https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Efficient and Stable Coupling of the SuperdropNet Deep Learning-based Cloud Microphysics (v0.1.0) to the ICON Climate and Weather Model (v2.6.5)

Caroline Arnold[1,2,3], Shivani Sharma[2,3,4], Tobias Weigel[1,2,3], and David Greenberg[2,3]

[1]German Climate Computing Center DKRZ, Hamburg, Germany
[2]Helmholtz Zentrum Hereon, Geesthacht, Germany
[3]Helmholtz AI, Munich, Germany
[4]International Max-Planck-Research School on Earth System Modeling, Hamburg, Germany

Correspondence: Caroline Arnold (arnold@dkrz.de), Shivani Sharma (shivani.sharma@hereon.de)

Abstract.

Machine learning (ML) algorithms can be used in Earth System models (ESMs) to emulate sub-grid-scale processes. Due to the statistical nature of ML algorithms and the high complexity of ESMs, these hybrid ML-ESMs require careful validation. Simulation stability needs to be monitored in fully coupled simulations, and the plausibility of results needs to be evaluated in suitable experiments.

We present the coupling of SuperdropNet, a machine learning model for emulating warm rain processes in cloud microphysics, into ICON 2.6.5. SuperdropNet is trained on computationally expensive droplet based simulations and can serve as an inexpensive proxy within weather prediction models. SuperdropNet emulates the collision-coalescence of rain and cloud droplets in a warm rain scenario and replaces the collision-coalescence process in the two-moment cloud microphysics scheme. We address the technical challenge of integrating SuperdropNet, developed in Python and PyTorch, into ICON, written in Fortran, by implementing three different coupling strategies: embedded Python via the C Foreign Function Interface, pipes, and coupling of program components via YetAnotherCoupler (YAC). We validate the emulator in the warm bubble scenario and find that SuperdropNet runs stable within the experiment. In comparing experiment outcomes from the bulk moment scheme and SuperdropNet, we find that the results are physically consistent, and discuss differences that are observed for several diagnostic variables.

In addition, we provide a quantitative and qualitative computational benchmark for three different coupling strategies—embedded Python, coupler YAC, and pipes—and find that embedded Python is a useful software tool for validating hybrid ML-ESMs.

1 Introduction

Machine learning (ML) is increasingly used in Earth system models (ESMs) to emulate sub-grid-scale processes that are typically parameterized or neglected due to their high computational cost (Dueben et al., 2021; Christensen and Zanna, 2022; Irrgang et al., 2021; Gentine et al., 2018). When sub-grid-scale processes are replaced by ML algorithms, the improvement can

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

aim at speeding up the overall simulation by emulating the existing parameterization. Recent examples include the emulation of gravitational wave drag (Chantry et al., 2021), cloud microphysics (Brenowitz et al., 2022), the ocean in a coupled climate model (Sonnewald et al., 2021), and cloud radiative effects (Meyer et al., 2022).

Other studies aim to improve the overall description of the Earth system by providing a better parameterization. ML algorithms can be trained on high-resolution ESM output or even on separately simulated processes to emulate resolved processes in a low-resolution simulation, e.g. for gravity waves (Dong et al., 2023), cloud cover parameterizations (Grundner et al., 2022), general parameterizations Brenowitz and Bretherton (2018), sub-grid-scale momentum transport (Yuval and O'Gorman, 2023), effects of cloud resolving simulations (Rasp et al., 2018), ozone distributions (Nowack et al., 2018), and radiative transfer (Belochitski and Krasnopolsky, 2021).

Many parameterizations in ESMs can be removed at higher resolutions if the process can be completely resolved, such as the convective parameterizations. On the other hand, some others would need to be parameterized even for 1-km scale weather models. Cloud microphysical processes fall in this category. Processes dealing with the droplet interactions that lead to precipitation are lumped together and referred to as cloud microphysical processes. Due to high particle counts even at small grid sizes and our incomplete understanding of processes that occur at a molecular level in clouds (Morrison et al., 2020), we cannot expect cloud microphysical parameterizations to become obsolete in the near future for high resolution models.

The parameterization of these processes suffers from a unique accuracy/speed trade-off. The most accurate droplet based Lagrangian schemes such as the superdroplet method (Shima et al., 2009) are computationally expensive. The commonly used bulk moment schemes represent the complex particle size distributions as only the first two moments, referring to the total droplet concentration and the liquid water content of the hydrometeors. For modelling the droplet collisions in a warm-rain scenario, ICON uses the well studied bulk moment scheme developed in Seifert and Beheng (2001). To bridge this gap and to make the use of more complex microphysical schemes feasible within operational models, a data-driven approach can be employed. We present here the integration of the recently released SuperdropNet (Sharma and Greenberg, 2023), an ML algorithm for emulating warm rain processes in cloud microphysics, into ICON 2.6.5. SuperdropNet is trained on zero dimensional box model superdroplet simulations from McSnow 1.1.0 (Brdar and Seifert, 2018) in a warm rain scenario and replaces the warm rain processes in the two-moment scheme available in ICON 2.6.5 (Seifert and Beheng, 2006).

Due to the statistical nature of ML algorithms and the complex nonlinear interactions in ESMs, hybrid systems of numerical ESMs and ML algorithms require careful validation and verification (Dueben et al., 2022; Brenowitz and Bretherton, 2019). Stand-alone ML algorithms are first trained on a dataset and then validated on a hold-out test dataset that is not seen during training. This test set is within the distribution of the training data. When an ML algorithm is coupled to an ESM, it may encounter conditions outside of the range of the training data, and the required extrapolation could lead to instabilities (Yuval et al., 2021). Thus, the so-called *offline* performance of an ML algorithm is often not a good indicator of its *online* performance (Brenowitz et al., 2020b; Rasp, 2020). Stability is a major concern when introducing ML emulators into ESMs. It can be improved by adapting the training procedure (Brenowitz and Bretherton, 2018; Brenowitz et al., 2020a; Qu and Shi, 2023; Rasp, 2020) or by fulfilling physical constraints in the network architecture (Beucler et al., 2021; Yuval et al., 2021). Careful

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

validation setups can help the scientific community to build trust in so-called black box ML algorithms (McGovern et al., 2019).

