, \widehat{U}_n = \widehat{F}_n(Z_n)$, and (iii) estimate C from the pseudo-observations. Then, given estimated models $\widehat{C}, \widehat{F}_1, \dots, \widehat{F}_n$ for the copula and marginal distributions, we can generate synthetic data as follows:

1. Simulate random variables $U_1, \dots, U_n$ from the estimated copula $\widehat{C}$.
2. Define $Z_1 = \widehat{F}_1^{-1}(X_1), \dots, Z_n = \widehat{F}_n^{-1}(X_n)$.

2.3.2 Parametric copula families

In practice, it is common to only consider sub-families of copulas that are conveniently parametrized. There is a variety of such parametric copula families. Such families can be derived from existing models for multivariate distributions by inverting the equation of Sklar's theorem:

$$C(u_1, \dots, u_n) = F\left(F_1^{-1}(u_1), \dots, F_n^{-1}(u_n)\right). \tag{2}$$

For example, we can take F as the joint distribution function of a multivariate Gaussian and $F_1, \dots, F_n$ as the corresponding marginal distributions. Then equation 2 yields a model for the copula called *Gaussian copula,* which is parametrized by a correlation matrix. The Gaussian copula model subsumes all possible dependence structure in a multivariate Gaussian distribution. The benefit comes from the fact that we can combine a given copula with any type of marginal distributions, not just the ones the copula was derived from. That way, we can build flexible models with arbitrary marginal distributions and Gaussian-like dependence. The same principle applies to other multivariate distributions and many copula models have been derived, most prominently the Student's *t* copula and Archimedean families. A comprehensive list can be found in Joe (2014).

2.3.3 Vine copula models

When there are more than two variables ($n > 2$) the type of dependence structures these models can generate is rather limited. Gaussian and Student copulas only allow for symmetric dependencies between variables. Quite often, dependence is asymmetric, however. For example, dependence between Z_1 and Z_2 may be stronger when both variables take large values. Many Archimedean families allow for such asymmetries but require all pairs of variables to have the same type and strength of dependence.

Vine copula models (Aas et al., 2009; Czado, 2019) are a popular solution to this issue. The idea is to build a large dependence model from only two-dimensional building blocks. We can explain this with a simple example with just three variables Z_1, Z_2, Z_3. We can model the dependence between Z_1 and Z_2 by a two-dimensional copula $C_{1,2}$ and the dependence between Z_2 and Z_3 by another, possibly different, copula $C_{2,3}$. These two copulas already contain some information about the dependence between Z_1 and Z_3, the part of the dependence that is induced by Z_2. The missing piece is the dependence between Z_1 and Z_3 after the effect of Z_2 has been removed. Mathematically, this is the conditional dependence between Z_1 and Z_3 given Z_2 and can be modeled by yet another two-dimensional copula $C_{1,3|2}$. The principle is easily extended to an arbitrary number of variables $Z_1, \ldots, Z_n$. Algorithms for simulation and selecting the right conditioning order and parametric families for each (conditional) pair are given in Dißman et al. (2013).

Because all two-dimensional copulas can be specified independently, such models are extremely flexible and allow for highly heterogenous dependence structures. Using parametric models for pair-wise dependencies remain a limiting factor, however. If necessary, it is also possible to use nonparametric models for the two-dimensional building blocks. Here, the joint distribution of pseudo-observations $\widehat{U}_1$, $\widehat{U}_2$ is estimated by a suitable kernel density estimator (see Nagler et al., 2017).

2.3.4 Implementation

Here we use the Synthia software (Meyer and Nagler, 2021) to fit three different copula types: Gaussian, Vine-parametric, Vine-nonparametric. Vine-parametric fits a parametric model for each pair in the model from the catalogue of Gaussian, Student, Clayton, Gumbel, Frank, Joe, BB1, BB6, BB7, BB8 copula families and their rotations (see Joe, 2014, for details on these families) using the AIC criterion. Vine-nonparametric uses transformation local quadratic likelihood fitting as explained in Nagler et al. (2017). Each copula model is fitted to the training set X_{train}. To evaluate the impact of copula-augmented datasets, we generate synthetic profiles with augmentation factors of 1x, 5x, and 10x the number of profiles in the source training dataset (i.e. 10 000 profiles). These are then used to create *augmented* versions of training datasets, defined as X'_{train}, each containing the source plus the synthetically generated profiles (i.e. with 20 000 profiles, or double the amount of training data, for 1x augmentation factor, and 60 000 and 110 000 profiles for 5x and 10x augmentation factors respectively). As the generation of new profiles with copula models is random, the generation is also repeated 10 times for each case to allow for meaningful statistics to be computed.

