Neural General Circulation Models

Dmitrii Kochkov[1*†], Janni Yuval[1*†], Ian Langmore[1†], Peter Norgaard[1†], Jamie Smith[1†], Griffin Mooers[1], James Lottes[1], Stephan Rasp[1], Peter Düben[3], Milan Klöwer[4], Sam Hatfield[3], Peter Battaglia[2], Alvaro Sanchez-Gonzalez[2], Matthew Willson[2], Michael P. Brenner[1,5], Stephan Hoyer[1*†]

[1]Google Research, Mountain View, CA.
[2]Google DeepMind, London, UK.
[3]European Centre for Medium-Range Weather Forecasts, Reading, UK.
[4]Earth, Atmospheric and Planetary Sciences, Massachusetts Institute of Technology.
[5]School of Engineering and Applied Sciences, Harvard University.

*Corresponding author(s). E-mail(s): dkochkov@google.com; janniyuval@google.com; shoyer@google.com;
†These authors contributed equally to this work.

Abstract

General circulation models (GCMs) are the foundation of weather and climate prediction. GCMs are physics-based simulators which combine a numerical solver for large-scale dynamics with tuned representations for small-scale processes such as cloud formation. Recently, machine learning (ML) models trained on reanalysis data achieved comparable or better skill than GCMs for deterministic weather forecasting. However, these models have not demonstrated improved ensemble forecasts, or shown sufficient stability for long-term weather and climate simulations. Here we present the first GCM that combines a differentiable solver for atmospheric dynamics with ML components, and show that it can generate forecasts of deterministic weather, ensemble weather and climate on par with the best ML and physics-based methods. NeuralGCM is competitive with ML models for 1-10 day forecasts, and with the European Centre for Medium-Range Weather Forecasts ensemble prediction for 1-15 day forecasts. With prescribed sea surface temperature, NeuralGCM can accurately track climate metrics such as global mean temperature for multiple decades, and climate forecasts with 140 km resolution exhibit emergent phenomena such as realistic frequency and trajectories of tropical cyclones. For both weather and climate, our approach offers

orders of magnitude computational savings over conventional GCMs. Our results show that end-to-end deep learning is compatible with tasks performed by conventional GCMs, and can enhance the large-scale physical simulations that are essential for understanding and predicting the Earth system.

Introduction

Solving the equations of the Earth's atmosphere with general circulation models (GCMs) is the basis of weather and climate prediction [1, 2]. Over the past 70 years, GCMs have been steadily improved with better numerical methods and more detailed physical models, while exploiting faster computers to run at higher resolution. Inside GCMs, the unresolved physical processes such as clouds, radiation and precipitation are represented by semi-empirical parameterizations. Tuning GCMs to match historical data remains a manual process [3], and GCMs retain many persistent errors and biases [4–6]. The difficulty of reducing uncertainty in long-term climate projections and estimating distributions of extreme weather events [7] presents major challenges for climate mitigation and adaptation [8].

Recent advances in machine learning (ML) have presented an alternative for weather forecasting [9–12]. These models rely solely on ML techniques, using roughly 40 years of historical data from the European Center for Medium-Range Weather Forecasts (ECMWF) reanalysis v5 (ERA5) [13] for model training and forecast initialization. ML methods have been remarkably successful, demonstrating state-of-the-art deterministic forecasts for 1-10 day weather prediction at a fraction of the computational cost of traditional models [11, 12]. ML atmospheric models also require considerably less code, for example GraphCast [12] has 5417 lines vs 376 578 lines for NOAA's FV3 atmospheric model [14] (see Appendix A for details).

Nevertheless, ML approaches have noteworthy limitations compared to GCMs. Existing ML models have focused on deterministic prediction, and surpass deterministic numerical weather prediction in terms of the aggregate metrics for which they are trained [11, 12]. However, they do not produce calibrated uncertainty estimates [11], which is essential for useful weather forecasts [1]. Deterministic ML models using a mean-squared-error loss are rewarded for averaging over uncertainty, producing unrealistically blurry predictions when optimized for multi-day forecasts [10, 12]. Unlike physical models, ML models misrepresent derived (diagnostic) variables such as geostrophic wind [15] and have not demonstrated the ability to run for the long timescales necessary for climate modeling.

Hybrid models that combine GCMs with machine learning are appealing because they build on the interpretability, extensibility and successful track record of traditional atmospheric models. In the hybrid model approach, an ML component replaces or corrects the traditional physical parameterizations of a GCM [16]. Up until now, the ML component in such models has been trained "offline," by learning parameterizations independently of their interaction with dynamics. These components are then inserted into an existing GCM. The lack of coupling between ML components and the governing equations during training potentially causes serious problems such

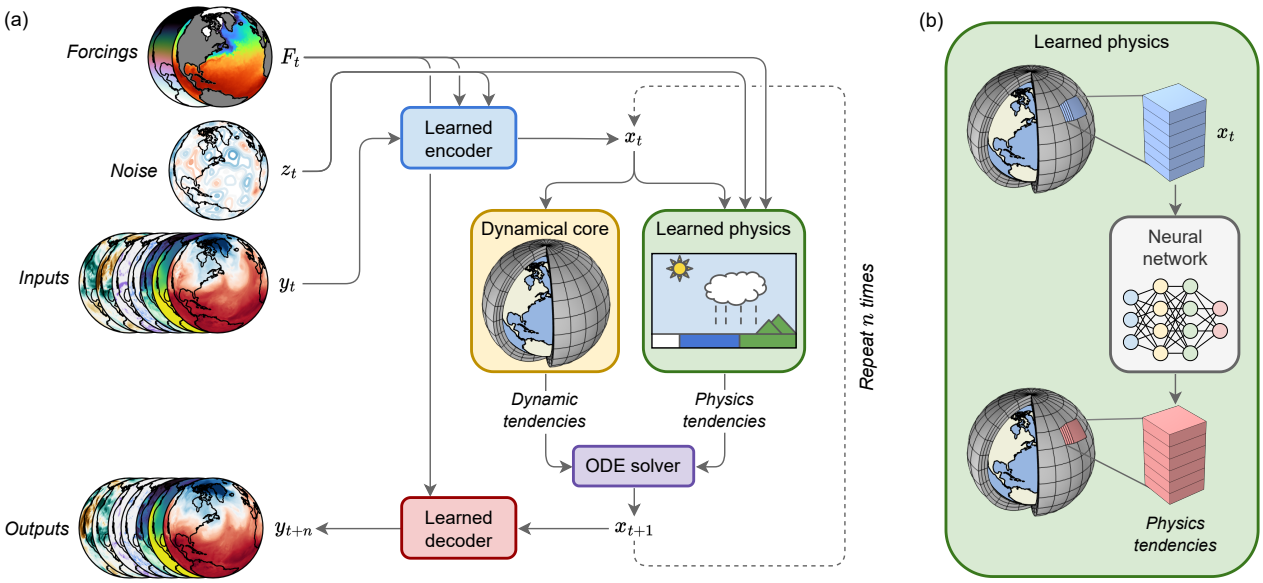

Fig. 1 Structure of the NeuralGCM model. (a) Overall model structure, showing how forcings F_t, noise z_t (for stochastic models), and inputs y_t are encoded into the model state x_t. Model state is fed into the dynamical core, and alongside forcings and noise into the learned physics module. This produces tendencies (rates of change) used by an implicit-explicit ODE solver to advance the state in time. The new model state x_{t+1} can then be fed back into another time step, or decoded into model predictions. (b) Inset of the learned physics module, which feeds data for individual columns of the atmosphere into a neural network used to produce physics tendencies in that vertical column.

as instability and climate drift [17]. So far, hybrid models have mostly been limited to idealized scenarios such as aquaplanets [18, 19]. Under realistic conditions, ML corrections have reduced biases of very coarse GCMs [20, 21], but performance remains considerably worse than state-of-the-art models.

Neural general circulation models

Here we present NeuralGCM, the first fully-differentiable hybrid general circulation model of the Earth's atmosphere. NeuralGCM is trained on forecasting up to 5-day weather trajectories sampled from ERA5. Differentiability enables end-to-end "online training" [22], with ML components optimized in the context of interactions with the governing equations for large-scale dynamics, which we find enables accurate and stable forecasts. NeuralGCM produces physically consistent forecasts with accuracy comparable to best-in-class models across a range of time-scales, from 1-15 day weather to decadal climate prediction.

A schematic of NeuralGCM is shown in Fig. 1. The two key components of NeuralGCM are a differentiable dynamical core for solving the discretized governing dynamical equations, and a learned physics module that parameterizes physical processes with a neural network, described in full detail in Appendix B and C. The

dynamical core simulates large-scale fluid motion and thermodynamics under the influence of gravity and the Coriolis force. The learned physics module predicts the effect of unresolved processes such as cloud formation, radiative transport and precipitation on the simulated fields using a neural network.

Our differentiable dynamical core is implemented in JAX, a library for high-performance code in Python that supports automatic differentiation [23]. The dynamical core solves the hydrostatic primitive equations with moisture, using a horizontal pseudo-spectral discretization and vertical sigma coordinates [24, 25]. We evolve seven prognostic variables: vorticity and divergence of horizontal wind, temperature, surface pressure, and three water species (specific humidity, and specific ice and liquid cloud water content).

Our learned physics module uses the single-column approach of GCMs [2], whereby information only from a single atmospheric column is used to predict the impact of unresolved processes occurring within that column. We use an encode-process-decode neural network architecture [26] built out of fully-connected layers, with weights shared across all atmospheric columns (C.4). The inputs to the neural network include the prognostic variables in the atmospheric column, total incident solar radiation, sea ice concentration and sea surface temperature (C.1). We also provide horizontal gradients of the prognostic variables, which we found improves performance [27]. All inputs are standardized to have zero mean and unit variance using statistics precomputed during model initialization. The outputs are the prognostic variable tendencies scaled by the fixed unconditional standard deviation of the target field (C.5).

To interface between ERA5 [13] data stored in pressure coordinates and the sigma coordinate system of our dynamical core, we introduce encoder and decoder components (Appendix D). These components perform linear interpolation between pressure levels and sigma coordinates levels. We additionally introduce learned corrections to both encoder and decoder steps, using the same column-based neural network architecture as the learned physics module. Importantly, the encoder enables us to eliminate the gravity waves from initialization shock [28], which otherwise contaminate forecasts. To align horizontal grids, we conservatively regrid ERA5 data from a regular longitude-latitude grid to the native Gaussian grid of NeuralGCM.

Figure 1(a) shows the sequence of steps that NeuralGCM takes to make a forecast. First, it encodes ERA5 data at $t = t_0$ on pressure levels to initial conditions on sigma coordinates. To perform a time step, the dynamical core and learned physics [Fig. 1(b)] then compute tendencies, which are integrated in time using an implicit-explicit ODE solver [29] (Appendix E). This is repeated to advance the model from $t = t_0$ to $t = t_{final}$. Finally, the decoder converts predictions back to pressure levels.

The time-step size of the ODE solver is limited by the CFL condition on dynamics, and can be small relative to the time-scale of atmospheric change. Evaluating learned physics is approximately 1.5× as expensive as a time step of the dynamical core. Accordingly, following the typical practice for GCMs, we hold learned physics tendencies constant for multiple ODE time-steps to reduce computational expense, typically corresponding to 30 minutes of simulation time.

Training

The differentiable dynamical core in NeuralGCM allows an end-to-end training approach, whereby we advance the model multiple time steps before employing stochastic gradient descent to minimize discrepancies between model predictions and ERA5 data (G.2). We gradually increase the rollout length from 6 hours to 5 days (Appendix G), which we found to be critical because our models are not accurate for multi-day prediction early in training. The extended backpropagation through hundreds of simulation steps enables our neural networks to take into account interactions between the learned physics and the dynamical core. We train deterministic and stochastic NeuralGCM models, each of which uses a distinct training protocol, described in full detail in G.4-G.6.

We train deterministic NeuralGCM models using a combination of three loss functions to encourage accuracy and sharpness while penalizing bias. During the main training phase, all losses are defined in a spherical harmonics basis. We use a standard MSE loss for prompting accuracy, modified to progressively filter out contributions from higher total wavenumbers at longer lead times. This filtering approach tackles the "double penalty problem" [30] as it prevents the model from being penalized for predicting high-wavenumber features in incorrect locations at later times, especially beyond the predictability horizon. A second loss term encourages the spectrum to match the training data using squared loss on the total wavenumber spectrum of prognostic variables. These first two losses are evaluated on both sigma and pressure levels. Finally, a third loss term discourages bias by adding MSE on the batch-averaged mean amplitude of each spherical harmonic coefficient. The combined action of the three training losses allow the resulting models trained on 3-day rollouts to remain stable during years-to-decades long simulations (Section 5). Before final evaluations, we perform additional fine-tuning of just the decoder component on short rollouts of 24 hours (G.5).

Stochastic NeuralGCM models incorporate inherent randomness into the forecasts, which enables us to produce ensemble weather forecasts. Our stochastic loss is based on the the Continuous Ranked Probability Score (CRPS) [31]. CRPS consists of mean absolute error that encourages accuracy, balanced by a similar term that encourages ensemble spread. For each variable we use a sum of CRPS in grid-space and CRPS in the spherical harmonic basis below a maximum cutoff wavenumber. As illustrated in Fig. 1, we inject noise by sampling from Gaussian random fields with learned spatial and temporal correlation (C.2) that are fed alongside other features into the learned encoder and learned physics module. For training, we generate two ensemble members per forecast, which suffices for an unbiased estimate of CRPS.

Results

We train a range of NeuralGCM models at horizontal resolutions with grid spacing of 2.8°, 1.4°, and 0.7°, each utilizing 32 evenly spaced vertical levels on sigma coordinates. We evaluate the performance of NeuralGCM at a range of timescales appropriate for weather forecasting and climate simulation. For weather, we compare against the best-in-class conventional physics-based weather models, ECMWF's high resolution model

(ECMWF-HRES) and ensemble prediction system (ECMWF-ENS), and two of the recent ML-based approaches, GraphCast [12] and Pangu [11]. For climate, we compare against a Global Cloud Resolving Model and Atmospheric Model Inter-comparison Project (AMIP) runs.

Medium-range global weather forecasting

Our evaluation setup focuses on quantifying accuracy and physical consistency, following WeatherBench2 [9]. We regrid all forecasts to a 1.5° grid using conservative regridding, and average over all 732 forecasts made at noon and midnight UTC in the year 2020, which was held-out from training data for all ML models. NeuralGCM, GraphCast, and Pangu compare to ERA5 as ground-truth, whereas ECMWF ENS/HRES compare to the ECMWF operational analysis (i.e., HRES at 0-hour lead time), to avoid penalizing the operational forecasts for different biases than ERA5.

Accuracy

We use ECMWF's ensemble (ENS) model as a reference baseline as it achieves the best performance across the majority of lead times [9]. We assess accuracy using (1) root mean squared error (RMSE), (2) bias, (3) continuous ranked probability score (CRPS) and (4) spread-skill ratio with results shown in Fig. 2. We provide more in-depth evaluations including scorecards, metrics for additional variables and levels, and maps in Appendix H.

Deterministic models that produce a single weather forecast for given initial conditions can be compared effectively using RMSE skill at short lead times. For the first 1-3 days, depending on the atmospheric variable, RMSE is minimized by forecasts that accurately track the evolution of weather patterns. At this timescale we find that NeuralCGM-0.7° and GraphCast achieve best results, with slight variations across different variables (Fig. 2a). At longer lead times RMSE rapidly increases due to chaotic divergence of nearby weather trajectories, making RMSE less informative for deterministic models. Root-mean-squared bias (RMSB) calculates persistent errors over time, which provides an indication of how models would perform at much longer lead times. Here NeuralGCM models also compare favorably against previous approaches (Fig. 2c), with notably much less bias for specific humidity in the tropics (Fig. 2d).

Ensembles are essential for capturing intrinsic uncertainty of weather forecasts, especially at longer lead times. Beyond about 7 days, the ensemble means of ECMWF-ENS and NeuralGCM-ENS forecasts have considerably lower RMSE than the deterministic models, indicating that these models better capture the average of possible weather. A better metric for ensemble models is CRPS, which as a proper scoring rule can only be minimized by predicting the true marginal probability distributions [31]. Our stochastic model (NeuralGCM-ENS) running at 1.4° resolution has lower error compared to ECMWF-ENS across almost all variables, lead times and vertical levels for ensemble-mean RMSE, RSMB and CRPS (Fig. 2a,c,e and Appendix H), with similar spatial patterns of skill (Fig. 2b,f). Like ECMWF-ENS, NeuralGCM-ENS has a spread-skill ratio of approximately one (Fig. 2d), which is a necessary condition for calibrated forecasts [32].

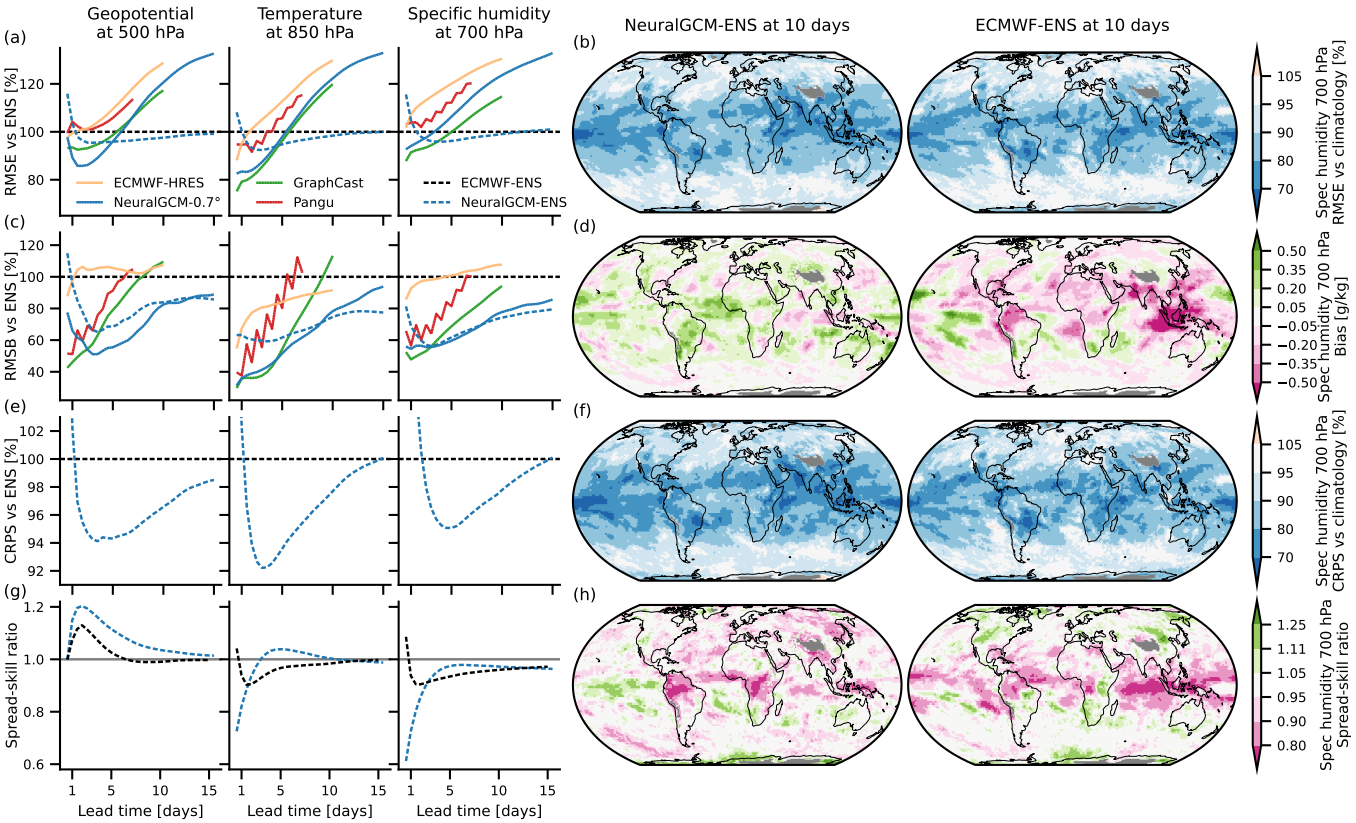

Fig. 2 Weather forecasting accuracy scores for deterministic and stochastic models. (a) RMSE and (c) RMSB for all models on the main WeatherBench variables, as a percent of the error of ECMWF-ENS. Deterministic and stochastic models are shown in solid and dashed lines respectively. (e) CRPS relative to ECMWF-ENS and (g) skill-spread ratio for ENS and NeuralGCM-ENS models. Spatial distributions of (b) RMSE, (d) Bias, (f) CRPS and (h) spread-skill ratio for NeuralGCM-ENS and ECMWF-ENS models for 10-day forecasts of specific humidity at 700 hPa. Spatial plots of RMSE and CRPS are relative to a probabilistic climatology with an ensemble member for each of the years 1990-2019.

Physical consistency

Case study. An important characteristic of forecasts is their resemblance to realistic weather patterns. Figure 3 shows a case study that illustrates the performance of NeuralGCM on three types of important weather phenomena: tropical cyclones, atmospheric rivers and the inter-tropical convergence zone. The top panel shows that all the ML models make significantly blurrier forecasts than source data ERA5 and physics-based ECMWF-HRES forecast, but NeuralCGM-0.7° outperforms the pure ML models, despite its coarser resolution (0.7° vs 0.25° for GraphCast and Pangu). Blurry forecasts correspond to physically inconsistent atmospheric conditions, and misrepresent extreme weather. Similar trends hold for other derived variables of meteorological interest (Appendix H.2). Ensemble mean predictions, both from NeuralGCM

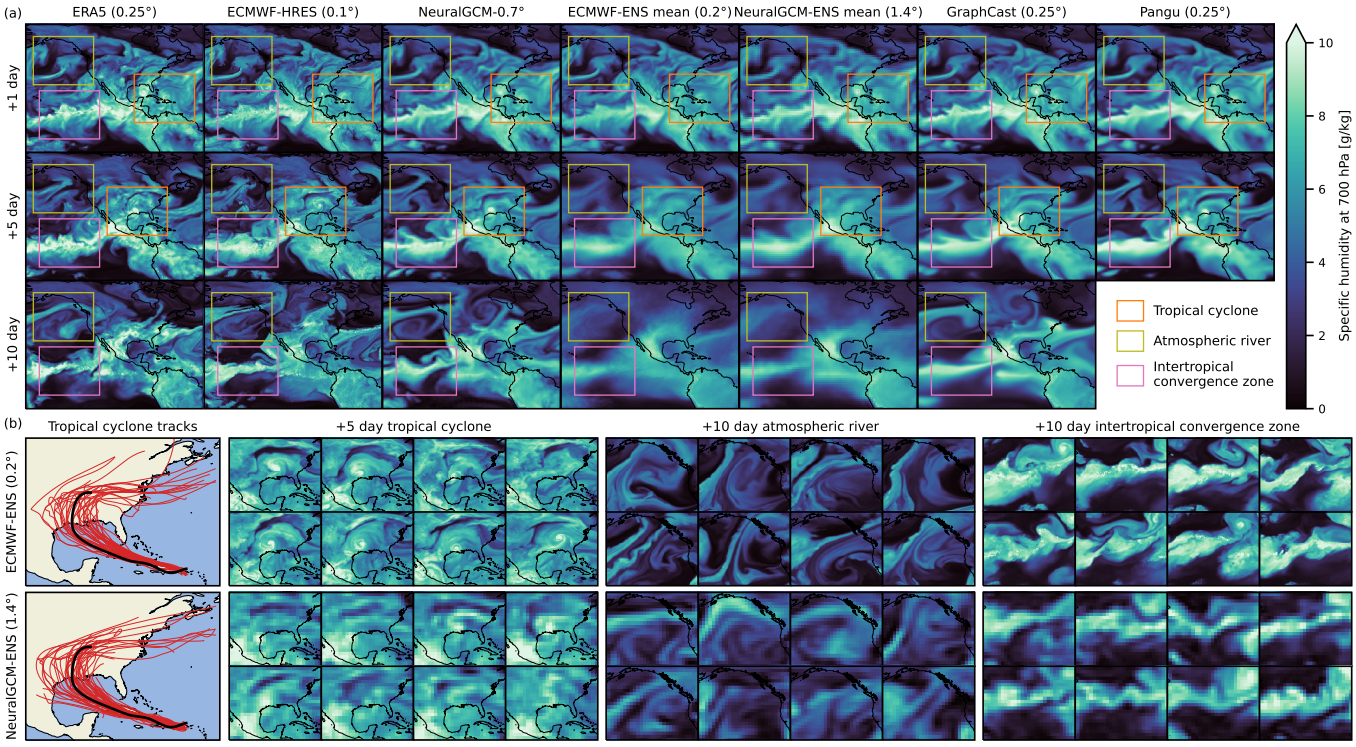

Fig. 3 Case study of a medium-range weather forecast. All forecasts are initialized at 2020-08-22T12z, chosen to highlight Hurricane Laura, the most damaging Atlantic hurricane of 2020. (a) Specific humidity at 700 hPa for 1-day, 5-day and 10-day forecasts from each model over North America and the North-East Pacific ocean. (b) Forecasts from individual ensemble members from ECMWF-ENS and NeuralGCM-ENS over regions of interest, including predicted tracks of Hurricane Laura from each of the 50 ensemble members (Appendix I.2). The track from ERA5 is plotted in black.

and ECMWF, are closer to ERA5 in an average sense, and thus are inherently smooth at long lead times. In contrast, as shown in the bottom panel and in Appendix H.3, individual realizations from the ECMWF and NeuralGCM ensembles remain sharp, even at long lead times. Like ECMWF-ENS, NeuralGCM-ENS produces a statistically representative range of future weather scenarios for each weather phenomena, despite its 8× coarser resolution.

Spectra. We can quantify the blurriness of different forecast models via their power spectra. Appendix Fig. H13 and Fig. H14 show that the power spectra of NeuralCGM-0.7° is consistently closer to ERA5 than the other ML forecast methods, but is still blurrier than ECMWF's physical forecasts. The spectra of NeuralGCM forecasts is also roughly constant over the forecast period, in stark contrast to GraphCast, which worsens with lead time. The spectrum of NeuralGCM becomes more accurate with increased resolution (H19), which suggests the potential for further improvements of NeuralGCM models trained at higher resolutions.

