

001
002
003
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046

Supplemental Information: *GEN²: A Generative Prediction-Correction Framework for Long-time Emulations of Spatially-Resolved Climate Extremes*

Mengze Wang^{1†}, Benedikt Barthel Sorensen^{1†},
Themistoklis P. Sapsis^{1*}

¹*Department of Mechanical Engineering, Massachusetts Institute of Technology, Street, Cambridge, 02139, MA, USA.

*Corresponding author(s). E-mail(s): sapsis@mit.edu;
†These authors contributed equally to this work.

047 **Supplementary Methods**

048 1. Stochastic Emulator
049 2. Machine Learned Debiasing
050 3. Data Post-processing and Evaluation Metrics

052

053

054 **Supplementary Notes**

055

056 **Supplementary Figures**

057 1. Linearity of PCA coefficients as functions of the global mean temper-
058 ature.
059 2. Raw Wheeler-Kiladis spectrum of zonal wind.
060 3. Machine-learning correction of the bias in quantiles.
061 4. Machine-learning correction of the bias in two-point correlations.

062

063

064 **Supplementary Tables**

065 1. RMSE of single-point statistics of U, V .
066 2. RMSE of single-point statistics of T, Q .
067 3. RMSE of two-point correlation of U, V .
068 4. RMSE of two-point correlation of T, Q .

069

070

071

072

073

074

075

076

077

078

079

080

081

082

083

084

085

086

087

088

089

090

091

092

Supplementary Methods

1 Stochastic Emulator

Here we describe in detail the first part of our climate modeling framework, the linear stochastic emulation. In summary, the emulator takes as input a time series of the global mean temperature $T_g(t)$ and outputs a time series of the local state of the climate $\mathbf{u}(\mathbf{x}, t) = [u_1, u_2, u_3, u_4]^\top = [U(\mathbf{x}, t), V(\mathbf{x}, t), T(\mathbf{x}, t), Q(\mathbf{x}, t)]^\top$ where U, V, T, Q are the zonal and meridional wind speeds, temperature and humidity respectively and the spatial dimensions $\mathbf{x} = (\theta, \varphi)$ are the longitude and latitude, $\theta \in [-\pi/2, \pi/2]$ and $\varphi \in [0, 2\pi]$. The time step size of t is three hours for ERA5 dataset and one day for the CMIP6 MPI model. Consistent with the formulation of modern climate models – and to reduce the data to a manageable size – our model operates at a fixed altitude, and thus the spatial dimension is 2D. We focus here exclusively on the near-surface climate, but our model could be directly applied to any altitude.

Stated succinctly, our approach consider a principal component analysis (PCA) of the climate data $\mathbf{u}(\mathbf{x}, t) = \sum_j a_j(t) \phi_j(\mathbf{x})$ and attempts to model the temporal coefficients $a_j(t)$ for a given spatial basis $\phi_j(\mathbf{x})$. Our emulator is therefore built on three fundamental assumptions:

1. The PCA basis $\phi_j(\mathbf{x})$ computed from the climate during a sufficiently long time period (e.g. historical and SSP5-8.5 scenario) remains an efficient basis for describing other future climate change scenarios;
2. The seasonal mean and variance of the coefficients $a_j(t)$ vary linearly with global mean temperature.
3. The statistics of daily fluctuations, given the season, are independent of the year and the climate change scenarios.

The construction of the emulator can be divided into two distinct steps: dimensionality reduction and stochastic modeling of PCA time series. The emulator is then nudged towards the observation data to facilitate machine-learning-based debiasing. We now describe each of these steps in detail.

1.1 Dimensionality Reduction

First we describe how the spatial PCA basis $\phi_j(\mathbf{x})$, which provides the structure for our emulator, is computed. Given a dataset consisting of N years, we extract the climatological mean $\bar{\mathbf{u}}(\mathbf{x}, t)$, defined as phase-average of \mathbf{u} on the same calendar day (e.g. Jan 1st),

$$\bar{\mathbf{u}}(\mathbf{x}, t) = \frac{1}{N} \sum_{n=0}^{N-1} \mathbf{u}(\mathbf{x}, t + nT), \quad 1 \leq t \leq T. \quad (1)$$

, where the period T is one year. When the emulator is trained on the daily maximum data from the MPI model, the climatological mean (1) only quantifies the seasonal cycle. For the three-hourly ERA5 data, $\bar{\mathbf{u}}$ accounts for not only the seasonal variation but also the diurnal cycles. To obtain the scaling of each state variable, we compute

139 the global-and-time-averaged standard deviation,

140

$$141 \quad \sigma_{g,k} = \left[\frac{1}{\mathcal{T}S} \int_0^{\mathcal{T}} \int_S (u_k(\mathbf{x}, t) - \bar{u}_k(\mathbf{x}, t))^2 \cos \theta d\theta d\varphi dt \right]^{1/2}. \quad (2)$$

142

143

144 The notations \mathcal{T} and S are the duration of training window and the Earth's surface,
145 respectively. The data are then centered to have zero climatological mean and scaled
146 by the global standard deviation,
147

148

$$149 \quad u'_k(\mathbf{x}, t) = (u_k(\mathbf{x}, t) - \bar{u}_k(\mathbf{x}, t)) / \sigma_{g,k}. \quad (3)$$

150

151 Now that each component of q'_k has the same order of magnitude, we construct its
152 spatial covariance function,
153

$$154 \quad \mathcal{R}_{jk}(\mathbf{x}, \mathbf{x}^*) = \frac{1}{\mathcal{T}} \int_0^{\mathcal{T}} u'_j(\mathbf{x}, t) u'_k(\mathbf{x}^*, t) dt, \quad j, k = 1, 2, 3, 4. \quad (4)$$