2.4 Target generation

Target generation is used to generate outputs Y from corresponding inputs X using a physical model. Here, outputs Y are computed using a simple toy model based on Schwarzschild's equation (e.g. Petty, 2006) to estimate the downwelling longwave radiation under the assumption that atmospheric absorption does not vary with wavelength as:

$$\frac{dL^{\downarrow}}{dz} = a(z)\left[B(z\,) - L^{\downarrow}\right] \tag{3}$$

where z is the geometric height, B is the Planck function at level z (i.e. $B = \sigma_{\mathrm{SB}} T^4$, where σ_{SB} is the Stefan-Boltzmann constant; giving the flux in W m^{-2} emitted from a horizontal black body surface), and a is the rate at which radiation is intercepted/emitted. A common approximation is to treat longwave radiation travelling at all angles as if it were all travelling with a zenith angle of 53 degrees (Elsasser, 1942): in this case $a = D\beta_e$ where β_e is the extinction coefficient of the medium, and $D = 1/\cos(53) = 1.66$ is the diffusivity factor, which accounts for the fact that the effective path length of radiation passing through a layer of thickness Δz is on average $1.66\Delta z$ due to the multiple different angles of propagation. In the context of ML, $a(z)$ and $B(z)$ are known and $F(z)$ is to be predicted. Here we use the difference in two atmospheric pressures expressed in sigma coordinates ($\Delta\sigma$, where σ is the pressure p at a particular height divided by the surface pressure p_0) instead of z. The layer optical depth $\tau = \beta_e \Delta z$ is calculated from the total-column gas optical depth τ_g and cloud layer optical depth τ_c as $\tau = \tau_c + \tau_g\, \Delta\sigma_i$, since $\Delta\sigma$ is the fraction of mass of the full atmospheric column in layer i. Then, as the downwelling flux at the top of the atmosphere is 0, the equation is discretized as follows assuming B and a are constant within a layer:

$$L^{\downarrow}{}_{i-1/2} = L^{\downarrow}{}_{i+1/2}\,(1 - \epsilon_i) + B_i \epsilon_i, \tag{4}$$

where B_i is the Planck function of layer i, $\epsilon_i = 1 - e^{-a_i \Delta z} = 1 - e^{D\tau}$ is the emissivity of layer i, $L^{\downarrow}{}_{i+1/2}$ is the downwelling flux at the top of layer i, and $L^{\downarrow}{}_{i-1/2}$ is the downwelling flux at the bottom of layer i. We compute $L^{\downarrow}$ in W m^{-2} from T in K, p in Pa, and τ_c using the source X or augmented X' data. To reduce, and thus simplify, the number of quantities used in the physical model and ML emulator (section 2.5), τ_c is pre-computed and used instead of vertical profiles of liquid and ice mixing ratios (q_l and q_l) and effective radius (r_l and r_l in m) as $\frac{3}{2}\frac{\Delta p}{g}\left(\frac{q_l}{\rho_l r_l} + \frac{q_i}{\rho_i r_i}\right)$, where ρ_l is the density of liquid water (1 000 kg m^{-3}), ρ_i is the density of ice (917 kg m^{-3}), g is the standard gravitational acceleration (9.81 m s^{-2}). For τ_g we use a constant value of 1.7 determined by minimizing the absolute error between profiles computed with this simple model and the comprehensive atmospheric radiation scheme ecRad (Hogan and Bozzo, 2018).

2.5 Estimator training

As the goal of this paper is to determine whether the use of synthetic data improves the prediction of ML emulators, here we implement a simple feedforward neural network (FNN). FNNs are one of the simplest and most common neural networks used in ML (Goodfellow et al., 2016) and have been previously used for similar weather and climate applications (e.g. Chevallier et al., 1998; Krasnopolsky et al., 2002). FNNs are composed of artificial neurons (conceptually derived from biological neurons) connected with each other where information moves forward from the input nodes, through hidden nodes. The multilayer perceptron (MLP) is a type of FNN composed of at least three layers of nodes: an input layer, a hidden layer, and an output layer with all but the input nodes using a nonlinear activation function.

Here we implement a simple an MLP consisting of 3 hidden layers with 512 neurons each. This is implemented in TensorFlow (Abadi et al., 2015), and configured with elu activation function, Adam optimizer, Huber loss, 1 000 epochs limit, and early stopping with patience of 25 epochs. The MLP is trained with profiles of dry-bulb air temperature (T in K; Figure 2a), atmospheric pressure (p in hPa; Figure 2b), and layer cloud optical depth (τ_c; Figure 2c) as inputs, and profiles of longwave downwelling longwave radiation ($L^{\downarrow}$ in W m^{-2}; Figure 2d) as outputs. Inputs are normalized and both inputs and outputs are flattened into feature vectors. The baseline case (Figure 1 A) uses 10 000 input profiles without data augmentation for training and copula-based cases (Figure 1 B) use either 20 000, 60 000, or 110 000 profiles. The validation dataset $Y_{\mathrm{validation}}$ of 5 000 profiles is used as input for the early stopping mechanism while the test dataset Y_{test} of 10 000 profiles is used to compute error statistics described in section 3.2. Because of the stochastic nature of MLPs, training (and inference) is repeated 10 times for each case to allow for meaningful statistics to be computed. Given that the generation of random profiles in the case of augmented datasets is also repeated 10 times (see section 2.3.4), all cases that also use data generation comprise of 100 iterations in total (i.e. for each data generation run, the estimator is run 10 times).