To avoid devoting resources to the development of ML algorithms that fail in contact with reality, we encourage incorporating online testing into the iterative development of the ML emulator. Popular ML software libraries such as PyTorch (Paszke et al., 2019), Keras (Chollet et al., 2023), or Tensorflow (Abadi et al., 2016), are based on the Python language, while ICON is written in Fortran. Hence, to couple ML algorithms to Fortran-based ESMs like ICON, it is necessary to integrate the two programming languages with one another (Brenowitz and Bretherton, 2019). Hybrid ML-ESMs must be computationally powerful enough for verification and validation experiments without requiring rewriting the ML code in Fortran.

In Sect. 2.1 we introduce the warm bubble scenario, which serves as a test case for SuperdropNet. The ML algorithm itself is described in 2.2. Different strategies for integrating SuperdropNet into ICON are discussed in Sect. 3. The results and the impact of SuperdropNet on atmospheric processes and prognostic variables are presented in Sect. 4.2A computational and qualitative benchmark of three different strategies is included in Sect. 4.3.

2 Methods

2.1 Warm bubble scenario

We validate SuperdropNet in the warm bubble scenario, a testcase for cloud microphysics available in ICON 2.6.5. It describes an atmosphere temperature profile with a warm air bubble at the bottom that rises vertically. The test case operates on a torus grid of 22×20 cells with a domain length $5\,\mathrm{km}$, and a simulation time step of $20\,\mathrm{s}$ with a total simulation time of $120\,\mathrm{min}$. The experiment is computationally lightweight and runs on a single compute node. We test SuperdropNet in a warm atmosphere with no ice particle formation, as well as in a cold atmosphere that allows ice formation. All simulation parameters are summarized in Table 1. We transport the tracers required for two-moment cloud microphysics, i.e. first and second moment of the hydrometeors cloud water, cloud ice, rain, snow, graupel, and hail.

2.2 SuperdropNet cloud microphysics model

SuperdropNet is a machine learning emulator for superdroplet simulations in a warm rain scenario. It is a neural network consisting of fully connected layers and is trained to predict updates of the bulk moments for cloud and rain over different droplet size distributions. SuperdropNet is described in detail in (Sharma and Greenberg, 2023).

The superdroplet simulations used for training are generated with McSnow (Brdar and Seifert, 2018; Seifert and Rasp, 2020). In (Brdar and Seifert, 2018) McSnow was used for simulating ice particles, while in (Seifert and Rasp, 2020) it was simulating a warm rain scenario. Similarly for generating the training data for SuperdropNet, the processes are restricted to the warm rain scenario and describe only the conversion of cloud droplets into rain in a dimensionless control volume. As superdroplet simulations are stochastic in nature, we use multiple realizations of simulations to train SuperdropNet. Hence, given a set of initial conditions, SuperdropNet is completely deterministic in nature and the bulk moments estimated by it are

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Parameter	Description	Warm bubble	Cold bubble
L_D	Torus domain length	5000 m	
t_{dyn}	Dynamical time step	20 s	
t_{2mom}	Two-moment scheme time step	20 s	
z_{lev}	Atmospheric levels	70	
p_{srfc}	Surface pressure	1013.25 hPa	
T_0	Cold point of atmosphere	303.15 K	268.15 K
γ_0	Vertical temperature lapse rate	0.006 K/m	0.009 K/m
z_0	Altitude up to which γ_0 applies	3000 m	4000 m
γ_1	Lapse rate above z_0	0.000 01 K/m	0.0001 K/m
T_{perturb}	Temperature perturbation	10 K	5 K
ϕ_{bg}	Background relative humidity	0.7	
ϕ_{mx}	Maximum relative humidity	0.9	0.95
ξ	Half-width of temperature perturbation in x	12 500 m	
ζ	Half-width of temperature perturbation in z	200 m	250 m
x_0	Center of temperature perturbation in x	0 m	

Table 1. Experiment parameters for the warm bubble and the cold bubble test case.

the equivalent of averaged superdroplet simulations (Sharma and Greenberg, 2023). The microphysical processes accounted for are accretion, autoconversion and self-collection of rain and cloud droplets. In ICON, the droplet collisions corresponding to warm rain processes are treated in a separate module where the process rates for accretion, autoconversion, and self-collection of rain and cloud droplets are calculated. The parameterization scheme is localized, i.e the process rates calculated for a grid cell depend only on the rain and cloud moments corresponding to that grid cell. Other microphysical processes and the vertical transport are accounted for in separate modules, which implies that the parameterization in ICON is structured such that all individual grid points can be considered zero-dimensional boxes. Thus, the parameterization setup for droplet collisions in ICON mimics the training data for SuperdropNet. This justifies the choice of using a test scenario in ICON for online coupling and testing of SuperdropNet.

Note that only the warm rain processes are replaced with SuperdropNet. In a cold atmosphere, SuperdropNet can still be coupled to ICON, but since warm rain processes are not relevant there, including SuperdropNet is expected not to change the experiment results.

2.3 ICON program flow

To illustrate at which point of program execution ML-ESM coupling becomes necessary, we show the flowchart for a single ICON time step in Fig. 1, focusing only on the steps relevant to our application. Starting from the general ICON time loop,

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

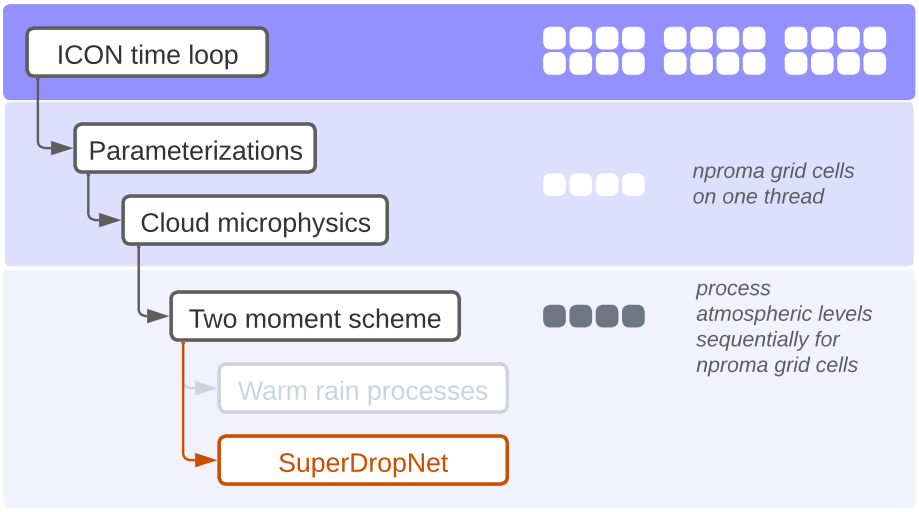


Figure 1. We replace the warm rain processes (gray) with a call to SuperdropNet (orange). At this point, each thread has access to an *ik-slice*, a specific representation in the cloud microphysics parameterization that corresponds to one atmospheric level for one block of grid cells.

where the full grid information is available, we enter the cloud microphysics parameterization. At this point, a given thread has access to one block of grid cells with block length nproma, and all threads work in parallel. The two-moment scheme has its own grid representation, called *ik-slices*, where the block of grid cells is again divided by atmospheric levels. In our experiment, we simply replace the warm rain processes with a call to SuperdropNet, which provides updated moments for cloud and water droplets.

