Supporting Information for "Use of neural networks for stable, accurate and physically consistent parameterization of subgrid atmospheric processes with good performance at reduced precision"

Janni Yuval[1], Paul A. O'Gorman[1]and Chris N. Hill[1]

[1]Department of Earth, Atmospheric and Planetary Sciences, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA

Contents of this file

1. Text S1 to S4
2. Figures S1 to S5
3. Tables S1 to S2

Here we present further information on the equations of motion for thermodynamic and moisture variables used in SAM and how these equations are modified when we omit precipitating water as a prognostic variable following the approach in Yuval and O'Gorman (2020) hereinafter referred to as YOG20 (text S1). We also describe the training data, training procedure and architecture of the neural networks (NNs) and their implementation in SAM (text S2), we show that the NN parameterization conserves the frozen

moist static energy (text S3), and we compare the offline performance of the NN and RF parameterizations (text S4).

Text S1. Equations of motion used for thermodynamic and moisture variables

The equations for these variables in SAM may be written as (Khairoutdinov & Randall, 2003)

$$\frac{\partial h_{\rm L}}{\partial t} = -\frac{1}{\rho_0}\frac{\partial}{\partial x_i}(\rho_0 u_i h_{\rm L} + F_{h_{\rm L} i}) - \frac{1}{\rho_0}\frac{\partial}{\partial z}(L_{\rm p} P_{\rm tot} + L_{\rm s} S) + (h_{\rm L})^{\rm tend}_{\rm rad}, \tag{S1}$$

$$\frac{\partial q_{\rm T}}{\partial t} = -\frac{1}{\rho_0}\frac{\partial}{\partial x_i}(\rho_0 u_i q_{\rm T} + F_{q_{\rm T} i}) + \frac{1}{\rho_0}\frac{\partial}{\partial z}(S) + (q_{\rm T})^{\rm tend}_{\rm mic}, \tag{S2}$$

$$\frac{\partial q_{\rm p}}{\partial t} = -\frac{1}{\rho_0}\frac{\partial}{\partial x_i}(\rho_0 u_i q_{\rm p} + F_{q_{\rm p} i}) + \frac{1}{\rho_0}\frac{\partial}{\partial z}(P_{\rm tot}) - (q_{\rm T})^{\rm tend}_{\rm mic}, \tag{S3}$$

where $h_{\rm L}$ is the liquid/ice water static energy ($= c_{\rm p}T + gz - L_{\rm c}(q_c + q_{\rm r}) - L_{\rm s}(q_{\rm i} + q_{\rm s} + q_{\rm g})$, and $q_{\rm c}$, $q_{\rm i}$, $q_{\rm r}$, $q_{\rm s}$ and $q_{\rm g}$ are mixing ratios of cloud water, cloud ice, rain, snow and graupel, respectively); $q_{\rm p}$ is the total precipitating water (rain + graupel + snow) mixing ratio; $q_{\rm T}$ is the non-precipitating water (water vapor + cloud water + cloud ice) mixing ratio; $\rho_0(z)$ is the reference density profile; $u_i = (u, v, w)$ is the three-dimensional wind; F_{Bi} is the diffusive flux of variable B; $P_{\rm tot}$ is the precipitation mass flux (due to rain, graupel and snow); S is the sedimentation mass flux; $(h_{\rm L})^{\rm tend}_{\rm rad}$ is the tendency due to radiative heating; $(q_{\rm T})^{\rm tend}_{\rm mic}$ is the non-precipitating water mixing ratio tendency due to autoconversion, collection, aggregation, evaporation and sublimation of precipitation. $L_{\rm c}$, $L_{\rm f}$ and $L_{\rm s}$ are the latent heat of condensation, fusion and sublimation, respectively; $L_{\rm p} = L_{\rm c} + L_{\rm f}(1 - \omega_{\rm p})$ is the effective latent heat associated with precipitation, where $\omega_{\rm p}$ is the precipitation partition function determining the ratio between ice and liquid phases.