3 Results

3.1 Copula

The quality of synthetic data is assessed in terms of summary statistics (e.g. Seitola et al., 2014) between the training X_{train} and copula-simulated X'_{train} datasets. For each copula type we compute a vector of summary statistics $S_i = f(\mathbf{P}_i)$ where f is the statistic function and $\mathbf{P}_i = \mathbf{D}\boldsymbol{w}_i$, with $\mathbf{D}$ a matrix of flattened source or simulated data and $\boldsymbol{w}$ a vector of random numbers from the ith iteration. Summary statistics are computed for mean, variance, and quantiles, iterating 100 times to allow for meaningful statistics to be computed. As we consider random linear combinations of variables in source and copula-generated data, we expect these summaries to coincide only if both marginal distributions and dependence between variables are captured. Figure 3 shows scatterplots of summary statistics S_i for (a) mean, (b) variance, (c) standard deviation, and (d) 10 %, (e) 50 % and (f) 90 % quantiles. Real NWP-SAF data are shown on the x-axis and copula-generated data on the y-axis with each point corresponding to a random projection as described earlier (100 points in total). For a perfect copula model, we expect all points to fall on the main diagonal where $x = y$. Figure 3 shows that for all copula models, synthetically-generated data are close to the real data, with larger errors in variance and standard deviation.

Qualitatively, we can evaluate copula-generated profiles in terms of their overall shape and smoothness across multiple levels, and range and density at each level. To this end we plot a side-by-side comparison of source (Figure 4, left panel) and Gaussian-copula generated(Figure 4, right panel) profiles showing the median profile and random selection of 90 profiles grouped in batches of 3 (i.e. each having 30 profiles) for the central 0-25 % and outer 25-50 %, 50-100 % quantiles, calculated with band depth statistics (López-Pintado and Romo, 2009). Simulated profiles of dry-bulb air temperature (Figure 4b) appear less smooth than the real ones across levels (Figure 4a); however, both density and range are simulated well at each level.

Simulated profiles of atmospheric pressure (Figure 4d) are simulated well: they are smooth across all levels with similar range and density (Figure 4c). The highly non-Gaussian and spikey profiles of cloud optical depth (Figure 4e) make qualitative comparisons difficult, however simulated profiles (Figure 4f) have similar range and density, with high density for low values and most range between levels 80 and 120.