Since the call to the ML component is not at the grid level, but operates on *ik-slices* far down in the nested structure of the ICON program flow, we need to call SuperdropNet several times per time step – once for each block of grid cells and once for each atmospheric level. Note that saturation adjustments and evaporation are handled outside of the parts of the ICON code replaced by SuperdropNet.

3 Integrating SuperdropNet in ICON

There are several ways to integrate Python machine learning components into Fortran code (Partee et al., 2022). Based on a pre-selection of suitable methods, we have implemented three strategies, so-called Fortran-Python bridges. For convenience, we add a namelist to ICON that allows the selection of the coupling strategy. We perform the experiment with all three methods on the DKRZ Levante system.

3.1 Embedding Python as a dynamic library

Using the techniques in (Brenowitz, N., 2023), we develop a dynamic library based on Python code. The library is generated using the C Foreign Function Interface (CFFI) Rigo and Fijalkowski (2018) and is linked to ICON at compile time. At runtime,

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Python code is executed from the library. Employment of CFFI results in Python and Fortran sharing their address space, hence passing memory pointers is sufficient to access the same data. Jobs are run in a homogeneous setting, with Python code executed on the same CPU compute node as ICON.

3.2 Using the coupling software YAC

YetAnotherCoupler (YAC) (Hanke et al., 2023, 2016) is commonly used to couple different ICON components, e.g., atmosphere, ocean and I/O. YAC provides Python bindings so that external Python programs can be coupled with little effort to ICON.

YAC requires a definition of fields that are to be exchanged, and an exchange schedule that cannot be below the time step of ICON. For the warm-bubble scenario, we set the block length to the number of grid cells (880) and define two exchange fields per atmospheric level, one for the ICON-to-Python exchange, and one for the reverse exchange. This yields a total of 140 fields, that are exchanged at each time step. A smaller block length would require the developer to define more exchange fields, such that bulk moments in each grid cell can be exchanged at every time step.

Data transfer is building on Message Passing Interface (MPI) routines that are integrated in YAC. This offers the flexibility to use heterogeneous jobs, i.e., running ICON on CPU nodes and ML inference on GPU nodes. Due to current limitations of the DKRZ Levante system, we were not able to test the performance in a heterogeneous setting. With ICON shifting to GPUs, we foresee that in the future homogeneous jobs will be run on GPU nodes.

3.3 Pipes

We implemented a coupling between n ICON processes and one Python process running on the same node using FIFO (first-in-first-out) pipes. The first ICON MPI rank on the node will spawn a separate Python process that runs a worker script. Each rank also creates two pipes, one for each direction of communication (input and output to the Python worker). The worker iterates over all input pipes, performs the warm rain calculation on data being available and writes results back to the corresponding ICON process via its output pipe.

While this solution does not incur the potential overhead of using MPI to communicate locally, it is not a full shared memory solution relying on pointers exclusively. The corresponding extensions to ICON and the Python worker script are optimized to do as few memory copies as possible, though naturally some copying cannot be avoided when interacting with the pipes. As FIFO pipes only work on a local node, no cross-node setups are possible, such as running ICON and Python on different types of nodes (CPU, GPU). As the Python worker runs as a separate process on a dedicated core, the number of cores available to ICON is also marginally reduced by one.

3.4 Other methods

We note that the selection of methods in Sects. 3.1–3.3 is by no means encompassing all the available tools and summarize here the alternatives to the best of our knowledge:

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Three software libraries developed at ECMWF (Bonanni et al., 2022), the Cambridge Institute for Computing in Climate Science (Elafrou et al., 2023), and NVIDIA (Alexeev, D., 2023) address ML inference directly by exposing the Tensorflow and Pytorch APIs for Fortran, respectively. This adds the benefit of not requiring a Python runtime environment at the time of execution. Since we require flexibility to use Python code beyond ML inference, and data exchange is done here via RAM comparable to the approach described in Sect. 3.1, we did not investigate these libraries further.

During development, we noted that integrating SmartSim (Partee et al., 2022) would require a rewrite of the ICON startup routine that is beyond the scope of this project. On a similar note, the coupling routines developed in WRF-ML for the open source Weather Research and Forecasting (WRF) model cannot easily be adjusted to work with ICON (Zhong et al., 2023).

The Fortran-Keras bridge (Ott et al., 2020) allows for ML inference in Fortran based on ML algorithms developed in the Keras framework. This limits flexibility, since only those network layers and functionalities supported by the library can be used. On a similar note, the implementation of the ML algorithm in Neural Fortran (Curcic, 2019) is contingent on the library. We chose to forego these methods since we desire the flexibility to use any novel Pytorch developments without depending on their integration into an external library.

4 Results

4.1 Experiment description

Using the three coupling techniques described in Sects. 3.1–3.3, we integrate SuperdropNet in ICON. The experiment results are the same since the same network is called, but the impact on computational performance is different. We run the warm bubble scenario and the cold bubble scenario, both with a representation of warm rain processes using SuperdropNet, as well as using the existing bulk moment scheme in the two-moment cloud microphysics module.

We compare the effect of replacing warm rain processes with SuperdropNet on the experiment outcome in Sect. 4.2. In Sect. 4.3, we compare the impact on computational performance that is incured by integrating SuperdropNet for all three coupling techniques.