The precipitating water mixing ratio q_p is a variable that varies on short time scales and is often not used as a prognostic variable in climate models, which typically have a coarser grid and larger time step than the high-resolution simulation. Therefore, we do not include q_p in the NN parameterization scheme that is presented in this study to make our results more generally applicable. Following YOG20, we define a new prognostic energy variable (H_L) that does not include the precipitating water (q_p):

$$H_\mathrm{L} = c_\mathrm{p}T + gz - L_\mathrm{c}q_\mathrm{c} - L_\mathrm{s}q_\mathrm{i}, \tag{S4}$$

Neglecting variations of L_p in the horizontal and in time (which are small compared to the vertical variations in L_p), allows us to write the prognostic equation for H_L as

$$\begin{aligned}\frac{\partial H_\mathrm{L}}{\partial t} = &-\frac{1}{\rho_0}\frac{\partial}{\partial x_i}(\rho_0 u_i H_\mathrm{L}) - \frac{1}{\rho_0}\frac{\partial}{\partial z}(L_\mathrm{s}S) - L_\mathrm{p}(q_\mathrm{T})^\mathrm{tend}_\mathrm{mic} + (h_\mathrm{L})^\mathrm{tend}_\mathrm{rad} \\ &- \frac{1}{\rho_0}\frac{\partial F_{H_\mathrm{L}i}}{\partial x_i} + \frac{1}{\rho_0}\frac{\partial L_\mathrm{p}}{\partial z}(\rho_0 w q_\mathrm{p} + F_{q_\mathrm{p}z} - P_\mathrm{tot})\end{aligned} \tag{S5}$$

where $F_{H_\mathrm{L}i} = F_{h_\mathrm{L}i} + L_\mathrm{p}F_{q_\mathrm{p}i}$, and the last term on the right hand side results from heating due to phase changes of precipitation. We define the sum of the total radiative heating and the heating from phase changes of precipitation as $(H_\mathrm{L})^\mathrm{tend}_\mathrm{rad-phase} = (h_\mathrm{L})^\mathrm{tend}_\mathrm{rad} + \frac{1}{\rho_0}\frac{\partial L_\mathrm{p}}{\partial z}(\rho_0 w q_\mathrm{p} + F_{q_\mathrm{p}z} - P_\mathrm{tot})$.

To find an expression for surface precipitation we assume that at coarse resolution (when the NN parameterization is used) we can neglect surface diffusive and horizontal fluxes of q_p and the time derivative of q_p in Equation (S3). Using these assumptions and vertically integrating Equation (S3) over the column gives an expression for the surface precipitation rate due to rain, graupel and snow:

$$P_\mathrm{tot}(z=0) = \int_0^\infty \rho_0 (q_\mathrm{T})^\mathrm{tend}_\mathrm{mic} dz. \tag{S6}$$

Following conventions used in SAM, we add any sedimentation at the surface to the surface precipitation rate and the total surface precipitation rate is calculated as:

$$P_{\rm tot}(z=0)+S(z=0)=\int_0^\infty \rho_0 (q_{\rm T})^{\rm tend}_{\rm mic} dz + (q_{\rm T})^{\rm subg-flux}_{\rm sed}(z=0)+S^{\rm resolved}(z=0), \quad \text{(S7)}$$

where $S^{\rm resolved}(z=0)$ is calculated online using resolved fields in SAM.

Text S2. Training and implementation

The data set for the NN parameterization is obtained from 337.5 days of 3-hourly snapshots of model output taken from the hi-res simulation. This data was split into train and test datasets, where the first 320.625 days (95% of the data) were used for training, and the last 16.875 days (5% of the data) were used as a test dataset. To easily upload all data into the RAM during the training procedure, and to decrease the correlation between different training samples for each 3-hourly snapshot that was used, we reduced the training data set size for the coarse-graining factor of x8 by randomly subsampling 30 (out of 72) atmospheric columns at each latitude for each snapshot. This results in training dataset size of $13,856,040$, where a sample is defined as an individual atmospheric column for a given horizontal location and time step. When using a coarse-graining factor of x16, we use all 36 longitudinal grid points at each latitude, giving $8,313,660$ training samples. We note that the split to train and test datasets is slightly different compared to YOG20 where we used 10% of the data as a test dataset.

The NNs training is implemented in Python using PyTorch (Paszke et al., 2017). The NNs weights and biases are optimized by the Adam optimizer (Kingma & Ba, 2014) combined with a cyclic learning rate (Smith, 2017). We use 1024 samples in each batch

and train over 8000 batches before completing a full cycle in the learning rate. We use 12 epochs, where the first seven epochs are trained with a minimal learning rate of 0.0002 and a maximal learning rate of 0.002, and five additional epochs are trained after reducing both the minimum and maximum learning rates by a factor of 10. We did a learning rate range test (Smith, 2017) before training and after seven epochs to find optimal learning rates. The NNs are stored as netcdf files, and then implemented in SAM using a Fortran module. The results presented in this work are for NNs with 128 nodes at each hidden layer and rectified linear unit activations (ReLu) except in the output layer where no activation function was used. Unless stated differently, NNs have five densely connected layers. Training typically takes 20 minutes when using 10 CPU cores. We note that we do not use batch normalization (Ioffe & Szegedy, 2015) since it leads to unstable simulations. The reason that using batch normalization leads to unstable simulations is that some neurons have vanishingly small variance, which leads to divergence of prognostic fields, typically within very few time steps.