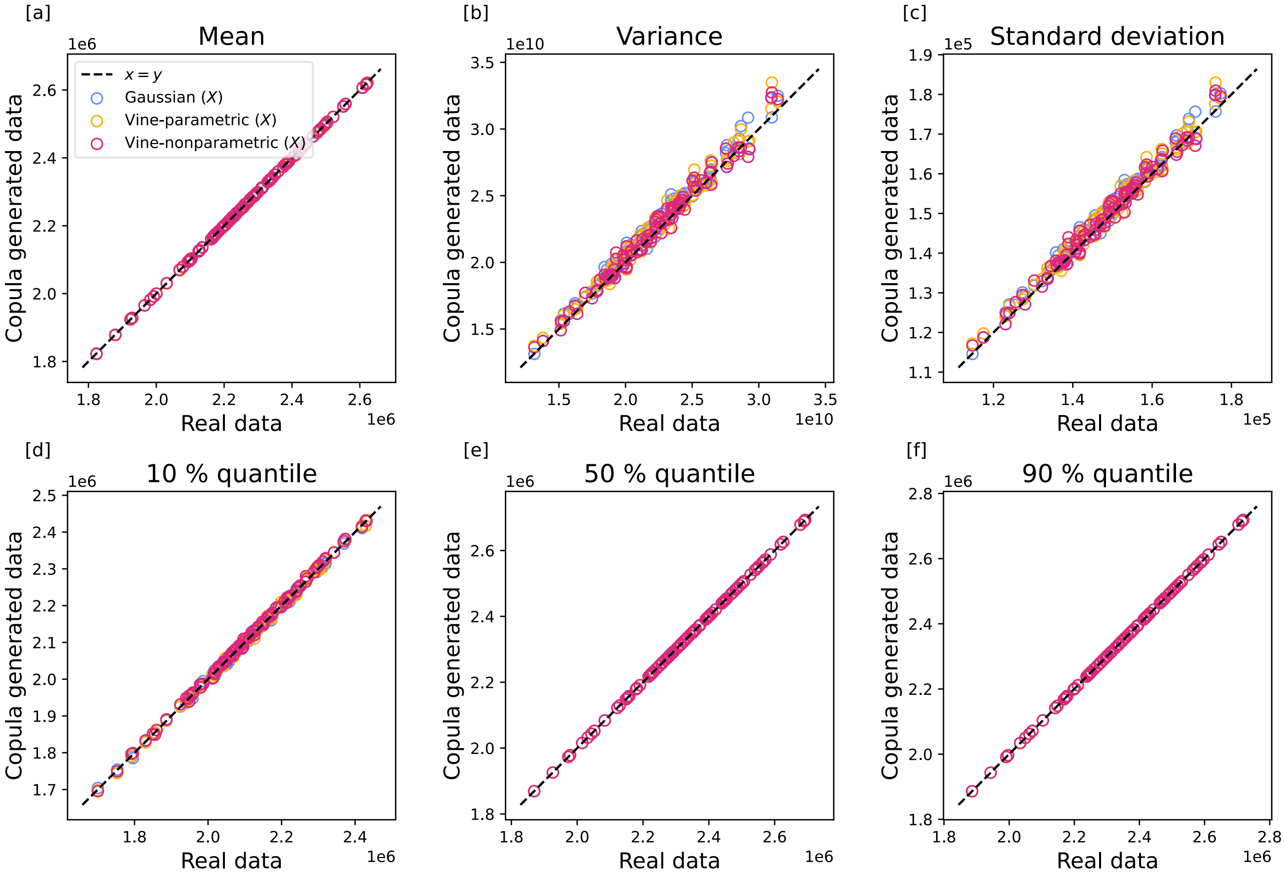

Figure 3. Summary statistics $\boldsymbol{S_i}$ from 100 iterations for (a) mean, (b) variance, (c) standard deviation, and (d) 10 %, (e) 50 %, and (f) 90 % quantiles. Each point corresponds to a statistic for single iteration in arbitrary units. The x axis represents the projection of the true data $\boldsymbol{X}_{\mathbf{train}}$ while the y axis that of the copula generated data $\boldsymbol{X}'_{\mathbf{train}}$. Results reported for Gaussian, Vine-parametric, Vine-nonparametric copulas (see legend for keys).

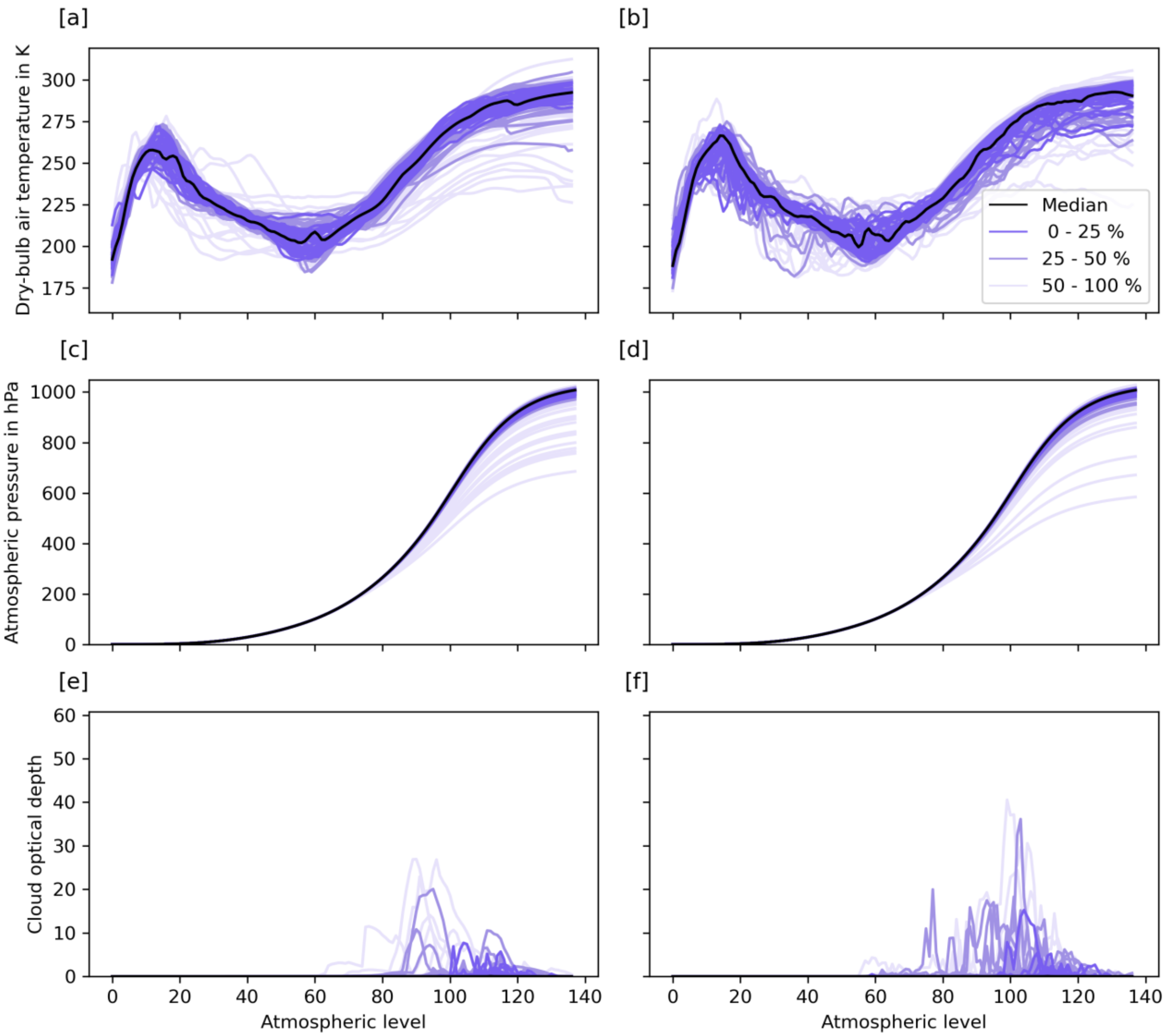

Figure 4. Profiles of (left) real and (right) Gaussian copula-generated data of (a-b) dry-bulb air temperature, (c-d) atmospheric pressure, (e-f) cloud optical depth. Median profile shown in black and random selection of 90 profiles grouped in batches of 3 (i.e. each having 30 profiles) for the central 0-25 % and outer 25-50 %, 50-100 % calculated with band depth statistics (López-Pintado and Romo, 2009).