4.2 Comparison of the bulk moment scheme and SuperdropNet

4.2.1 Rain rates

Figure 2 shows the grid-averaged rain rate in the warm bubble scenario deriving from warm rain processes using ICON's two-moment bulk cloud microphysics, with a comparison to SuperdropNet microphysics. Since SuperdropNet was trained on particle-based simulations that avoid certain statistical approximations of bulk moment schemes, we do not expect the rain rates in both scenarios to match. Due to the experimental setup, it is not possible to identify with certainty which model produces the more accurate rain rates. We do note, however, that SuperdropNet yields physically plausible rain rates. The rain rate obtained using SuperdropNet evolves in a predictable way, i.e, there is no rain at the beginning of the simulation, which eventually builds up to a peak and then slowly rescinds. At the end of the simulation, the rain rate is zero for both the simulations. Also,

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

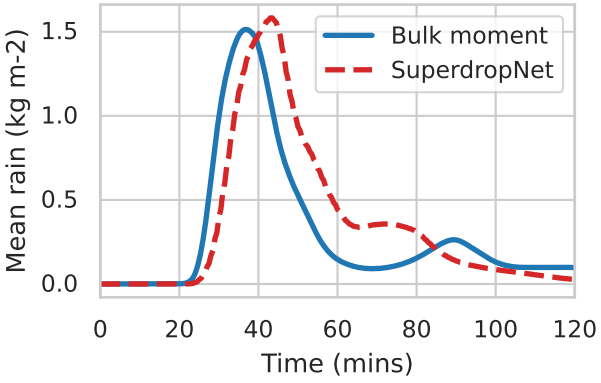


Figure 2. Grid-averaged rain in the warm bubble scenario for the bulk moment scheme used in ICON two-moment cloud microphysics, and for SuperdropNet.

no negative values are observed and neither does coupling with SuperdropNet result in dramatically different values. This emphasizes that SuperdropNet is stable over longer simulation runs and overall behaves as a realistic ML based emulator for droplet collisions. One of the key differences in the evolution of the rain rate with the two different parameterizations is that the onset of rain is slightly delayed with SuperdropNet coupling which indicates a slower conversion of cloud droplets to rain droplets.

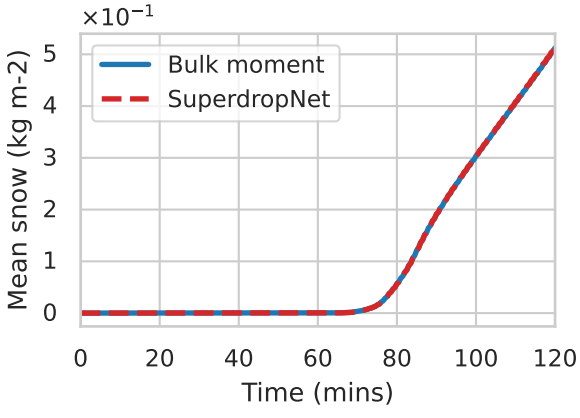


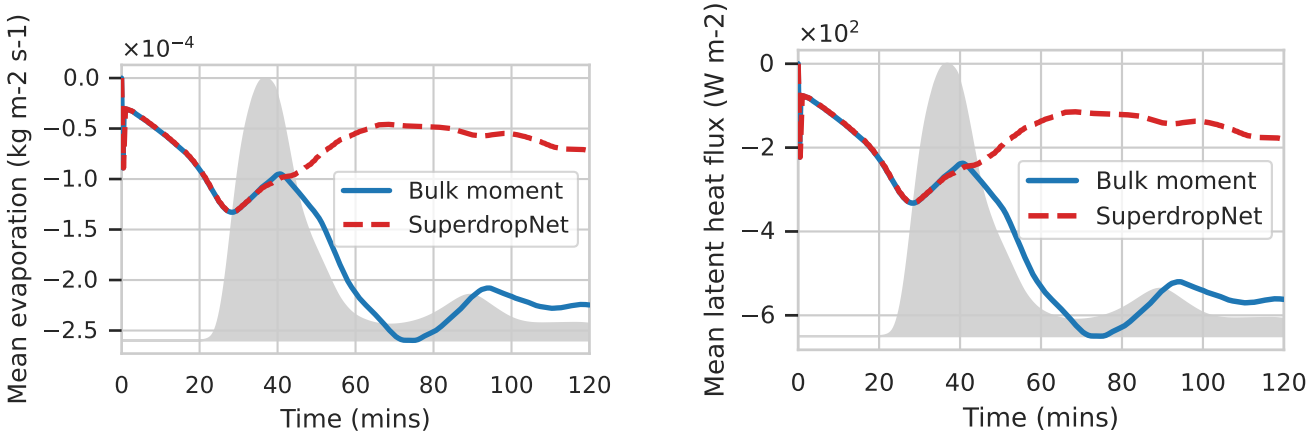
Figure 3. Grid-averaged snow for the cold bubble scenario for the bulk moment scheme used in ICON two-moment cloud microphysics, and for SuperdropNet.

As a sanity check, we perform the cold bubble experiment using both the bulk moment scheme and SuperdropNet for the warm rain processes. In this scenario, warm rain processes are not relevant for the cloud microphysics, and we expect that

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

including SuperdropNet does not affect processes with frozen particles. Figure 3 shows the grid-averaged snow rate. Both schemes show identical snow rates, which confirms that there are no undesired side-effects from coupling SuperdropNet.

4.2.2 Heat Transport Fluxes



(a) Evaporation.

(b) Latent heat flux.

Figure 4. Grid-averaged surface heat fluxes for the bulk moment scheme used in ICON two-moment cloud microphysics, and for SuperdropNet. The gray area shows the grid-integrated rain obtained using the bulk-moment scheme. High negative values indicate a higher heat flux.

Figure 4 shows the grid-averaged surface heat fluxes as they evolve with time during the coupled warm-bubble simulation in ICON. While in the beginning both the bulk moment scheme and SuperdropNet produce similar fluxes, the values diverge approximately after about 30 minutes, corresponding to the onset of rain. Generally, it can be seen that coupling SuperdropNet with the warm-bubble leads to reduction in the magnitude of evaporative and latent heat fluxes. This difference between the magnitude of fluxes is also reflected in the evolution of winds during the simulation. Winds are the primary source of energy transport and Fig. 5 shows the evolution of meridional winds in the simulation. After approximately 40 minutes, which roughly corresponds to the end of the first rainfall with both parameterizations, the wind patterns are markedly different for the bulk moment and the SuperdropNet parameterizations. The winds appear much stronger in case of the bulk moment parameterization across the vertical column. The reduced magnitude of winds in SuperdropNet coupling corresponds to reduced heat fluxes in Fig. 4.