Prior to training, each input (feature) of both NNs were standardized by removing the mean and rescaling to unit variance. The outputs of NN2 were standardized in a similar way, except that for the diffusivity the mean and variance were calculated across all 15 vertical levels. A different approach was used for the outputs of NN1 in order to weight the effects of the different tendencies and fluxes consistently. We first define for each training sample and subgrid process the equivalent tendencies at all vertical levels associated with the fluxes by calculating tendencies due to the predicted fluxes. We then multiplied all q_{T} tendencies by L_{c} such that all tendencies have units of J s^{-1}.

Next, we calculated the variance of each of the tendencies associated with each of the outputs (variance was calculated across all 30 levels of each process). We normalized these variances by the variance of $(H_{\mathrm{L}})^{\mathrm{tend}}_{\mathrm{rad-phase}}$ which had the smallest variance. The resulting normalized variances were 3.29 for $(H_{\mathrm{L}})^{\mathrm{subg-flux}}_{\mathrm{adv}}$, 22.30 for $(q_{\mathrm{T}})^{\mathrm{subg-flux}}_{\mathrm{adv}}$, 14.93 for $(q_{\mathrm{T}})^{\mathrm{tend}}_{\mathrm{mic}}$, 2.26 for $(q_{\mathrm{T}})^{\mathrm{subg-flux}}_{\mathrm{sed}}$ and 1.00 for $(H_{\mathrm{L}})^{\mathrm{tend}}_{\mathrm{rad-phase}}$. Finally, the outputs of NN1 were standardized by removing the mean and rescaling to the normalized variances listed above. This output normalization weights each output such that it is proportional to its effect on the prognostic variables. In the reduced precision simulations, we reduce the precision of NN scaled inputs and scaled (direct) outputs, but we do not reduce the precision of these means and scaling factors.

Text S3. Conservation of frozen moist static energy

We define the frozen moist static energy (FMSE; Kuang and Bretherton (2006)) as

$$\mathrm{FMSE} = H_{\mathrm{L}} + L_{\mathrm{c}} q_{\mathrm{T}} = c_{\mathrm{p}} T + g z + L_{\mathrm{c}} q_{\mathrm{v}} - L_{\mathrm{f}} q_{\mathrm{i}}, \tag{S8}$$

where q_{v} is the mixing ratio of water vapor and all other variables are defined in text S1. To get an evolution equation for the column integrated FMSE we multiply equation S2 by the latent heat of condensation, then sum the multiplied equation with equation S5,

and integrate this sum in the vertical with density weighting which gives

$$\begin{aligned}\frac{\partial}{\partial t}\int_0^\infty \rho_0(L_\mathrm{c}q_\mathrm{T}+H_\mathrm{L})dz &= (L_\mathrm{c}F_{q_\mathrm{T}z}+F_{H_\mathrm{L}z})(z=0)\\ &+ L_\mathrm{f}S(z=0)\\ &+ \int_0^\infty \rho_0(H_\mathrm{L})^{\mathrm{tend}}_{\mathrm{rad-phase}}dz\\ &- \int_0^\infty \rho_0(1-\omega_\mathrm{p})L_\mathrm{f}(q_\mathrm{T})^{\mathrm{tend}}_{\mathrm{mic}}dz\\ &- \int_0^\infty L_\mathrm{c}\nabla_h\cdot(\rho_0 u_h q_\mathrm{T}+F_{q_\mathrm{T}h})+\nabla_h\cdot(\rho_0 u_h H_\mathrm{L}+F_{H_\mathrm{L}h})dz \quad \text{(S9)}\end{aligned}$$

where the subscript h refers to the two horizontal coordinates. The terms in the right hand side of equation S9 represent FMSE tendencies due to: (1) surface turbulent fluxes, (2) the latent heat of fusion when ice sediments reach the surface, (3) radiative heating and heating from phase changes of precipitation, (4) the latent heat of fusion when condensate is converted to graupel or snow, and (5) horizontal advection and horizontal diffusion.