Water budget. In NeuralGCM, advection is handled by the dynamical core, while the ML parameterization models local processes within vertical columns of the atmosphere. Thus, unlike pure ML methods, local sources and sinks can be isolated from tendencies due to horizontal transport and other resolved dynamics. This makes our results more interpretable and facilitates the diagnosis of the water budget. Specifically, we diagnose precipitation minus evaporation (P-E, see H.6) rather than directly predicting these as in ML-based approaches [12]. For short weather forecasts, the mean of P-E has a realistic spatial distribution that is very close to ERA5 data (Fig. H21c-e). The P-E rate distribution of NeuralGCM-0.7° closely matches ERA5 distribution in the extratropics (Fig. H21b), though it underestimates extreme events in the tropics (Fig. H21a). Note that the current version of NeuralGCM directly predicts tendencies for an atmospheric column, and thus cannot distinguish between precipitation and evaporation.

Geostrophic wind balance. We examined the extent to which NeuralGCM, GraphCast, and ECMWF-HRES capture the geostrophic wind balance, the near-equilibrium between the dominant forces that drive large-scale dynamics in the mid-latitudes [33]. A recent study [15] highlighted that Pangu misrepresents the vertical structure of the geostrophic and ageostrophic winds, and noted a deterioration at longer lead times. Similarly, we observe that GraphCast exhibits an error that worsens with lead time. In contrast, NeuralGCM more accurately depicts the vertical structure of the geostrophic and ageostrophic winds, as well as their ratio, compared to GraphCast across various rollouts, when compared against ERA5 data (Fig. H15). However, ECMWF-HRES still exhibits a slightly closer alignment to ERA5 data than NeuralGCM does. Within NeuralGCM, the representation of the geostrophic wind's vertical structure only slightly degrades in the initial few days, showing no noticeable changes thereafter, particularly beyond day 5.

Generalizing to unseen data. Physically consistent weather models should still perform well in a different climate and for weather conditions for which they were not trained. To test generalization, we compare versions of NeuralCGM-0.7° and GraphCast trained through 2017 on five years of weather forecasts beyond the training period (2018-2022) in Fig. H20. Unlike GraphCast, NeuralGCM does not show a clear trend of increasing error when initialized further into the future from the training data.

Climate Simulations

Although our deterministic NeuralGCM models are trained to predict weather up to 3 days ahead, they are generally capable of simulating the atmosphere far beyond medium-range weather timescales. For extended climate simulations we prescribe historical sea surface temperature (SST) and sea ice concentration. These simulations feature many emergent phenomena of the atmosphere on timescales from months to decades.

For climate simulations with NeuralGCM, we use 2.8° and 1.4° deterministic models, which are relatively inexpensive to train (Appendix G.7) and allow us to explore a larger parameter space to find stable models. Previous studies found that running extended simulations with hybrid models is challenging due to numerical instabilities

and climate drift [17]. To quantify stability in our selected models, we run multiple initial conditions and report how many of them finish without instability.

Seasonal cycle and emergent phenomena

To assess the capability of NeuralGCM to simulate various aspects of the seasonal cycle, we run two-year simulations with NeuralGCM-1.4°. for 37 different initial conditions spaced every 10 days for the year 2019. Out of these 37 initial conditions, 35 successfully complete the full two years without instability. We compare results from NeuralGCM-1.4° for 2020 to ERA5 data and to outputs from GFDL's X-SHiELD global cloud resolving model, which is coupled to an ocean model nudged towards reanalysis [34]. This X-SHiELD run has been used as a target for training ML climate models [20]. For fair comparison we evaluate models after regridding predictions to 1.4° resolution.

Figure 4a displays the temporal variation of the global mean temperature throughout 2020, as captured by 35 simulations from NeuralGCM, in comparison to the ERA5 reanalysis and standard climatology benchmarks. The seasonality and variability of the global mean temperature from NeuralGCM are quantitatively similar to those observed in ERA5. The ensemble mean temperature RMSE for NeuralGCM stands at 0.16K when benchmarked against ERA5, which is a significant improvement over the climatology's RMSE of 0.45K. We find that NeuralGCM accurately simulates the seasonal cycle, as evidenced by metrics such as the annual cycle of the global precipitable water (Fig. I24a) and global total kinetic energy (Fig. I24b). Furthermore, the model captures essential atmospheric dynamics, including the Hadley circulation and the zonal-mean zonal wind (Fig. I22), as well as the spatial patterns of eddy kinetic energy in different seasons (Fig. I25), and the distinctive seasonal behaviors of monsoon circulation (Fig. I23; additional details are provided in I.1).

Next, we compare the annual biases of a single NeuralGCM realization with a single realization of X-SHiELD (the only one available), both initiated in mid-October 2019. We consider 19 January 2020 to 17 January 2021, the time frame for which X-SHiELD data is available. Global cloud-resolving models, such as X-SHiELD, are considered state-of-the-art, especially for simulating the hydrological cycle, due to their resolution being capable of resolving deep convection [35]. The annual bias in precipitable water for NeuralGCM (RMSE of 1.09 mm) is substantially smaller than the biases of both X-SHiELD (RMSE of 1.74 mm) and climatology (RMSE of 1.36 mm; Fig. 4h-j). Moreover, NeuralGCM shows a lower temperature bias in the upper and lower troposphere than X-SHiELD (Fig. I27). We also indirectly compare precipitation bias in X-SHiELD with precipitation minus evaporation bias in NeuralGCM-1.4°, which shows slightly larger bias and grid-scale artifacts for NeuralGCM (Fig. I26).

Finally, to assess NeuralGCM capability of generating tropical cyclones in an annual model integration, we use the tropical cyclone tracker TempestExtremes [36], as described in Appendix I.2. Figure 4d,e,f shows that NeuralGCM, even at a coarse resolution of 1.4°, produces realistic trajectories and numbers of tropical cyclones (83 versus 86 in ERA5 for the corresponding period; the average number of TCs when averaging over all initial conditions in NueralGCM is 88.7 and standard deviation of

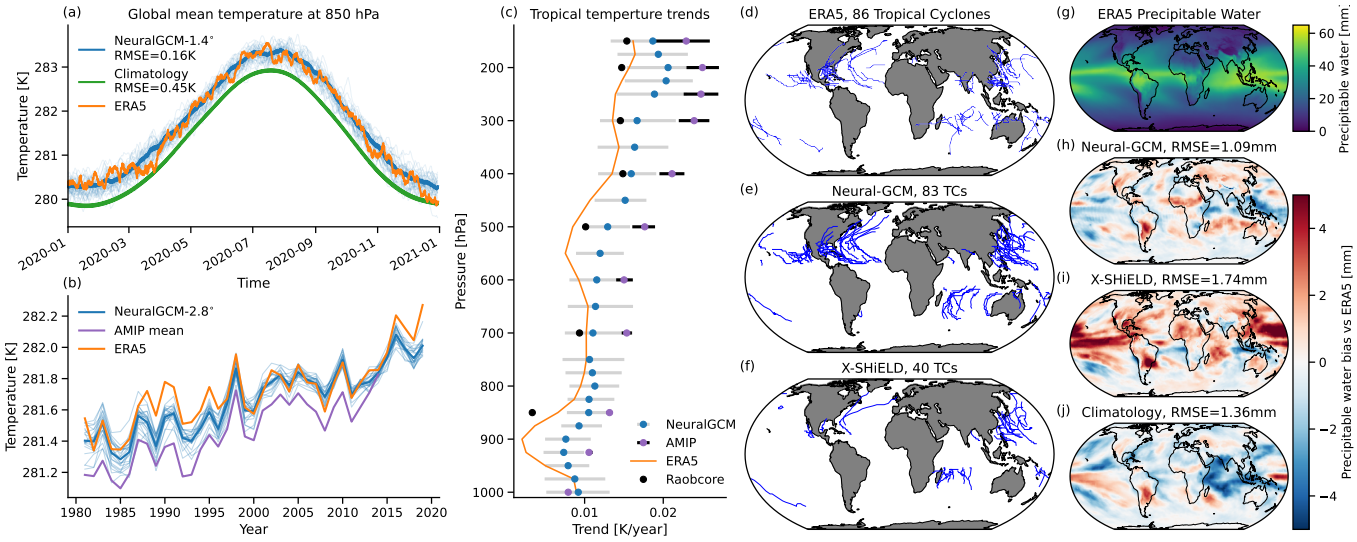

Fig. 4 (a) Global mean temperature for ERA5 for 2020 (orange), climatology (defined as the averaged temperature between 1990-2019; green), and for NeuralGCM-1.4° for 2020 for 35 simulations initialized every 10 days during 2019 (thick blue represents the ensemble mean; thin blue lines indicate different initial conditions). (b) Yearly global mean temperature for ERA5 (orange), mean over 22 CMIP6 AMIP experiments for 1981-2014 (purple; model details are found in Appendix I.3), and NeuralGCM-2.8° for 22 AMIP-like simulations with prescribed SST initialized every 10 days during 1980 (thick blue represents the ensemble mean, and thin blue lines indicate different initial conditions). (c) Vertical profiles of tropical (20S-20N) temperature trends for the period 1981-2014. The orange line shows ERA5 reanalysis, and the black dots show adjusted radiosonde data (Raobcore). The blue dots shows the mean trends for NeuralGCM-2.8° 22 AMIP-like runs and purple dots are the mean trends from CMIP6 AMIP runs (see appendix I.3 for full list of models used), where the grey (black) whiskers show the 25th and 75th percentiles for NeuralGCM-2.8° (CMIP6 AMIP runs). (d,e,f) Tropical cyclone tracks for (d) ERA5, (e) NeuralGCM-1.4° and (f) X-SHiELD. (g) Mean precipitable water for ERA5 and the precipitable water bias in (h) NeuralGCM-1.4°; initialized 90 days before mid-January 2020 similarly to X-SHiELD, (i) X-SHiELD and (j) climatology (averaged between 1990-2019). In panels d-i quantities are calculated between mid-January 2020 and mid-January 2021 (when X-SHiELD model data is available) and all models were regridded to a 256x128 Gaussian grid before computation and tracking.

6.9 and more detailed analysis in Fig. I30), while X-SHiELD, when regridded to 1.4° resolution, substantially underestimates the number (40).

Decadal simulations - AMIP-like experiments

To assess the capability of NeuralGCM to simulate historical temperature trends, we conduct AMIP-like simulations over a duration of 40 years with NeuralGCM-2.8°. Out of 37 different runs with initial conditions spaced every 10 days during the year 1980, 22 simulations were stable for the entire 40-year period, and our analysis focuses on these results. We compare to 22 simulations run with prescribed SST from the Coupled Model Intercomparison Project Phase 6 [37], listed in Appendix I.3.

We find that all 40-year simulations of NeuralGCM, as well as the mean of the 22 AMIP runs, accurately capture the global warming trends observed in ERA5 data (Fig. 4b). There is a strong correlation in the year-to-year temperature trends with ERA5 data, suggesting that NeuralGCM effectively captures the impact of SST forcing on climate. When comparing spatial temperature biases in 2014, the last year for which some AMIP data is available, we find that the NeuralGCM ensemble exhibits less temperature bias than the AMIP ensemble (Fig. I28).

Lastly, we investigated the vertical structure of tropical warming trends, which climate models tend to overestimate in the upper troposphere [38]. As shown in Fig. 4c, the trends, calculated by linear regression, of NeuralGCM are closer to ERA5 than those of AMIP runs. In particular, the bias in the upper troposphere is reduced. NeuralGCM does show a wider spread in its predictions than the AMIP runs, even at levels near the surface where temperatures are typically more constrained by prescribed SST.

Discussion

NeuralGCM is a differentiable hybrid atmospheric model that combines the strengths of traditional GCMs with machine learning for weather forecasting and climate simulation. NeuralGCM is the first ML-based model to make accurate ensemble weather forecasts, with better CRPS than state-of-the-art physics-based models. It is also the first hybrid model that achieves comparable spatial bias to global cloud resolving models, can simulate realistic tropical cyclone tracks, and that can run AMIP-like simulations with realistic historical temperature trends. Overall, NeuralGCM demonstrates that incorporating machine learning is a viable alternative to building increasingly detailed physical models for improving GCMs.

Compared to traditional GCMs with similar skill, NeuralGCM is compute efficient and low-complexity. NeuralGCM runs at 8-40$\times$ coarser horizontal resolution than ECMWF IFS and global cloud resolving models, which enables 3-5 orders of magnitude savings in compute. For example, NeuralGCM-1.4° simulates 70 000 simulation days in 24 hours using a single TPU vs 19 simulated days on 13 824 CPU cores with X-SHiELD. This can be leveraged for previously impractical tasks like large ensemble forecasting. NeuralGCM's dynamical core uses global spectral methods [24], and learned physics is parameterized with fully-connected neural networks acting on single vertical columns. Substantial headroom exists to pursue higher accuracy using advanced numerical methods and ML architectures.

Our results provide strong evidence for the disputed hypothesis [39, 40] that learning to predict short-term weather is an effective way to tune parameterizations for climate. NeuralGCM models trained on 72-hour forecasts are capable of realistic multi-year simulation. When provided with historical sea surface temperatures, they capture essential atmospheric dynamics such as seasonal circulation, monsoons, and tropical cyclones. We note that stability and long-term climate consistency is a challenge for conventional climate models, and similarly for NeuralGCM understanding the mechanisms leading to stable models remains a topic of ongoing research.

The NeuralGCM approach is compatible with incorporating either more physics or more ML, as required for operational weather forecasts and climate simulations.

For weather forecasting, we expect that end-to-end learning [41] with observational data will allow for better and more relevant predictions, including key variables such as precipitation. Such models could include neural networks acting as corrections to traditional data assimilation and model diagnostics. In contrast, climate modeling will require incorporating processes that are likely not learned on weather timescales, such as ocean dynamics, biogeochemistry and radiative transport. Climate change also introduces strong generalization challenges, e.g., in the form of unprecedented temperatures and greenhouse gas concentrations.

Models based on physical laws and empirical relationships are ubiquitous in science. We believe the differentiable hybrid modeling approach of NeuralGCM has the potential to transform simulation for a wide range of applications, such as materials discovery, protein folding, and multiphysics engineering design.

Acknowledgments. We thank Anna Kwa, Alex Merose, and Kunal Shah for assistance with data acquisition and handling, Leonardo Zepeda-Núñez for feedback on the manuscript, and John Anderson, Christopher Van Arsdale, Rei Chemke, Gideon Dresdner, Justin Gilmer, Jason Hickey, Nicholas Lutsko, Grey Nearing, Adam Paszke, John Platt, Sameera Ponda, Mike Pritchard, Daniel Rothenberg, Fei Sha and Octavian Voicu for productive discussions.

References

[1] Bauer, P., Thorpe, A. & Brunet, G. The quiet revolution of numerical weather prediction. *Nature* **525**, 47–55 (2015). URL http://dx.doi.org/10.1038/nature14956.

[2] Balaji, V. *et al.* Are general circulation models obsolete? *Proc. Natl. Acad. Sci. U. S. A.* **119**, e2202075119 (2022). URL http://dx.doi.org/10.1073/pnas.2202075119.

[3] Hourdin, F. *et al.* The art and science of climate model tuning. *Bull. Am. Meteorol. Soc.* **98**, 589–602 (2017). URL https://doi.org/10.1175/BAMS-D-15-00135.1.

[4] Bony, S. & Dufresne, J.-L. Marine boundary layer clouds at the heart of tropical cloud feedback uncertainties in climate models. *Geophysical Research Letters* **32** (2005).

[5] Webb, M. J., Lambert, F. H. & Gregory, J. M. Origins of differences in climate sensitivity, forcing and feedback in climate models. *Climate Dynamics* **40**, 677–707 (2013).

[6] Sherwood, S. C., Bony, S. & Dufresne, J.-L. Spread in model climate sensitivity traced to atmospheric convective mixing. *Nature* **505**, 37–42 (2014).

[7] Fischer, E. M., Beyerle, U. & Knutti, R. Robust spatially aggregated projections of climate extremes. *Nature Climate Change* **3**, 1033–1038 (2013).

[8] Field, C. B. *Managing the risks of extreme events and disasters to advance climate change adaptation: special report of the intergovernmental panel on climate change* (Cambridge University Press, Cambridge, UK, 2012).

[9] Rasp, S. *et al.* WeatherBench 2: A benchmark for the next generation of data-driven global weather models (2023). Preprint at http://arxiv.org/abs/2308.15560.

[10] Keisler, R. Forecasting global weather with graph neural networks. *arXiv preprint arXiv:2202.07575* (2022).

[11] Bi, K. *et al.* Accurate medium-range global weather forecasting with 3d neural networks. *Nature* **619**, 533–538 (2023).

[12] Lam, R. *et al.* Graphcast: Learning skillful medium-range global weather forecasting. *arXiv preprint arXiv:2212.12794* (2022).

[13] Hersbach, H. *et al.* The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society* **146**, 1999–2049 (2020).

[14] Zhou, L. *et al.* Toward convective-scale prediction within the next generation global prediction system. *Bull. Am. Meteorol. Soc.* **100**, 1225–1243 (2019). URL https://journals.ametsoc.org/view/journals/bams/100/7/bams-d-17-0246.1.xml.

[15] Bonavita, M. On the limitations of data-driven weather forecasting models. *arXiv preprint arXiv:2309.08473* (2023).

[16] Bretherton, C. S. Old dog, new trick: Reservoir computing advances machine learning for climate modeling. *Geophys. Res. Lett.* **50** (2023). URL https://agupubs.onlinelibrary.wiley.com/doi/10.1029/2023GL104174.

[17] Brenowitz, N. D. & Bretherton, C. S. Spatially extended tests of a neural network parametrization trained by coarse-graining. *Journal of Advances in Modeling Earth Systems* **11**, 2728–2744 (2019).

[18] Rasp, S., Pritchard, M. S. & Gentine, P. Deep learning to represent subgrid processes in climate models. *Proceedings of the National Academy of Sciences* **115**, 9684–9689 (2018).

[19] Yuval, J. & O'Gorman, P. A. Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature communications* **11**, 3295 (2020).

[20] Kwa, A. *et al.* Machine-learned climate model corrections from a global storm-resolving model: Performance across the annual cycle. *Journal of Advances in Modeling Earth Systems* **15**, e2022MS003400 (2023).

[21] Arcomano, T., Szunyogh, I., Wikner, A., Hunt, B. R. & Ott, E. A hybrid atmospheric model incorporating machine learning can capture dynamical processes not captured by its physics-based component. *Geophysical Research Letters* **50**, e2022GL102649 (2023).

[22] Um, K., Brand, R., Fei, Y. R., Holl, P. & Thuerey, N. Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers. *Advances in Neural Information Processing Systems* **33**, 6111–6122 (2020).

[23] Bradbury, J. *et al.* JAX: composable transformations of Python+NumPy programs (2018). URL http://github.com/google/jax.

[24] Bourke, W. A Multi-Level Spectral Model. I. Formulation and Hemispheric Integrations. *Mon. Weather Rev.* **102**, 687–701 (1974). URL https://journals.ametsoc.org/view/journals/mwre/102/10/1520-0493_1974_102_0687_amlsmi_2_0_co_2.xml.

[25] Durran, D. R. *Numerical methods for fluid dynamics: With applications to geophysics* Second edn, Vol. 32 (Springer, New York, 2010).

[26] Battaglia, P. W. *et al.* Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261* (2018).

[27] Wang, P., Yuval, J. & O'Gorman, P. A. Non-local parameterization of atmospheric subgrid processes with neural networks. *Journal of Advances in Modeling Earth Systems* **14**, e2022MS002984 (2022).

[28] Daley, R. Normal mode initialization. *Reviews of Geophysics* **19**, 450–468 (1981).

[29] Whitaker, J. S. & Kar, S. K. Implicit–explicit runge–kutta methods for fast–slow wave problems. *Monthly weather review* **141**, 3426–3434 (2013).

[30] Gilleland, E., Ahijevych, D., Brown, B. G., Casati, B. & Ebert, E. E. Intercomparison of spatial forecast verification methods. *Weather and forecasting* **24**, 1416–1430 (2009).

[31] Gneiting, T. & Raftery, A. E. Strictly Proper Scoring Rules, Prediction, and Estimation. *J. Am. Stat. Assoc.* **102**, 359–378 (2007). URL https://doi.org/10.1198/016214506000001437.

[32] Fortin, V., Abaza, M., Anctil, F. & Turcotte, R. Why should ensemble spread match the RMSE of the ensemble mean? *J. Hydrometeorol.* **15**, 1708–1713 (2014). URL http://journals.ametsoc.org/doi/10.1175/JHM-D-14-0008.1.

[33] Holton, J. R. *An introduction to dynamic meteorology* Fifth edn (Elsevier Academic Press, Waltham, MA, USA, 2004).

[34] Cheng, K.-Y. *et al.* Impact of warmer sea surface temperature on the global pattern of intense convection: insights from a global storm resolving model. *Geophysical Research Letters* **49**, e2022GL099796 (2022).

[35] Stevens, B. *et al.* Dyamond: the dynamics of the atmospheric general circulation modeled on non-hydrostatic domains. *Progress in Earth and Planetary Science* **6**, 1–17 (2019).

[36] Ullrich, P. A. *et al.* Tempestextremes v2. 1: A community framework for feature detection, tracking, and analysis in large datasets. *Geoscientific Model Development* **14**, 5023–5048 (2021).

[37] Eyring, V. *et al.* Overview of the coupled model intercomparison project phase 6 (cmip6) experimental design and organization. *Geoscientific Model Development* **9**, 1937–1958 (2016).

[38] Mitchell, D. M., Lo, Y. E., Seviour, W. J., Haimberger, L. & Polvani, L. M. The vertical profile of recent tropical temperature trends: Persistent model biases in the context of internal variability. *Environmental Research Letters* **15**, 1040b4 (2020).

[39] Ruiz, J. J., Pulido, M. & Miyoshi, T. Estimating model parameters with ensemble-based data assimilation: A review. *Journal of the Meteorological Society of Japan. Ser. II* **91**, 79–99 (2013).

[40] Schneider, T., Lan, S., Stuart, A. & Teixeira, J. Earth system modeling 2.0: A blueprint for models that learn from observations and targeted high-resolution simulations. *Geophysical Research Letters* **44**, 12–396 (2017).

[41] Sutskever, I., Vinyals, O. & Le, Q. V. Sequence to sequence learning with neural networks. *Advances in neural information processing systems* **27** (2014).

[42] Phillips, N. A. A coordinate system having some special advantages for numerical forecasting. *J. Meteor.* **14**, 184–185 (1957).

[43] Jouppi, N. *et al. Tpu v4: An optically reconfigurable supercomputer for machine learning with hardware support for embeddings*, ISCA '23 (Association for Computing Machinery, New York, NY, USA, 2023). URL https://doi.org/10.1145/3579371.3589350.

[44] Henry, G., Tang, P. T. P. & Heinecke, A. *Leveraging the bfloat16 artificial intelligence datatype for higher-precision computations* (IEEE, Kyoto, Japan, 2019). URL https://ieeexplore.ieee.org/document/8877427/.

[45] Xu, Y. *et al.* GSPMD: General and scalable parallelization for ML computation graphs (2021). URL http://arxiv.org/abs/2105.04663.

[46] Douglas, S., Vikram, S., Bradbury, J., Zhang, Q. & Johnson, M. shmap (shard_map) for simple per-device code. https://jax.readthedocs.io/en/latest/jep/14273-shard-map.html (2023). URL https://jax.readthedocs.io/en/latest/jep/14273-shard-map.html. Accessed: 2023-10-21.

[47] Palmer, T. N. *et al.* Stochastic parametrization and model uncertainty. https://www.ecmwf.int/sites/default/files/elibrary/2009/11577-stochastic-parametrization-and-model-uncertainty.pdf (2009). URL https://www.ecmwf.int/sites/default/files/elibrary/2009/11577-stochastic-parametrization-and-model-uncertainty.pdf.

[48] Jablonowski, C. & Williamson, D. L. The pros and cons of diffusion, filters and fixers in atmospheric general circulation models. *Numerical techniques for global atmospheric models* 381–493 (2011).

[49] Jablonowski, C. & Williamson, D. L. in *The pros and cons of diffusion, filters and fixers in atmospheric general circulation models* (eds Lauritzen, P., Jablonowski, C., Taylor, M. & Nair, R.) *Numerical Techniques for Global Atmospheric Models* 381–493 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2011). URL https://doi.org/10.1007/978-3-642-11640-7_13.

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

[51] May, R. M. *et al.* Metpy: A meteorological python library for data analysis and visualization. *Bulletin of the American Meteorological Society* **103**, E2273 – E2284 (2022). URL https://journals.ametsoc.org/view/journals/bams/103/10/BAMS-D-21-0125.1.xml.

[52] Roberts, M. J. *et al.* Impact of model resolution on tropical cyclone simulation using the highresmip–primavera multimodel ensemble. *Journal of Climate* **33**, 2557–2583 (2020).

Appendices

Appendix A Lines of code in atmospheric models

To measure of the complexity of different code-bases, we counted the number of "core model" lines of code, excluding files devoted to tests, examples, input/output, coupling between different frameworks and running models.

A.1 fv3atm

We counted a total of 376 578 lines of core model code for fv3atm, the atmospheric component of the NOAA's Unified Forecast System (UFS) weather model, which we downloaded from https://github.com/NOAA-EMC/fv3atm on 7 November, 2023.

Excluding files with "test" or "example" in their paths, we counted the following number of lines in Fortran files in the two major modules of fv3atm (ending in .f or .f90):

1. 42 387 lines in the FV3 dynamical core (atmos_cubed_sphere/model)
2. 334 191 lines in the Common Community Physics Package (CCPP) Physics (ccpp/physics/physics)

A.2 GraphCast

We counted a total of 5417 lines of core model code for GraphCast [12], which we downloaded from https://github.com/deepmind/graphcast on 7 November, 2023.

A.3 NeuralGCM

We counted a total of 20 136 lines of core model code for NeuralGCM, split between two major modules:

1. 8609 lines in the spectral dynamical core
2. 11 527 lines in the machine learning code

Appendix B Dynamical core of NeuralGCM

The dynamical core provides NeuralGCM with strong physics priors based on well understood and easy to simulate phenomena. In section B.1 we provide more details on spatial discretization of the atmospheric state in NeuralGCM. In section B.2 we summarize the governing equations of the dynamical core. In section B.3 we provide references to numerical implementations and rationale for our choices.

B.1 Discretization of the dynamical core

Our dynamical core uses a Gaussian grid and sigma coordinates [42] to discretize the computational domain. Gaussian grids enable fast and accurate transformations between the grid space representation and spherical harmonics basis. They result in equiangular longitude lines and unequal spacing latitudes defined by the Gaussian quadrature. Terrain-following sigma coordinates discretize the vertical direction by the fraction of the surface pressure, and thus correspond to non-stationary vertical height since surface pressure changes with time. Cell boundaries in sigma coordinates take values $\sigma \in [0, 1]$, with $\sigma = 0$ corresponding to the top of the atmosphere ($p = 0$ pressure boundary) and $\sigma = 1$ representing the earth's surface.