265 3.2 Machine learning

To evaluate whether ML emulators trained on augmented datasets have lower prediction errors compared to the baseline, here we use the test dataset X_{test} of 10 000 profiles defined in section 2.2. Statistics are computed based on a vector of random variables representing differences $\boldsymbol{d} = (d_1, \dots, d_i)$ between the physically predicted baseline Y_{test} and ML emulated Y'_{test} (i.e. $\boldsymbol{d} = Y_{\text{test}} - Y'_{\text{test}}$) for $1, \dots, N$ profiles. From this, the mean bias error (MBE $=\frac{1}{N}\sum_{i=1}^{N} d_i$) and mean absolute error 270 (MAE $= \frac{1}{N}\sum_{i=1}^{N}|d_i|$) are computed.

Boxplots of MBE and MAE are shown in Figure 5. Summary MBE and MAE for ML emulators with lowest MAE using an augmentation factor of 10x are reported in Table 2. A qualitative side-by-side comparison of baseline and ML-predicted profiles using Gaussian copula-generated profiles with augmentation factor of 10x and are shown in Figure 6.

MBEs (Figure 5a) across all copula types and augmentation factors are generally improved, with median MBEs and respective spreads decreasing with larger augmentation factors. Overall, the Gaussian copula model performs better than the Vine-parametric or Vine-nonparametric models. MAEs (Figure 5b) show a net improvement from the baseline across all copula models and augmentation factors. When using an augmentation factor of 1x (i.e. with double the amount of training data), the median MAE is reduced to approximately 1.1 W m^{-2} from a baseline of approximately 1.4 W m^{-2} and further reduced with increasing augmentation factors. In the best case, corresponding to an augmentation factor of 10x (i.e. with an additional 100 000 synthetic profiles), the copula and ML emulator combinations with lowest MAE (Table 2) show that MBEs are reduced from a baseline of 0.08 W m^{-2} to -0.02 and -0.05 W m^{-2} for Gaussian and Vine-nonparametric respectively but increased to 0.10 W m^{-2} for Vine-parametric. MAEs are reduced from a baseline of 1.17 W m^{-2} to 0.45, 0.56 and 0.44 W m^{-2} for Gaussian, Vine-parametric, Vine-nonparametric copula type respectively.

The ML training configuration with the lowest overall MBE and MAE combination during inference corresponds to a Gaussian copula and augmentation factor of 10x (Table 2). Errors between the physically predicted Y_{test} and ML predicted Y'_{test} are shown for the baseline (Figure 6a) and Gaussian copula (Figure 6b). These are shown grouped by their central 0-25 % and outer 25-50 %, 50-100 %. Qualitatively most ML generated profiles show improvements from to the baseline. For the most central 25 % profiles are within ±20 W m^{-2} for the Gaussian copula case, and about ±40 W m^{-2} for the baseline case. Near surface errors (levels 130-BOA) are reduced to approximately ±5 W m^{-2} from approximately ±10 W m^{-2}.

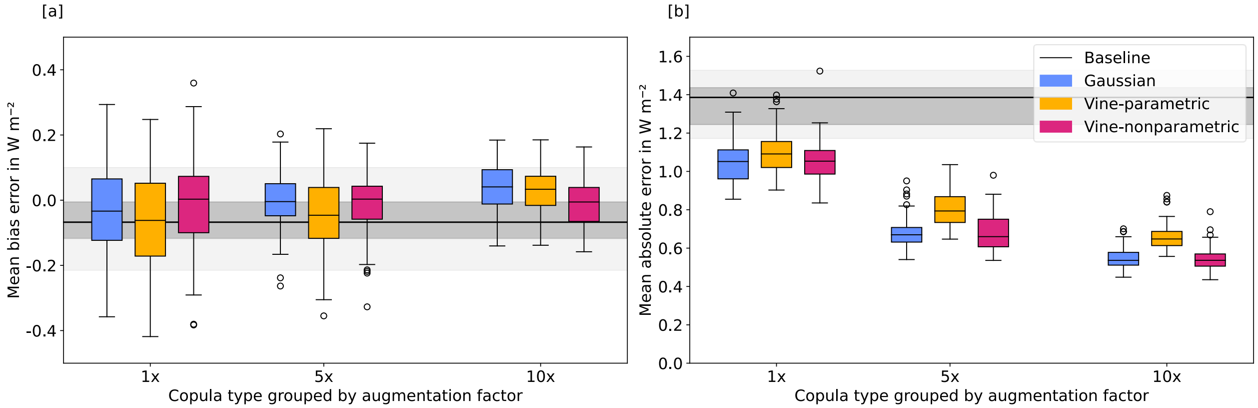

Figure 5. Errors grouped by different copula types (Gaussian: blue, Vine-parametric: yellow, Vine-nonparametric: red) and augmentation factors (1x, 5x, 10x) for the mean bias error (MBE; a) and mean absolute error (MAE; b). The median for the baseline case is shown in black and the range shaded in grey.