The derivation of equation S9 is unaffected by the presence of the NN parameterization because: (1) the NN parameterization predicts vertical advective fluxes rather than their associated tendencies and these fluxes are set to zero at the lower boundary, (2) the NN parameterization predicts the diffusivity instead of diffusive tendencies, and (3) the NN parameterization uses equations 4 and 5 in the main paper to diagnose the subgrid fluxes or tendencies of H_L due to sedimentation and cloud microphysics. The NN parameterization does not affect the horizontal fluxes. We thus conclude from equation S9 that the NN parameterization conserves the column integrated FMSE in the absence of radiative heating, heating from phase changes of precipitation, surface turbulent fluxes, conversion of condensate to graupel or snow and sedimentation that reaches the surface.

Text S4. Offline comparison between random-forest parameterization and neural-network parameterization

Overall, the input and output structures of the NN and RF parameterizations have similarities, though several important differences exist. The inputs for RF-tend (which is analogous to NN1) are similar, except that in RF-tend all 48 vertical levels are used as inputs. Furthermore, RF-tend predicts the net tendency summed over all processes included in NN1 of H_{L} and q_{T} at all 48 vertical levels (with the exception that radiative heating was predicted up to $z = 11.8$km), rather than predicting separate fluxes or tendencies for different physical processes. As a test, we also tried to use NN1 with a similar input and output structure to RF-tend and thus have it predict the net tendency over all processes rather than separate fluxes and sinks for each process. However, this approach does not ensure physical consistency in an NN parameterization (see section 3.2 of the main paper) unlike an RF which predict averages over training samples and thus automatically conserves energy. Although the simulation in this test was stable, the quality of online results varied substantially when different grid spacings were used. Future work could further compare two identical structures of RFs and NNs. Further details about RF-tend can be found in the supplementary information of YOG20.

NN2 and RF-diff have the same unscaled outputs and their offline results can be easily compared directly. To compare the offline results between RF-tend and NN1 we use the unscaled tendencies of H_{L} and q_{T} as calculated from the outputs predicted by NN1, such that we compare the same target values. Offline performance is measured here as the coefficient of determination (R^2) as applied to the test dataset. We find that the NN

parameterization outperforms the RF parameterization across all variables (Figure S4). To give a better intuition for the offline performance of the NN parameterization, the predictions and true values for q_{T} tendencies for specific columns and times are shown in Figure S5.

The RF and NN parameterizations have some differences in their test (text S2) and train datasets. Since the RF parameterization requires more memory during training, we used less training samples (5,000,000 for x8-RF compared to 13,856,040 for x8-NN). Using the same number of samples as in the NN lead to memory errors when training on nodes with 108Gb of random-access memory. Since the R^2 of the RF parameterization is roughly constant when increasing the number of training samples above 5,000,000 (Fig. S8 in YOG20), it is unlikely that adding more samples would substantially change the offline result of RF parameterization.

References

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Khairoutdinov, M. F., & Randall, D. A. (2003). Cloud resolving modeling of the ARM summer 1997 IOP: Model formulation, results, uncertainties, and sensitivities. *Journal of the Atmospheric Sciences*, *60*(4), 607–625.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kuang, Z., & Bretherton, C. S. (2006). A mass-flux scheme view of a high-resolution simulation of a transition from shallow to deep cumulus convection. *Journal of the*

Atmospheric Sciences, *63*(7), 1895–1909.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., ... Lerer, A. (2017). Automatic differentiation in pytorch. In *Neural information processing systems workshop, 2017.*

Smith, L. N. (2017). Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)* (pp. 464–472).

Yuval, J., & O'Gorman, P. A. (2020). Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature Communications*, *11*(1), 1–10.