In this work we have trained a lineup of models that make forecasts at varying horizontal resolutions: 2.8°, 1.4°, and 0.7°, corresponding to truncated linear Gaussian grids TL63, TL127, TL255. The number in the grid name corresponds to the maximum total wavenumber of spherical harmonic that the grid can represent. These grids provide a framework for transforming data from grid space (nodal) to spherical harmonic representations with minimal loss of information. When solving model equations we use cubic truncation Gaussian grids T62, T125 and T253, that capture a similar number of spherical harmonics, while avoiding aliasing errors and minimizing the need to increase array dimensions above a multiple of 128, which is expensive on the Google TPU. See Table B.1 for resolution details. All models use 32 equidistant sigma levels for vertical discretization. We suspect that using higher vertical resolution with assimilation data from more levels could further improve the performance.

Grid name	Longitude nodes	Latitude nodes	Max total wavenumber
TL63	128	64	63
TL127	256	128	127
TL255	512	256	255
T62	190	95	62
T125	379	190	125
T254	766	383	254

Table B1 Spatial and spectral resolutions of horizontal grids used by NeuralGCM.

B.2 Primitive equations

The dynamical core of NeuralGCM solves the primitive equations, which represent a combination of (1) momentum equations, (2) the second law of thermodynamics,

(3) a thermodynamic equation of state (ideal gas), (4) continuity equation and (5) hydrostatic approximation. For solving the equations we use a divergence-vorticity representation of the horizontal winds, resulting in equations for the following seven prognostic variables: divergence δ, vorticity ζ, temperature T, logarithm of the surface pressure $\log p_s$, as well as 3 moisture species (specific humidity q, specific cloud ice q_{c_i} and specific liquid cloud water content q_{c_l}). To facilitate efficient time integration of our models we split temperature T into a uniform reference temperature on each sigma level $\bar{T}_\sigma$ and temperature variations per level $T'_\sigma = T_\sigma - \bar{T}_\sigma$. The resulting equations are:

$$
\begin{aligned}
\frac{\partial \zeta}{\partial t} &= -\nabla \times \left((\zeta + f)\mathbf{k} \times \mathbf{u} + \dot{\sigma} \frac{\partial \mathbf{u}}{\partial \sigma} + RT' \nabla \log p_s \right) \\
\frac{\partial \delta}{\partial t} &= -\nabla \cdot \left((\zeta + f)\mathbf{k} \times \mathbf{u} + \dot{\sigma} \frac{\partial \mathbf{u}}{\partial \sigma} + RT' \nabla \log p_s \right) - \nabla^2 \left(\frac{||\mathbf{u}||^2}{2} + \Phi + R\bar{T} \log p_s \right) \\
\frac{\partial T}{\partial t} &= -\mathbf{u} \cdot \nabla T - \dot{\sigma} \frac{\partial T}{\partial \sigma} + \frac{\kappa T \omega}{p} = -\nabla \cdot \mathbf{u} T' + T' \delta - \dot{\sigma} \frac{\partial T}{\partial \sigma} + \frac{\kappa T \omega}{p} \\
\frac{\partial q_i}{\partial t} &= -\nabla \cdot \mathbf{u} q_i + q_i \delta - \dot{\sigma} \frac{\partial q_i}{\partial \sigma} \\
\frac{\partial \log p_s}{\partial t} &= -\frac{1}{p_s} \int_0^1 \nabla \cdot (\mathbf{u} p_s) d\sigma = -\int_0^1 \left(\delta + \mathbf{u} \cdot \nabla \log p_s \right) d\sigma
\end{aligned}
\tag{B1}
$$

with horizontal velocity vector $\mathbf{u} = \nabla(\Delta^{-1}\delta) + \mathbf{k} \times \nabla(\Delta^{-1}\zeta)$, Coriolis parameter f, upward-directed unit vector parallel to the z-axis $\mathbf{k}$, ideal gas constant R, heat capacity at constant pressure C_p, $\kappa = \frac{R}{C_p}$, diagnosed vertical velocity in sigma coordinates $\dot{\sigma}$, diagnosed change in pressure of a fluid parcel $\omega \equiv \frac{dp}{dt}$, diagnosed geopotential Φ, diagnosed virtual temperature T_ν and each moisture species denoted as q_i.

Diagnostic quantities are computed as follows:

$$\dot{\sigma}_{k+\frac{1}{2}} = -\sigma_{k+\frac{1}{2}} \frac{\partial \log p_s}{\partial t} - \frac{1}{p_s} \int_0^{\sigma_{k+\frac{1}{2}}} \nabla \cdot (p_s \mathbf{u}) d\sigma \tag{B2}$$

$$\frac{\omega_k}{p_s \sigma_k} = \mathbf{u}_k \cdot \nabla \log p_s - \frac{1}{\sigma_k} \int_0^{\sigma_k} \left(\delta + \mathbf{u} \cdot \nabla \log p_s \right) d\sigma \tag{B3}$$

$$\Phi_k = \Phi_s + R \int_{\log \sigma_k}^{0} T_\nu d \log \sigma \tag{B4}$$

$$T_\nu = T(1 + (\frac{R_{vap}}{R} - 1)q - q_{c_i} - q_{c_l}) \tag{B5}$$

where $\Phi_s = g z_s$ is the geopotential at the surface.

B.3 Numerics

Our choice of the numerical schemes for interpolation, integrals and diagnostics exactly follows Durran's book [25] §8.6, with the addition of moisture species (which are

advected by the wind and only affect the dynamics through through their effect on the virtual temperature). We use semi-implicit time-integration scheme, where all right hand side terms are separated into groups that are treated either explicitly or implicitly. This avoids severe time step limitations due to fast moving gravity waves.

Our choice of dynamical core was also informed by our desire to run efficiently on machine learning accelerators, in particular Google TPUs [43]. TPUs have dedicated hardware for low-precision matrix-matrix multiplication, which conveniently is well suited for the bottleneck in spectral methods, which are forward and inverse spherical harmonic transformations. Accordingly, we use single-precision arithmetic throughout. We found that full single precision for spherical harmonic transformations was not required to obtain accurate results even on our largest grid sizes, and according use only three passes of bfloat16 matrix-multiplication rather than the six passes that would required for full single precision [44]. Our implementation supports parallelism across spatial dimensions (x, y, and z) for running on multiple accelerator cores, using XLA SPMD [45], with JAX's `shard_map` for parallelizing key bottlenecks including matrix-multiplications in spherical harmonic transforms [46].

Appendix C Learned physics of NeuralGCM

In NeuralGCM, effects of physical processes not accounted by the dynamical core are approximated by neural networks. Associated tendencies of the atmospheric state are computed in two steps: (1) features extraction and normalization (sections C.1, C.3) (2) neural network forward pass and rescaling (sections C.4, C.5).

C.1 Input features for all models

The core input features to the neural network include the vertical structure of the divergence, vorticity, wind vector, temperature variation, specific humidity, specific cloud ice water content, specific cloud liquid water content and (log) surface pressure.

Additionally, our model incorporates various supplementary features to capture critical information relevant to the atmospheric conditions. These features include spatial derivatives of the core features (∂_ϕ, ∂_θ and ∇^2), a land-sea mask, incoming solar radiation, orography (along with spatial gradients), cosine, and sine of the latitude, pressure levels (i.e., sigma level times the pressure surface), and an 8-dimensional location-specific embedding vector for each horizontal grid-point. This embedding vector aims to represent static location-specific information. It is initialized to random values and optimized during training.

We use two embedding modules in NeuralGCM that aim to extract helpful representations that are used as input features by the main network described in C.4. Both are computed by small neural networks that use the same weights shared across all spatial locations to promote feature learning. Each embedding module takes a restricted set of inputs. Vertical embedding network C.4.1 aims to extract common features across the vertical structure of the atmosphere. Surface embedding network C.4.2 aims to estimate the state of the atmosphere's surface boundary. To accomplish this, it receives an additional surface-related inputs, in particular sea surface temperature (SST) and sea ice concentration.

When running weather forecast evaluation the provided SST and sea ice concentration remain constant throughout each forecast, with values taken from the day before the initial time of the forecast launch. This approach ensures that the model relies only on data available at the forecast initialization.

When running seasonal and climate evaluation forecasts, we do update the SST and sea ice concentration from ERA5 data at 6-hour intervals for the 2.8° resolution model and 12-hour intervals for the 1.4° resolution model. This simulates a one way coupling to an ocean model that follows historical evolution.

In some NeuralGCM variations we experimented with adding core features from the previous time step as additional inputs. We did not find this modification to have any significant effect on model performance.

C.2 Additional input features for stochastic models

Stochastic models accept ten additional space-time correlated Gaussian Random Fields (GRF). The fields are initialized and evolved independently of the model state. Every forecast uses a different seed to form independent GRFs. This provides sources of randomness needed to generate an ensemble of statistically independent forecasts.

Each GRF is constructed in a spherical harmonic basis, with adjustable length scale λ and time scale γ.

$$Z(t) = \sum_{l,m} Z_{l,m}(t) Y_{l,m},$$

where $Y_{l,m}$ are the spherical harmonics, and $Z_{l,m}$ are scalar Gaussian processes satisfying (with R_e the earth's radius)

$$\begin{aligned} \mathbb{E}\left[Z_{l,m}(t)\right] &\equiv 0, \\ \mathbb{E}\left[Z_{l,m}(t) Z_{l',m'}(t')\right] &= \begin{cases} F_0 \exp\left\{-\left(\frac{\lambda}{R_e}\right)^2 \frac{l(l+1)}{2}\right\} \exp\{-|t-t'|/\gamma\} & \text{l=l', m=m'} \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

The normalization constant F_0 is chosen so that the global mean variance is 1. This is the same construction used by ECMWF in [47]. In NeuralGCM-ENS each ensemble member uses 10 random fields with initial length scales ranging from 85km to 10,000km and initial time scales from 30 minutes to 60 hours. These values were fixed for the first 1000 training iterations and optimized afterwards.

C.3 Normalization of input features

We normalize all input features to be approximately distributed with zero mean and unit variance to improve training dynamics. We do so by estimating the mean and the standard deviation of all of the features at initialization and adding appropriate shift and rescale transformations before feeding features into the neural network. All features were normalized uniformly across all atmospheric levels, except for specific

humidity q which is normalized per-level. Features corresponding to trainable parameters (termed learned features in section C.1), Gaussian random field features (section C.2) and embedding features were skipped during normalization as they are normally distributed by construction.

C.4 Network architecture

We use Encode-Process-Decode (EPD) architecture[26] with 5 fully connected MLP blocks in the "Process" component. All input features are concatenated together before being passed to the "Encode" layer, which is a linear layer that maps input features to a latent vector of size 384. All "Process" blocks use 3-layer MLP block with 384 hidden units to compute updates to the latent vector. Finally, linear "Decode" layer maps the vector of size 384 to the output vector, which is is then split into per-level values for different variables.

In all models we predict tendencies of the wind vector, temperature and moisture species at all levels.

C.4.1 Vertical embedding network

We use 1D convolutional network to compute vertical embeddings. We use 5 layers with 64 hidden and 32 output channels. Input channels correspond to the 8 atmospheric variables (divergence, vorticity, two components of the wind vector, temperature and three moisture species), aligned by their vertical location in the atmosphere.

C.4.2 Surface embedding network

The surface embeddings are calculated by aggregating outputs from three different embeddings corresponding to distinct surface types: land, sea, and sea ice. The "sea embedding" is a neural network that receives the sea surface temperature (SST) as input. The settings for the "land embedding" and "sea ice embedding" vary slightly across different models. For the deterministic models with resolutions of 1.4° and 0.7°, we employ a constant learned embedding for both land embedding and sea ice embedding. Conversely, for the 2.8° resolution deterministic model and the 1.4° ensemble model, we use neural networks that take the lowest-level atmospheric temperature and moisture as inputs. The output from each network is an embedding of size 8. Each of these smaller networks is a fully connected network comprising three hidden layers with 16 hidden neurons, followed by a readout linear layer that produces an embedding of size 8. The aggregation step is determined based on the relative fraction of each surface type at each grid point.

C.5 Network output scaling

When predicting learned physics tendencies, the outputs of the neural network are scaled by 0.01 standard deviation of the tendencies for each variable., which are estimated from ERA5 data using finite-difference method at 1-hour intervals. The standard deviation is estimated using 10 snapshots averaged over the globe.

Appendix D Encoder and Decoder of NeuralGCM

To interface NeuralGCM that uses sigma coordinate representation of the atmosphere with ERA5 data we use Encoder and Decoder modules.

D.1 Encoder

We use Encoder to obtain a model state in sigma coordinates from an ERA5 snapshot on pressure coordinates in three steps. First, we compute surface pressure for each (longitude, latitude), by calculating the pressure levels at which geopotential field matches that of the surface. Next, we linearly interpolate all relevant atmospheric variables to NeuralGCM's sigma coordinates. Finally, we add a correction computed using a neural network of the same structure as in learned physics module (section C.4) to the interpolation results. The last step significantly alleviates the initialization shock, which otherwise contaminates forecasts with rapid oscillations.

When computing the correction, input features to the network are extracted from input data on pressure levels. We omit embedding features from the Encoder network inputs. Network outputs include corrections to divergence, vorticity, temperature, logarithm of the surface pressure and all moisture species. Before being combined with linear interpolation, network outputs are scaled by 0.02 standard deviation of corresponding variables.

D.2 Decoder

We use Decoder to map model state on sigma coordinates back to ERA5 snapshot on pressure coordinates in three steps. First, we diagnose geopotential from temperature and moisture using Eq. B4. Next, we interpolate the results to pressure levels of ERA5. For pressure levels that correspond to values above the Earth's surface, we uses linear interpolation. To extrapolate data from sigma coordinates (which are terrain-following) to pressure coordinates (that extend below the Earth's surface) we use linear extrapolation for geopotential and temperature. For all other variables we uses boundary values for extrapolation. These choices aim to approximate a more complicated procedure used by ECMWF for pressure extrapolation. Finally, we add a correction computed using a neural network of the same structure as in learned physics module (section C.4) to the interpolation results.

Similarly to the Encoder, embedding features are omitted. Network outputs include corrections to the horizontal wind vector, temperature, geopotential and all moisture species. Before being combined with the result of interpolation, network outputs are scaled by 0.02 standard deviation of corresponding variables.

Appendix E Time integration

In NeuralGCM, the state of the atmosphere is advanced in time by integrating model equations that combine effects from the dynamical core and learned physics parameterizations. This is done iteratively using Implicit-Explicit integration scheme[29] described in E.1. Integration time step varies with resolution, as shown in Table E3. This results in iterative updates to the model state every 4-30 minutes, depending

on model resolution. In contrast, data-drive methods commonly make predictions at 6-hour jumps [10, 12]. Throughout time integration, dynamical core tendencies are computed at every time step, while learned physics tendencies are only recomputed once every 60 minutes for our lowest resolution (2.8°) model and every 30 minutes for all others. This is done to avoid excessive backpropagation through the neural networks in learned physics. At higher resolutions it might be advantageous to include more frequent updates to learned physics tendencies to be able to account for short-time processes (rather than statistical effect that varies smoothly in time). Similar to traditional spectral GCMs we introduce spectral filters to improve numerical stability[48], which are described in E.2.

E.1 Time integration scheme

As is typical for atmospheric models, in NeuralGCM we use semi-implicit ODE solvers to the solve the primitive equations, by partitioning dynamical tendencies into "implicit" and "explicit" terms. "Implicit" tendencies include linear terms of Eq. B1 that give rise to the low amplitude, fast moving gravity waves. These terms are treated implicitly, allowing for longer stable time steps, while the rest of the terms are computed explicitly.

Rather than the traditional semi-implicit leapfrog method, we use implicit-explicit Runge-Kutta methods to avoid the complexity of keeping track of multiple time-steps and time-filtering required by the traditional semi-implicit leapfrog method. Specifically, we use the semi-implicit Lorenz three cycle scheme (SIL3), which was developed specifically for global spectral weather models [29].

1/3	1/3				1/3	1/6	1/6		
2/3	1/6	1/2			1/3	0	1/3		
1	1/2	−1/2	1		1	3/8	0	3/8	1/4
	1/2	−1/2	1	0		3/8	0	3/8	1/4

Table E2 Butcher tableau for the IMEX SIL3 scheme.

E.2 Filtering

During time integration we use two exponential filters of different strengths ("hard" and "soft"). These filters correspond to hyper-diffusion, a standard component of spectral atmospheric models used to stabilize dynamics [49]. Each transform a scalar field x_{hml} in spherical harmonic representation as:

$$x_{hml} \to x_{hml} * e^{-a\left(\frac{k-c}{1-c}\right)^{2p}} \tag{E6}$$

with filter attenuation a, filter cutoff c, filter order p, and normalized total wavenumber $k \equiv \frac{l}{l_{max}}$.

Filter parameters used by different NeuralGCM models are summarized in Table E3, where filter attenuation is specified via attenuation time α and time step dt via

$a = \frac{\alpha}{dt}$. Both hard and soft filters are applied to the model state at the end of each integration step. We additionally apply hard filter to the outputs of learned physics parameterizations to avoid injection of high frequency noise in each model step. The filtering strength sets the true length scale of the simulation, which is generally slightly larger than the grid spacing.

Model resolution	Time step [minutes]	Filter	Attenuation time [minutes]	Order	Cutoff
2.8°	12	hard	4	10	0.4
		soft	120	3	0.0
1.4°	6	hard	8	6	0.4
		soft	120	3	0.0
0.7°	3.75	hard	4	6	0.4
		soft	120	3	0.0

Table E3 Time step and filtering parameters of NeuralGCM models.

Appendix F Evaluation metrics

Evaluation metrics compare forecasts X with ground truth Y. In most cases, Y is ERA5 reanalysis. However, for evaluating ECMWF-HRES and ECMWF-ENS forecasts, we use the lead time = 0 (or "analysis") from ECMWF-HRES. This prevents data-driven approaches from having an unfair advantage. All evaluations reported in the paper were done after regridding X and Y to 1.5°. We use WeatherBench2 library [9] to standardize model evaluation. Our primary evaluation metrics include: root mean square error (RMSE), root mean squared bias (RSMB), continuous ranked probability score (CRPS) and skill-spread ratio.

F.1 Root mean square error (RMSE)

RMSE compares $Y(t+\tau)$, the ground truth at time $t+\tau$, with $X(t \to t+\tau)$, a forecast initialized at time t (to $Y(t)$), at lead time τ hours into the future. This is reported separately for each variable $v \in \mathcal{V}$ and pressure level p, but averaged over initial times.

$$\mathcal{E}_{RMSE}(\tau, v, p) := \sqrt{\frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \|X(t \to t+\tau, v, p) - Y(t+\tau, v, p)\|^2_{nodal}},$$

where $\mathcal{T}$ are the set of 12-hour spaced times in 2020. The *nodal* norm above is area weighted, to avoid polar regions (where points are more dense) from having an undue influence:

$$\|X(t \to t+\tau, v, p) - Y(t+\tau, v, p)\|^2_{RMSE} \tag{F7}$$

$$:= \frac{1}{IJ}\sum_{i=1}^{121}\sum_{j=1}^{240} w(i)|X(t \to t+\tau, v, p, i, j) - Y(t+\tau, v, p, i, j)|^2. \tag{F8}$$

where (i, j) are the (latitude, longitude) indices and $w(i) \propto (\sin \phi_{i+0.5} - \sin \phi_{i-0.5})$.

For models that produce a single deterministic forecast X, MSE $\equiv \mathbb{E}\left[\|X(t+\tau) - Y(t+\tau)\|^2\right]$ (and hence RMSE) is minimized by $X \equiv \mathbb{E}[Y]$. So the best RMSE scores will be had by a forecast that predicts the (conditional) mean $\mathbb{E}[Y(t+\tau) \,|\, Y(t)]$. We primarily use this metric for assessing accuracy of deterministic forecasts at shorter lead times, when the distribution of $Y(t+\tau)$ is sharply peaked. As lead time τ increases, the distribution $Y(t+\tau)$ spreads out, as true dynamics is depends on information not fully contained in the initial state $Y(t)$. When this happens, $\mathbb{E}[Y(t+\tau) \,|\, Y(t)]$ becomes blurry and unphysical.

For models that generate ensemble forecasts we compute RMSE scores using ensemble mean $\mathbb{E}_i[X_i]$. In this setting RMSE remains informative even at later times, as the ensemble mean is taken explicitly. It's important to note that RMSE of the ensemble mean alone is not sufficient to assess the skill of the forecasting system. An ensemble of identical blurry realizations could still achieve good RMSE skill, but fail to capture extreme events and provide probabilistic insight.

F.2 Root mean squared bias (RMSB)

Biases estimate persistent differences between forecasts $X(t+\tau)$ and ground truth $Y(t+\tau)$ averaged over time. This is reported separately for each variable $v \in \mathcal{V}$ and pressure level p, lead time τ, and (latitude, longitude) coordinates (i, j).

$$\mathcal{E}_{bias}(\tau, v, p, i, j) := \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \left[X(t \to t+\tau, v, p, i, j) - Y(t+\tau, v, p, i, j)\right],$$

RMSB is computed by taking RMSE of the bias and aggregating it over spatial dimensions.

$$\mathcal{E}_{RMSB}(\tau, v, p) := \sqrt{\frac{1}{IJ} \sum_{i=1}^{121} \sum_{j=1}^{240} w(i) \|\mathcal{E}_{bias}\|^2}$$

F.3 Continuous Ranked Probability Score (CRPS)

Ideally, evaluation metrics should be minimized when $X(t) \sim Y(t)$, rather than $\mathbb{E}Y(t)$ as is the case for MSE. One such metric is CRPS. The generic CRPS in one dimension is constructed using ground truth Y, and two independent forecasts X and X'. It takes the form

$$\mathbb{E}|X - Y| - \frac{1}{2}\mathbb{E}|X - X'|.$$

The skill term $\mathbb{E}|X - Y|$ penalizes forecasts that deviate from Y, while the spread term $\mathbb{E}|X - X'|$ encourages well dispersed forecasts. As it turns out, CRPS is minimized just when X has the same distribution as Y [31]. Following [9], we extend CRPS to

N dimensions by summing over coordinates.

$$\mathcal{E}_{CRPS}(\tau, v, p) := \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{1}{IJ} \sum_{i=1}^{121} \sum_{j=1}^{240} w(i)\, \mathcal{C}(X(\ldots, i, j), X'(\ldots, i, j), Y(\ldots, i, j)),$$

$$\mathcal{C}(x, x', y) := \frac{1}{2} \left(|x - y| + |x' - y| - |x - x'| \right).$$

The resultant "CRPS" is minimized by any distribution X such that the marginals X_n, have the same distribution as Y_n. CRPS will therefore not require correct forecasts, but will not penalize them either.

F.4 Spread-Skill ratio

Spread-skill ratio represents the ratio of the ensemble spread (standard deviation) to skill (RMSE) of the ensemble mean. If ensemble members X are distributed identically to the ground truth Y, then the spread-skill ratio should be equal to 1. We additionally multiply the spread by $\sqrt{1 + \frac{1}{N}}$, where N is the ensemble size, to account for the biased estimation of RMSE of the ensemble mean for finite ensembles [32]. Similar to other evaluate metrics we compute spread-skill ratio for ECMWF-ENS and NeuralGCM-ENS models for all lead times by averaging over initial time. When reporting global spread-skill ratio we perform spatial aggregation prior to computing the ratio. For spatial visualizations spread-skill ratio is computed for each latitude, longitude and level independently.

Appendix G Training

We train NeuralGCM using adam[50], optimizing model parameters to minimize the loss function between predictions and coarsened ERA5 trajectories. Optimizer parameters are summarized in G.1. For all models we progressively extend the lead time horizon over which the loss is computed as the model trains. The schedule and the data are discussed in G.2. Before computing the loss we rescale errors between trajectories to address the differences in magnitudes of the target distributions (see G.3 for details). Finally, the two loss functions used to train deterministic and stochastic NeuralGCM are discussed in G.4 and G.6 respectively. Computational resources and training times are reported in G.7.

G.1 Optimizer settings

All NeuralGCM models were trained using adam[50]. We used values $\beta_1 = 0.9$, $\beta_2 = 0.95$ and $\epsilon = 10^{-6}$ for all experiments based on empirical evidence from initial results at coarsest resolution. Learning rate was adjusted as a function of training iterations, linearly increasing to its maximum value over the first 2000 training steps, then remaining constant until training step 15000 when exponential decay with rate 0.5 was initiated with decay time set to 10000 steps. The only exception to these parameters was decoder fine-tuning stage described in G.5, where warm-up is completed by

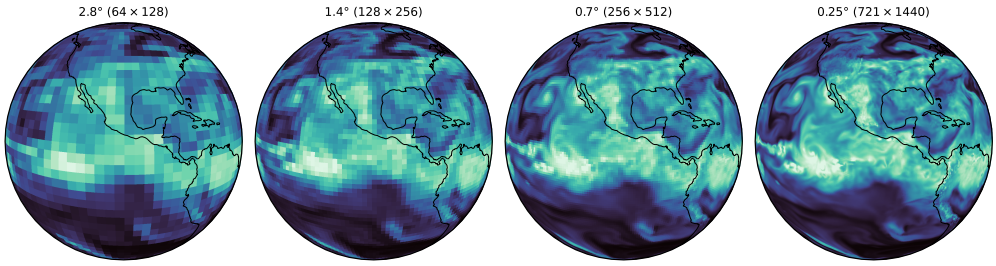

Fig. G1 Visualization of specific humidity at 700 hPa from ERA5 conservatively regridded to Gaussian grids 2.8°, 1.4°, and 0.7°, and the original ERA5 data.

step 1000 and learning rate decays with rate 0.5 every 1000 steps starting from step 2000. Peak learning rates and total number of steps for different model configurations are reported in Table G4.

	NeuralGCM-2.8°	NeuralGCM-1.4°	NeuralGCM-0.7°	NeuralGCM-ENS(1.4°)
Peak learning rate	0.002	0.002	0.001	0.001
Number of training steps	38000	26000	25000	43000

Table G4 Learning rate and number training steps used for NeuralGCM at different resolutions.

Model	Unroll lengths	Transition boundaries
NeuralGCM-2.8°	(12, 24, 36, 48, 60, 72)	(2000, 5656, 10392, 16000, 22360)
NeuralGCM-1.4°	(12, 24, 36, 48, 60, 72)	(2000, 5656, 10392, 16000, 22360)
NeuralGCM-0.7°	(6, 12, 18, 24, 36, 48, 60)	(500, 2000, 4500, 8000, 12500, 18000)
NeuralGCM-ENS (1.4°)	(6, 12, 18, 24, 36, 48, 60, 72, 96, 120)	(500, 2000, 4500, 8000, 12500, 18000, 24500, 32000, 40500)

Table G5 Training curriculum used for NeuralGCM at different resolutions.