Evaporative heat flux is proportional to the differences in humidity near the surface and the mean rain mass near the surface, while latent heat flux also increases proportionally with the mean rain mass. Figure 6 shows the vertical profile of specific humidity at different timesteps during the simulation. For the first 40 minutes of the experiment, both parameterization schemes produce similar specific humidity profiles but this changes during the later part of the simulation. Close to the surface, it can be observed that the bulk moment parameterization produces a stronger humidity gradient in comparison to SuperdropNet. This

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

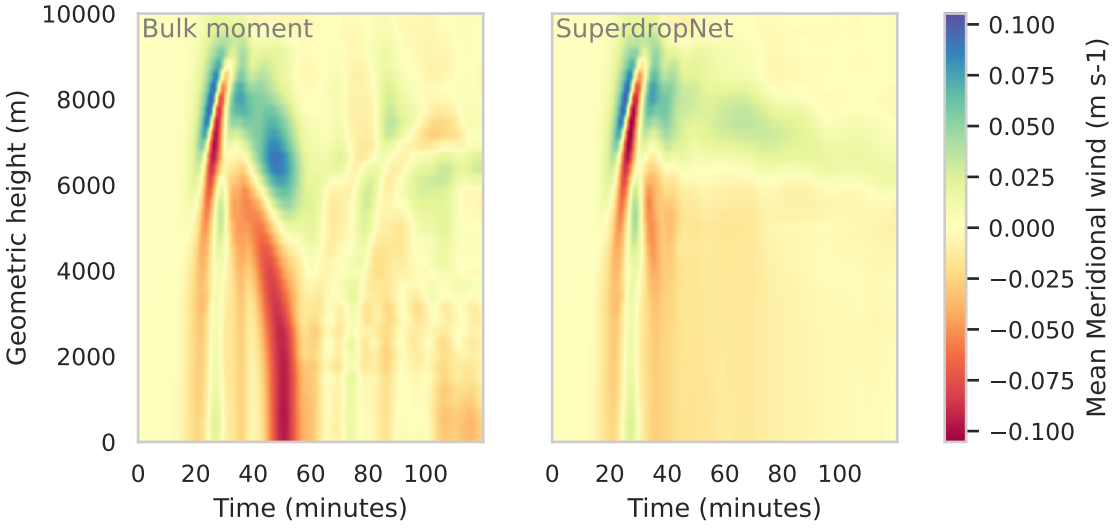


Figure 5. Averaged meridional winds for the bulk moment scheme used in ICON two-moment cloud microphysics *(left)* and for SuperdropNet *(right)*.

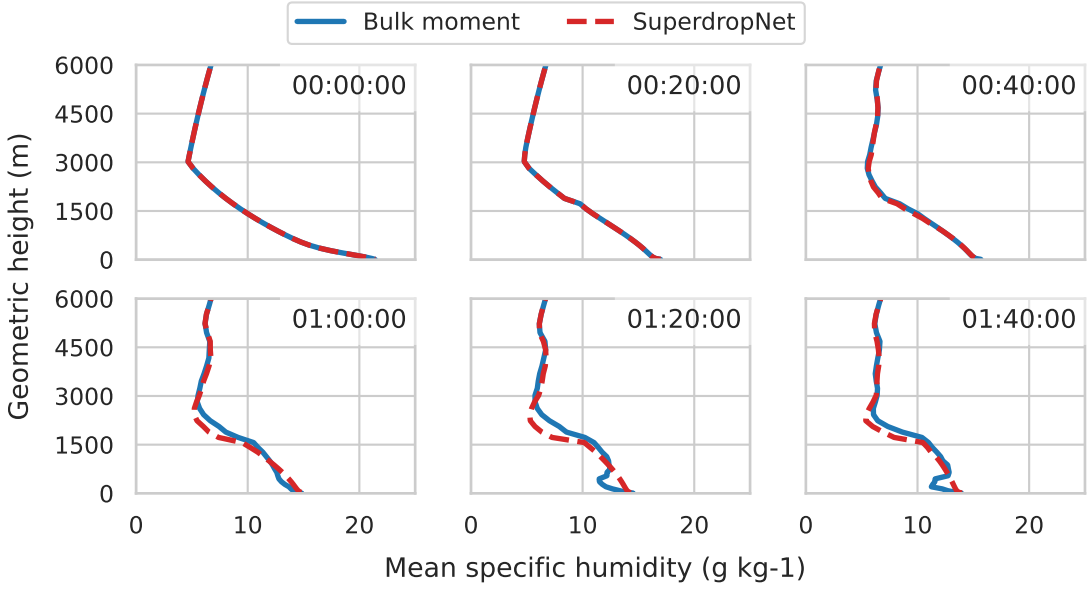


Figure 6. Vertical profile of the specific humidity at different times for the bulk moment scheme and for SuperdropNet.

difference in the specific humidity gradient possibly results in a higher evaporative flux for the bulk moment coupling than the SuperdropNet coupled simulation.

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

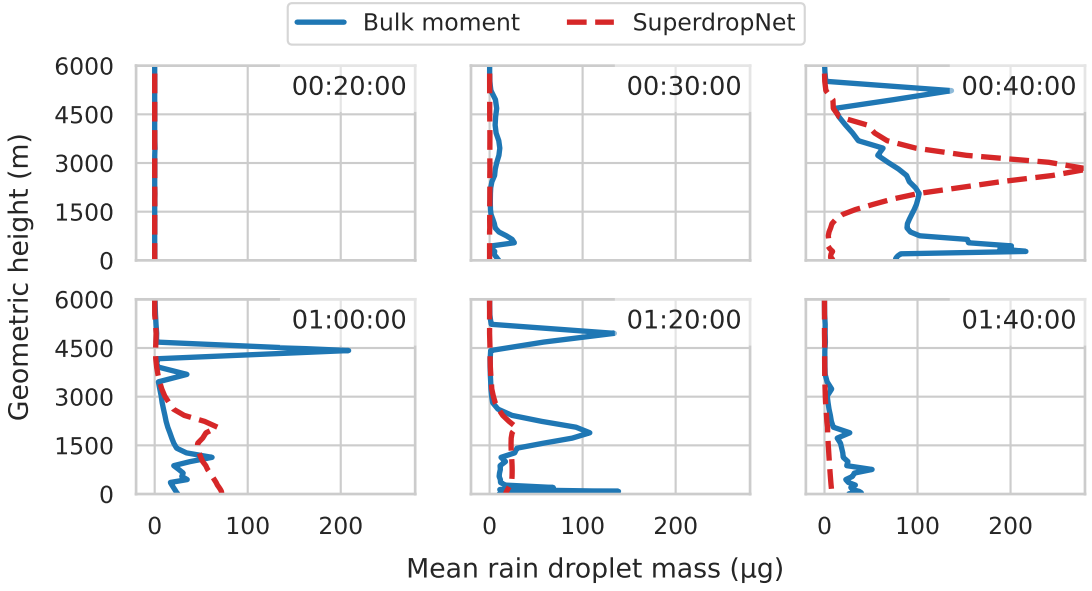


Figure 7. Vertical profile of the rain droplet mass, calculated as the ratio of the specific rain content and the number concentration of rain droplets at different times for the bulk moment scheme and for SuperdropNet.