Table 2. Mean bias error (MBE) and mean absolute error (MAE) for baseline, and copula and ML emulator combination with lowest MAE. Baseline case trained using 10 000 real profiles and copula cases trained using augmented dataset of 110 000 profiles (10 000 real and 100 000 synthetic), i.e. with an augmentation factor of 10x.

Case name	MBE in W m^{-2}	MAE in W m^{-2}
Baseline	0.08	1.17
Gaussian	**-0.02**	0.45
Vine-parametric	0.10	0.56
Vine-nonparametric	-0.05	**0.44**

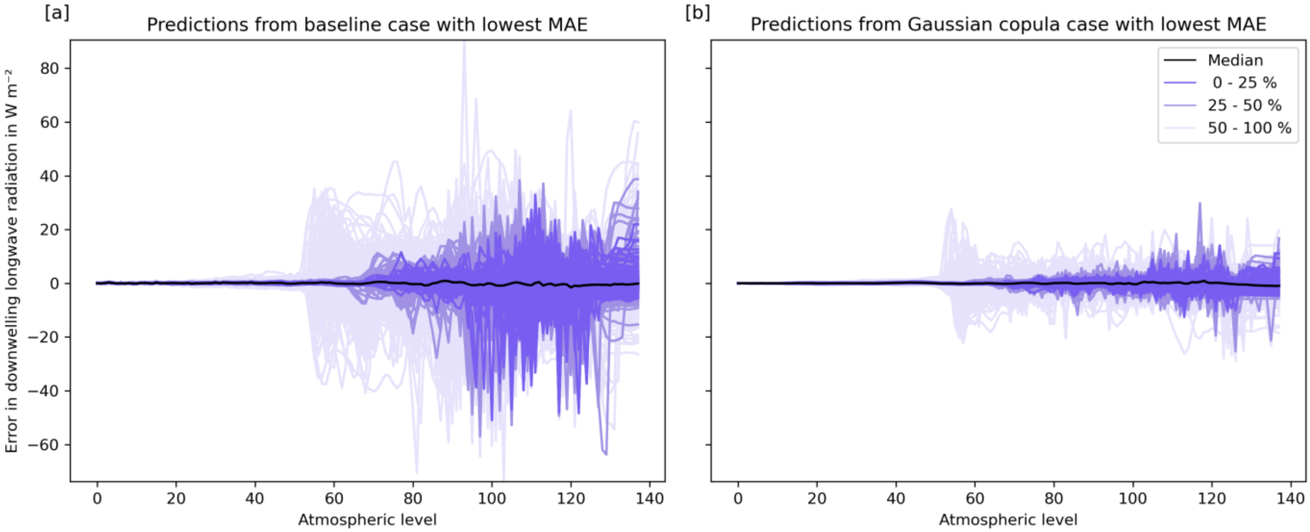

Figure 6. Prediction errors for (a) baseline emulator and (b) data augmentation emulator using 110 000 profiles (10x augmentation factor; Gaussian copula). The median (most central) profile is shown in black and the most central 25 %, and outer 25 – 50 % and 50 – 100 % profiles are computed using band depth statistics and shown in shades of blue.

4 Discussion and conclusion

Results from a qualitative comparison of synthetically generated profiles (Figure 4) shows that synthetic profiles tend to be less smooth and noisier than the real ones. Nevertheless the machine learning evaluation shows that errors for emulators trained with augmented datasets are cut by up to 75 % for the mean bias error (from 0.08 to -0.02 W m^{-2};Table 2) and by up to 62 % for the mean absolute error (from 1.17 to 0.44 W m^{-2};Table 2).

Here we show how copula-based models may be used to improve the prediction of ML emulators by generating augmented datasets containing statistically similar profiles in terms of their individual behaviour and dependence across variables (e.g.

dry-bulb air temperature at a specific level and across several levels). Although the focus of this paper is to evaluate copula-based data generation models to improve predictions of ML emulators, we speculate that same or similar methods of data generation have the potential to be used in several other ML-related applications such as to: (i) test architectures (e.g. instead of cross validation, one may generate synthetic datasets of different size to test the effect of sample size on different ML architectures); (ii) generate data for un-encountered conditions (e.g. for climate change scenarios, by extending data ranges, or relaxing marginal distributions); (iii) data compression (e.g. by storing reduced parameterized versions of the data if the number of samples is much larger than the number of features).

Although so far, we have only highlighted the main benefits of copula-based models, several limiting factors should also be considered. A key factor for very high-dimensional data is that both Gaussian and Vine copula models scale quadratically in the number of features – in terms of both memory and computational complexity. This can be alleviated by imposing structural constraints on the model, for example using structured covariance matrix or truncating the vine after a fixed number of trees. However, this limits their flexibility and adds some arbitrariness to the modelling process. A second drawback compared to GANs is that the model architecture cannot be tailored to a specific problem, like images. For such cases, a preliminary data compression step as in Tagasovska et al. (2019) may be necessary.