155

156 The PCA modes are acquired by solving the eigenvalue problem,
157

158

$$159 \quad \int_S \sum_k \mathcal{R}_{jk}(\mathbf{x}, \mathbf{x}^*) \phi_k(\mathbf{x}^*) \cos \theta d\theta d\varphi = \lambda \phi_j(\mathbf{x}), \quad j = 1, 2, 3, 4, \quad (5)$$

160

161 This set of equations has multiple solutions $(\lambda^{(i)}, \phi^{(i)})$, $i = 1, 2, 3, \dots$, which are the
162 PCA eigenvalues and mode shapes, respectively. Without loss of generality we rank
163 the eigenpairs such that the eigenvalues, which represent variance, satisfy $\lambda_1 > \lambda_2 >$
164 $\dots > \lambda_I$. The temporal PCA coefficients which govern the time dependence of the
165 spatial PCA modes are found by projecting the normalized fluctuation field onto $\phi^{(i)}$,
166

167

$$168 \quad a_i(t) = \int_S \sum_k u'_k(\mathbf{x}, t) \phi_k^{(i)}(\mathbf{x}) \cos \theta d\theta d\varphi. \quad (6)$$

169

170 The state of the climate can then be expressed as superposition of PCA modes,
171

172

$$173 \quad u_k(\mathbf{x}, t) = \bar{u}_k(\mathbf{x}, t) + \sigma_{g,k} \sum_{i=1}^I a_i(t) \phi_k^{(i)}(\mathbf{x}). \quad (7)$$

174

175

176 When the number of PCA modes I is equal to the number of grid points or the
177 number of snapshots, whichever is smaller, we recover the full field, and any smaller
178 value of I represents a truncation. In this work, we always retain 500 PCA modes,
179 which represent 79.6% of the total variance of 1979-2018 ERA5 data and 78.2% of
180 1950-2100 MPI data. To reiterate, we assume that the climatological mean $\bar{u}_k(\mathbf{x}, t)$,
181 global standard deviation $\sigma_{g,k}$, and PCA mode shapes $\phi_k^{(i)}$ are unchanged with time
182

183
184

or future scenarios. As a result, we focus purely on modeling $a_i(t)$, and the emulated state is written as, 185
186
187

$$\hat{q}_k(\mathbf{x}, t) = \bar{u}_k(\mathbf{x}, t) + \sigma_{g,k} \sum_{i=1}^{500} \hat{a}_i(t) \phi_k^{(i)}(\mathbf{x}). \quad (8)$$

Here the notations with $\hat{\cdot}$ are emulated quantities. 191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207

1.2 Stochastic emulator of PCA time series

1.2.1 Seasonal Decomposition

Our goal is to construct a time series of $\hat{a}_i(t)$ that *statistically* resembles the reference data $a_i(t)$. Although we have removed the climatological mean, the statistics of $a_i(t)$ still exhibit seasonal variation that is important to take into account. Therefore, we divide $a_i(t)$ into four seasons $a_{s,i}(t)$ – of approximately equal length – and model them separately where the additional subscript $s = 1, 2, 3, 4$ represents winter (Dec-Feb), spring (Mar-May), summer (Jun-Aug) and autumn (Sep-Nov). The number of days in each season is 90, 92, 92, and 91 respectively. 208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230

1.2.2 Formulation and Estimation of Model Parameters

We postulate a decomposition of the time series of PCA coefficients, 208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230

$$\hat{a}_{s,i}(t) = \hat{\mu}_{s,i}(T_{s,g}) + \hat{\sigma}_{s,i}(T_{s,g}) \hat{\eta}_{s,i}(t), \quad (9)$$

which is a superposition of the seasonal mean $\hat{\mu}_{s,i}$ and fluctuations parameterized through an envelope of the seasonal variance $\hat{\sigma}_{s,i}^2$. The seasonal mean and variance are assumed to be functions of the global mean temperature $T_{s,g}$, defined as the seasonal average of the daily T_g . The time-dependent daily fluctuations in each season are modelled as autoregressive Gaussian processes $\hat{\eta}_{s,i}(t)$. We will now discuss the formulation and computation of each of these terms in detail. 208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230

Linear Regression of Seasonal Mean and Variance. For each season s and each mode i , in the n th year, we compute the $T_{s,g}$ as well as the seasonal mean $\mu_{s,i}$ and variance $\sigma_{s,i}^2$ of the PCA coefficients of the reference data $a_{s,i}(t)$. Note that for each s, i , and n , the mean $\mu_{s,i}$ and variance $\sigma_{s,i}^2$ are constants – we generally omit explicit notation of the year n to avoid notational clutter. Grouping these values by season s and mode i allows us to perform a linear regression using $\{\mu_{s,i}(n), T_{s,g}(n)\}$ and $\{\sigma_{s,i}^2(n), T_{s,g}(n)\}$ 208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230

$$\begin{aligned} \hat{\mu}_{s,i}(T_{s,g}) &= \hat{p}_{s,i,0} + \hat{p}_{s,i,1} T_{s,g} \\ \hat{\sigma}_{s,i}^2(T_{s,g}) &= \hat{q}_{s,i,0} + \hat{q}_{s,i,1} T_{s,g}, \end{aligned} \quad (10)$$

an assumption which is justified by the linear trends which have been observed in data by a number of sources [1, 2] and illustrated in figure 1. 208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230