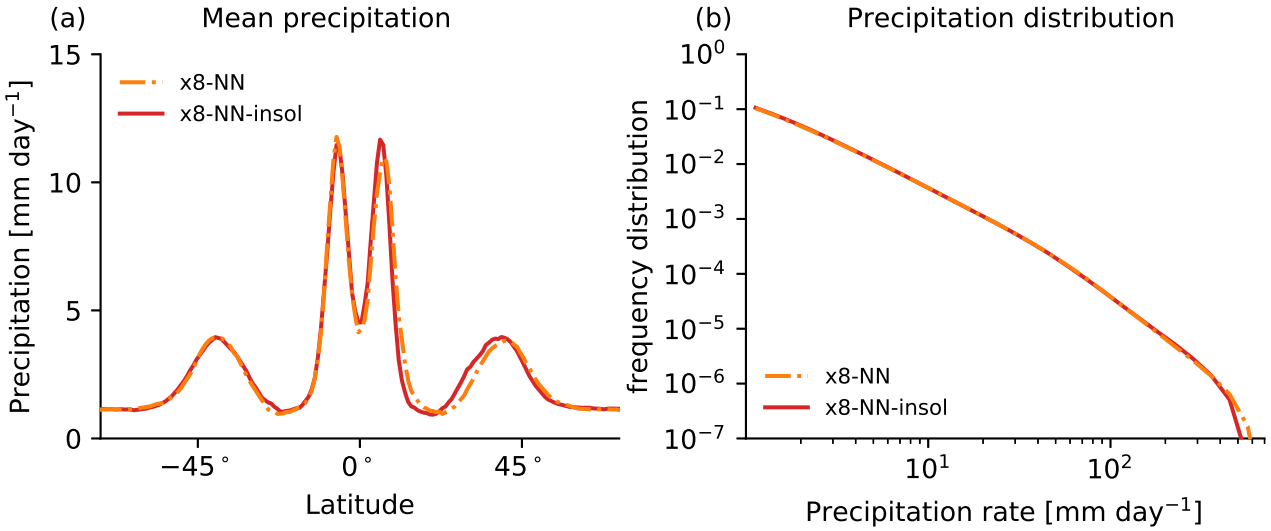

Figure S1. (a) Zonal- and time-mean precipitation rate as a function of latitude and (b) frequency distribution of 3-hourly precipitation rate for coarse-resolution simulations with the default NN parameterization which uses distance to the equator as an input feature (x8-NN; orange dash-dotted) and an alternative NN parameterization that instead uses top of atmosphere insolation as an input feature (x8-NN-insol; red).

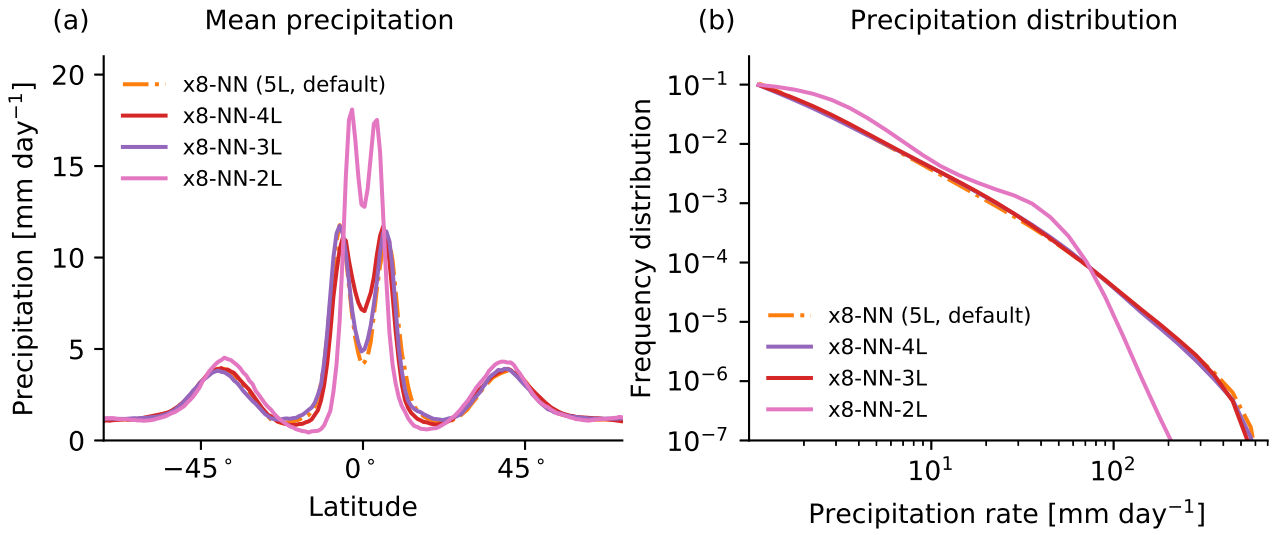

Figure S2. (a) Zonal- and time-mean precipitation rate as a function of latitude and (b) frequency distribution of 3-hourly precipitation rate for the coarse-resolution simulation with the NN parameterization using 5 layers which is the default NN used in the paper (x8-NN; orange dash-dotted), 4 layers (x8-NN-4L; red), 3 layers (x8-NN-3L; purple) and 2 layers (x8-NN-2L; purple). All networks were trained with the same protocol and learning rates.