G.2 Training data and unroll schedules

To train NeuralGCM models at 2.8°, 1.4° and 0.7° resolutions we regridded ERA5 data to the corresponding Gaussian grids. We use a conservative regridding scheme, which effectively aggregates contributions by the relative area overlap. Final 2.8° and 1.4° models were trained on data from years 1979 through 2017 and evaluated on 2018 during the development cycle. 0.7° model and stochastic model variations were trained with data from 1979 through 2019. None of the models had any exposure to data from 2020 prior to running final model evaluations.

During training we also change lead time unroll length as a function of training iterations, increasing the prediction horizon as the model improves over the course of training. The values are reported in Table G5.

G.3 Variable rescaling for losses

Before computing losses, trajectories of all variables are rescaled to address: (1) differences in natural scales of atmospheric variables, and (2) growth of their deviations from the ground truth as a function of lead time. The former is accomplished by dividing each atmospheric variable by standard deviation of the temporal difference between snapshots that are 24 hours apart, similar to [10]. We further adjusted scaling factors across variables to better balance out the initial skill of the model by the following additional scaling factors: geopotential 2, specific humidity 0.66, logarithm of the surface pressure (internal model representation) 5 and cloud moisture species 0.05. Our rationale for additional balancing is to prevent the loss landscape being dominated by just a few atmospheric variables. Standard deviations were estimated based on 10-60 snapshots of 24-hour differences in ERA5 data averaged over longitude, latitude, level and sample. The only trajectory that was normalized per-level was specific humidity as its values vary significantly with level. When estimating scales for variables in internal representation such as divergence, vorticity, log surface pressure, corresponding quantities were compute from ERA5 data using linear interpolation to sigma coordinates and appropriate transformations relating them to the data variables.

Each trajectory is also rescaled to account for a near linear increase in expected error variance with lead time τ. The scaling factor is $(1 + (\tau/24))^{-1/2}$ for all losses except the spectral losses, which use $(1 + (\tau/40)^4)^{-1/2}$.

Finally, in the case of filtered mean squared loss an additional time-dependent rescaling is performed which described in detail in G.4.1.

G.4 Loss for deterministic models

Our deterministic models are trained in two stages that differ in loss objectives and parameters being optimized: (1) primary training and (2) decoder fine tuning. During both stages, the loss terms take the general form of a mean squared error (MSE) on rescaled variables (Section G.3) with varying definitions of distances ρ_*

$$\mathcal{M}_*(\tau) := \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \sum_{\sigma, v} \sum_{\substack{\tau' \in \mathcal{D} \\ \tau' \leq \tau}} \rho_*(X(t \to t + \tau', v, \sigma), Y(t + \tau', v, \sigma))^2, \tag{G9}$$

where the lead time τ, in hours, is selected from

$$\mathcal{D} := \{6, 12, 18, 24, 36, 48, 60, 72, 96, 120\}.$$

During the primary stage we used a combination of three loss types to account for different types of discrepancies between NeuralGCM predictions and ERA5 data: accuracy (G.4.1), sharpness (G.4.2) and biases (G.4.2). All losses, except for the bias loss, are imposed on both "data"(i.e., pressure levels) and "model"(i.e., sigma levels) representations. This results in a training objective that is a sum of five loss terms:

$$\mathcal{L}_{Deterministic} = \lambda_{data}\mathcal{M}_{Data} + \lambda_{spec}\mathcal{M}_{DataSpec} + \lambda_{model}\mathcal{M}_{Model}$$

$$+ \lambda_{spec}\mathcal{M}_{ModelSpec} + \lambda_{bias}\mathcal{M}_{MSBias},$$

where loss scales $\lambda_{data} = 20$, $\lambda_{spec} = 0.1$, $\lambda_{model} = 1$ and $\lambda_{bias} = 2$ were selected empirically. We found that even though the contributions corresponding to "spec" and "bias" terms were small (compared to the total loss), they had a positive effect on sharpness of NeuralGCM predictions. This is possibly related to the fact that the dynamical core simulates an energy cascade, maintaining the spectral energy of the simulated fields.

After the main training phase is complete, we run a short decoder fine-tuning optimization discussed in G.5 to remove spectral artifacts that arise from truncation errors in transformations from spherical harmonic to grid-point space. During this training phase we only update parameters of the decoder component of the model.

G.4.1 Accuracy loss: filtered MSE

Traditional mean squared error loss between forecast X and targets Y can be computed on pressure levels p and in spherical harmonic basis indexed by total (l), and zonal (m), wavenumbers using $\rho_{Data}(X, Y) = \|X - Y\|_{modal}$, where

$$\|\epsilon\|^2_{modal} := \sum_l \sum_m |\epsilon(l, m)|^2. \tag{G10}$$

Up to discretization error, this is equivalent to $\|\cdot\|^2_{nodal}$, the area-weighted MSE in grid-point space representation.

When used to evaluate forecasts at longer lead time, this loss suffers from the "double penalty problem"[30] as it penalizes the model for predicting sharp features that are slightly misplaced. We address this issue by using "filtered MSE", which applies filtering to X and Y prior to computing the norm G10.

The filtering step aims to retain components of X and Y that we expect to be able to predict and drop components that cross the predictability horizon due to chaotic dynamics. We estimate such filtering parameters by analyzing the normalized MSE growth between ECMWF-HRES and ERA5 as a function of time for each variable and total wavenumber l separately Fig. G2(a). We chose a predictability threshold of relative error of 0.12, and determined the largest value of l^* for each lead time that should be included in the loss. Then we determined attenuation parameters for the exponential filters of order 12 that would remove total wavenumbers higher than l^* for each group, shown in Fig. G2(b). The effect of applying filters is shown in Fig. G2(c).

The *Model* terms are defined similarly to their Data counterparts, except they are computed in the encoded "model" space. This encourages the encoder/decoder round-trip to be the identity. It was found to enhance stability.

G.4.2 Sharpness loss: Spectrum MSE

Spectrum MSE is defined by the distance

$$\rho_{DataSpec}(X(t \to t + \tau, v, \sigma, l), Y(t + \tau, v, \sigma, l))^2$$

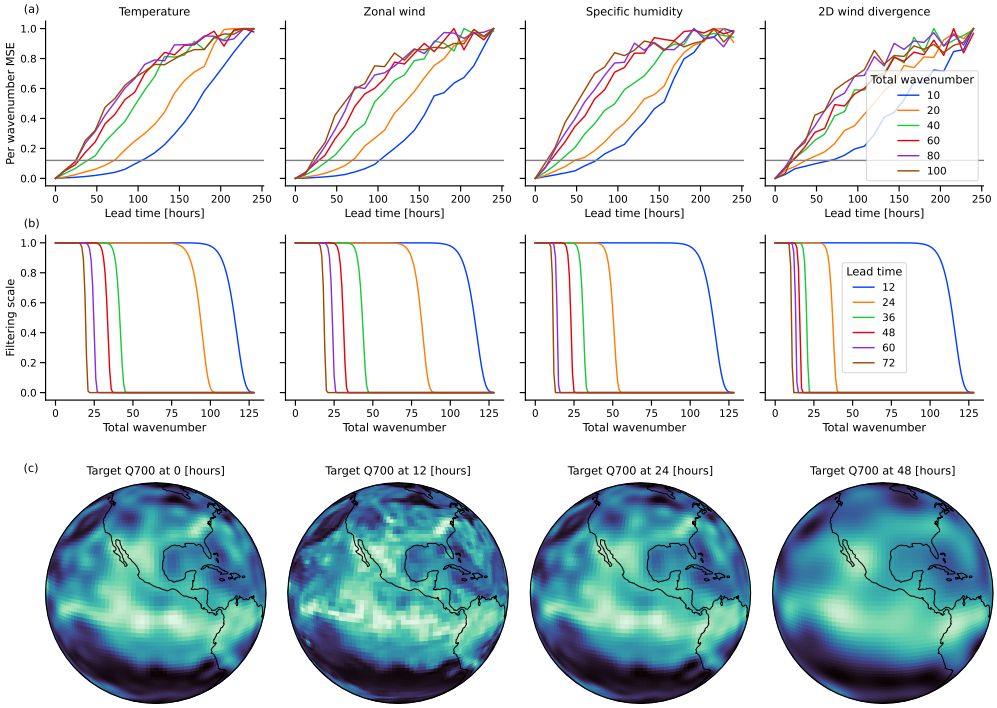

Fig. G2 Estimation of predictability based on ECMWF-HRES errors. (a) Normalized mean squared error between ECMWF-HRES and ERA5 for each total wavenumber as a function of time for temperature, zonal wind, specific humidity and horizontal wind divergence. Grey line corresponds to threshold that was used to estimate the predictability horizon. (b) Resulting filtering profiles for varying lead times and different variables. (c) Visualization of effective resolution at which errors are estimated at varying lead times. Zero hour forecasts are additionally blurred in the loss so the model does not need to match exact initial conditions.

$$= \sum_{l}^{\tilde{l}} \left[\mathcal{S}(X)(t \to t+\tau, v, \sigma) - \mathcal{S}(Y)(t+\tau, v, \sigma) \right],$$

where $S(X)^2(..., l) = \sum_m X(..., l, m)^2$ is the spectral energy at total wavenumber l and $\tilde{l}$ corresponds to a spectral cutoff for the loss. For NeuralGCM models at 2.8°, 1.4°, 0.7° we used $\tilde{l}$ of 42, 80 and 120 respectively.

G.4.3 Bias loss: spectral bias MSE

Spectral bias MSE is slightly different from the previous two terms, as it averages over the batch and lead time dimensions prior to computing the distance norm:

$$\mathcal{M}_{MSBias}(\tau) = \sum_{\sigma,v} \left\| \frac{1}{|\mathcal{T}|} \sum_{t\in\mathcal{T}} \sum_{\substack{\tau'\in\mathcal{D} \\ \tau'\le\tau}} [X(t \to t+\tau, v, \sigma) - Y(t+\tau, v, \sigma)] \right\|^2_{modal}$$

This loss term discourages accumulation of bias.

G.5 Decoder fine-tuning

The final stage of training deterministic NeuralGCMs is focused on optimizing only the "Decoder" component. After the primary training phase, "Decoder" outputs contain high-frequency artifacts that are not captured by the losses imposed in spherical harmonics representation (because numerical spherical harmonics transform has non-zero kernel). To address that we freeze all of the "Encoder" and "learned physics" components of the model and optimize the "Decoder" parameters using traditional MSE computed in grid-space representation on predictions at 6, 12, 18 and 24 hours (12 and 24 hours for NeuralGCM-1.4°). We use 4000 gradient descent steps, but generally find that loss and evaluation metrics plateau after the first 1000 training steps.

G.6 Loss for stochastic models

The CRPS loss function is defined using the same notation as Equation (G9) as the sum of modal and nodal terms. The CRPS evaluation metric is defined similarly [9].

For any variable v, and initial/lead times t and τ, the *modal* CRPS term on pressure level p is

$$\begin{aligned}\mathcal{C}_{modal}(t,p,v,\tau) = \frac{1}{2}\sum_{l,m}\Big[&|X(t\to t+\tau,\ldots,l,m) - Y(t+\tau,\ldots,l,m)| \\ &+ |X'(t\to t+\tau,\ldots,l,m) - Y(t+\tau,\ldots,l,m)| \\ &- |X(t\to t+\tau,\ldots,l,m) - X'(t\to t+\tau,\ldots,l,m)|\Big],\end{aligned}$$

where X and X' denote two independent ensemble members. $\mathcal{C}_{modal}$ is minimized just when the forecast has the correct spherical harmonic components. CRPS contributions from lower wavenumbers l encourage forecasts with correct long range correlations, which we found to be critical for training models with good performance. We excluded wavenumbers l and m from modal CRPS loss above a maximum wavenumber of 80, because we filter out higher wavenumbers for stability in our dynamical core.

The *nodal* CRPS loss is defined similarly, but uses an area-weighted sum over latitude/longitude points, as in (F7). These two terms are combined to give

$$\mathcal{L}_{CRPS}(\tau) := \sum_{t\in\mathcal{T}} \sum_{p,v} \sum_{\substack{\tau'\in\mathcal{D} \\ \tau'\leq\tau}} \left[\mathcal{C}_{modal}(t,\tau',v) + \mathcal{C}_{nodal}(t,\tau',v)\right].$$

G.7 Training resources

Table G6 lists the approximate computational resources used to train each version of NeuralGCM. In all cases, we use data parallism, running our model with one single example on each TPU device, with a batch of examples distributed across multiple TPUs. For NeuralGCM-0.7° and NeuralGCM-ENS, we additionally distribute across model dimensions.

	NeuralGCM-2.8°	NeuralGCM-1.4°	NeuralGCM-0.7°	NeuralGCM-ENS (1.4°)
Training time	1 day	1 week	3 weeks	10 days
Device	16 TPU v4	16 TPU v4	256 TPU v4	128 TPU v5e
Parallelism	batch=16	batch=16	batch=16 x=2 y=2 z=4	batch=16 x=2 z=2 ensemble=2

Table G6 Resource requirements for different NeuralGCM models trained on Google Cloud TPUs.

Appendix H Additional weather evaluations

H.1 Accuracy

We assess the accuracy of deterministic NeuralGCM, along with other ML models and ECMWF-HRES, by quantifying their skill using the root mean square error (RMSE) and root-mean-square-bias (RMSB), both computed against the relevant ground truth data. As NeuralGCM was trained to predict ERA5 data, we evaluate its errors against ERA5 as the ground truth. In contrast, the ECMWF-HRES model utilizes ECMWF-HRES analysis as input, leading us to compare ECMWF-HRES against ECMWF-HRES analysis, thereby reducing the reported ECMWF-HRES errors. We additionally include scores for the NeuralGCM-ENS mean. This model is not as accurate as NeuralGCM-0.7° at short lead times, but indicates the time when the RMSE skill starts to be dominated by forecast uncertainty.

In Fig. H3 we plot RMSE and RMSB scorecard comparisons of NeuralGCM-0.7° and NeuralGCM-ENS against ECMWF-HRES and GraphCast across all core atmospheric variables (geopotential, temperature, specific humidity and u-component of wind) and 13 pressure levels. Similar to previous ML approaches we find that NeuralGCM-0.7° generally outperforms ECMWF-HRES across all scores (Fig. H3(a)), except for the top level of the atmosphere. When compared to GraphCast, we find that on short lead times NeuralGCM-0.7° performs similarly on RMSE, with some

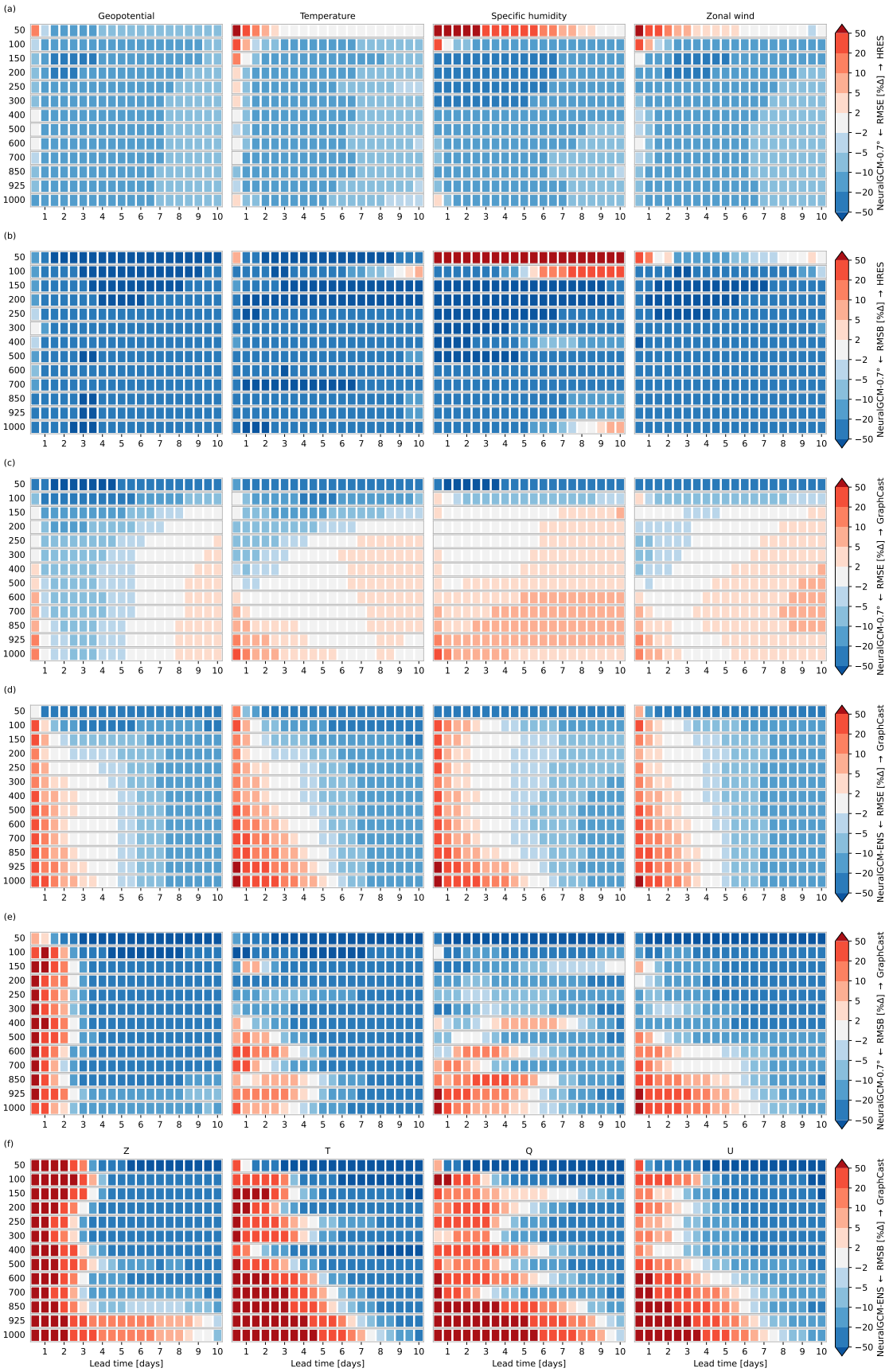

Fig. H3 Scorecards comparing NeuralGCM-0.7° and NeuralGCM-ENS models against ECMWF-HRES and GraphCast on RMSE and RSMB scores across atmospheric variables and pressure levels. (a) RMSE scorecard for NeuralGCM-0.7° vs ECMWF-HRES, (b) RSMB scorecard for NeuralGCM-0.7° vs ECMWF-HRES, (c) RMSE scorecard for NeuralGCM-0.7° vs GraphCast, (d) RMSE scorecard for NeuralGCM-ENS vs GraphCast, (e) bias RMSE scorecard for NeuralGCM-0.7° vs GraphCast, (f) bias RMSE scorecard for NeuralGCM-ENS vs GraphCast.

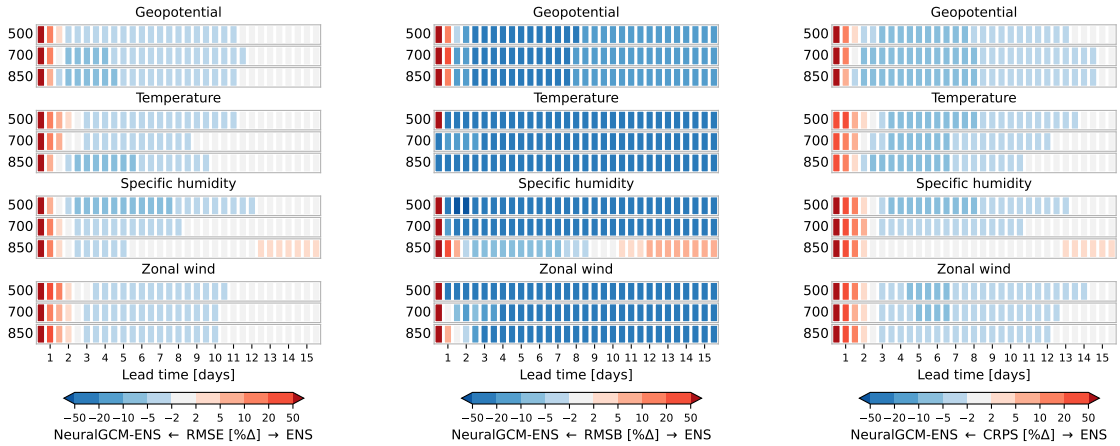

Fig. H4 Scorecards comparing NeuralGCM-ENS against ECMWF-ENS on RMSE (a), RMSB (b) and CRPS (c) scores across atmospheric variables and pressure levels.

variables better modeled by GraphCast (specific humidity) and some by NeuralGCM-0.7° (geopotential) (Fig. H3(c)). At longer lead times (5-6 days) NeuralGCM-ENS achieves better RMSE scores than GraphCast across the board, underlining the utility of stochastic models at such timescales. The general trend of NeuralGCM-0.7° to outperform GraphCast at higher levels and being slightly worse at lower levels of the atmosphere is potentially caused by an additional loss weight factor introduced in GraphCast that discounts levels at lower pressure.

Contrary to RMSE, RMSB is not a priori expected to rapidly increase with lead time. NeuralGCM-0.7° model maintains consistently lower bias than ECMWF-HRES for the 10 days (Fig. H3(b)). GraphCast achieves lowest bias at very early lead times, but quickly degrades afterwards (Fig. H3(e)). NeuralGCM-ENS model has slightly higher biases compared to NeuralGCM-0.7° model, likely due to coarser spatial resolution. In Fig. H4 we provide similar RMSE and RMSB scorecards comparing NeuralGCM-ENS to ECMWF-ENS on 15 day.

Figures H5, H6, H7 and H8 compare NeuralGCM models against ECMWF-ENS across all core atmospheric variables and 3 pressure levels (500, 700 and 850 hPa), for which we had ECMWF-ENS data to compare against. Beyond two days, NeuralGCM-ENS has lower or similar score (within 1%) compared to than ECMWF-ENS across all metrics (RMSE, RSMB and CRPS), with the exception of specific humidity at 850 hPa at longer lead times. ECMWF-ENS has better scores at very early lead times, which is likely linked to relatively coarse resolution (1.4° vs 0.2°).

Figures H9, H10, H11 and H12 compare spatial metrics for 10-day forecasts from NeuralGCM-ENS and ECMWF-ENS across all core atmospheric variables on WeatherBench2 pressure levels. Overall, NeuralGCM-ENS and ECMWF-ENS have similar spatial patterns of skill relative to climatology as measured by RMSE and CRPS. Spatial biases for Neural-GCM are better overall than for ECMWF-ENS, and are also noticeably more evenly distributed. Comparing spread-skill ratio shows that unlike ECMWF-ENS, NeuralGCM-ENS is not consistently under-dispersed in the tropics.

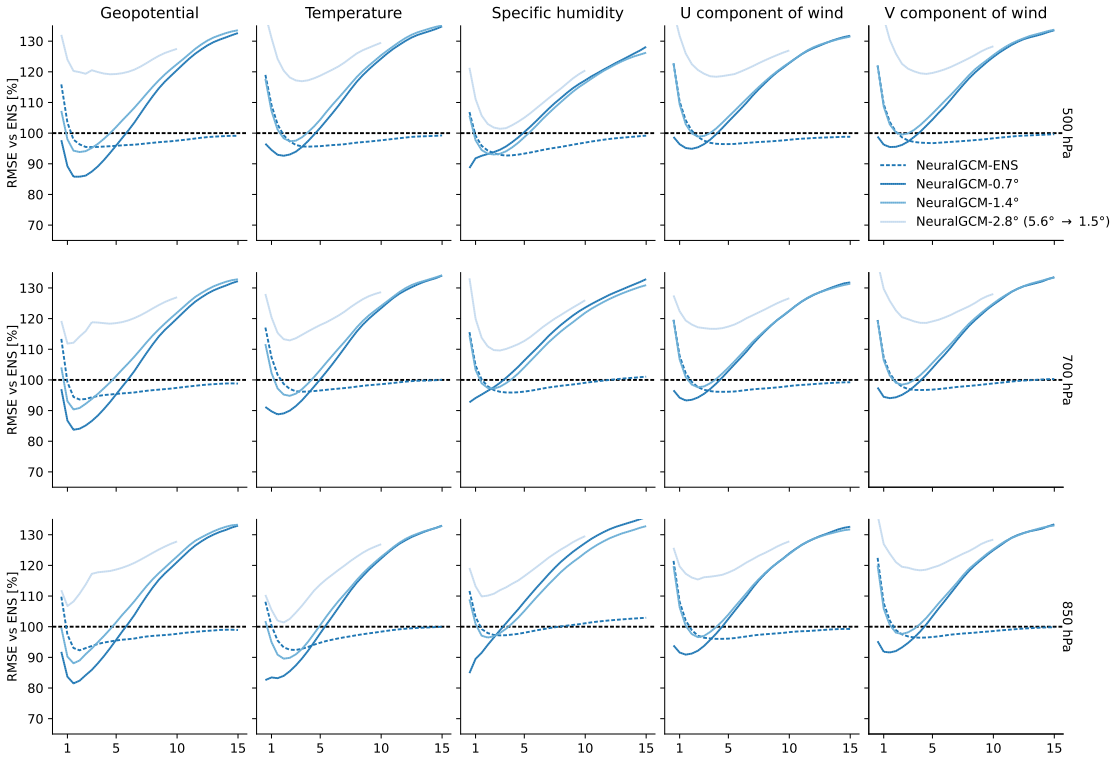

Fig. H5 Root-mean-squared-error (RMSE) relative to ECMWF-ENS for all NeuralGCM forecasts in 2020.

H.2 Derived variables and spectra

To better understand the consistency of ML weather forecasts, we calculate a variety of the variables that can be derived from geopotential, temperature, horizontal wind velocity and specific humidity:

1. **Lapse rate** is given by

$$\frac{\partial T}{\partial z} = \left(\frac{\partial T}{\partial p}\right)\left(\frac{1}{g}\frac{\partial \Phi}{\partial p}\right)^{-1},$$

   where T is temperature, p is pressure, g is the gravitional constant and Φ is geopotential.
2. **Wind speed** is given by

$$\sqrt{u^2 + v^2}$$

   where u and v denote zonal and meridional wind velocity.