231 **Time Lagged Cross-Mode Covariance.** After extracting the linear trends of the
 232 seasonal mean and standard deviation in response to the global mean temperature,
 233 we remove these trends from the true PCA coefficients, resulting in the residuals
 234 $\eta_{s,i} = (a_{s,i} - \hat{a}_{s,i}) / \hat{\sigma}_{s,i}$. To accurately capture the spatio-temporal dynamics, our
 235 model must reflect not only the contemporaneous correlations between different modes,
 236 e.g. $\eta_{s,1}(t)$ and $\eta_{s,2}(t)$, but also their correlations across time. To this end, we define
 237 the *time-lagged cross-mode covariance*,

238

$$239 \quad \Sigma_s(m) = \frac{1}{T_s} \int_{\mathcal{T}_s} \boldsymbol{\eta}_s(t) \boldsymbol{\eta}_s^\top(t + m\Delta t) dt, \quad m = 0, 1, \dots, M, \quad (11)$$

240

241 where $\boldsymbol{\eta}_s = [\eta_{s,1}, \eta_{s,2}, \dots, \eta_{s,M}]^\top$ is the vector of fluctuations of each PCA mode,
 242 $M\Delta t$ is maximum time lag considered, and \mathcal{T}_s represents the set of time indices
 243 corresponding to season s across all training years.

244 Now we want to model the observed fluctuations $\eta_{s,i}(t)$ as a multivariate Gaussian
 245 process $\hat{\eta}_{s,i}(t)$, which has the same covariance matrix $\Sigma_s(m)$ as $\eta_{s,i}(t)$. To further
 246 simplify our notation, the subscript s will be omitted. Mathematically, we seek to
 247 construct an autoregressive model of order M ,

248

$$249 \quad \hat{\boldsymbol{\eta}}(t) = \hat{\boldsymbol{\Psi}}_1 \hat{\boldsymbol{\eta}}(t - \Delta t) + \hat{\boldsymbol{\Psi}}_2 \hat{\boldsymbol{\eta}}(t - 2\Delta t) + \dots + \hat{\boldsymbol{\Psi}}_M \hat{\boldsymbol{\eta}}(t - M\Delta t) + \boldsymbol{\epsilon}(t) \quad (12)$$

250

251 where the noise term is a multivariate Gaussian random vector $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \hat{\mathbf{R}})$. The
 252 unknown matrices $\hat{\boldsymbol{\Psi}}_1, \hat{\boldsymbol{\Psi}}_2, \dots, \hat{\boldsymbol{\Psi}}_M, \hat{\mathbf{R}}$ are solved such that the simulated process (12)
 253 satisfy the given covariance matrices with different time lags $\Sigma(0), \Sigma(1), \dots, \Sigma(M\Delta t)$.
 254 By multiplying both sides of (12) by $\hat{\boldsymbol{\eta}}(t - i\Delta t)$ and averaging in time, we can derive
 255 a set of equations, the so-called Yule-Walker equations,

256

$$257 \quad \begin{bmatrix} \Sigma(0) & \Sigma^\top(1) & \dots & \Sigma^\top(M-1) \\ \Sigma(1) & \Sigma(0) & \dots & \Sigma^\top(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ \Sigma(M-1) & \Sigma(M-2) & \dots & \Sigma(0) \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\Psi}}_1^\top \\ \hat{\boldsymbol{\Psi}}_2^\top \\ \vdots \\ \hat{\boldsymbol{\Psi}}_M^\top \end{bmatrix} = \begin{bmatrix} \Sigma(1) \\ \Sigma(2) \\ \vdots \\ \Sigma(M) \end{bmatrix}. \quad (13)$$

258

259 which may be readily solved for the $\hat{\boldsymbol{\Psi}}_j$ [3]. The corresponding noise covariance is then
 260 given by

261

$$262 \quad \hat{\mathbf{R}} = \Sigma(0) - \sum_{m=1}^M \hat{\boldsymbol{\Psi}}_m \Sigma(m). \quad (14)$$

263

264 After solving equations (13,14), the matrices $\hat{\boldsymbol{\Psi}}_1, \hat{\boldsymbol{\Psi}}_2, \dots, \hat{\boldsymbol{\Psi}}_M, \hat{\mathbf{R}}$ are substituted
 265 into the autoregressive model (12) to simulate the daily fluctuations. The complete
 266 procedures for running the emulator are summarized in Algorithm 1.

267

268 1.3 Nudging the Stochastic Emulator

269 The stochastic emulator introduced previously was designed to capture the second-
 270 order statistics of the leading PCA modes. While this emulator has demonstrated
 271

Algorithm 1 Stochastic emulator of global climate.	277
Input: Temporal evolution of global mean temperature $T_g(t)$	278
Output: Emulated statistics of climate variables	279
Step 1: Emulate seasonal mean and variance;	280
• Compute seasonal global mean temperature $T_{s,g}$ for each season s ;	281
• For each mode i , predict the seasonal mean $\hat{\mu}_{s,i}(T_{s,g})$ and variance $\hat{\sigma}_{s,i}^2(T_{s,g})$.	282
Step 2: Generate stochastic daily fluctuations;	283
• At every time step t , sample a Gaussian random vector $\epsilon \sim \mathcal{N}(\mathbf{0}, \hat{\mathbf{R}})$;	284
• Compute the vector autoregressive process $\hat{\eta}(t)$ according to equation (12).	285
Step 3: Construct time series of spatial fields;	286
• Combine seasonal mean and variance with daily fluctuations to obtain $\hat{a}_i(t)$;	287
• Multiply PCA coefficients by their mode shapes and superpose all the modes;	288
• Denormalize by σ_g and \bar{u} to compute U, V, T, Q in physical space (8).	289
Step 4: Estimate statistics of interest;	290
• Average the spatial fields of U, V, T, Q over window to calculate statistics;	291
• If needed, input $T_g(t)$ from a different ensemble member, repeat steps 1-3 and average over multiple members.	292
	293
	294
	295
	296
	297
	298
	299
	300
	301
	302
	303
	304
	305
	306
	307
	308
	309
	310
	311
	312
	313
	314
	315
	316
	317
	318
	319
	320
	321
	322