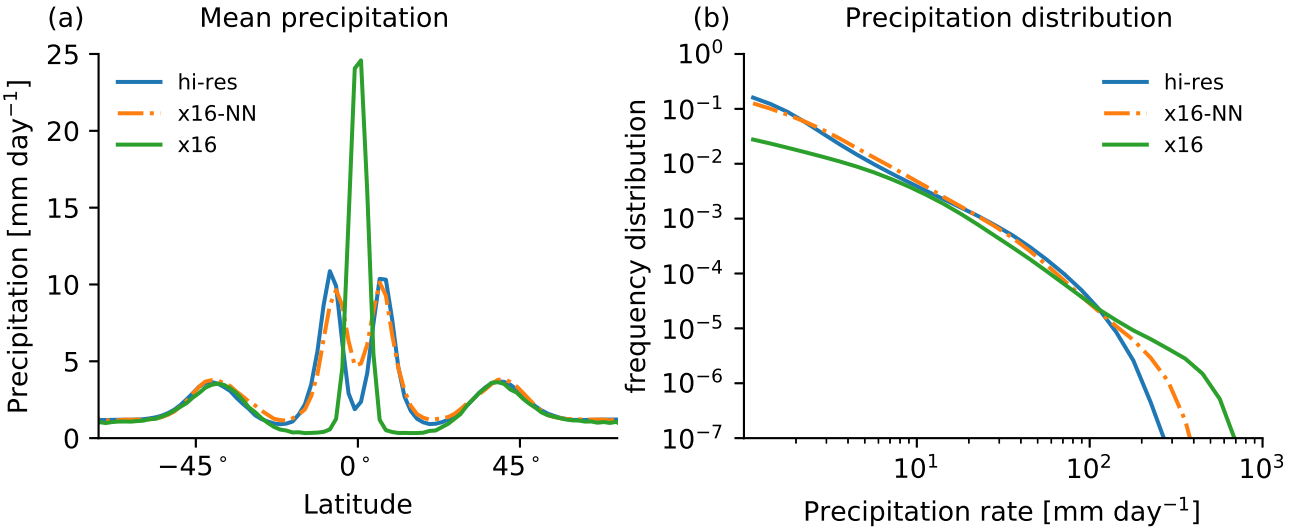

Figure S3. (a) Zonal- and time-mean precipitation rate as a function of latitude and (b) frequency distribution of 3-hourly precipitation rate for the high-resolution simulation (hi-res; blue), the coarse-resolution simulation at 192km horizontal grid spacing without the NN parameterization (x16; green) and with the NN parameterization (x16-NN; orange dash-dotted). For hi-res, the precipitation is coarse-grained to the grid spacing of x16 prior to calculating the frequency distribution.