Similarly, in Fig. 7 the evolution of mean rain droplet mass ($\bar{X}_r$) is shown. The differences in $\bar{X}_r$ close to the surface as calculated using the bulk moment scheme vs. SuperdropNet become more visible after 40 minutes. In general with the bulk moment parameterization $\bar{X}_r$ values are higher than those with the superdroplet parameterization close to the surface. Since both evaporative and latent heat fluxes are propotional to the mean rain mass, higher $\bar{X}_r$ in bulk moment coupling results in higher heat fluxes. Throughout the vertical column, the SuperdropNet parameterization usually corresponds to lower $\bar{X}_r$, except at the 40 minutes time step where the high $\bar{X}_r$ value near the 3000 m height also corresponds to a higher amount of the vertically integrated rain rate as seen in figure 2.

Note that the warm-bubble scenario in ICON is highly sensitive to tiniest fluctuations within the assumptions made for cloud microphysics parameterization. Since many other complex phenomena are simplified and the focus is only on the formation and dissipation of a single cloud, small deviations in the approximation of the cloud and rain moments lead to changes in other diagnostic variables that can accumulate over time.

4.3 Computational performance upon including SuperdropNet

4.3.1 Benchmark

We run the experiments on the Levante compute system at the German Climate Computing Center on compute nodes equipped with 2 AMD 7763 CPUs with a total of 128 cores and 256 GB main memory. The nodes are connected with a Mellanox Infiniband HDR100 fabric.

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Experiment		$t_{2\mathrm{mom}}$ (s)	Nodes
Bulk moment scheme (Fortran)		1.25	1
SuperdropNet (Pytorch)	CFFI	24.1	1
	Pipes	62.6	1
	YAC	49.5	2

Table 2. Time spent in the two-moment scheme in the ICON warm-bubble scenario, using the bulk-moment scheme (Fortran), and SuperdropNet (Pytorch) coupled to ICON.

SuperdropNet provides a significant speedup by emulating processes that would otherwise be computationally infeasible to include in ICON, but when adding a Python component to the existing highly optimized Fortran code we expect an impact on computational performance. Table 2 summarizes the total time spent in the calculation of the two-moment scheme in the ICON warm bubble scenario, using the bulk moment scheme and SuperdropNet coupled to ICON through three different coupling strategies. The fastest time to solution is provided by including SuperdropNet via embedded Python, i.e. the C Foreign Function Interface (CFFI) (Sect. 3.1). Coupling SuperdropNet via YAC (Sect. 3.2) increases the relative runtime by a factor of two compared to embedded Python. Note that when coupling with YAC, the ICON and the Python main program run on two different computational nodes, which doubles the amount of computational resources required for the experiment. In the current configuration, YAC can only be used when the block length is equal to the grid size, which limits us to small experiments like the bubble scenarios. Coupling SuperdropNet and ICON using pipes is almost three times slower than embedded Python. On a qualitative note, implementing the coupling via pipes requires changes to core components of ICON beyond the cloud microphysics parameterization and may be an additional challenge for ML developers.

4.3.2 Detailed evaluation for coupling with embedded Python

Process	Time (s)	Fraction
Time reported by ICON	5.0×10^{-4}	100%
Time reported by Python	4.8×10^{-4}	96%
↪ out of which time reported for data transfer	4.2×10^{-5}	8.5%
↪ out of which time reported for inference	4.4×10^{-4}	87%

Table 3. Processes when coupling SuperdropNet to ICON via embedded Python and their associated duration. Machine learning inference is executed on a CPU node of the Levante compute system at the German Climate Computing Center.

We now turn to the fastest coupling scheme, embedded Python, and investigate the contribution of the individual steps to the total runtime. By including SuperdropNet, we incur computational cost for data exchange and for machine learning inference.

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Table 3 summarizes the contribution of the individual parts, measured with a block length of $\mathrm{nproma} = 44$ grid cells using the ICON timer module. ICON averages the execution time across a total of $496,800$ calls to SuperdropNet. Most of the time can be attributed to model inference, while the actual data transfer is less significant. This could be attributed to the fact that ML inference has to be done on CPU. On a node equipped with an NVIDIA A100 GPU, we measure an inference time of $267\,\mu\mathrm{s}$. This corresponds to 33% of the inference time reported on a CPU (see Table 3). Note however that a heterogeneous setup, where moments are transfered to and from the GPU nodes via the Mellanox Infiniband network, would likely lead to a larger overall wall time. Given the successful efforts of porting ICON to GPU, a future experiment could be run exclusively on GPUs. By only applying SuperdropNet when at least one input moment is nonzero, we are already reducing the number of calls to ML inference to improve performance.

5 Conclusions

We have coupled SuperdropNet, a machine learning algorithm emulating warm rain processes in a two-moment cloud microphysics scheme, to ICON. In the warm bubble experiment, the ML emulator is stable, and the results are physically sound.

The strategies to bridge ICON and Python provide flexibility for the development of the ML component and account for the fact that ML development is done iteratively. Both embedded Python and YAC can be integrated with little programming overhead into ICON. For a later ML emulator, that replaces a full parameterization at the grid level, YAC can be used regardless of the block length. Coupling via pipes is comparatively slow and does not scale well. Since it requires an extensive rewrite of core components of ICON, we would not recommend it for implementation. Out of the three coupling strategies we tested, embedded Python provided the fastest performance. It can be used independent of the ICON grid to execute any Python code at any level of the ICON time loop.