effectiveness in representing the conditional Gaussian distribution of certain variables, such as temperature [4], it inherently struggles to reproduce the non-Gaussian characteristics of climate data, including extreme events associated with higher-order PCA modes. A common approach to addressing this limitation involves using machine-learning models to debias the emulator. However, due to the stochastic nature of the emulated spatiotemporal data, instantaneous matches with reference data are not achievable. For instance, an emulated wind speed field on January 1st, 2025, would significantly differ from the corresponding ground truth dataset, whether sourced from ERA5 or CMIP6. Ideally, an infinite ensemble of realizations could be produced by the emulator, enabling selection of instances closest to reference observations for training a debiasing model. However, this method is impractical. A more realistic alternative is the nudging approach [5–8], where the emulator is forced by the deviation from the reference data to produce a time series of fields that approximately maintain the emulator’s statistical characteristics while closely aligning with the observed ground truth. Herein we interpret how to nudge the stochastic emulator. In the following, we detail how to implement nudging within the stochastic emulator framework.

Recall the formulation of the emulator (8,9),

$$\hat{a}_{s,i}(t) = \hat{\mu}_{s,i}(T_{s,g}) + \hat{\sigma}_{s,i}(T_{s,g}) \hat{\eta}_{s,i}(t) \quad (15)$$

$$\hat{q}_k(\mathbf{x}, t) = \bar{u}_k(\mathbf{x}, t) + \sigma_{g,k} \sum_{i=1}^{500} \hat{a}_i(t) \phi_k^{(i)}(\mathbf{x}). \quad (16)$$

The only stochastic component is the time series of daily fluctuations $\hat{\eta}_{s,i}$. All other parts are deterministic and constructed to align with the reference data. Therefore we focus on nudging $\hat{\eta}_{s,i}$, given the true fluctuations $\eta_{s,i}$. Hereafter we omit any subscripts to simplify the notation.

323 The nudged emulator, denoted as $\boldsymbol{\nu}$, is designed to follow the dynamics of the
 324 free-running emulator $\hat{\boldsymbol{\eta}}$, while driven by the deviation from the reference data,
 325

$$326 \quad \dot{\boldsymbol{\nu}} = \dot{\hat{\boldsymbol{\eta}}} - \frac{1}{\tau} (\boldsymbol{\nu} - \boldsymbol{\eta}). \quad (17)$$

327

328 The relaxation time scale τ is a constant that is independent from the season or the
 329 PCA mode. Equation (17) has a closed-form solution,
 330

$$331 \quad \boldsymbol{\nu}(t) = \boldsymbol{\nu}(0)e^{-t/\tau} + \int_0^t e^{-(t-s)/\tau} \left(\dot{\hat{\boldsymbol{\eta}}}(s) + \frac{1}{\tau} \boldsymbol{\eta}(s) \right) ds. \quad (18)$$

332

333 The time derivative term $\dot{\hat{\boldsymbol{\eta}}}$ is approximated using the first-order Euler scheme and
 334 computed from the free-running emulator data. Combining the nudged time series of
 335 daily fluctuations $\boldsymbol{\nu}$ with seasonal mean and variance, we obtain the complete nudged
 336 PCA time series and the spatiotemporal fields,
 337

$$338 \quad \hat{a}_{s,i}^{\nu}(t) = \hat{\mu}_{s,i}(T_{s,g}) + \hat{\sigma}_{s,i}(T_{s,g}) \hat{\nu}_{s,i}(t) \quad (19)$$

339

$$340 \quad \hat{q}_k^{\nu}(\mathbf{x}, t) = \bar{u}_k(\mathbf{x}, t) + \sigma_{g,k} \sum_{i=1}^{500} \hat{a}_i^{\nu}(t) \phi_k^{(i)}(\mathbf{x}). \quad (20)$$

341

342 The interpretation of nudging and the selection of τ have been thoroughly discussed
 343 in [9]. Briefly speaking, the relaxation timescale τ serves to separate the time scales
 344 between slow and fast dynamics. The feedback term in equation (17) drives the slow
 345 dynamics of $\boldsymbol{\nu}$ towards the reference trajectory $\boldsymbol{\eta}$ in the state space, while allowing
 346 the fast dynamics of $\boldsymbol{\nu}$ to freely evolve. Thus, when pairs of the nudged and reference
 347 data are used for training a machine-learning model, we are essentially learning a map
 348 that corrects the fast features of the imperfect emulator and improve the performance
 349 on extreme events. In our case, the relaxation timescale is set as $\tau = 6$ hrs, consistent
 350 with previous work [10]. Minor adjustments of τ , such as to 3 or 12 hours, do not
 351 significantly alter the results.
 352

353 The feedback term in (17), although driving the nudged emulator towards the
 354 reference, introduces artificial dissipation not present in the free-running emulator.
 355 Such an effect leads to a distribution of $\boldsymbol{\nu}$ and $\hat{\boldsymbol{\eta}}$ that is slightly different from the
 356 free-running emulator. In order for a neural network trained on the nudged dataset to
 357 generalize to unseen free-running emulator data, this discrepancy must be remedied.
 358 To this end, we rescale the nudged solution $\hat{\boldsymbol{\eta}}$ in each season so that its mean and
 359 variance match those of the free-running emulator $\hat{\boldsymbol{\eta}}$ at each grid point.
 360