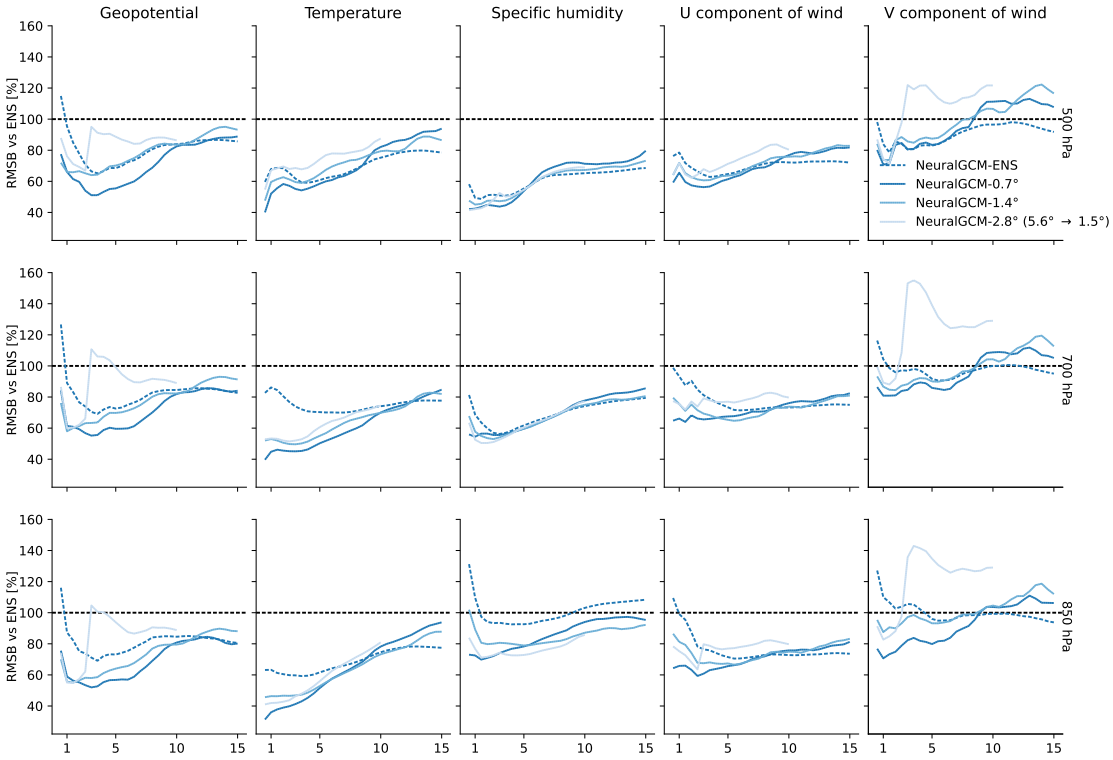

Fig. H6 Root-mean-squared-bias (RMSB) relative to ECMWF-ENS for all NeuralGCM forecasts in 2020.

3. **Divergence** is given by

$$\nabla_p \cdot \mathbf{u} = \partial_x u + \partial_y v,$$

where ∂_x and ∂_y denote the partial derivatives in the zonal and meridional directions.

4. **Vorticity** is given by

$$\hat{\mathbf{k}} \cdot (\nabla_p \times \mathbf{u}) = \partial_x v - \partial_y u.$$

5. **Vertical velocity** at pressure level p_0 under the assumption of hydrostatic balance $\partial p/\partial z = -\rho g$ is given by [25]

$$-\int_0^{p_0} dp \nabla_p \cdot \mathbf{u} = -\int_0^{p_0} dp[\partial_x u + \partial_y v].$$

6. **Eddy kinetic energy** is given by

$$\frac{1}{2}\left[(u-\bar{u})^2 + (v-\bar{v})^2\right],$$

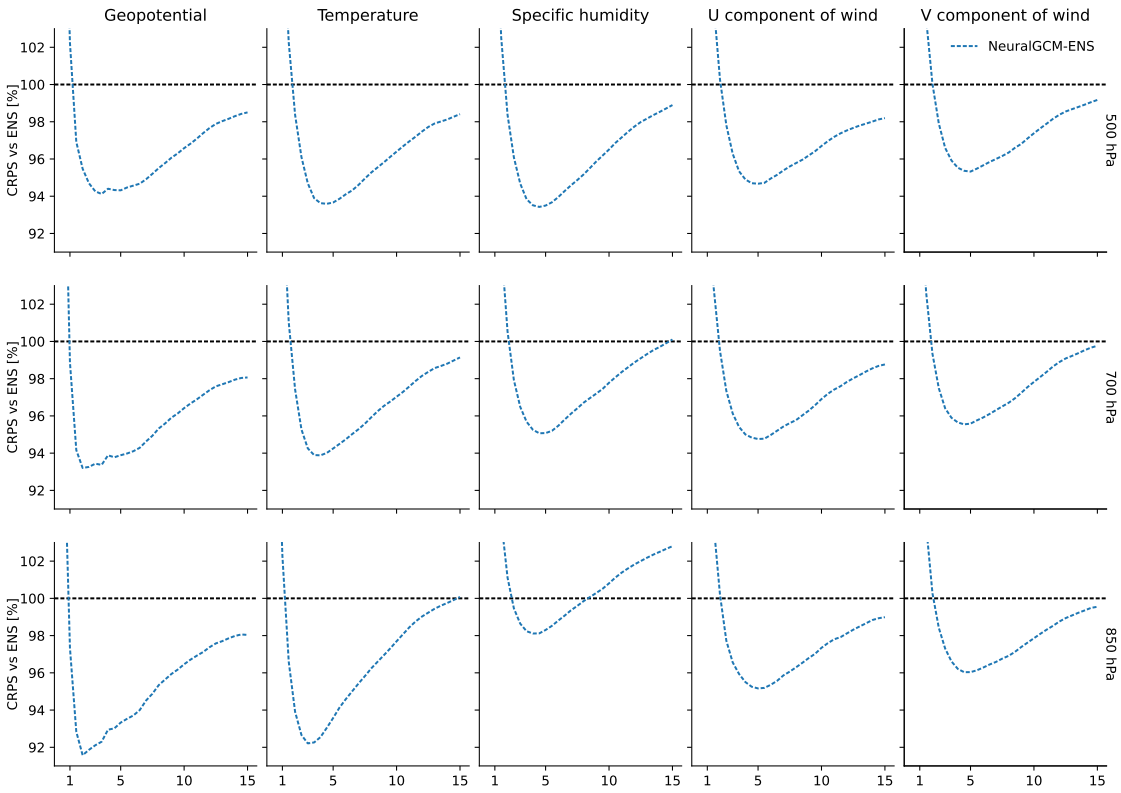

Fig. H7 Continuous ranked probability score (CRPS) relative to ECMWF-ENS for NeuralGCM-ENS forecasts in 2020.

where $\bar{u}$ and $\bar{v}$ denote the longitudinal mean of zonal and meridional wind velocity, respectively.

7. **Geostrophic wind** is a horizontal vector given by

$$\frac{1}{2\Omega \sin \phi} \hat{\mathbf{k}} \times \nabla_p \Phi = \frac{1}{2\Omega \sin \phi} \left[-\partial_y \Phi, \partial_x \Phi \right],$$

where Ω is the rotational speed of the Earth, ϕ is latitude in radians, ∇_p is the gradient on constant pressure levels and Φ is geopotential [15].

8. **Ageostrophic wind** is the horizontal wind vector minus geostrophic wind.
9. **Total column moisture** quantities are given by a vertical integral over pressure levels,

$$\frac{1}{g} \int dpq,$$

where g is the gravitational constant and q is the desired moisture species (humidity/vapor, cloud liquid or cloud ice).

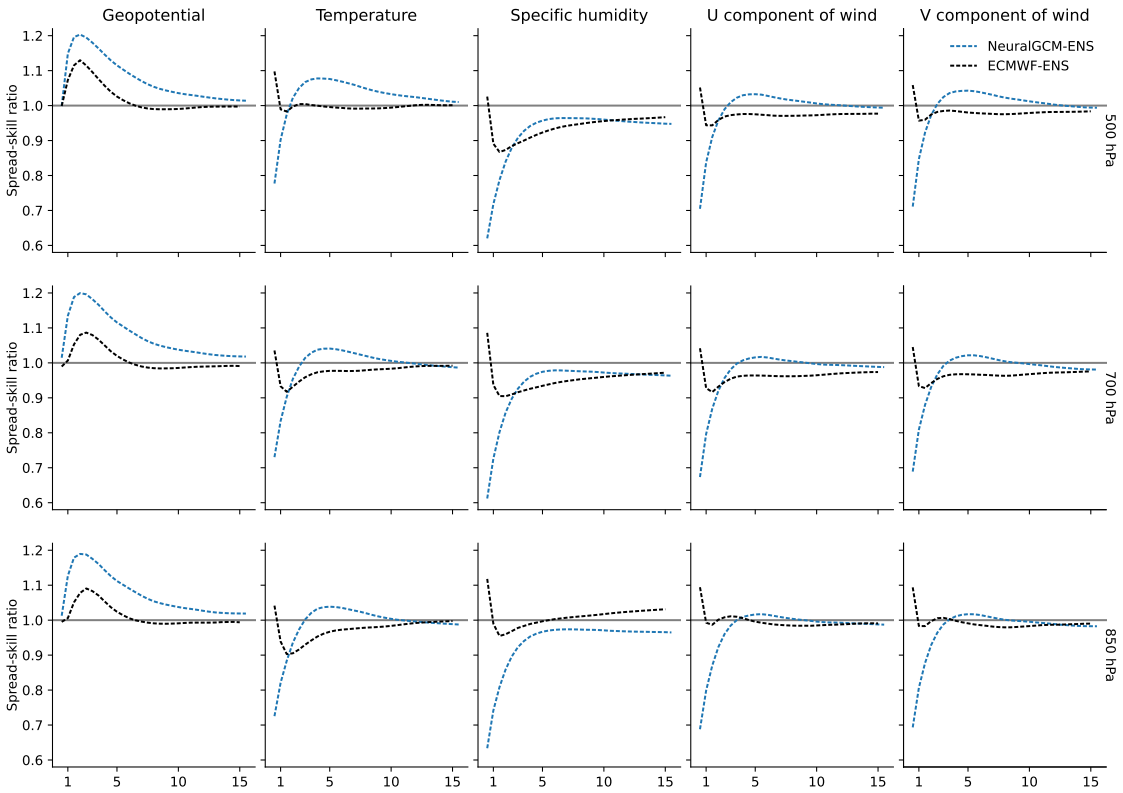

Fig. H8 Spread-skill-ratio for NeuralGCM-ENS and ECMWF-ENS forecasts in 2020.

10. **Integrated vapor transport** is given by

$$\frac{1}{g}\sqrt{\left(\int_{p_0}^{p_1} dpqu\right)^2 + \left(\int_{p_0}^{p_1} dpqv\right)^2},$$

where q is specific humidity, u is zonal wind, v is meridional wind, $p_0 = 300\,\text{hPa}$ is the minimum pressure and $p_1 = 1000\,\text{hPa}$ is the maximum pressure [12].

11. **Relative humidity** is given by

$$\frac{q/(1-q)}{\epsilon e_s/(p-e_s)}$$

where $\epsilon = 0.622$ and e_s the saturation vapor pressure in hPa is

$$e_s = 6.112\exp[17.67(T-273.15)/(T-29.65)],$$

where T is the temperature in Kelvin, which is the formula implemented by MetPy [51].

All derivatives are calculated using second-order finite differences, and vertical integrals are calculated using trapezoidal integration. Code for calculating these variables has been added to the WeatherBench2 codebase [9].

Fig. H13 and Fig. H14 plots a variety of example 7-day forecast fields and power spectra averaged over many weather forecasts, based archived weather forecasts from WeatherBench2. To consistently compare predictions from models with different resolutions, all calculations are performed after conservative regridding to a 1.5° equiangular grid on 37 pressure levels. Variables are excluded if calculating them requires field not archived as part of WeatherBench2 for a particular forecast model, including vertical integrals or derivatives if not all pressure levels are available. We omit NeuralGCM-ENS because conservative regridding from a 1.4° Gaussian to a 1.5° equiangular grid introduces aliasing artifacts that are particularly evident for derived variables.

H.3 Visualization of ensemble weather forecasts

To verify that NeuralGCM-ENS produces reasonable looking forecasts of all variables, we plot maps of forecast fields in Fig. H16 and vertical profiles in Fig. H17 and Fig. H18. Here we compare to ERA5 after conservative horizontal regridding to the native resolution of NeuralGCM-ENS. The forecasts look qualitatively similar to ERA5, except the cloud variables (specific cloud liquid water content and specific cloud ice water content) occasionally take on small in magnitude negative values, which is not physically possible. Fixing this issue represents an improvement opportunity for improving future versions of NeuralGCM.

H.4 Evaluation of lower resolution models

In Fig. H5,H6 we compare RMSE skill and root mean squared bias of NeuralGCM models at 0.7°, 1.4° and 2.8° resolutions, evaluated on the year 2020. Similar to the main text we normalize the results against ECMWF-ENS for easier comparison. Across all atmospheric variables we find that increasing the level of detail improves RMSE. Note that the skill of NeuralGCM-2.8° is evaluated on 5.6° and then rescaled by the ratio of ECMWF-HRES errors at 1.5° and 5.6° resolutions to estimate model performance without upsampling coarse predictions.

Fig. H19 compares power spectra of core atmospheric variables for NeuralGCM models of different resolution at 1 and 7 days into the forecast.

H.5 Forecast generalization over five years

One motivation for incorporating strong physics priors into NeuralGCM is to improve performance on out of sample data, e.g., caused by climate change or systematic changes in Earth observing systems. Because the final version of NeuralGCM-0.7° was trained on data through 2019, to produce additional data for comparisons we also consider a developmental version NeuralGCM-0.7°-2017 only trained through 2017 data. This model uses the same architecture described previously with three major differences:

1. It does not include a surface embedding or surface forcing features (sea surface temperature, sea ice concentration).
2. It does not include the memory feature.
3. Its loss is scaled differently in time, like $(1+(\tau/24))^{-1}$ instead of $(1+(\tau/40)^4)^{-1/2}$.

In Fig. H20 we compare accuracy trends of GraphCast, NeuralGCM-0.7°, NeuralGCM-0.7°-2017 and ECMWF-HRES evaluated on ERA5 data from years 2018-2022. In this experiment GraphCast and NeuralGCM-0.7°-2017 variants were trained on data until 2017. ECMWF-HRES and NeuralGCM models display little variation over evaluation years, while GraphCast accuracy degrades by a few percent when evaluated on 2022 forecasts (5 years more recent than the training domain), consistent with the degraded performance noted in the GraphCast paper. Forecasts of geopotential and at short lead times have the largest increases in error.

H.6 Diagnosing precipitation

To diagnose precipitation minus evaporation rate $(P-E)$ we calculate:

$$P - E = \frac{1}{g}\int_0^1 \sum_{\mathrm{i}} \left(\frac{dq}{dt}\right)_{\mathrm{i}}^{\mathrm{NN}} p_{\mathrm{s}} d\sigma \tag{H11}$$

where p_{s} is the surface pressure, and $\sum_{\mathrm{i}}(\frac{dq}{dt})_{\mathrm{i}}^{\mathrm{NN}}$ is the sum of the water species tendencies predicted by the neural network. To compare against reanalysis data we use the precipitation rate and evaporation rate from ERA5. Fig. H21a,b shows the daily distribution of precipitation minus evaporation rates from NeuralGCM-0.7° forecasts alongside ERA5. The distribution of precipitation minus evaporation rates has been normalized such that the area under the points is equal to one.

We observe that the precipitation rate distribution from NeuralGCM-0.7° aligns closely with the ERA5 distribution in the extratropics. However, NeuralGCM-0.7° tends to underestimate extreme events in the tropics. For weather forecasting, we note that the average forecast (averaged across all initial conditions for the third day of prediction) aligns well with the spatial distribution seen in ERA5 (Fig. H21c,d). To underscore the differences between ERA5's $P-E$ and NeuralGCM-0.7°'s diagnosed $P-E$, we present snapshots of daily $P-E$. While there are overarching similarities between these snapshots, it's evident that in the tropics, NeuralGCM-0.7° tends to moderate extreme events (Fig. H21e,f).

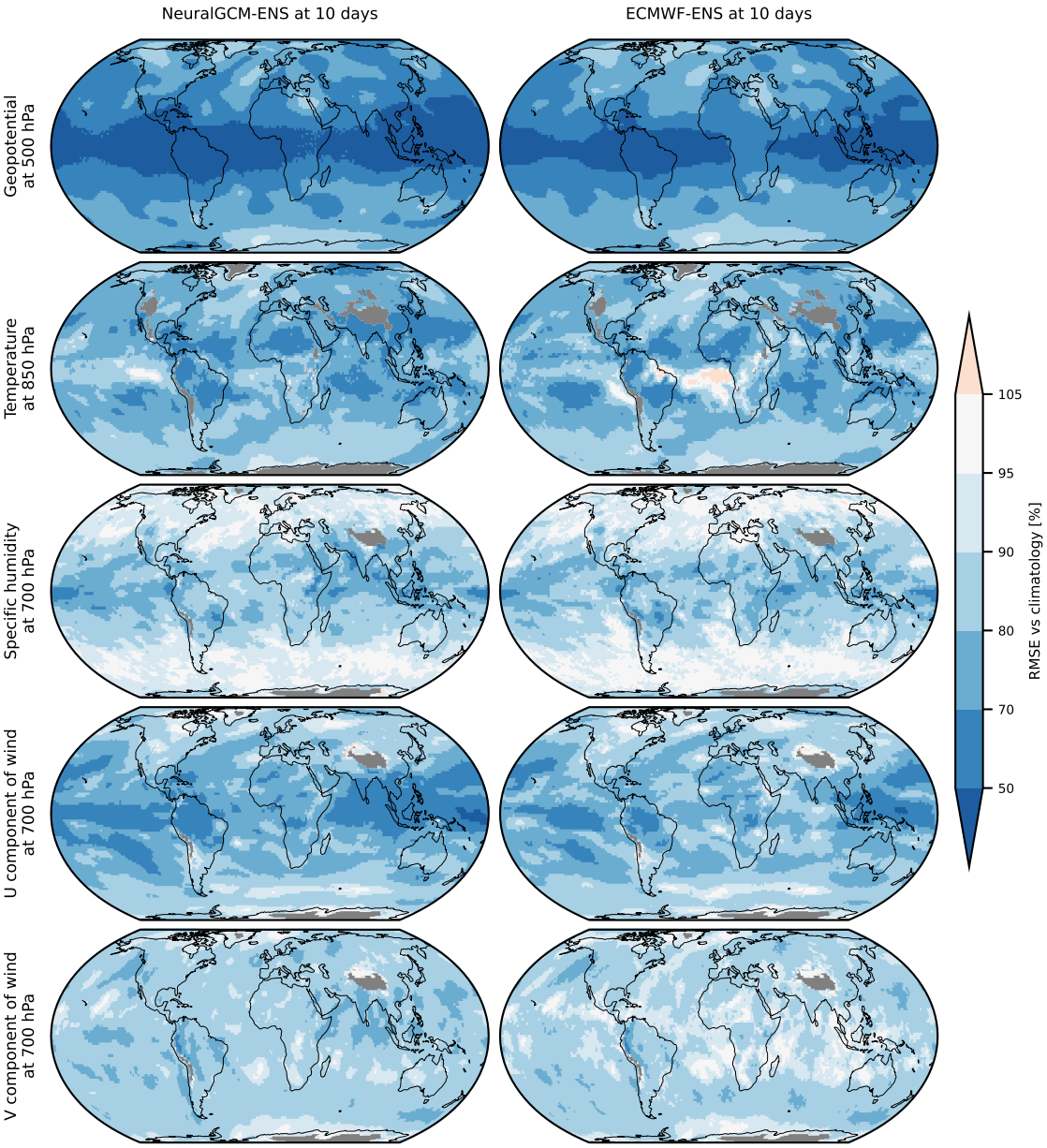

Fig. H9 Maps of root-mean-squared-error for NeuralGCM-ENS and ECMWF-ENS relative to 1990-2019 climatology for all forecasts in 2020.

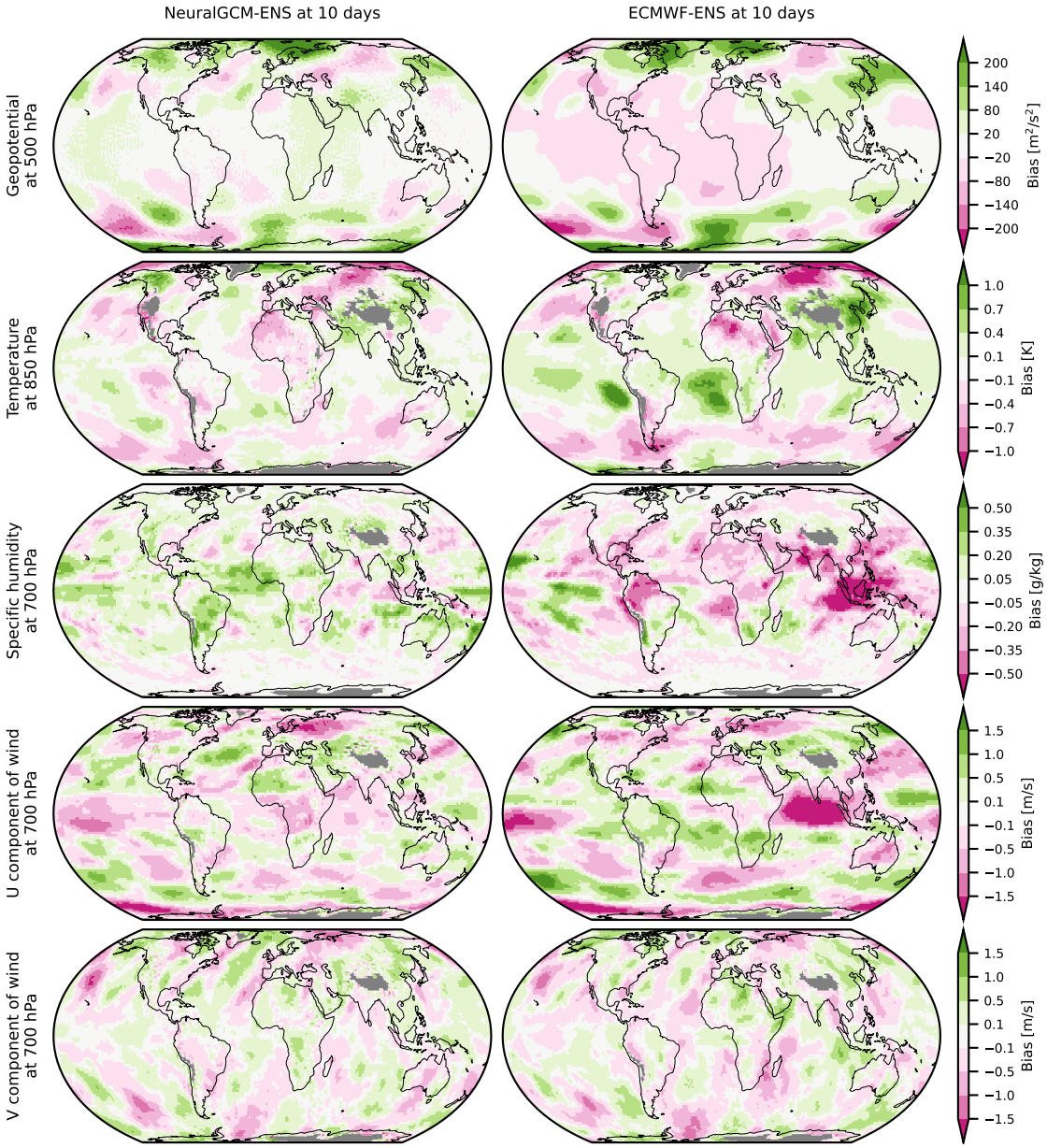

Fig. H10 Maps of average bias for NeuralGCM-ENS and ECMWF-ENS for all forecasts in 2020.

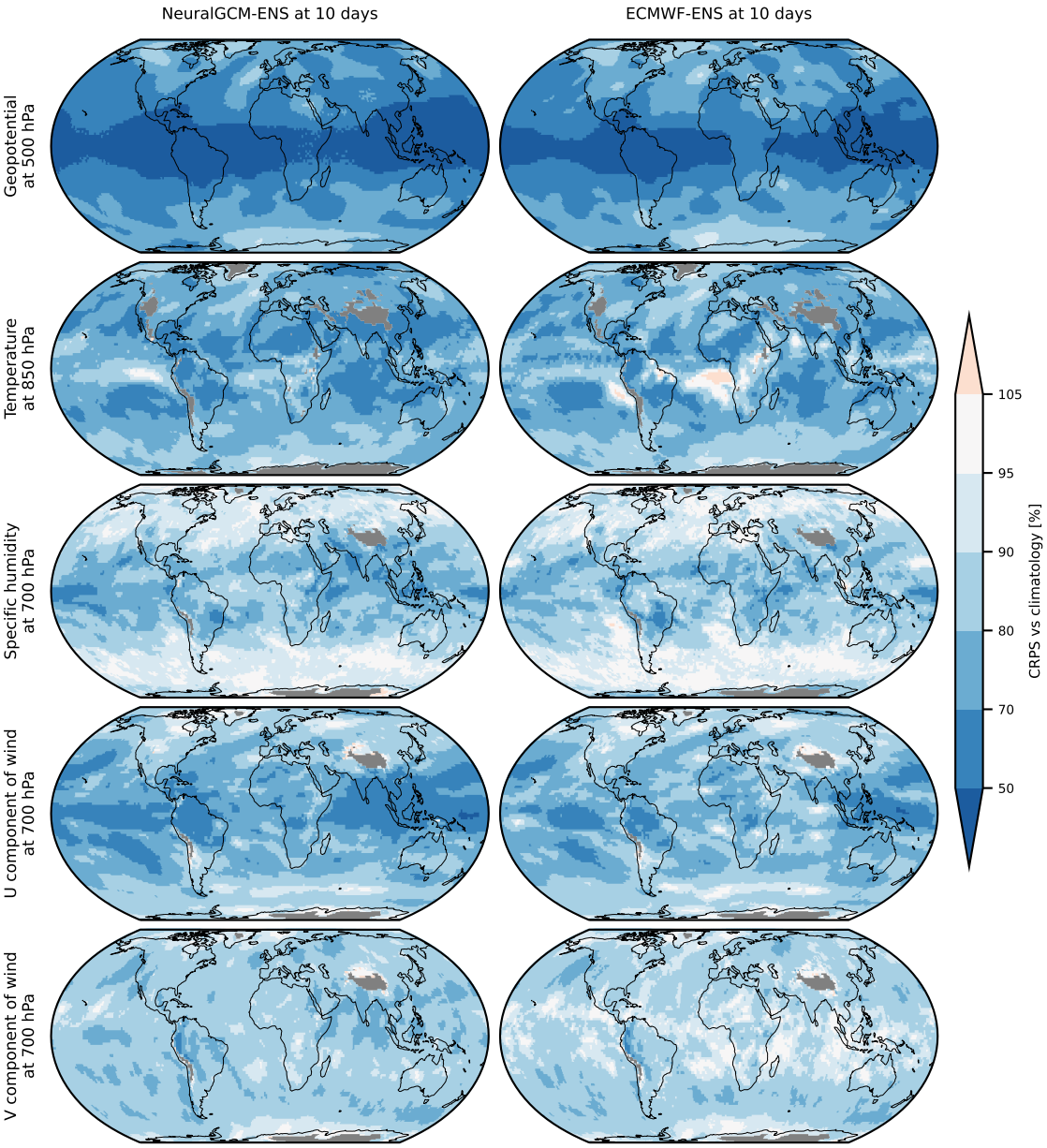

Fig. H11 Maps of average CRPS for NeuralGCM-ENS and ECMWF-ENS relative to a probabilistic climatology sampled from the years 1990-2019 for all forecasts in 2020.