As highlighted here, data augmentation for ML emulators may be of particular interest to scientists and practitioners looking to achieve a better generalization of their ML emulators (i.e. synthetic data may act as a regularizer to reduce overfitting; Shorten and Khoshgoftaar, 2019). Although a comprehensive analysis of prediction errors using different ML architectures is out of scope, our work is a first step towards further research in this area. Moreover, although we did not explore the generation of data for un-encountered conditions (e.g. by extending the range of air temperature profiles while keeping a meaningful dependency across other quantities and levels), the use of copula-based synthetic data generation may prove useful to make emulators more resistant to outliers (e.g. in climate change scenario settings) and should be investigated in future research.

Acknowledgments

We thank three anonymous reviewers for their useful comments and feedback.

Code and data availability

The key software used in this paper are Synthia (available under MIT licence at https://github.com/dmey/synthia) and TensorFlow (available under Apache 2.0 licence at https://www.tensorflow.org/). Software, data, and tools are archived with a Singularity (Kurtzer et al., 2017) image deposited on Zenodo as described in the scientific reproducibility section of Meyer et al. (2020). Users wishing to download or reproduce the results described in this paper can download the archive at https://doi.org/10.5281/zenodo.5081927 and optionally run Singularity on their local or remote systems.

Author contribution

Conceptualization, D.M.; Data curation, D.M.; Formal analysis, D.M., T.N.; Investigation, D.M.; Methodology, D.M., T.N., R.H.; Software, D.M.; Resources, D.M.; Validation, D.M.; Visualization, D.M.; Writing – original draft preparation, D.M., T.N.; Writing – review & editing, D.M., T.N., R.H..

Competing interests

The authors declare no conflict of interest.

References

Aas, K., Czado, C., Frigessi, A., and Bakken, H.: Pair-copula constructions of multiple dependence, Insurance: Mathematics and Economics, 44, 182–198, https://doi.org/10/bmnchh, 2009.

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

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

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

Cheruy, F., Chevallier, F., Morcrette, J.-J., Scott, N. A., and Chédin, A.: Une méthode utilisant les techniques neuronales pour le calcul rapide de la distribution verticale du bilan radiatif thermique terrestre, 322, 665–672, https://hal.archives-ouvertes.fr/hal-02954375, 1996.

Chevallier, F., Ruy, F. C., Scott, N. A., and Din, A. C.: A Neural Network Approach for a Fast and Accurate Computation of a Longwave Radiative Budget, 37, 13, https://doi.org/10.1175/1520-0450(1998)037<1385:ANNAFA>2.0.CO;2, 1998.

Chevallier, F., Morcrette, J.-J., Chéruy, F., and Scott, N. A.: Use of a neural-network-based long-wave radiative-transfer scheme in the ECMWF atmospheric model, Q.J.R. Meteorol. Soc., 126, 761–776, https://doi.org/10.1002/qj.49712656318, 2000.

Czado, C.: Analyzing Dependent Data with Vine Copulas: A Practical Guide With R, Springer International Publishing, Cham, https://doi.org/10.1007/978-3-030-13785-4, 2019.

Dißmann, J., Brechmann, E. C., Czado, C., and Kurowicka, D.: Selecting and estimating regular vine copulae and application to financial returns, Computational Statistics & Data Analysis, 59, 52–69, https://doi.org/10/ghfn5n, 2013.

Elsasser, W. M.: Heat transfer by infrared radiation in the atmosphere, Blue Hill Meteorological Observatory, 1942.

Eresmaa, R. and McNally, A. P.: Diverse profile datasets from the ECMWF 137-level short-range forecasts, EUMETSAT Satellite Application Facility (NWP SAF), 2014.

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

Goodfellow, I., Bengio, Y., and Courville, A.: Deep learning, MIT Press, Cambridge, 775 pp., 2016.

Hocking, J., Vidot, J., Brunel, P., Roquet, P., Silveira, B., Turner, E., and Lupu, C.: A new gas absorption optical depth parameterisation for RTTOV version 13, Geosci. Model Dev., 14, 2899–2915, https://doi.org/10.5194/gmd-14-2899-2021, 2021.

Hogan, R. J. and Bozzo, A.: A Flexible and Efficient Radiation Scheme for the ECMWF Model, J. Adv. Model. Earth Syst., 10, 1990–2008, https://doi.org/10.1029/2018MS001364, 2018.

Hogan, R. J. and Matricardi, M.: Evaluating and improving the treatment of gases in radiation schemes: the Correlated K-Distribution Model Intercomparison Project (CKDMIP), 13, https://doi.org/10.5194/gmd-13-6501-2020, 2020.

Huntingford, C., Jeffers, E. S., Bonsall, M. B., Christensen, H. M., Lees, T., and Yang, H.: Machine learning and artificial intelligence to aid climate change research and preparedness, Environ. Res. Lett., 14, 124007, https://doi.org/10.1088/1748-9326/ab4e55, 2019.

Joe, H.: Dependence Modeling with Copulas, 1st Edition., Chapman and Hall/CRC, https://doi.org/10.1201/b17116, 2014.

Krasnopolsky, V. M. and Lin, Y.: A Neural Network Nonlinear Multimodel Ensemble to Improve Precipitation Forecasts over Continental US, Advances in Meteorology, 2012, 1–11, https://doi.org/10.1155/2012/649450, 2012.

Krasnopolsky, V. M., Chalikov, D. V., and Tolman, H. L.: A neural network technique to improve computational efficiency of numerical oceanic models, 21, https://doi.org/10.1016/S1463-5003(02)00010-0, 2002.