361 2 Machine Learned Debiasing

362

363 2.1 Conditional Score-based Diffusion model

364 Here we describe the training strategy and network architecture used in the ML
 365 correction step of our model. Our model relies on the framework introduced by
 366

Barthel Sorensen et al. [10], which aims to learn a deterministic map from the nudged trajectory to the reference trajectory, 369
370

$$\mathbf{u} = \mathcal{F}(\hat{\mathbf{q}}^\nu). \quad (21) \quad 371$$

In practice, such a mapping is not necessarily deterministic. There could exist multiple reference state \mathbf{u} that are close to the same nudged state $\hat{\mathbf{q}}^\nu$. Therefore, we generalize this framework by learning a conditional probability distribution function, 374
375
376

$$p(\mathbf{u} | \hat{\mathbf{q}}^\nu). \quad (22) \quad 377$$

If the mapping is actually deterministic, the conditional probability distribution will collapse to a Dirac delta function $\delta(\mathbf{u} - \mathcal{F}(\hat{\mathbf{q}}^\nu))$. Once the conditional PDF (22) is learned, we can provide the free-running emulation $\hat{\mathbf{q}}$ as the conditional information to generate debiased estimations of the state variables $\hat{\mathbf{u}}$, 380
381
382
383

$$\hat{\mathbf{u}}(\mathbf{x}, t) \sim \mathcal{G}_{\theta, 2}[\hat{\mathbf{q}}(\mathbf{x}, t)] \quad (23) \quad 384$$

Although learning and sampling high-dimensional PDFs were long considered intractable, these tasks have recently become practical thanks to advances in deep generative models. In this study, we adopt conditional score-based diffusion model [11, 12] that has been demonstrated effective for geophysical datasets [13]. Other frameworks, such as flow matching [14] and stochastic interpolant [15], could likewise address the debiasing problem considered here. The choice of the generative model is beyond the scope of this work and will be investigated in the future. 387
388
389
390
391
392
393

Our implementation of score-based diffusion model follows that of Bischoff and Deck [13]. To simplify the notation, we will use \mathbf{q} to represent the nudged emulation $\hat{\mathbf{q}}^\nu$. The diffusion model consists of a forward diffusion process, which maps the data distribution to normal distribution, and a reverse denoising process that transforms Gaussian noise to a sample or image of the climate state. Specifically, given an initial condition $\mathbf{u}(t = 0) \sim p_{\text{data}}(\mathbf{u} | \mathbf{q})$ drawn from the training data, the forward diffusion process is defined by the stochastic differential equation (SDE), 394
395
396
397
398
399
400
401
402
403

$$d\mathbf{u} = g(t)d\mathbf{W}, \quad (24) \quad 404$$

where the diffusion coefficient $g(t)$ is a non-negative prescribed function and \mathbf{W} is a Wiener process. Note that the diffusion time t is within $[0, 1]$ and should be distinguished from the physical time t . At any time t , the solution to the SDE (24) is a “noised” image $\mathbf{u}(t)$, which follows a normal distribution conditioned on $\mathbf{u}(0)$, 404
405
406
407
408
409
410
411
412
413
414

$$\mathbf{u}(t) \sim \mathcal{N}(\mathbf{u}(0), \sigma^2(t)) = p(\mathbf{u}(t) | \mathbf{u}(0)), \quad (25) \quad 415$$