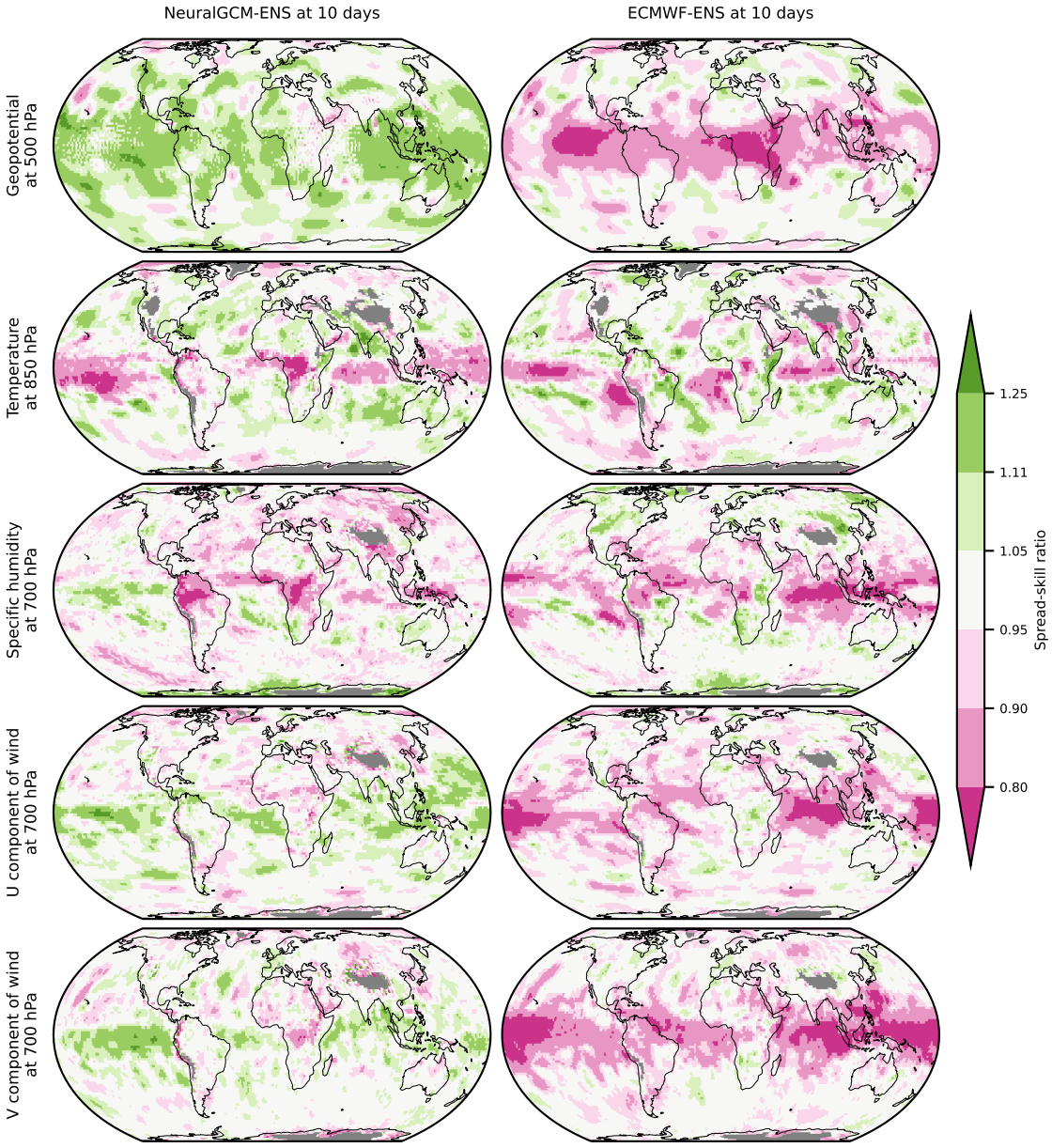

Fig. H12 Maps of spread-skill ratio for NeuralGCM-ENS and ECMWF-ENS relative to a probabilistic climatology sampled from the years 1990-2019 for all forecasts in 2020.

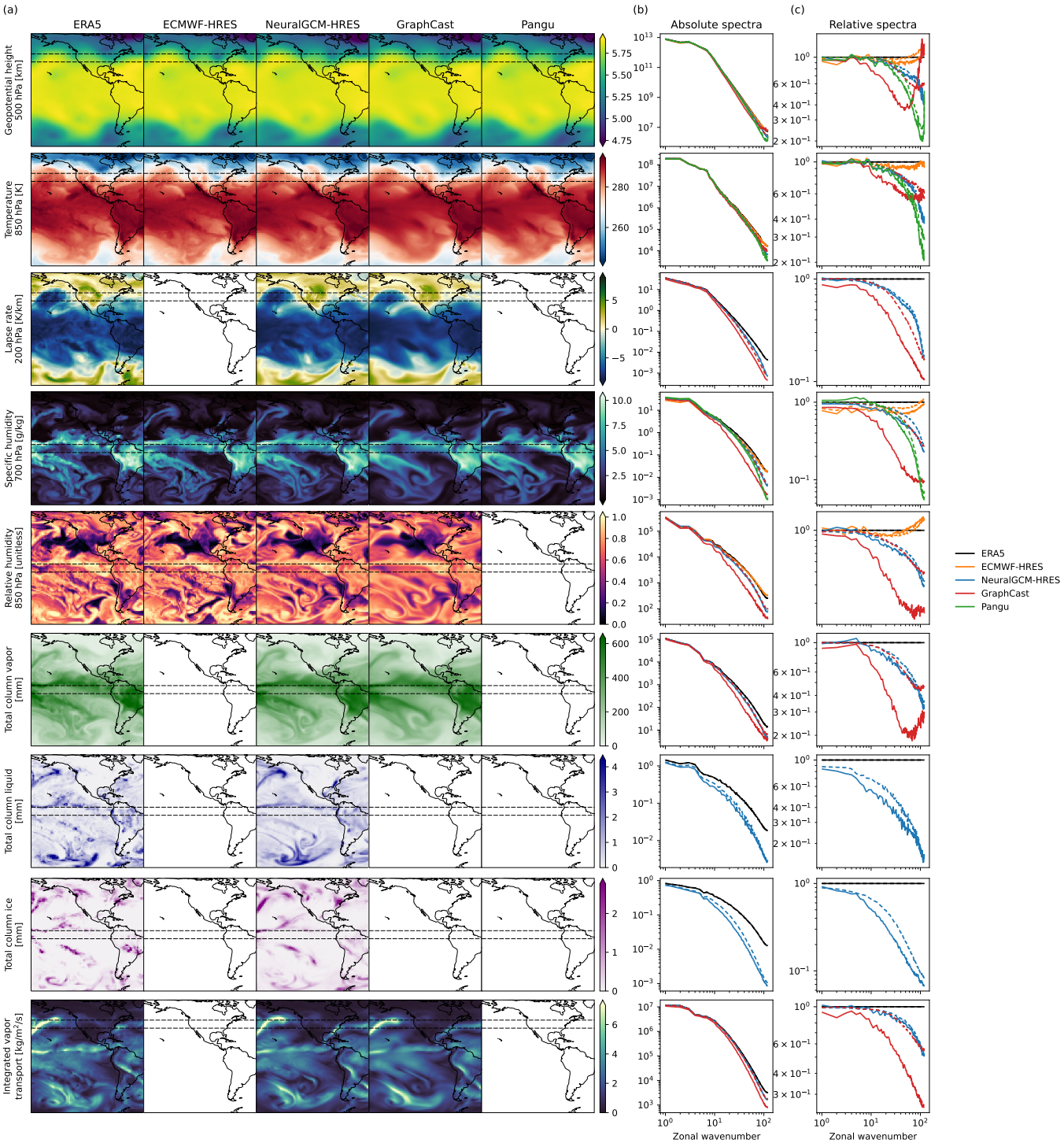

Fig. H13 Forecasts and power spectra for forecasts of thermodynamic variables. (a) Example 7-day forecasts initialized at 2020-01-01T00. (b) Absolute spectral density for 1-day (dashed) and 7-day (solid) forecasts of the indicated field, averaged over all forecasts initialized in 2020 and either over the tropics ($[-15°, 15°]$ North) or the extratropics ($[25°, 55°]$ North) as indicated by the area between dashed lines in panel (a). (c) Normalized spectral density, given by the absolute spectral density divided by the spectral density of ERA5.

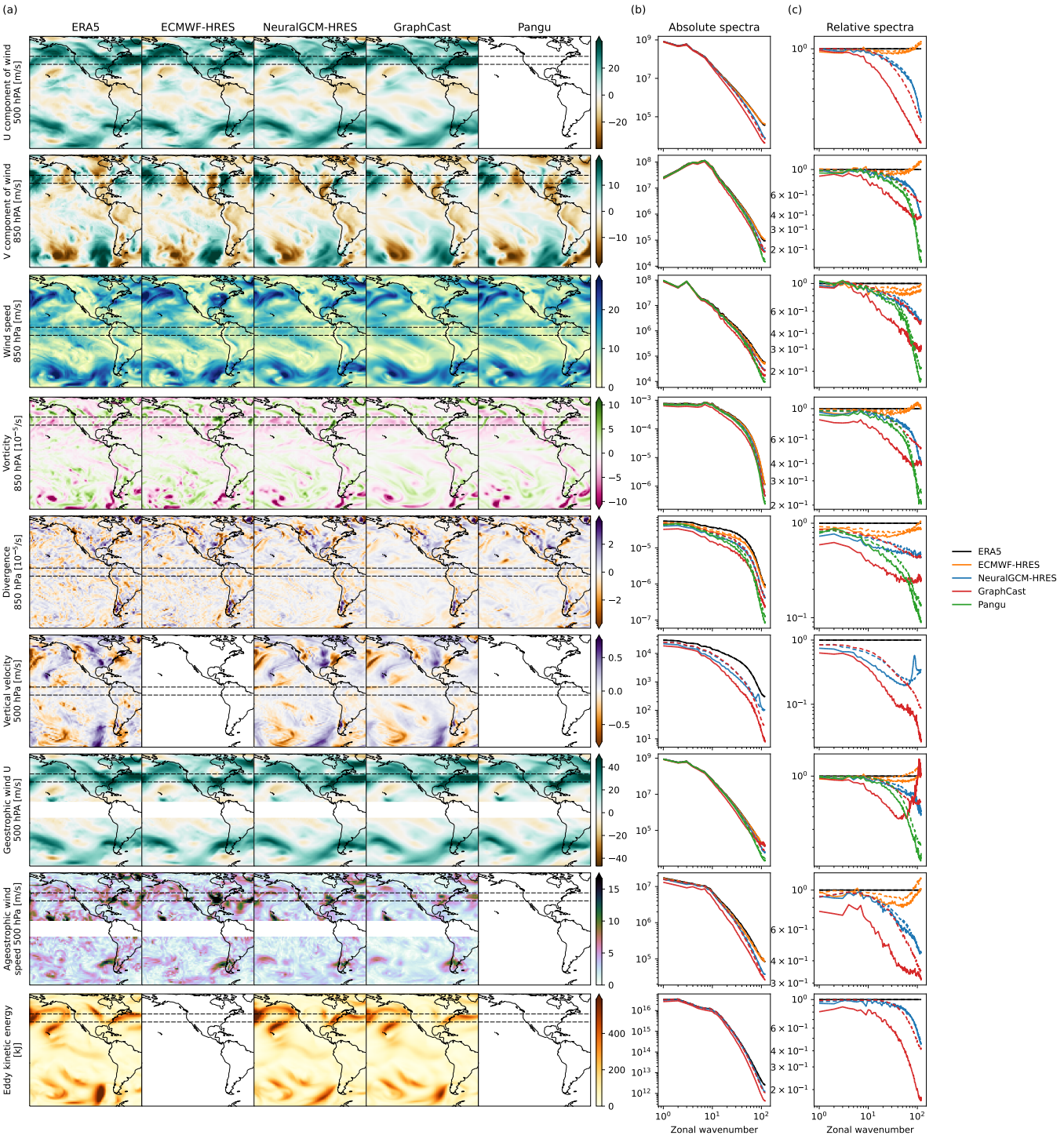

Fig. H14 Forecasts and power spectra for forecasts of wind variables, like in Fig. H13. Note that vertical velocity is recalculated for GraphCast from horizontal wind, rather than using the vertical velocity prediction directly output by GraphCast

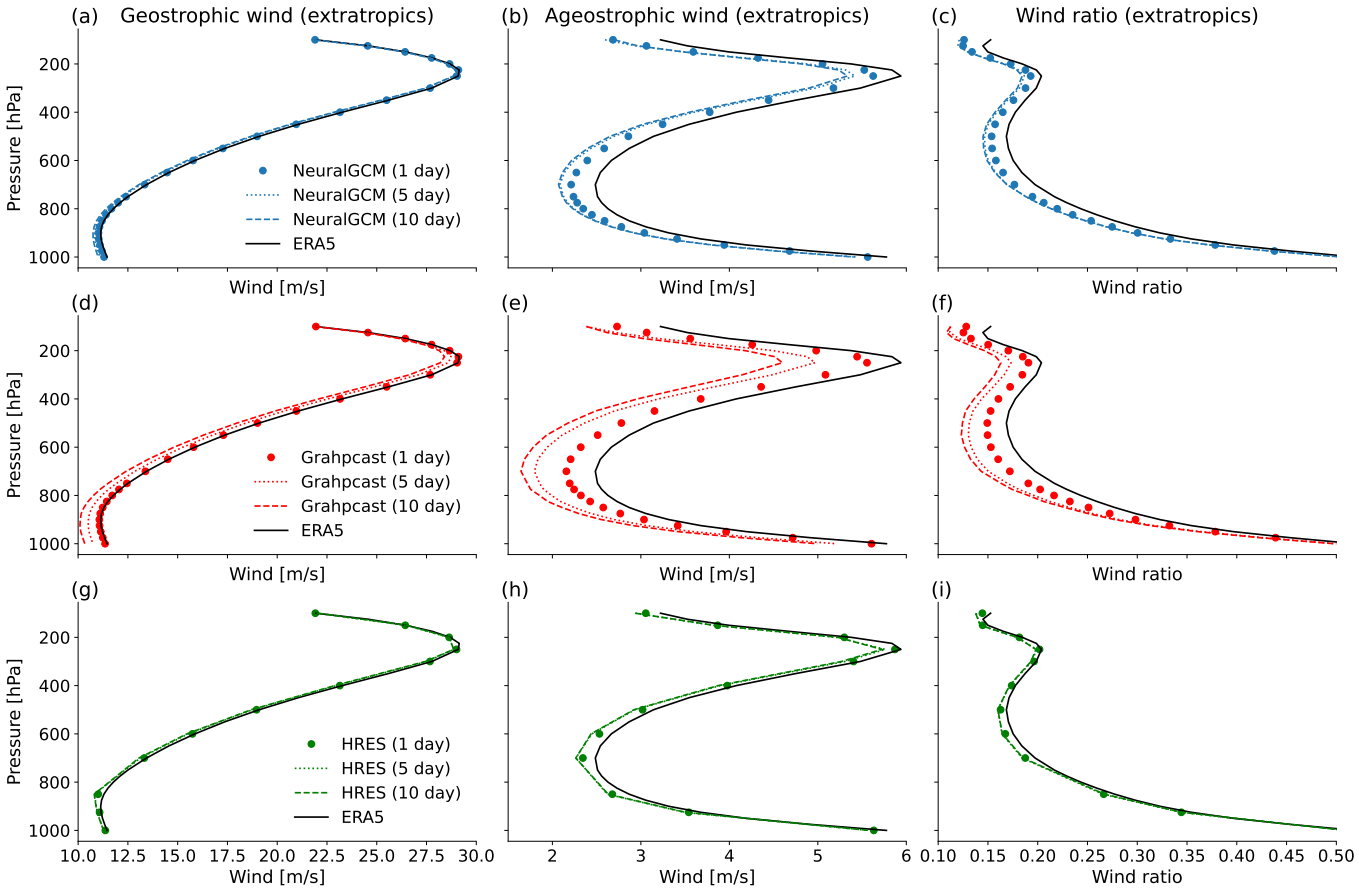

Fig. H15 Vertical profiles of the extratropical intensity (averaged between latitude 30°-70° in both hemispheres) and over initial conditions of (a,d,g) geostrophic wind, (b,e,h) ageostrophic wind and (c,f,i) the ratio of the intensity of ageostrophic wind over geostrophic wind for ERA5 (black continuous line in all panels), (a,b,c) NeuralGCM, (d,e,f) GraphCast and (g,h,i) ECMWF-HRES at lead times of 1 day, 5 days and 10 days.

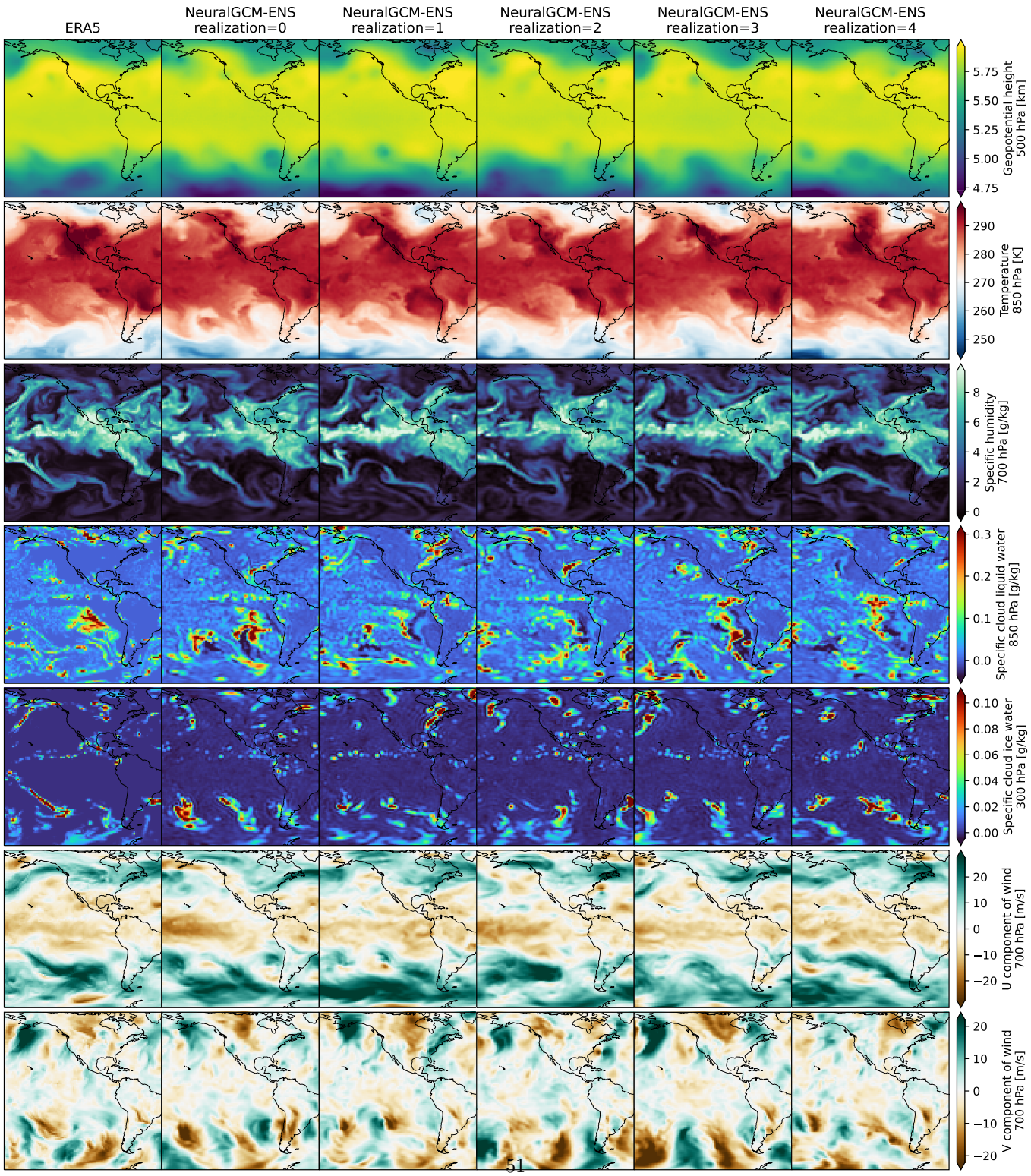

Fig. H16 Maps of five ensemble realizations of +15 day forecast fields from NeuralGCM-ENS initialized at 2020-08-22T12z, compared to ERA5.

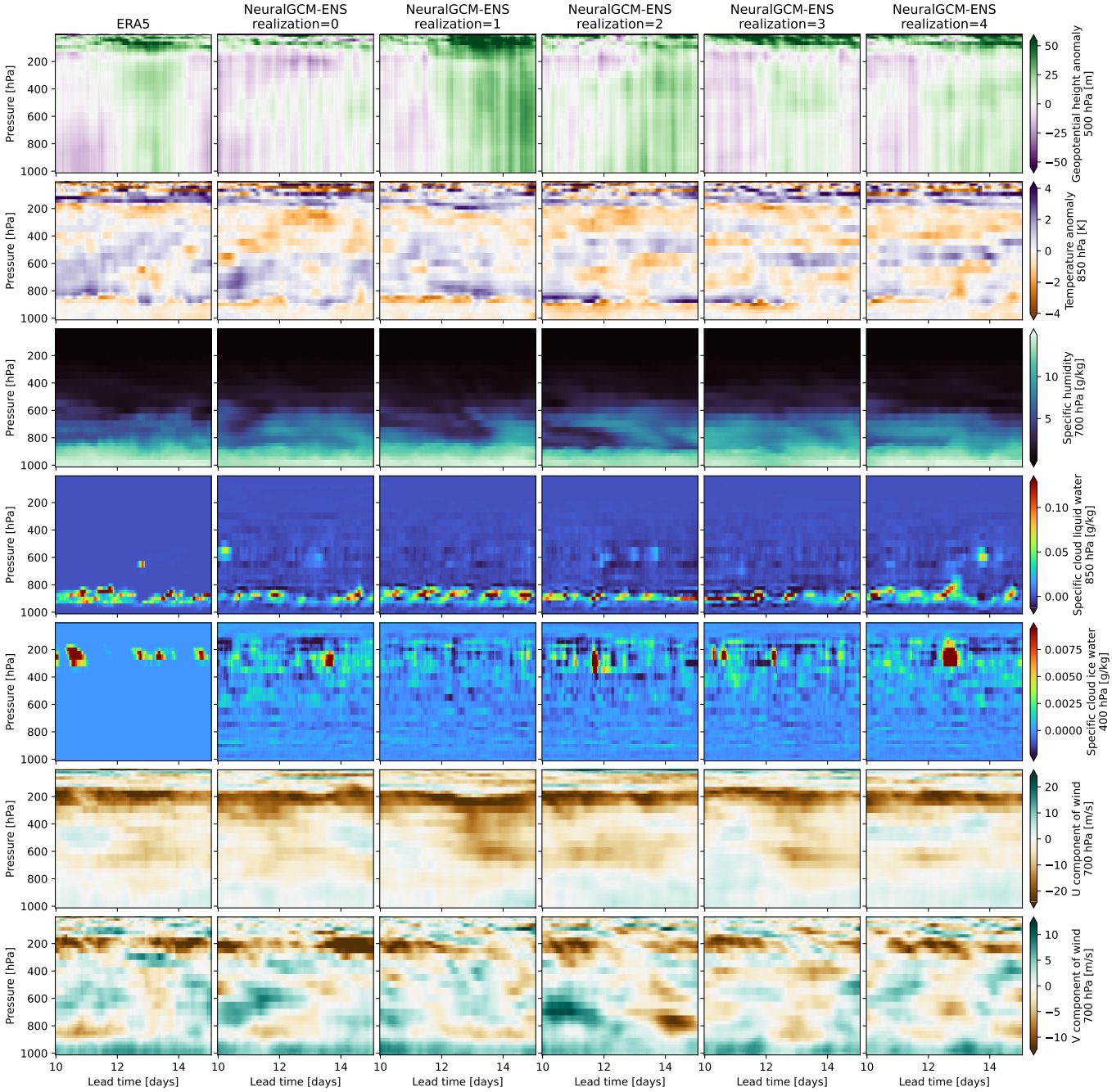

Fig. H17 Vertical profiles of five ensemble realizations of 10-15 day forecast fields at 0° N 0° W over the tropical Pacific ocean, from NeuralGCM-ENS initialized at 2020-08-22T12z, compared to ERA5.