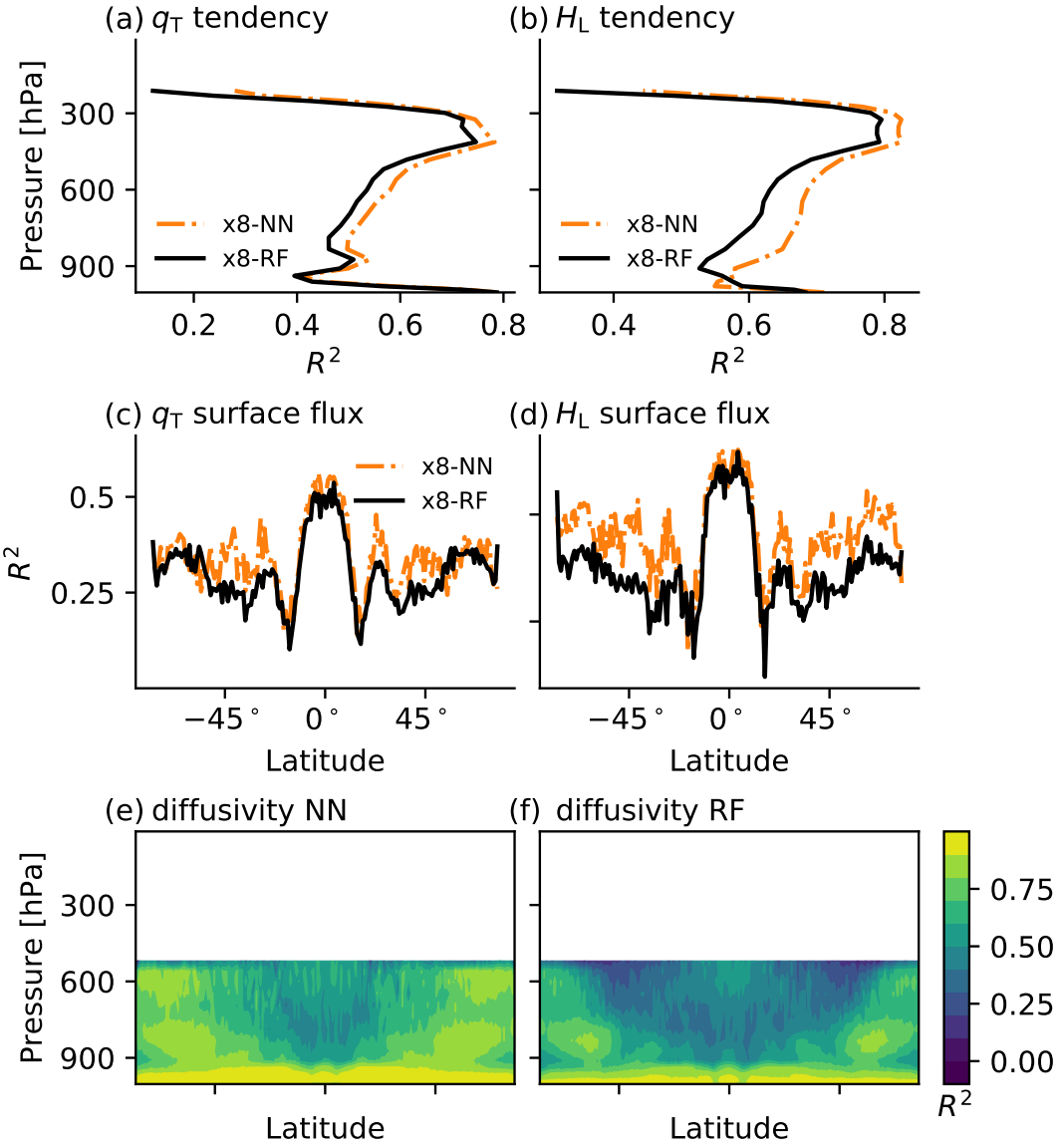

Figure S4. Coefficient of determination (R^2) for offline performance of the x8-NN and x8-RF parameterizations for the (a) subgrid tendency of non-precipitating water mixing ratio (q_T) as a function of pressure, (b) subgrid tendency of liquid/ice water static stability energy (H_L) as function of pressure, (c) subgrid surface energy flux as a function of latitude, (d) subgrid surface moisture flux as a function of latitude, (e-f) diffusivity as a function of latitude and pressure for the (e) NN and (f) RF. In panels a-d orange dash-dotted lines show results for NN and black lines show results for RF. R^2 is calculated based on the samples from the test datasets.