where the variance $\sigma^2(t)$ depends on $g(t)$, 416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
7710
7711
7712
7713
7714
7715
7716
7717
7718
7719
7720
7721
7722
7723
7724
7725
7726
7727
7728
7729
7730
7731
7732
7733
7734
7735
7736
7737
7738
7739
7740
7741
7742
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769
7770
7771
7772
7773
7774
7775
7776
7777
7778
7779
77710
77711
77712
77713
77714
77715
77716
77717
77718
77719
77720
77721
77722
77723
77724
77725
77726
77727
77728
77729
77730
77731
77732
77733
77734
77735
77736
77737
77738
77739
77740
77741
77742
77743
77744
77745
77746
77747
77748
77749
77750
77751
77752
77753
77754
77755
77756
77757
77758
77759
77760
77761
77762
77763
77764
77765
77766
77767
77768
77769
77770
77771
77772
77773
77774
77775
77776
77777
77778
77779
77780
77781
77782
77783
77784
77785
77786
77787
77788
77789
77790
77791
77792
77793
77794
77795
77796
77797
77798
77799
777100
777101
777102
777103
777104
777105
777106
777107
777108
777109
777110
777111
777112
777113
777114
777115
777116
777117
777118
777119
777120
777121
777122
777123
777124
777125
777126
777127
777128
777129
777130
777131
777132
777133
777134
777135
777136
777137
777138
777139
777140
777141
777142
777143
777144
777145
777146
777147
777148
777149
777150
777151
777152
777153
777154
777155
777156
777157
777158
777159
777160
777161
777162
777163
777164
777165
777166
777167
777168
777169
777170
777171
777172
777173
777174
777175
777176
777177
777178
777179
777180
777181
777182
777183
777184
777185
777186
777187
777188
777189
777190
777191
777192
777193
777194
777195
777196
777197
777198
777199
777200
777201
777202
777203
777204
777205
777206
777207
777208
777209
777210
777211
777212
777213
777214
777215
777216
777217
777218
777219
777220
777221
777222
777223
777224
777225
777226
777227
777228
777229
777230
777231
777232
777233
777234
777235
777236
777237
777238
777239
777240
777241
777242
777243
777244
777245
777246
777247
777248
777249
777250
777251
777252
777253
777254
777255
777256
777257
777258
777259
777260
777261
777262
777263
777264
777265
777266
777267
777268
777269
777270
777271
777272
777273
777274
777275
777276
777277
777278
777279
777280
777281
777282
777283
777284
777285
777286
777287
777288
777289
777290
777291
777292
777293
777294
777295
777296
777297
777298
777299
777300
777301
777302
777303
777304
777305
777306
777307
777308
777309
777310
777311
777312
777313
777314
777315
777316
777317
777318
777319
777320
777321
777322
777323
777324
777325
777326
777327
777328
777329
777330
777331
777332
777333
777334
777335
777336
777337
777338
777339
777340
777341
777342
777343
777344
777345
777346
777347
777348
777349
777350
777351
777352
777353
777354
777355
777356
777357
777358
777359
777360
777361
777362
777363
777364
777365
777366
777367
777368
777369
777370
777371
777372
777373
777374
777375
777376
777377
777378
777379
777380
777381
777382
777383
777384
777385
777386
777387
777388
777389
777390
777391
777392
777393
777394
777395
777396
777397
777398
777399
777400
777401
777402
777403
777404
777405
777406
777407
777408
777409
777410
777411
777412
777413
777414
777415
777416
777417
777418
777419
777420
777421
777422
777423
777424
777425
777426
777427
777428
777429
777430
777431
777432
777433
777434
777435
777436
777437
777438
777439
777440
777441
777442
777443
777444
777445
777446
777447
777448
777449
777450
777451
777452
777453
777454
777455
777456
777457
777458
777459
777460
777461
777462
777463
777464
777465
777466
777467
777468
777469
777470
777471
777472
777473
777474
777475
777476
777477
777478
777479
777480
777481
777482
777483
777484
777485
777486
777487
777488
777489
777490
777491
777492
777493
777494
777495
777496
777497
777498
777499
777500
777501
777502
777503
777504
777505
777506
777507
777508
777509
777510
777511
777512
777513
777514
777515
777516
777517
777518
777519
777520
777521
777522
777523
777524
777525
777526
777527
777528
777529
777530
777531
777532
777533
777534
777535
777536
777537
777538
777539
777540
777541
777542
777543
777544
777545
777546
777547
777548
777549
777550
777551
777552
777553
777554
777555
777556
777557
777558
777559
777560
777561
777562
777563
777564
777565
777566
777567
777568
777569
777570
777571
777572
777573
777574
777575
777576
777577
777578
777579
777580
777581
777582
777583
777584
777585
777586
777587
777588
777589
777590
777591
777592
777593
777594
777595
777596
777597
777598
777599
777600
777601
777602
777603
777604
777605
777606
777607
777608
777609
777610
777611
777612
777613
777614
777615
777616
777617
777618
777619
777620
777621
777622
777623
777624
777625
777626
777627
777628
777629
777630
777631
777632
777633
777634
777635
777636
777637
777638
777639
777640
777641
777642
777643
777644
777645
777646
777647
777648
777649
777650
777651
777652
777653
777654
777655
777656
777657
777658
777659
777660
777661
777662
777663
777664
777665
777666
777667
777668
777669
777670
777671
777672
777673
777674
777675
777676
777677
777678
777679
777680
777681
777682
777683
777684
777685
777686
777687
777688
777689
777690
777691
777692
777693
777694
777695
777696
777697
777698
777699
777700
777701
777702
777703
777704
777705
777706
777707
777708
777709
777710
777711
777712
777713
777714
777715
777716
777717
777718
777719
777720
777721
777722
777723
777724
777725
777726
777727
777728
777729
777730
777731
777732
777733
777734
777735
777736
777737
777738
777739
777740
777741
777742
777743
777744
777745
777746
777747
777748
777749
777750
777751
777752
777753
777754
777755
777756
777757
777758
777759
777760
777761
777762
777763
777764
777765
777766
777767
777768
777769
777770
777771
777772
777773
777774
777775
777776
777777
777778
777779
777780
777781
777782
777783
777784
777785
777786
777787
777788
777789
777790
777791
777792
777793
777794
777795
777796
777797
777798
777799
7777100
7777101
7777102
7777103
7777104
7777105
7777106
7777107
7777108
7777109
7777110
7777111
7777112
7777113
7777114
7777115
7777116
7777117
7777118
7777119
7777120
7777121
7777122
7777123
7777124
7777125
7777126
7777127
7777128
7777129
7777130
7777131
7777132
7777133
7777134
7777135
7777136
7777137
7777138
7777139
7777140
7777141
7777142
7777143
7777144
7777145
7777146
7777147
7777148
7777149
7777150
7777151
7777152
7777153
7777154
7777155
7777156
7777157
7777158
7777159
7777160
7777161
7777162
7777163
7777164
7777165
7777166
7777167
7777168
7777169
7777170
7777171
7777172
7777173
7777174
7777175
7777176
7777177
7777178
7777179
7777180
7777181
7777182
7777183
7777184
7777185
7777186
7777187
7777188
7777189
7777190
7777191
7777192
7777193
7777194
7777195
7777196
7777197
7777198
7777199
7777200
7777201
7777202
7777203
7777204
7777205
7777206
7777207
7777208
7777209
7777210
7777211
7777212
7777213
7777214
7777215
7777216
7777217
7777218
7777219
7777220
7777221
7777222
7777223
7777224
7777225
7777226
7777227
7777228
7777229
7777230
7777231
7777232
7777233
7777234
7777235
7777236
7777237
7777238
77772