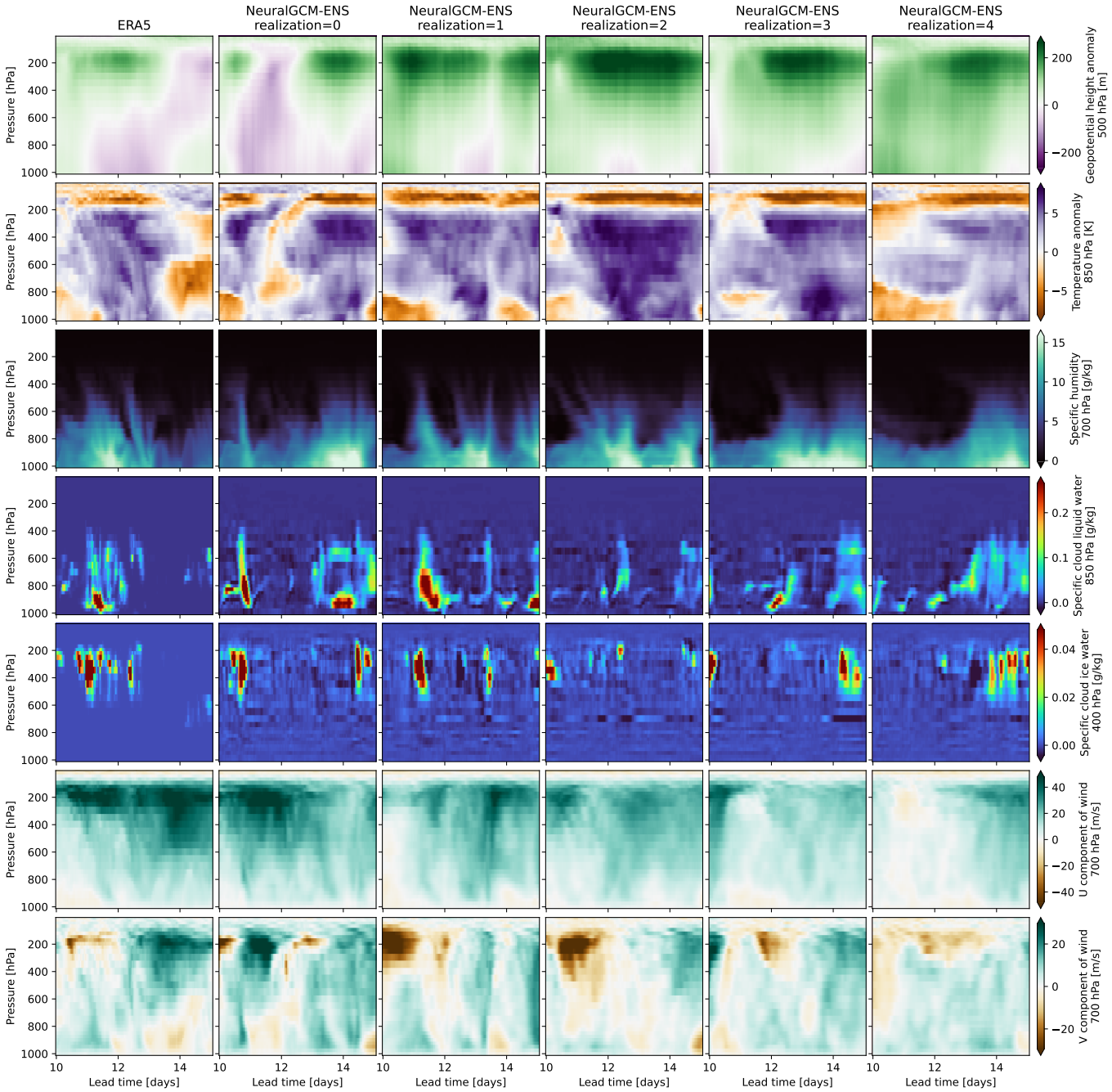

Fig. H18 Like Fig. H17, but at 42.3° N 71.1° W over Boston, MA, USA.

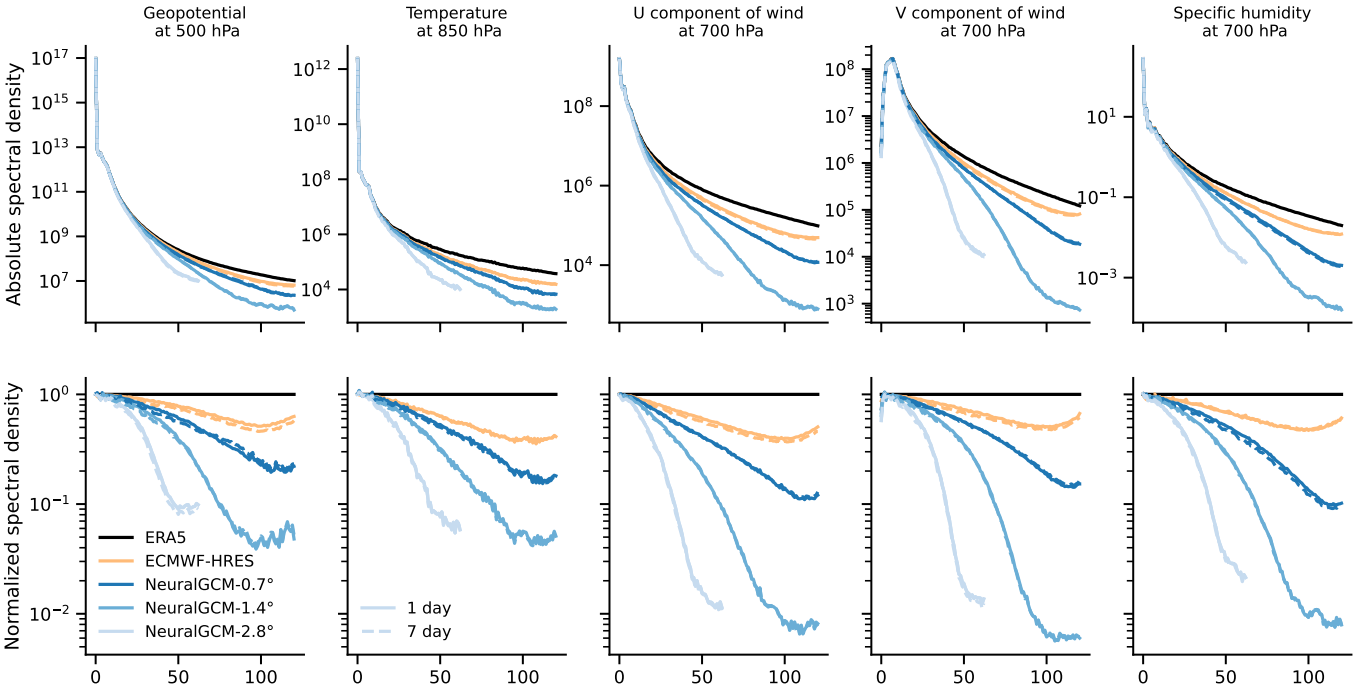

Fig. H19 Comparison of NeuralGCM models at different resolutions on zonal power spectra averaged over extratropics (15° to 55° latitude) for core atmospheric variables. (a) Absolute spectral density. (b) Spectral density normalized against ERA5.

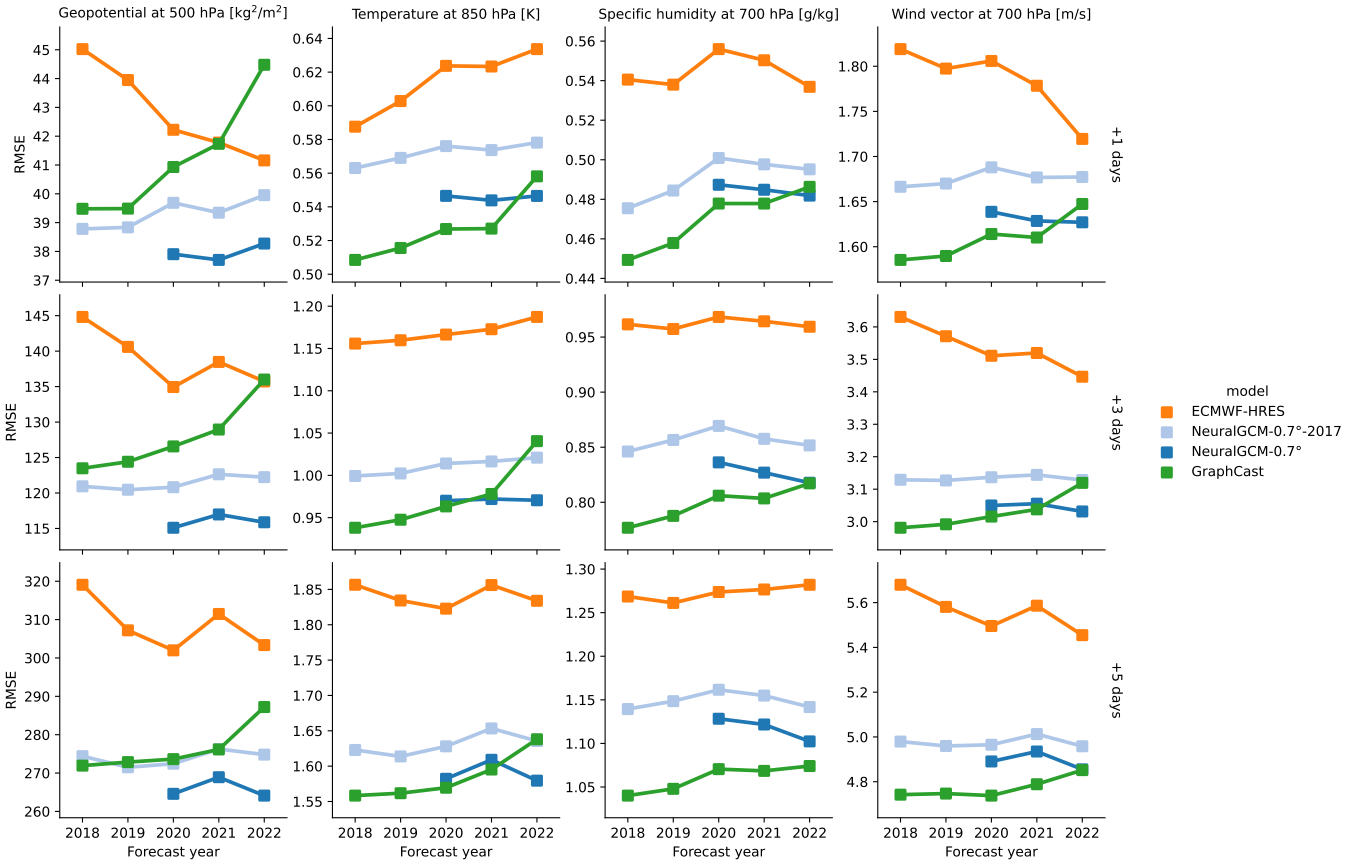

Fig. H20 RMSE for 1-, 3- and 5-day forecasts, starting in different forecast outside the training datasets for NeuralGCM and GraphCast. RMSE is averaged over forecasts initialized every 12 hours at midnight and noon UTC in the indicated year.

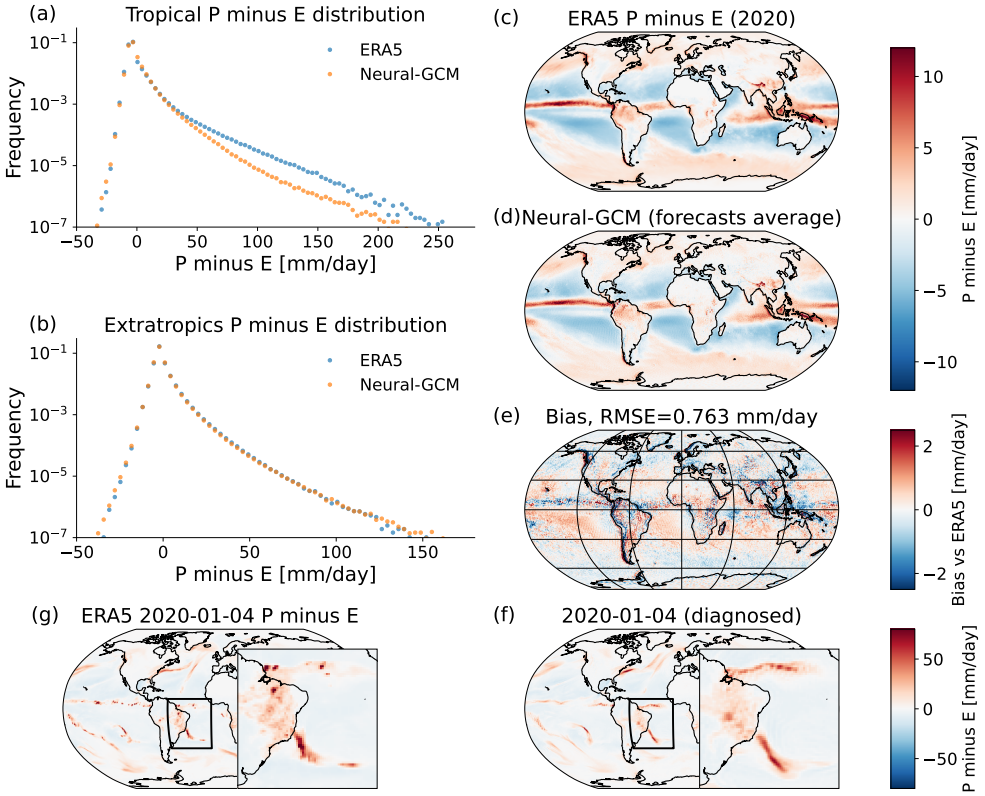

Fig. H21 Precipitation minus evaporation calculated from the third day of weather forecasts. (a) Tropical (latitudes $-20°$ t20°∘) Precipitation rate distribution, (b) Extratropical (latitudes $30°$ to $70°$ in both hemispheres), (c) mean precipitation minus evaporation for 2020 ERA5 and (d) NeuralGCM-0.7° (calculated from the third day of forecasts and averaged over initial conditions), (e) the bias between NeuralGCM-0.7° and ERA5, (f-g) Snapshot of daily precipitation for 01-04-2020 for (f) ERA5 and (g) NeuralGCM-0.7° (forecast initialized on 01-02-2020)

Appendix I Additional climate evaluations

I.1 Seasonal cycle

To assess the skill of our model for simulating seasonal cycles, we conduct a comprehensive comparison between NeuralGCM-1.4° resolution and ERA5. We ran 2-year simulations with 37 different initial conditions spaced at 10 days for the year 2019. Out of these 37 initial conditions, 35 successfully completed full two years without encountering model instability. The comparison is done for the year 2020, focusing on several key aspects. We begin by examining the global mean temperature at 850 hPa and find that NeuralGCM-1.4° closely resembles ERA5, both in terms of mean temperature and variability for all stable initial conditions (Fig. 4a).

To example the tropical circulation we analyze the Hadley cell seasonal cycle and its amplitude. The Hadley cell circulation is characterized by computing the mass streamfunction $\psi(\phi, p)$, which quantifies the mass transport between latitudes and altitudes. The mass streamfunction is

$$\psi(\phi, p) = \frac{2\pi cos(\phi)}{g} \int_{p}^{p_s} \bar{v} dp, \tag{I12}$$

where ϕ is the latitude, p is the pressure, g is the acceleration due to gravity, p_s is the surface pressure and $\bar{v}$ is the zonal mean meridional velocity. We find that NeuralGCM-1.4° is able to capture both the seasonal cycle and the amplitude of the Hadley cell circulation (Fig. I22). We also demonstrate in Fig. I23 that NeuralGCM-1.4° can accurately capture the spatial structure of the wind during the Indian monsoon and non-monsoon months.

To characterize the extratropical circulation, we plot zonal-mean zonal wind in different seasons and find that NeuralGCM-1.4° has a very similar structure compared to ERA5 (Fig. I22) with the exception that above 30hPa there are noticeable differences as NeuralGCM-1.4° does not optimize its predictions for these levels (since the upper-most level exist in sigma coordinates is ≈ 0.03). We also consider the extratropical storm tracks and we compare the seasonal cycle of the eddy kinetic energy (EKE). EKE is computed as:

$$\text{EKE} = \int_{150\text{hPa}}^{1000\text{hPa}} \frac{1}{2\text{g}} (\text{u}'^2 + \text{v}'^2) \text{dp}, \tag{I13}$$

where the prime symbol denotes deviations from the instantaneous zonal mean (i.e., $v' = v - \bar{v}$, where $\bar{v}$ represents the zonal mean of the variable), and g is the gravitational acceleration. We find that NeuralGCM-1.4° successfully captures the seasonal cycle of EKE, displaying the correct seasonal and spatial structure of EKE (Fig. I25).

We also illustrate the global annual cycle of atmospheric water and total kinetic energy ($\text{TKE} = \int_{150\text{hPa}}^{1000\text{hPa}} \frac{1}{2\text{g}} (\text{u}^2 + \text{v}^2) \text{dp}$), demonstrating that there is no visible drift in these quantities and that we achieve a realistic annual cycle (I24). This is in contrast to a recent attempt at hybrid modelling found that the atmosphere tend to dry out rapidly [20]. The ensemble mean of NeuralGCM-1.4°'s precipitable water accurately

captures the magnitude seen in ERA5 during 2020, yielding a smaller RMSE than the climatological value. However, the ensemble mean TKE of NeuralGCM-1.4° exhibits a slight bias, trending lower than the ERA5 values.

I.2 Tropical cyclone tracking

To assess NeuralGCM's capability to simulate Tropical Cyclones (TCs), we use the TempestExtremes tracker [36] to identify TC tracks. To compare NerualGCM TC results to ERA5 we first detect TCs in ERA5 native resolution (0.25°) using the default configuration of TempestExtremes.

The default configuration of TempestExtremes, used for the native ERA5 resolution (0.25°), utilizes sea level pressure (SLP) as the feature-tracking variable. Candidate TCs are initially identified based on SLP minima, and a closed contour criterion is applied, demanding an SLP increase of at least 2 hPa within a 5.5° Great Circle Distance (GCD) from the candidate point. Additionally, the difference between geopotential on the 300 and 500 hPa surfaces must decrease by 58.8 m^2s^{-2} within a 6.5° GCD from the candidate geopotential, taken as the maximum geopotential height difference within 1° GCD of the candidate location. These candidates are then linked over time to form TC paths. These paths have a maximum allowable distance of 8° between consecutive candidates, feature a maximum allowable gap size of 24 hours (representing time periods with no TC identification) and have a minimum trajectory length of 54 hours. For at least 10 time steps, the underlying orography has to be less than 150m, the storm formation is constrained within latitudes of $-50°$ to 50°, and the 10m wind magnitude has to exceed 10 m/s. Tracking is conducted at 6-hour output intervals for all resolutions.

One challenge with TempestExtremes is its tendency to yield significantly different numbers of Tropical Cyclones (TCs) when applied to data at varying resolutions [52]. To apply the tracker to NeuralGCM-1.4° and make meaningful comparisons with ERA5 data, we specify a set of tracking parameters that yield nearly identical TC tracks when ERA5 data is regridded to a resolution of 1.4°.

We fine-tuned these parameters so that TempestExtremes detected nearly identical TC tracks (as shown in Fig. I29a) and a similar number of TCs, with 88 in the native resolution (using the default parameters for tracking) and 84 in the 1.4° resolution (using the new set of parameters for tracking). The main differences is that we reduced the gradient requirements, necessitating an SLP increase of at least 0.7 hPa within a 5.5° GCD radius of the candidate node and the difference between geopotential on the 300 and 500 hPa surfaces must decrease by 25.8 m^2s^{-2} within a 6.5° GCD from the candidate geopotential, taken as the maximum geopotential height difference within 1° GCD of the candidate location. It's worth noting that, for the 1.4° tracking, we do not use the 10m wind criterion since this data was not available for the NeuralGCM simulation.

Next, we compare these results with TCs simulated by the X-SHiELD model (Fig. 4) when regridded to 1.4°, allowing for a fair comparison with NeuralGCM-1.4° at the same resolution. Since the X-SHiELD dataset lacks SLP information, we utilize vorticity for tracking TCs. In a manner similar to what is described in the previous

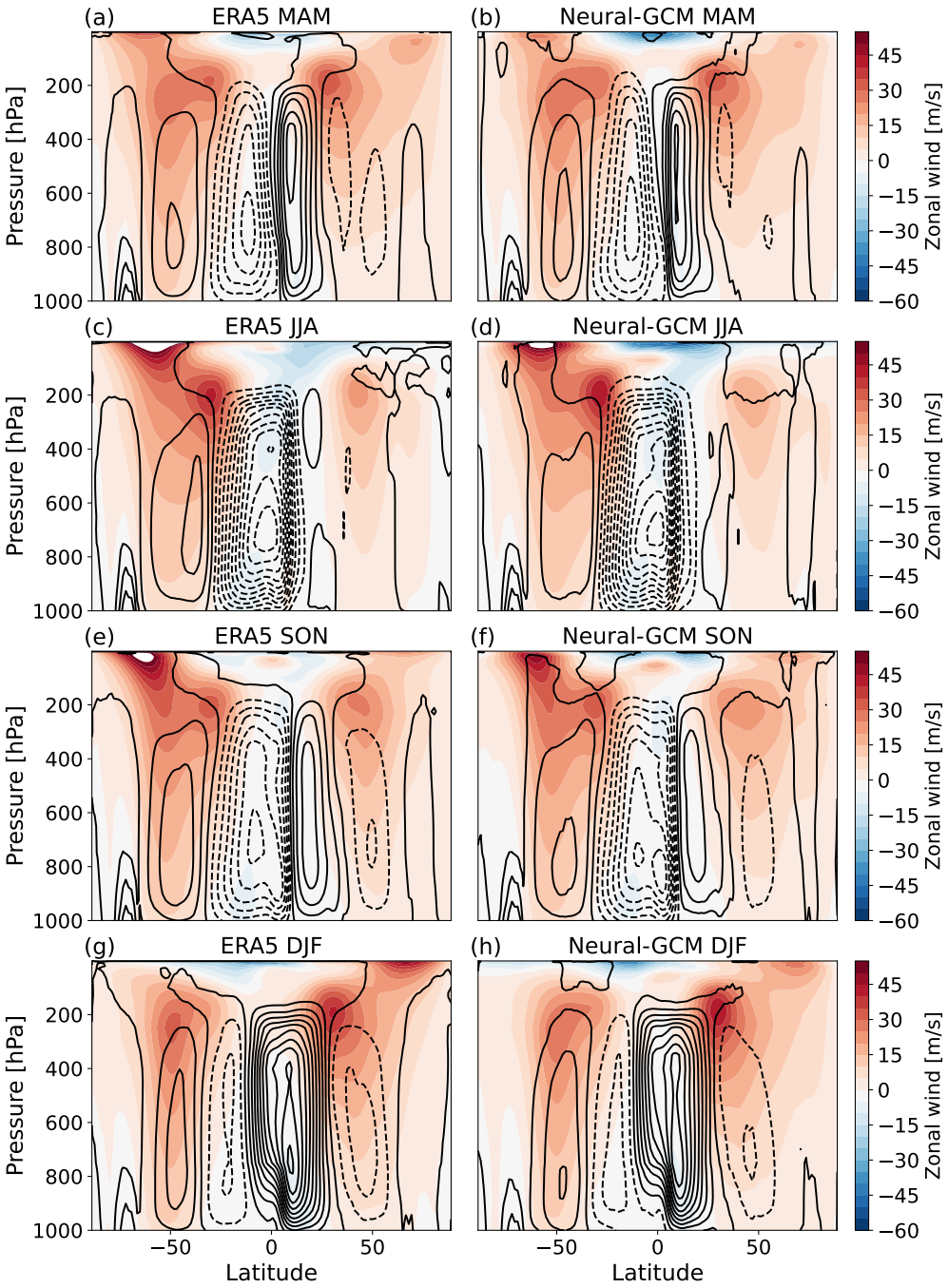

Fig. I22 The mass streamfunction (contours) and zonal-mean zonal wind (colors) as a function of pressure and latitude averaged over different seasons during 2020 for (a,c,e,f) ERA5 and (b,d,f,h) NeuralGCM-1.4° simulation initialized in October 18th 2019. Solid contours indicate positive values of mass streamfunction and dashed contours indicate negative values and contour intervals are 2×10^{10} kg/s.

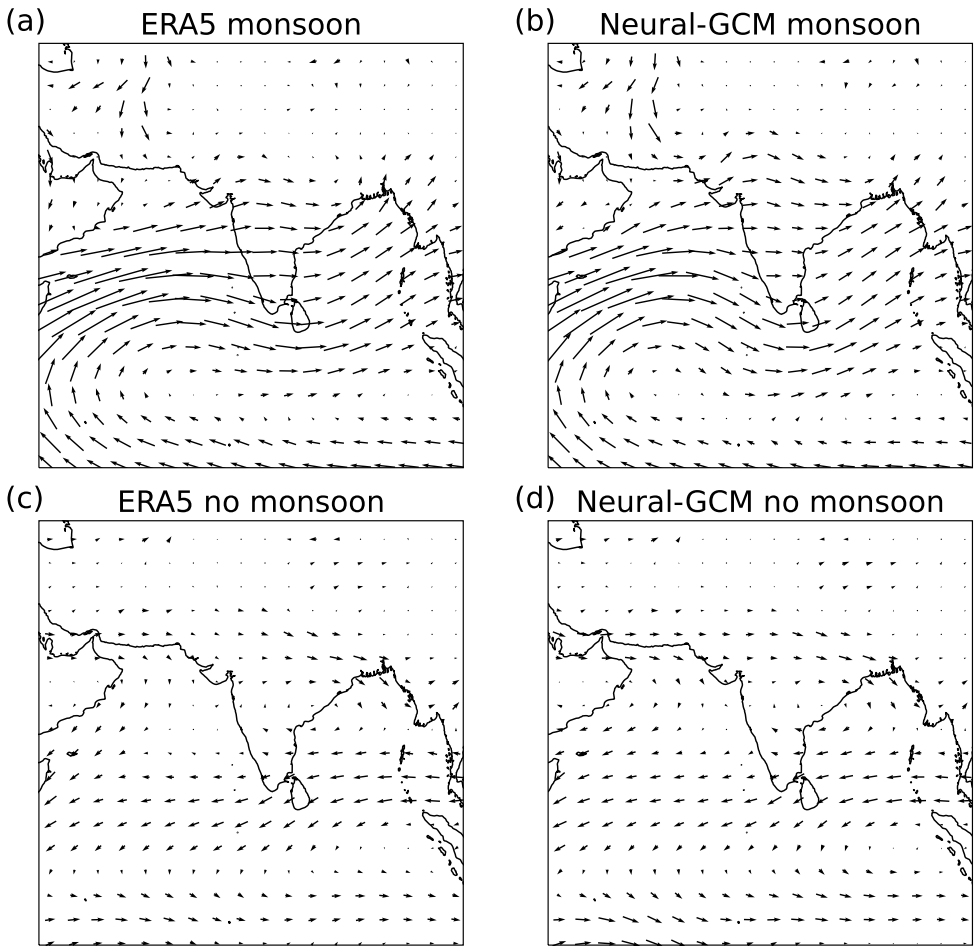

Fig. I23 Quiver plot of the time mean wind at 850 hPa for the Indian Monsoon months (defined here as June 15th to Septemper 15th; panels a and b) and no Monsoon months (defined here as January 1st to April 1st; panels b and d) for 2020 for (a,c) ERA5 (b,d) NeuralGCM-1.4° simulation (initialized in October 18th 2019)

paragraph, we identify a set of vorticity parameters that, when used with the TempestExtremes tracker on ERA5 data regridded to 1.4° resolution, yield results that closely matched TCs in native ERA5 data tracked using SLP as the criterion. We successfully fine-tune the vorticity parameters to ensure that TempestExtremes detects nearly identical TC tracks (as shown in Fig. I29b) when using the vorticity criterion compared to when using the tracker with native ERA5 resolution and the SLP criterion. This yields a similar number of TCs, with 88 in the native resolution and 86 when using the vorticity criterion in the 1.4° resolution. For vorticity tracking, we require a vorticity increase/decrease of at least 0.00065 s^{-1} within a 5.5° GCD radius of the candidate location. Additionally, we impose the requirement for a decrease in

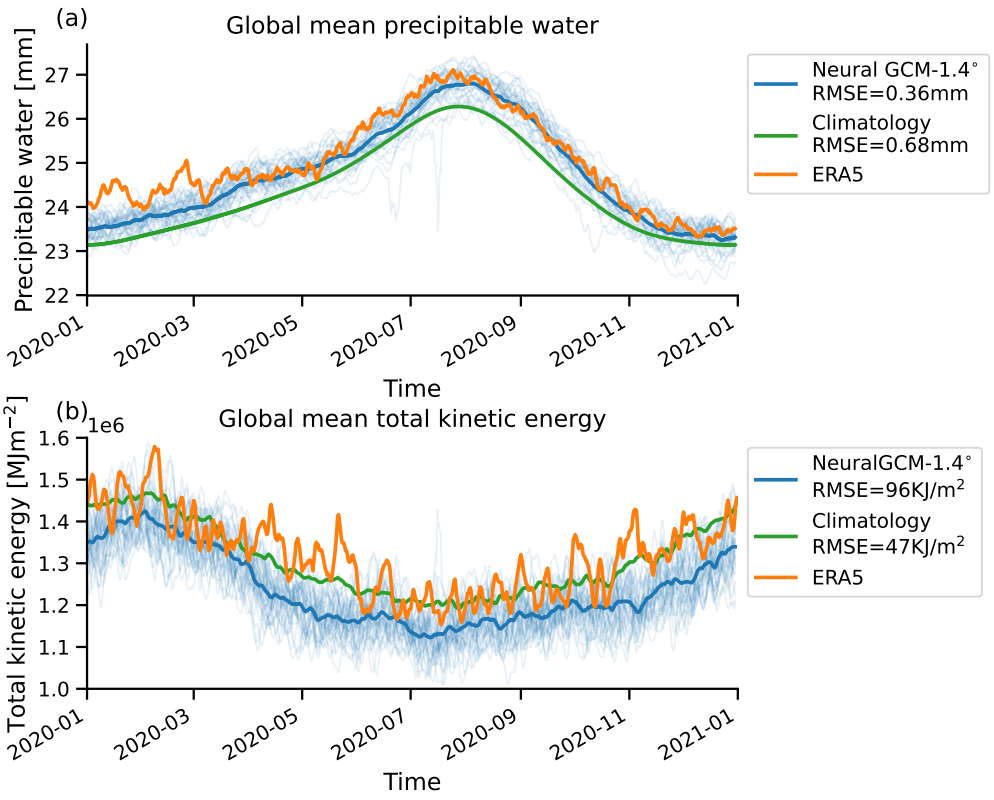

Fig. I24 Global mean (a) precipitable water and (b) total kinetic energy for 2020 ERA5 for 2020 (orange), climatology (defined as the averaged temperature between 1990-2019; green), and for NeuralGCM-1.4° for 2020 for simulations initialized every 10 days during 2019 (thick blue represents the ensemble mean, and thin blue lines indicate different initial conditions).

geopotential height difference between the 300 hPa and 500 hPa of 25.8 m^2s^{-2} within a 6.5° GCD from the candidate geopotential. (see Table I7).