We note that by coupling SuperdropNet to ICON we introduce a scheme that would otherwise be computationally intractable for cloud microphysics in a standard numerical simulations. A direct comparison of runtimes is therefore not possible. Note however that integrating a Python component will slow down the overall time to solution due to the incurred cost in network inference and data transfer. For applications that are more demanding than our warm bubble scenario test case, and if the ML component is thoroughly tested, a reimplementation in Fortran might increase performance, at the expense of loosing the flexibility of development.

A natural extension of this work are more complex modelling scenarios. This would involve training machine learning based emulators for other cloud microphysical processes and/or introduction of other hydrometeors apart from clouds and rain. Apart from droplet collisions, processes such as sedimentation of droplets and deep convection can be challenging to represent with bulk moment parameterization schemes. Hence, in the future we want to explore the possibility of creating ML based proxies for these processes while continuing to use hybrid ML-ESMs for continuous online testing.

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Code and data availability. The SuperdropNet (version 0.1.0) inference code, trained model weights, and modules describing the coupling between SuperdropNet inference and generic Fortran code, analysis scripts and Jupyter notebooks, as well as the experiment description files are available under the MIT license here: https://doi.org/10.5281/zenodo.10069121. The license file is included in the repository. The ICON model code used for the simulations in this paper is available under https://doi.org/10.5281/zenodo.8348256. It is based on the ICON release 2.6.5 and includes additional code for coupling SuperdropNet. Release versions of the code are available to individuals under license as described by MPI-M (2022). By downloading the ICON source code, the user accepts the license agreement that is included in the repository. Information about ICON licensing and obtaining access to different versions of ICON can be found at https://code.mpimet.mpg.de/projects/icon-license. The experiment results obtained with SuperdropNet (version 0.1.0) coupled to ICON (version 2.6.5) are available under https://doi.org/10.5281/zenodo.8348266. We used McSnow (version 1.1.0) for generating the training data in a warm rain scenario. McSnow is not publicly available. Access to McSnow can be granted upon agreeing to the ICON licensing terms by the developers of McSnow (Brdar and Seifert, 2018).

Author contributions. CA developed the embedded Python and YAC coupling software, performed the experiments, provided the visualizations, and led the writing of the manuscript as a whole. SS developed SuperdropNet. CA and SS defined and evaluated the experiments, curated software and data, and wrote the original draft. TW developed the pipes coupling software and contributed to the original draft. DG helped conceive the project, define the coupling task and contributed to the original draft. CA, SS, TW, and DG reviewed and edited the final manuscript.

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

Acknowledgements. We thank A.-K. Naumann and S. Rast for support with the ICON warm bubble scenario and helpful discussions, and we thank N.-A. Dreier and M. Hanke for support with integrating YAC. This work was supported by Helmholtz Association's Initiative and Networking Fund through Helmholtz AI [grant number: ZT-I-PF-5-01]. This work used resources of the Deutsches Klimarechenzentrum (DKRZ) granted by its Scientific Steering Committee (WLA) under project ID AIM.

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: A system for large-scale machine learning, in: 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283, https://arxiv.org/abs/1603.04467, 2016.

Alexeev, D.: PyTorch bindings for Fortran (v0.4), https://github.com/alexeedm/pytorch-fortran, 2023.

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

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

Bonanni, A., Hawkes, J., and Quintino, T.: infero: A lower-level API for Machine Learning inference in operations (version 0.2.0), https://github.com/ecmwf/infero, 2022.

Brdar, S. and Seifert, A.: McSnow: A Monte-Carlo Particle Model for Riming and Aggregation of Ice Particles in a Multidimensional Microphysical Phase Space, J. Adv. Model. Earth Sy., 10, 187–206, https://doi.org/10.1002/2017MS001167, 2018.

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.

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

Brenowitz, N. D., Beucler, T., Pritchard, M., and Bretherton, C. S.: Interpreting and Stabilizing Machine-Learning Parametrizations of Convection, J. Atmos. Sci., 77, 4357–4375, https://doi.org/10.1175/JAS-D-20-0082.1, 2020a.

Brenowitz, N. D., Henn, B., McGibbon, J., Clark, S. K., Kwa, A., Perkins, W. A., Watt-Meyer, O., and Bretherton, C. S.: Machine Learning Climate Model Dynamics: Offline versus Online Performance, https://doi.org/10.48550/arXiv.2011.03081, 2020b.

Brenowitz, N. D., Perkins, W. A., Nugent, J. M., Watt-Meyer, O., Clark, S. K., Kwa, A., Henn, B., McGibbon, J., and Bretherton, C. S.: Emulating Fast Processes in Climate Models, in: Machine Learning for the Physical Sciences, NEURIPS Workshop, https://doi.org/10.48550/arXiv.2211.10774, 2022.

Brenowitz, N.: Call Python from Fortran (v0.2.1), https://doi.org/10.5281/zenodo.7779572, https://github.com/nbren12/call_py_fort, 2023.

Chantry, M., Hatfield, S., Dueben, P., Polichtchouk, I., and Palmer, T.: Machine learning emulation of gravity wave drag in numerical weather forecasting, J. Adv. Model. Earth Sy., 13, e2021MS002 477, https://doi.org/10.1029/2021MS002477, 2021.

Chollet, F. et al.: Keras (v2.14.0), https://github.com/fchollet/keras, 2023.

Christensen, H. and Zanna, L.: Parametrization in Weather and Climate Models, in: Oxford Research Encyclopedia of Climate Science, https://doi.org/10.1093/acrefore/9780190228620.013.826, 2022.

Curcic, M.: A parallel Fortran framework for neural networks and deep learning, https://doi.org/10.48550/arXiv.1902.06714, 2019.

Dong, W., Fritts, D. C., Liu, A. Z., Lund, T. S., Liu, H.-L., and Snively, J.: Accelerating Atmospheric Gravity Wave Simulations Using Machine Learning: Kelvin-Helmholtz Instability and Mountain Wave Sources Driving Gravity Wave Breaking and Secondary Gravity Wave Generation, Geophys. Res. Lett., 50, https://doi.org/10.1029/2023GL104668, 2023.

Dueben, P., Modigliani, U., Geer, A., Siemen, S., Pappenberger, F., Bauer, P., Brown, A., Palkovic, M., Raoult, B., Wedi, N., and Baousis, V.: Machine learning at ECMWF: A roadmap for the next 10 years, ECMWF Technical Memoranda, https://doi.org/10.21957/ge7ckgm, 2021.