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

Krasnopolsky, V. M., Fox-Rabinovitz, M. S., and Belochitski, A. A.: Using Ensemble of Neural Networks to Learn Stochastic Convection Parameterizations for Climate and Numerical Weather Prediction Models from Data Simulated by a Cloud Resolving Model, Advances in Artificial Neural Systems, 2013, 1–13, https://doi.org/10.1155/2013/485913, 2013.

Kurtzer, G. M., Sochat, V., and Bauer, M. W.: Singularity: Scientific containers for mobility of compute, PLoS ONE, 12, e0177459, https://doi.org/10.1371/journal.pone.0177459, 2017.

López-Pintado, S. and Romo, J.: On the Concept of Depth for Functional Data, Journal of the American Statistical Association, 104, 718–734, https://doi.org/10.1198/jasa.2009.0108, 2009.

Meyer, D. and Nagler, T.: Synthia: Multidimensional synthetic data generation in Python, https://doi.org/10.21105/joss.02863, 2021.

Meyer, D., Schoetter, R., Riechert, M., Verrelle, A., Tewari, M., Dudhia, J., Masson, V., Reeuwijk, M., and Grimmond, S.: WRF-TEB: Implementation and Evaluation of the Coupled Weather Research and Forecasting (WRF) and Town Energy Balance (TEB) Model, J. Adv. Model. Earth Syst., 12, https://doi.org/10.1029/2019MS001961, 2020.

Meyer, D., Hogan, R. J., Dueben, P. D., and Mason, S. L.: Machine Learning Emulation of 3D Cloud Radiative Effects, https://arxiv.org/abs/2103.11919, 2021.

Nagler, T., Schellhase, C., and Czado, C.: Nonparametric estimation of simplified vine copula models: comparison of methods, 5, 99–120, https://doi.org/10/ghfn5m, 2017.

Nowack, P., Braesicke, P., Haigh, J., Abraham, N. L., Pyle, J., and Voulgarakis, A.: Using machine learning to build temperature-based ozone parameterizations for climate sensitivity simulations, Environ. Res. Lett., 13, 104016, https://doi.org/10.1088/1748-9326/aae2be, 2018.

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

Patki, N., Wedge, R., and Veeramachaneni, K.: The Synthetic Data Vault, in: 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Montreal, QC, Canada, 399–410, https://doi.org/10.1109/DSAA.2016.49, 2016.

Petty, G. W.: A First Course in Atmospheric Radiation, End of Line Clearance Book, Madison, Wis, 459 pp., 2006.

Rasp, S. and Lerch, S.: Neural Networks for Postprocessing Ensemble Weather Forecasts, Mon. Wea. Rev., 146, 3885–3900, https://doi.org/10.1175/MWR-D-18-0187.1, 2018.

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

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, N., and Prabhat: Deep learning and process understanding for data-driven Earth system science, Nature, 566, 195–204, https://doi.org/10.1038/s41586-019-0912-1, 2019.

Seitola, T., Mikkola, V., Silen, J., and Järvinen, H.: Random projections in reducing the dimensionality of climate simulation data, Tellus A: Dynamic Meteorology and Oceanography, 66, 25274, https://doi.org/10/ghfpd8, 2014.

Shorten, C. and Khoshgoftaar, T. M.: A survey on Image Data Augmentation for Deep Learning, J Big Data, 6, 60, https://doi.org/10.1186/s40537-019-0197-0, 2019.

Sklar, M.: Fonctions de repartition an dimensions et leurs marges, 8, 229–231, 1959.

Tagasovska, N., Ackerer, D., and Vatter, T.: Copulas as high-dimensional generative models: Vine copula autoencoders, in: Advances in neural information processing systems 32, edited by: Wallach, H., Larochelle, H., Beygelzimer, A., dAlché-Buc, F., Fox, E., and Garnett, R., Curran Associates, Inc., 6528–6540, 2019.

Trivedi, P. K. and Zimmer, D. M.: Copula Modeling: An Introduction for Practitioners, FNT in Econometrics, 1, 1–111, https://doi.org/10/fwkzd3, 2006.

Ukkonen, P., Pincus, R., Hogan, R. J., Nielsen, K. P., and Kaas, E.: Accelerating radiation computations for dynamical models with targeted machine learning and code optimization, 12, https://doi.org/10.1029/2020ms002226, 2020.

Veerman, M. A., Pincus, R., Stoffer, R., van Leeuwen, C. M., Podareanu, D., and van Heerwaarden, C. C.: Predicting atmospheric optical properties for radiative transfer computations using neural networks, Phil. Trans. R. Soc. A., 379, 20200095, https://doi.org/10.1098/rsta.2020.0095, 2021.

Wan, Z., Zhang, Y., and He, H.: Variational autoencoder based synthetic data generation for imbalanced learning, in: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017 IEEE Symposium Series on Computational Intelligence (SSCI), Honolulu, HI, 1–7, https://doi.org/10.1109/SSCI.2017.8285168, 2017.

Xu, L. and Veeramachaneni, K.: Synthesizing Tabular Data using Generative Adversarial Networks, 2018.