We apply these new SLP and vorticity parameters to detect TCs in the NeuralGCM-1.4° simulation. We find that tracking TCs with both sets of parameters produces similar TC tracks (Fig. I29c). These TC tracks also closely resemble TC tracks identified in ERA5 data in terms of number of TCs, as well as their locations and shapes (Fig. I29).

We also employ the tracker to derive TC tracks from both ECMWF-ENS and NeuralGCM-ENS, each at a resolution of 1.4°. However, since ECMWF-ENS lacks data for geopotential at 300 hPa, which is necessary for the conditions we used above, we track TCs for the ensemble using the sea-level pressure (SLP) criterion in conjunction with the vorticity criterion, as previously described for simulations at the 1.4° resolution.

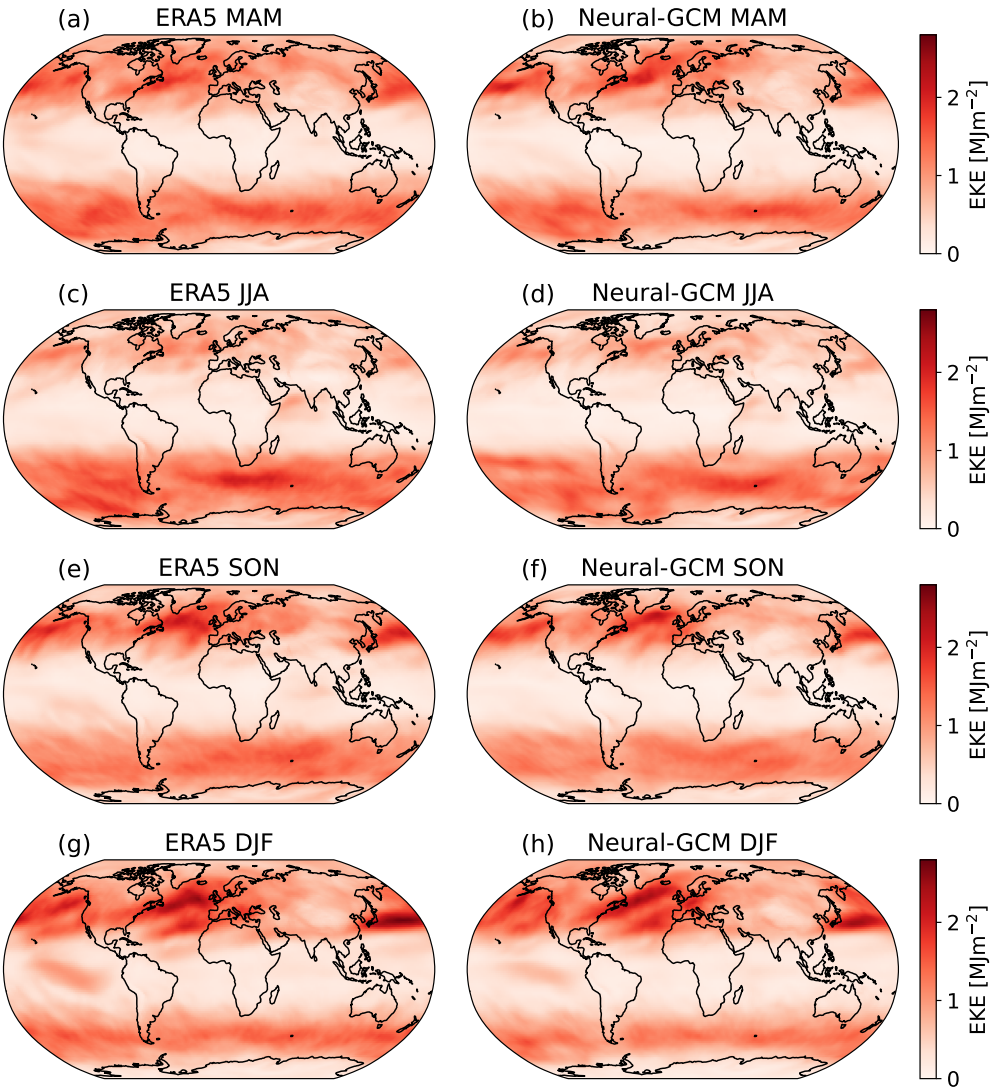

Fig. I25 Vertically integrated Eddy Kinetic Energy (EKE) as a function of longitude and latitude averaged over different seasons during 2020 for (a,c,e,f) ERA5 and (b,d,f,h) NeuralGCM-1.4° simulation (initialized in October 18th 2019).

I.3 CMIP6 models used in AMIP runs

We analyzed data from 22 CMIP6 models that were configured to use prescribed sea surface temperatures (SST), known as AMIP runs, taken from Google's Public Dataset program stored on Google Cloud Storage. Among these, for the following 17 models — BCC-CSM2-MR, CAMS-CSM1-0, CESM2, CESM2-WACCM, CanESM5, EC-Earth3, EC-Earth3-Veg, FGOALS-f3-L, GFDL-AM4, GFDL-CM4,

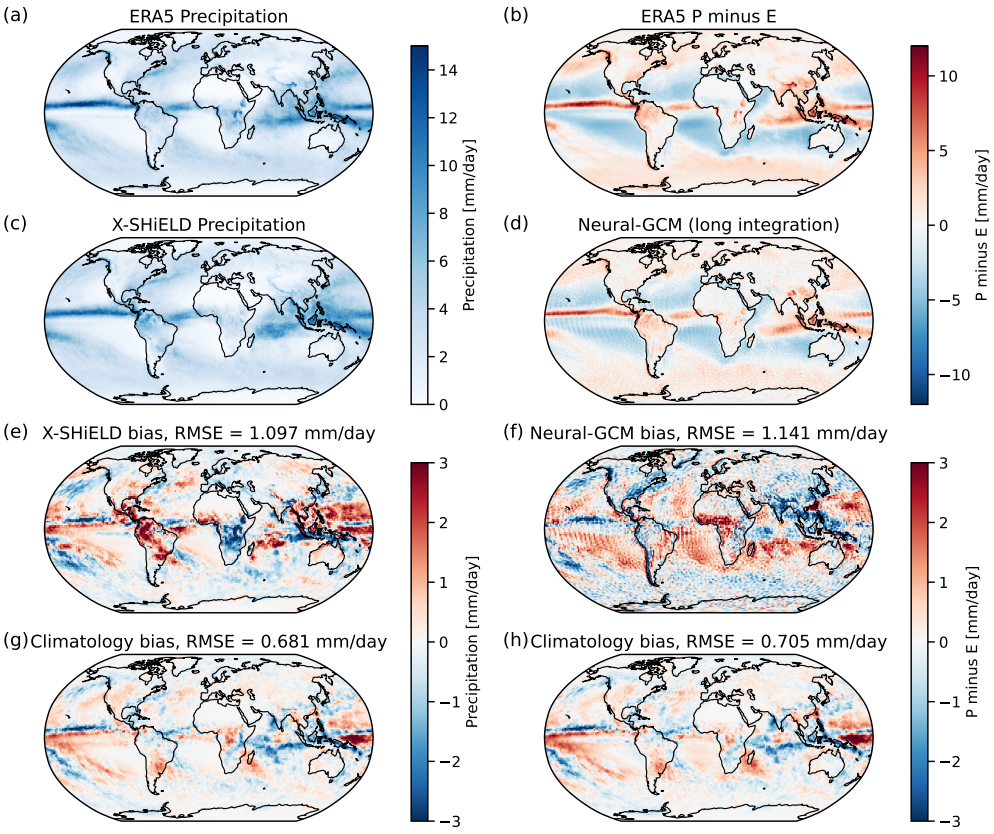

Fig. I26 Indirect comparison between precipitation bias in X-SHiELD and precipitation minus evaporation bias in NeuralGCM-1.4°. Mean precipitation calculated between 01-19-2020 to 01-17-2021 for (a) ERA5 (b) X-SHiELD and the biases in (c) X-SHiELD and (d) climatology (ERA5 data averaged over 1990-2019). Mean precipitation minus evaporation calculated between 01-19-2020 to 01-17-2021 for (a) ERA5 (b) NeuralGCM-1.4° (initialized in October 18th 2019) and the biases in (c) NeuralGCM-1.4° and (d) climatology (data averaged over 1990-2019).

	Criterion 1	Criterion 2
Sea level pressure tracking (0.25°)	SLP change of 200Pa over 5.5 GCD	Difference between geopotential surfaces at 300 and 500 hPa decrease by at least -58.8m^2/s^2 over 6.5 GCD
Sea level pressure tracking (1.4°)	SLP change of 60Pa over 5.5 GCD	Difference between geopotential surfaces at 300 and 500 hPa decrease by at least -25.8m^2/s^2 over 6.5 GCD
Vorticity tracking (1.4°)	Vorticity at 850hPa change of $\pm 0.00006 s^{-1}$ over 5.5GCD	Difference between geopotential surfaces at 300 and 500 hPa decrease by at least -25.8m^2/s^2 over 6.5 GCD
SLP ensemble tracking (1.4°)	SLP change of 60Pa over 5.5 GCD	Vorticity at 850hPa change of $\pm 0.00006 s^{-1}$ over 5.5GCD

Table I7 The parameters used for TC tracking when using different model resolutions.

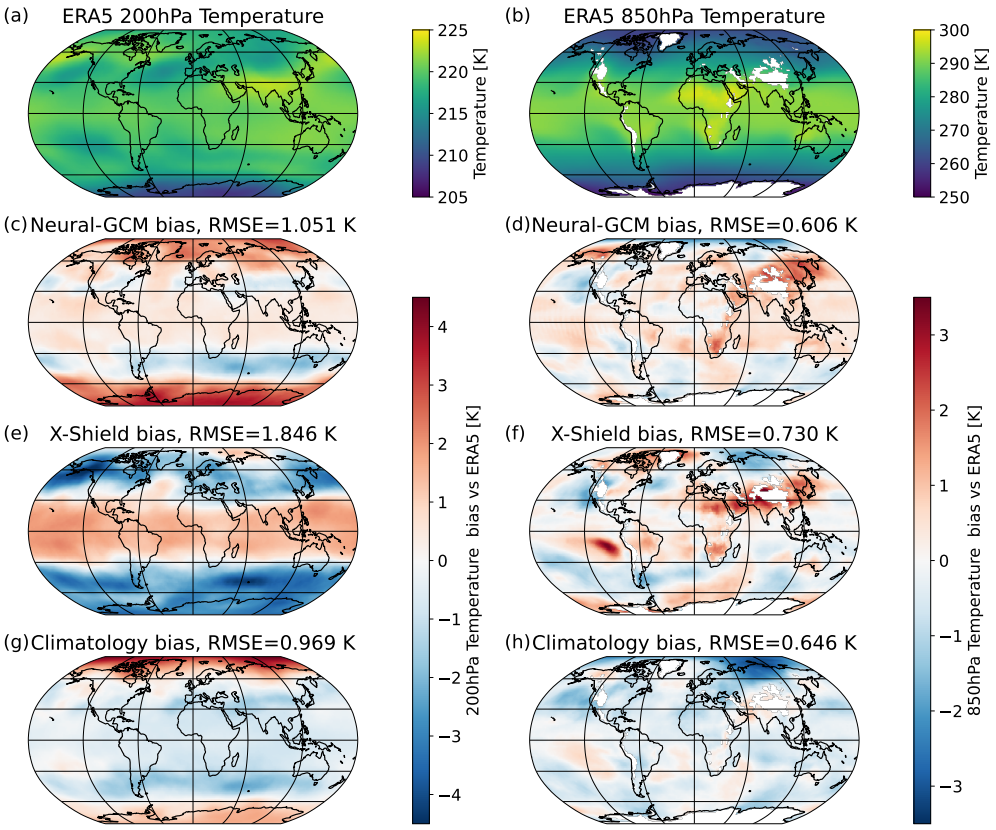

Fig. I27 Mean temperature between 01/19/2020 to 01/17/2020 for (a) ERA5 at 200hPa and (b) 850hPa. (c,d) the bias in the temperature for NeuralGCM-1.4°, (e,f) the bias in X-SHiELD and (g,h) the bias in climatology (calculated from 1990-2019). NeuralGCM-1.4° was initialized in 18th of October (similar to X-SHiELD)

GFDL-ESM4, GISS-E2-1-G, IPSL-CM6A-LR, MIROC6, MRI-ESM2-0, NESM3, and SAM0-UNICON — we utilized the "r1i1p1f1" variant identifier to which we had access. However, for the remaining five models, we resorted to using alternative variant identifiers: "r1i1p1f2" for CNRM-CM6-1 and CNRM-ESM2-1, "r2i1p1f3" for HadGEM3-GC31-LL, "r1i1p1f3" for HadGEM3-GC31-MM, and "r1i1p1f2" for UKESM1-0-LL. Additionally, although data from the E3SM model was available from Google's Public Dataset program, we chose not to include it in our analysis due to some temperature artifacts near Antarctica.

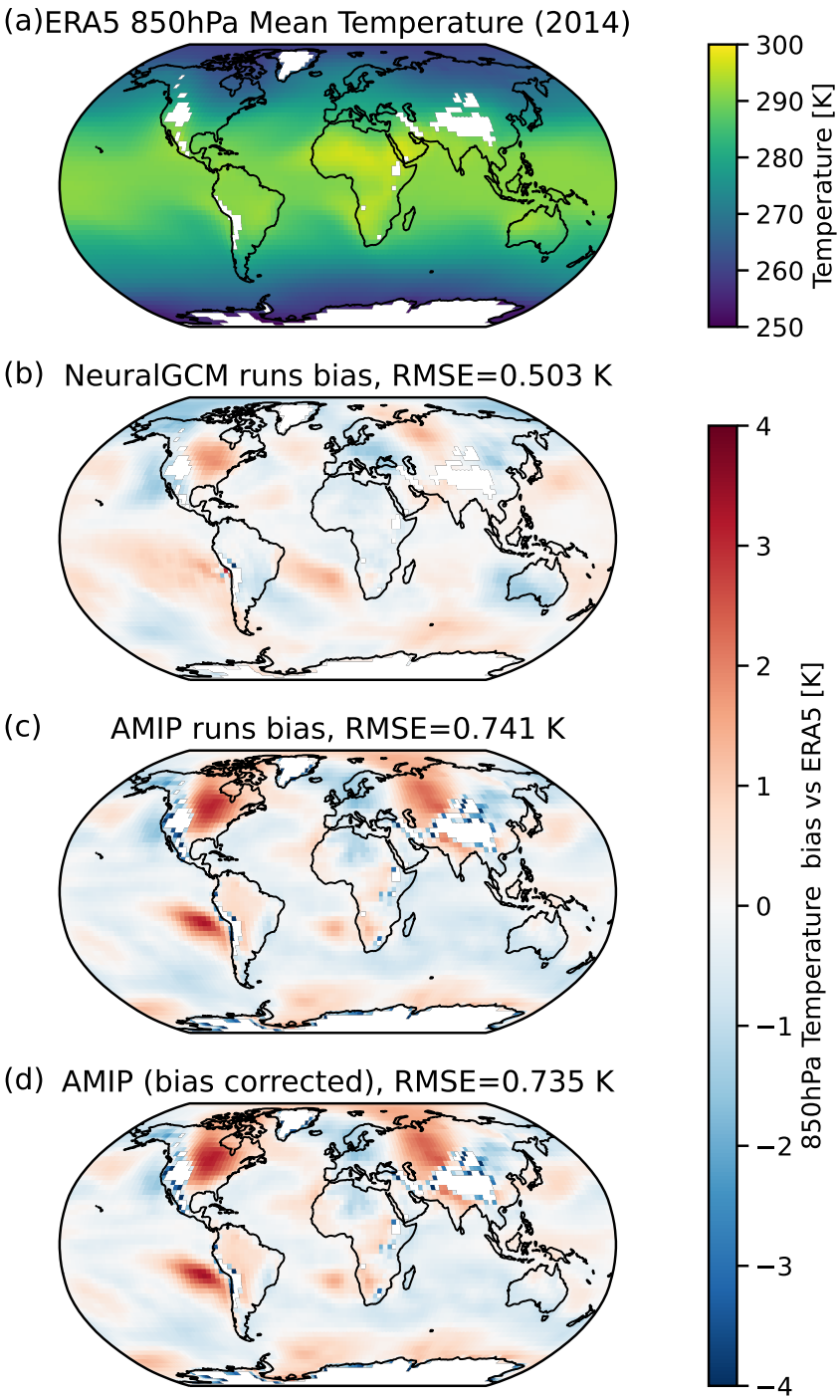

Fig. I28 Mean temperature for 2014 for (a) ERA5 at 850 hPa. (e) The bias in the temperature for NeuralGCM-2.8° averaged across 22 simulations with different initial conditions (all initialized in 1980). (f) the bias in AMIP simulations averaged across 22 models (models used are described in I.3). (g) The bias in AMIP simulations averaged across 22 models after removing the horizontal temperature bias at 850 hPa.

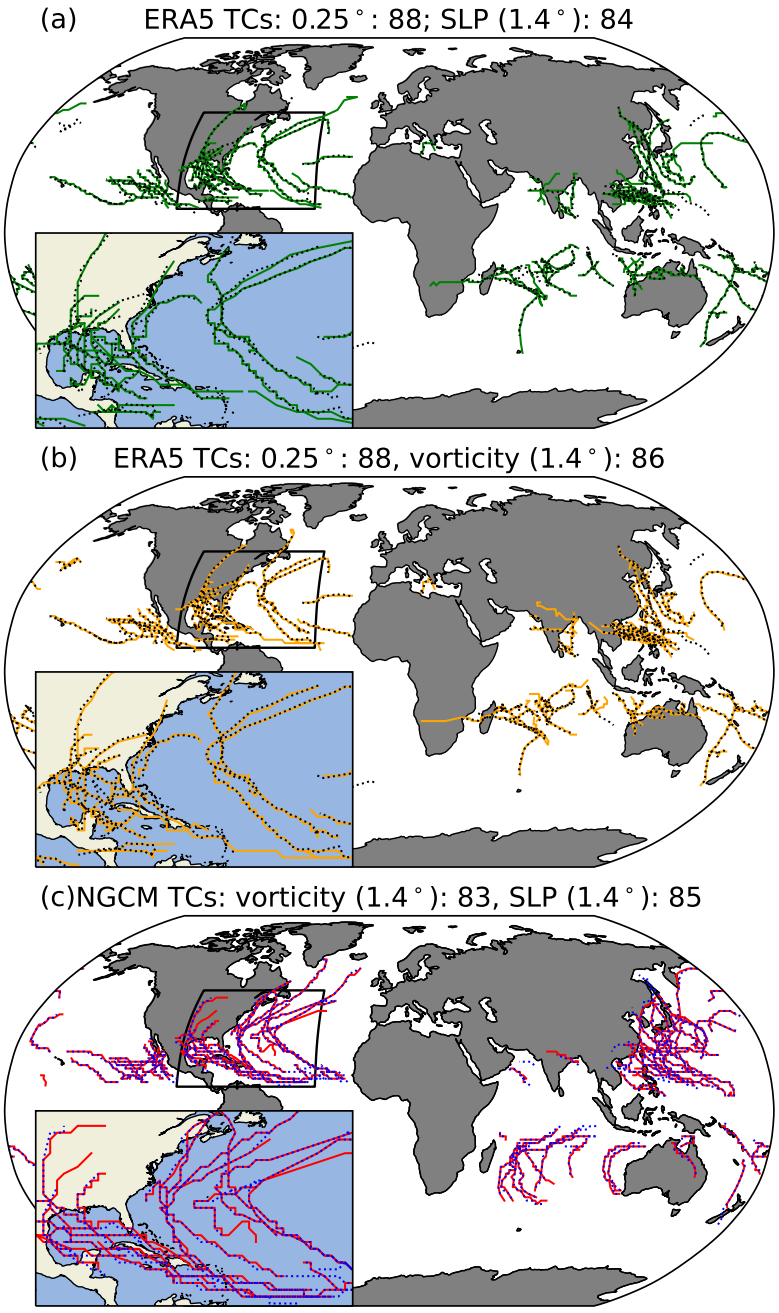

Fig. I29 Tropical Cyclone (TC) tracks identified from (a,b) ERA5 and (c) NeuralGCM-1.4° using different criterion's and resolutions. (a) TC tracks from ERA5 native resolution (0.25°) using sea level pressure criterion (SLP; dotted black) and ERA5 at 1.4° resolution using SLP modified criterion (green). (b) TC tracks from ERA5 native resolution (0.25°) using SLP criterion (dotted black) and ERA5 at 1.4° resolution using vorticity criterion (orange). (c) TC tracks from NeuralGCM-1.4° using SLP modified criterion (dotted blue) and vorticity criterion (red). The TC tracking was applied to all simulations between 01-19-2020 to 01-17-2021, which were the dates that were available for the X-SHiELD model. NeuralGCM-1.4° initialized on 10-18-2019 (similar to X-SHiELD). Insets show a zoom in into the Northern Atlantic region.

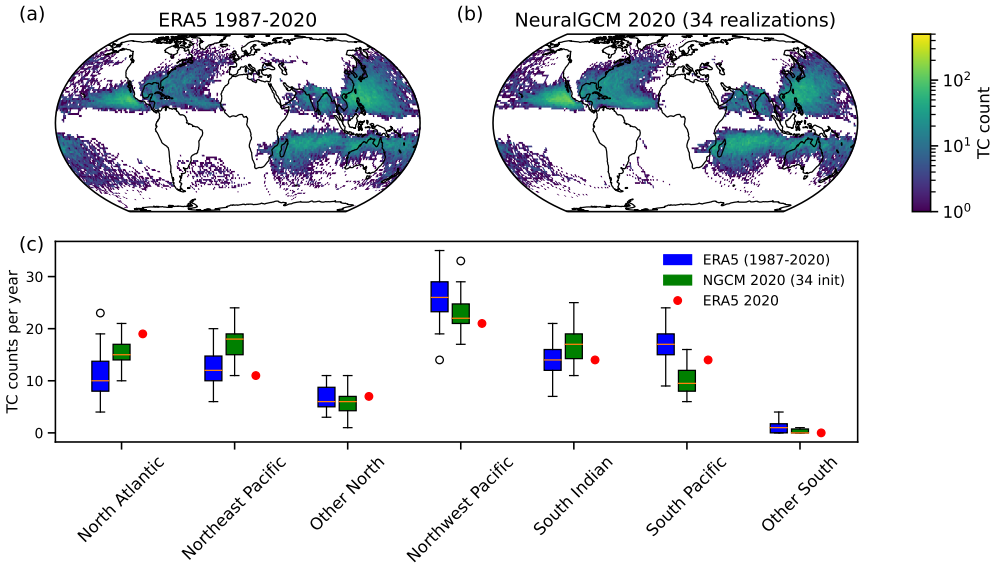

Fig. I30 Tropical Cyclone (TC) densities and annual regional counts. (a) TC density from ERA5 data spanning 1987-2020. (b) TC density from NeuralGCM-1.4° for 2020, generated using 34 different initial conditions all initialized in 2019. (c) Box plot depicting the annual number of TCs across different regions, based on ERA5 data (1987-2020), NeuralGCM-1.4° for 2020 (34 initial conditions), and red markers show ERA5 for 2020. In the box plots, the red line represents the median; the box delineates the first to third quartiles; the whiskers extend to 1.5 times the interquartile range (Q1 - 1.5IQR and Q3 + 1.5IQR), and outliers are shown as individual dots. Each year is defined from January 19th to January 17th of the following year, aligning with data availability from XSHiELD. For NeuralGCM simulations, the 3 initial conditions starting in January 2019 exclude data for January 17th, 2021, as these runs spanned only two years.