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Dueben, P. D., Schultz, M. G., Chantry, M., Gagne, D. J., Hall, D. M., and McGovern, A.: Challenges and Benchmark Datasets for Machine Learning in the Atmospheric Sciences: Definition, Status, and Outlook, Artificial Intelligence for the Earth Systems, 1, https://doi.org/10.1175/AIES-D-21-0002.1, 2022.

Elafrou, A., Orchard, D., and Cliffard, S.: fortran-pytorch-lib (commit: ffe833b66a6e1ce1c6cf023708d1f351a3a11f8b), https://github.com/Cambridge-ICCS/fortran-pytorch-lib, 2023.

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.

Grundner, A., Beucler, T., Gentine, P., Iglesias-Suarez, F., Giorgetta, M. A., and Eyring, V.: Deep Learning Based Cloud Cover Parameterization for ICON, J. Adv. Model. Earth Sy., 14, e2021MS002 959, https://doi.org/10.1029/2021MS002959, 2022.

Hanke, M., Redler, R., Holfeld, T., and Yastremsky, M.: YAC 1.2.0: new aspects for coupling software in Earth system modelling, Geosci. Model Dev., 9, 2755–2769, https://doi.org/10.5194/gmd-9-2755-2016, 2016.

Hanke, M., Dreier, N.-A., and Redler, R.: YetAnotherCoupler (YAC) (version 2.6.1), https://dkrz-sw.gitlab-pages.dkrz.de/yac/, 2023.

Irrgang, C., Boers, N., Sonnewald, M., Barnes, E. A., Kadow, C., Staneva, J., and Saynisch-Wagner, J.: Towards neural Earth system modelling by integrating artificial intelligence in Earth system science, Nature Machine Intelligence, 3, 667–674, https://doi.org/10.1038/s42256-021-00374-3, 2021.

McGovern, A., Lagerquist, R., Gagne, D. J., Jergensen, G. E., Elmore, K. L., Homeyer, C. R., and Smith, T.: Making the Black Box More Transparent: Understanding the Physical Implications of Machine Learning, B. Am. Meteorol. Soc., 100, 2175–2199, https://doi.org/10.1175/BAMS-D-18-0195.1, 2019.

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

Morrison, H., Lier-Walqui, M. v., Fridlind, A. M., Grabowski, W. W., Harrington, J. Y., Hoose, C., Korolev, A., Kumjian, M. R., Milbrandt, J. A., Pawlowska, H., Posselt, D. J., Prat, O. P., Reimel, K. J., Shima, S.-I., Diedenhoven, B. v., and Xue, L.: Confronting the Challenge of Modeling Cloud and Precipitation Microphysics, J. Adv. Model. Earth Sy., 12, e2019MS001 689, https://doi.org/10.1029/2019MS001689, 2020.

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, 104 016, https://doi.org/10.1088/1748-9326/aae2be, 2018.

Ott, J., Pritchard, M., Best, N., Linstead, E., Curcic, M., and Baldi, P.: A Fortran-Keras Deep Learning Bridge for Scientific Computing, Scientific Programming, 2020, Article ID 8888 811, https://doi.org/10.1155/2020/8888811, 2020.

Partee, S., Ellis, M., Rigazzi, A., Shao, A. E., Bachman, S., Marques, G., and Robbins, B.: Using Machine Learning at scale in numerical simulations with SmartSim: An application to ocean climate modeling, J. Comput. Sci.-Neth., 62, 101 707, https://doi.org/10.1016/j.jocs.2022.101707, 2022.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library, in: Advances in Neural Information Processing Systems 32, pp. 8024–8035, Curran Associates, Inc., http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf, 2019.

Qu, Y. and Shi, X.: Can a Machine Learning–Enabled Numerical Model Help Extend Effective Forecast Range through Consistently Trained Subgrid-Scale Models?, Artificial Intelligence for the Earth Systems, 2, https://doi.org/10.1175/AIES-D-22-0050.1, 2023.

https://doi.org/10.5194/egusphere-2023-2047
Preprint. Discussion started: 15 November 2023

Rasp, S.: Coupled online learning as a way to tackle instabilities and biases in neural network parameterizations: general algorithms and Lorenz 96 case study (v1.0), Geosci. Model Dev., 13, 2185–2196, https://doi.org/10.5194/gmd-13-2185-2020, 2020.

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

Rigo, A. and Fijalkowski, M.: C Foreign Function Interface for Python, https://cffi.readthedocs.io/en/release-1.14/, 2018.

Seifert, A. and Beheng, K.: A two-moment cloud microphysics parameterization for mixed-phase clouds. Part 1: Model description, Meteorol. Atmos. Phys., pp. 45–66, https://doi.org/10.1007/s00703-005-0112-4, 2006.

Seifert, A. and Beheng, K. D.: A double-moment parameterization for simulating autoconversion, accretion and selfcollection, Atmos. Res., 59–60, 265–281, https://doi.org/10.1016/s0169-8095(01)00126-0, 2001.

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

Sharma, S. and Greenberg, D.: SuperDropNet: Machine Learning Parameterization for Superdroplet Cloud Microphysics Scheme (in preparation), 2023.

Shima, S., Kusano, K., Kawano, A., Sugiyama, T., and Kawahara, S.: The super-droplet method for the numerical simulation of clouds and precipitation: a particle-based and probabilistic microphysics model coupled with a non-hydrostatic model, Q. J. Roy. Meteor. Soc., 135, 1307–1320, https://doi.org/10.1002/qj.441, 2009.

Sonnewald, M., Lguensat, R., Jones, D. C., Dueben, P. D., Brajard, J., and Balaji, V.: Bridging observations, theory and numerical simulation of the ocean using machine learning, Environ. Res. Lett., 16, 073 008, https://doi.org/10.1088/1748-9326/ac0eb0, 2021.

Yuval, J. and O'Gorman, P. A.: Neural-Network Parameterization of Subgrid Momentum Transport in the Atmosphere, J. Adv. Model. Earth Sy., 15, e2023MS003 606, https://doi.org/10.1029/2023MS003606, 2023.

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

Zhong, X., Ma, Z., Yao, Y., Xu, L., Wu, Y., and Wang, Z.: WRF–ML v1.0: a bridge between WRF v4.3 and machine learning parameterizations and its application to atmospheric radiative transfer, Geosci. Model Dev., 16, 199–209, https://doi.org/10.5194/gmd-16-199-2023, 2023.