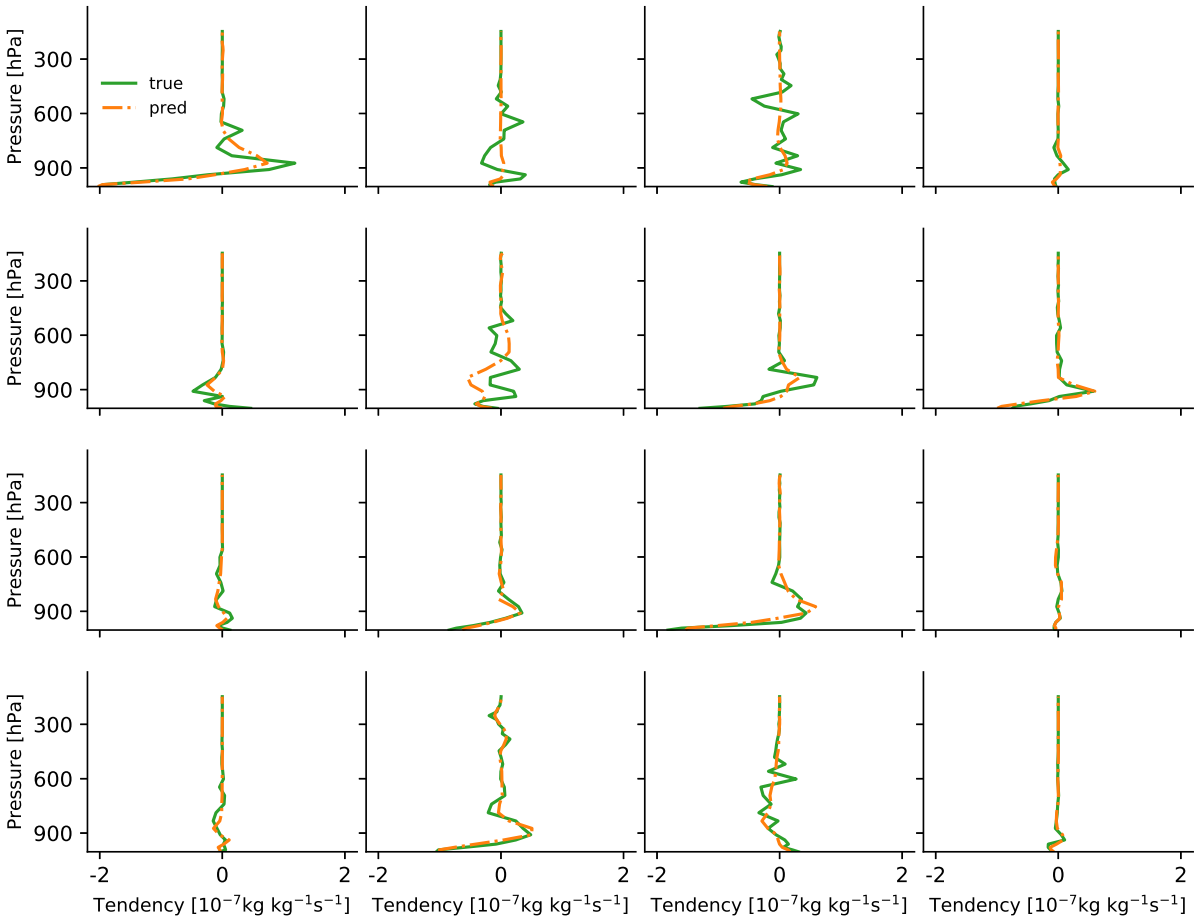

Figure S5. The true (green) and NN1 predicted (orange dash dotted) subgrid tendency of non-precipitating water mixing ratio (q_T; tendency calculated from the sum of subgrid vertical advection, sedimentation and microphysics) as a function of pressure for randomly chosen columns and times.

	R^2 relative to hi-res			R^2 relative to x8-NN			
	x8	x8-RF	x8-NN	x8-NN-7bits	x8-NN-5bits	x8-NN-3bits	x8-NN-1bits
Eddy kinetic energy	0.88	0.97	0.94	0.99	0.99	0.99	0.30
Zonal wind	0.87	0.98	0.93	0.99	0.99	0.99	0.70
Meridional wind	-0.01	0.87	0.80	0.98	0.99	0.95	0.78
Non-precipitating water	0.97	0.99	0.99	0.99	0.99	0.99	0.98
Precipitation	-3.47	0.92	0.88	0.94	0.99	0.86	0.79
Precip. frequency dist.	0.35	0.98	0.99	0.99	0.99	0.99	0.95

Table S1. Online performance as measured by the coefficient of determination (R^2) of the time- and zonal-mean of eddy kinetic energy, zonal wind, meridional wind, non-precipitating water, precipitation, and of the frequency distribution of 3-hourly precipitation. R^2 is calculated relative to hi-res for the coarse-resolution simulation with no ML parameterization (x8), with the RF parameterization (x8-RF) and with the NN parameterization (x8-NN). R^2 is calculated relative to x8-NN for the simulations with reduced-precision parameterizations with 7 (x8-NN-7bits), 5 (x8-NN-5bits), 3 (x8-NN-3bits) and 1 (x8-NN-1bits) bits in the mantissa. The eddy kinetic energy is defined with respect to the zonal and time mean. x8-NN-7bits corresponds to the bfloat16 half-precision format which has 7 bits in the mantissa, and the default parameterization (x8-NN) is single precision which has 23 bits in the mantissa. For hi-res, the precipitation is coarse-grained to the grid spacing of x8 prior to calculating the frequency distribution.

	$(H_{\mathrm{L}})^{\mathrm{subg-flux}}_{\mathrm{adv}}$	$(q_{\mathrm{T}})^{\mathrm{subg-flux}}_{\mathrm{adv}}$	$(q_{\mathrm{T}})^{\mathrm{subg-flux}}_{\mathrm{sed}}$	$(q_{\mathrm{T}})^{\mathrm{tend}}_{\mathrm{mic}}$	$(H_{\mathrm{L}})^{\mathrm{tend}}_{\mathrm{rad-phase}}$
x8-NN-2L	0.60	0.64	0.69	0.68	0.65
x8-NN-3L	0.63	0.67	0.79	0.78	0.77
x8-NN-4L	0.63	0.68	0.80	0.79	0.80
x8-NN (5L)	0.64	0.68	0.81	0.79	0.79
x16-NN (5L)	0.71	0.80	0.85	0.81	0.83

Table S2. Offline performance of NN1 as measured by R^2 for different NN architectures and coarse-graining factors. The offline performance is given for different outputs for NNs with 2,3,4 and 5 layers at x8 resolution (x8-NN-2L, x8-NN-3L, x8-NN-4L, x8-NN, respectively), and for 5 layers at x16 resolution (x16-NN). Note that x8-NN and x16-NN use the default of 5 layers. All vertical levels used are included when calculating R^2. All results are based on the test dataset.