415 The marginal distribution of $\mathbf{u}(t)$ after integrating out $\mathbf{u}(0)$ is defined as $p_t(\mathbf{u}(t)|\mathbf{q})$,
 416 which is generally non-Gaussian. The diffusion coefficient $g(t)$ is chosen such that at
 417 $t = 1$, the variance $\sigma^2(t = 1)$ has much larger magnitude than the original $\mathbf{u}(t = 0)$.
 418 Therefore, $\mathbf{u}(t = 0)$ has lost all memory of initial condition, and

419

$$420 \quad p(\mathbf{u}(1)|\mathbf{u}(0)) = p(\mathbf{u}(1)) = \mathcal{N}(\mathbf{0}, \sigma^2(1)). \quad (27)$$

421

422 According to Anderson's theorem [16], the reverse of equation (24) is also a diffusion
 423 process, running backward in time and governed by the following SDE,

424

$$425 \quad d\mathbf{u} = -g^2(t)\mathbf{s}(\mathbf{u}, \mathbf{q}, t)dt + g(t)d\bar{\mathbf{W}}, \quad (28)$$

426

427 where $\bar{\mathbf{W}}$ is a reverse-time Wiener process, and $\mathbf{s}(\mathbf{u}, t)$ is the conditional score,

428

$$429 \quad \mathbf{s}(\mathbf{u}, \mathbf{q}, t) = \nabla_{\mathbf{u}} \log p_t(\mathbf{u}(t)|\mathbf{q}). \quad (29)$$

430

431 If we have access to the score for all t , we can derive the reverse diffusion process,
 432 simulate it from $t = 1$ to $t = 0$, and generate samples that follow the data distribu-
 433 tion. To this end, we approximate the score by a neural network, $\mathbf{s}_\theta(\mathbf{u}, \mathbf{q}, t)$, which is
 434 obtained by minimizing the score-matching loss or Fisher's divergence,

435

$$436 \quad \mathcal{L}_{SM}(\theta) := \frac{1}{2} \mathbb{E}_{\substack{t \sim U(0,1) \\ \mathbf{u}(t), \mathbf{q} \sim p_t(\mathbf{u}(t)|\mathbf{q})}} \left[\sigma^2(t) \|\nabla_{\mathbf{u}} \log p_t(\mathbf{u}|\mathbf{q}) - \mathbf{s}_\theta(\mathbf{u}, \mathbf{q}, t)\|_2^2 \right], \quad (30)$$

438

439 where $U(0, 1)$ stands for a uniform distribution from 0 to 1. However, the loss function
 440 (30) cannot be directly optimized, since the true conditional score $\nabla_{\mathbf{u}} \log p_t(\mathbf{u}|\mathbf{q})$ is
 441 unknown. Taking advantage of the Gaussian property of the forward diffusion process
 442 (25,27), [11] showed that \mathcal{L}_{SM} is equal to the following loss up to an additive term,

443

$$444 \quad \mathcal{L}(\theta) := \frac{1}{2} \mathbb{E}_{\substack{t \sim U(0,1) \\ \mathbf{u}(0), \mathbf{q} \sim p_t(\mathbf{u}(0)|\mathbf{q}) \\ \mathbf{u}(t) \sim p(\mathbf{u}(t)|\mathbf{u}(0))}} \left[\sigma^2(t) \|\nabla_{\mathbf{u}} \log p_t(\mathbf{u}(t)|\mathbf{u}(0)) - \mathbf{s}_\theta(\mathbf{u}, \mathbf{q}, t)\|_2^2 \right]. \quad (31)$$

446

447 This expression only involves $\nabla_{\mathbf{u}} \log p_t(\mathbf{u}(t)|\mathbf{u}(0))$ which can be computed analytically
 448 from the forward diffusion process.

449 Once the conditional score is learned through training, it can be substituted into
 450 the backward SDE (equation 28) to generate a debiased sample. The backward SDE
 451 is simulated using Euler-Maruyama scheme. The complete procedures of training and
 452 sampling are summarized in Algorithm 2.

453

2.2 Network Architecture and Training Parameters

454 Before feeding the emulation and reference data into the diffusion model, we remove the
 455 true climatological mean and scale each variable (U, V, T, Q) by twice its own globally-
 456 averaged standard deviation. In other words, we focus on correcting the fluctuation
 457 fields provided by the emulator, and each variable is scaled to the same order of
 458

Algorithm 2 Conditional score-based diffusion model.	461
Training	462
Input: Reference data and nudged emulation $\{\mathbf{u}_i, \mathbf{q}_i\} \sim p(\mathbf{u} \mathbf{q})$	463
repeat	464
$\mathbf{u}(0), \mathbf{q} \sim p(\mathbf{u} \mathbf{q})$	465
$t \sim U(0, 1)$	466
$\mathbf{u}(t) \sim p(\mathbf{u}(t) \mathbf{u}(0))$	467
Take gradient descent step on $\nabla_{\theta} \left[\sigma^2(t) \ \nabla_{\mathbf{u}} \log p_t(\mathbf{u}(t) \mathbf{u}(0)) - \mathbf{s}_{\theta}(\mathbf{u}, \mathbf{q}, t)\ _2^2 \right]$	468
until converged	469
Output: Trained neural network $\mathbf{s}_{\theta}(\mathbf{u}, \mathbf{q}, t)$.	470
End	471
Sampling	472
Input: Snapshot of emulated state \mathbf{q}	473
$\mathbf{u}(1) \sim \mathcal{N}(\mathbf{0}, \sigma^2(1))$	474
for $t = 1$ to 0 do	475
Evaluate $\mathbf{s}_{\theta}(\mathbf{u}(t), \mathbf{q}, t)$	476
$\mathbf{u}(t - \Delta t) = \mathbf{u}(t) + g^2(t) \mathbf{s}_{\theta}(\mathbf{u}, \mathbf{q}, t) \Delta t - g(t) (\bar{\mathbf{W}}(t) - \bar{\mathbf{W}}(t - \Delta t))$	477
$t \leftarrow t - \Delta t$	478
end for	479
Output: Debiased snapshot $\hat{\mathbf{u}} = \mathbf{u}(t = 0)$	480
End	481
	482
	483
	484
	485
	486
	487
	488
	489
	490
	491
	492
	493
	494
	495
	496
	497
	498
	499
	500
	501
	502
	503
	504
	505
	506

magnitude. Regarding the diffusion coefficient $g(t)$ in equation (24), we adopt the “variance-exploding” schedule,

$$g(t) = \sigma_{\min} \left(\frac{\sigma_{\max}}{\sigma_{\min}} \right)^t \sqrt{2 \log \left(\frac{\sigma_{\max}}{\sigma_{\min}} \right)}, \quad (32)$$

where $\sigma_{\min} = 0.01$ and σ_{\max} is chosen as the maximum 2-norm distance between any two snapshots of \mathbf{u} .

The neural network architecture we adopted is a U-Net [13, 17], denoted as

$$\mathbf{s}_{\theta}(\mathbf{u}, \mathbf{q}, t) = \mathcal{U}(\mathbf{X}, t; \theta). \quad (33)$$

The first input X is a tensor of size (N_1, N_2, C_{in}) , where (N_1, N_2) are longitude and latitude dimensions, and C_{in} is the number of channels. In our case, $C_{in} = 8$, including (U, V, T, Q) from the nudged emulation and reference data, respectively. The output of our U-Net is another tensor of size (N_1, N_2, C_{out}) . The number of output channels is $C_{out} = 4$. The U-Net architecture consists of

1. A lifting layer which increases the number of channels from C_{in} to 32;
2. Three downsampling convolutional layers, each of which reduces the spatial dimension and increase the number of channels by a factor of 2;
3. Eight residual blocks [18] to promote continuity in the latent space;

507 4. Three nearest neighbor up sampling layer and convolution layers which mirror the
 508 downsampling operations;
 509 5. A Final projection layer that decreases the number of channels to C_{out} .

510

511 3 Data Post-processing and Evaluation Metrics

512

513 This section provides detailed definitions and calculation methods for all statistics
 514 and metrics presented in the main figures. Given that the climatological mean $\bar{\mathbf{u}}(\mathbf{x}, t)$
 515 (equation 1) is assumed known, our analysis focuses on evaluating the statistics of the
 516 fluctuation fields, $\mathbf{u} - \bar{\mathbf{u}}$. This approach enables a clearer comparison between reference
 517 statistics and GEN² prediction. In the following subsections, the fluctuations from the
 518 climatological mean, $\mathbf{u} - \bar{\mathbf{u}}$, will be simply written as \mathbf{u} for notational convenience.

519

520 3.1 Single-point and two-point statistics

521

522 Without loss of generality, we use the zonal wind speed $U(\mathbf{x}, t)$ as an example. To
 523 evaluate the statistics of U at location \mathbf{x} , we perform a time average (e.g. from 1979
 524 to 2018 for ERA5 dataset). If we have N_t time steps available, the mean and standard
 525 deviation are computed as,

526

$$527 \quad \mu_U(\mathbf{x}, t_j) = \frac{1}{N_t} \sum_{j=1}^{N_t} U(\mathbf{x}, t_j) \quad \text{and} \quad \sigma_U(\mathbf{x}) = \sqrt{\frac{1}{N_t - 1} \sum_{j=1}^{N_t} (U(\mathbf{x}, t_j) - \mu_U(\mathbf{x}, t_j))^2}. \quad (34)$$

530

531 To obtain the unbiased skewness, we first compute

532

$$533 \quad s_U(\mathbf{x}) = \frac{\frac{1}{N_t} \sum_{j=1}^{N_t} (U(\mathbf{x}, t_j) - \mu_U(\mathbf{x}, t_j))^3}{\left(\frac{1}{N_t} \sum_{j=1}^{N_t} (U(\mathbf{x}, t_j) - \mu_U(\mathbf{x}, t_j))^2\right)^{3/2}}, \quad (35)$$

534

535 which is then substituted into

536

$$537 \quad s_U^0(\mathbf{x}) = \frac{\sqrt{N_t(N_t - 1)}}{N_t - 2} s_U(\mathbf{x}). \quad (36)$$

538

539 The calculation of unbiased kurtosis also consists of two steps:

540

$$541 \quad k_U(\mathbf{x}) = \frac{\frac{1}{N_t} \sum_{j=1}^{N_t} (U(\mathbf{x}, t_j) - \mu_U(\mathbf{x}, t_j))^4}{\left(\frac{1}{N_t} \sum_{j=1}^{N_t} (U(\mathbf{x}, t_j) - \mu_U(\mathbf{x}, t_j))^2\right)^2}, \quad (37)$$

542

$$543 \quad k_U^0(\mathbf{x}) = \frac{N_t - 1}{(N_t - 2)(N_t - 3)} ((N_t + 1)k_U(\mathbf{x}) - 3(N_t - 1) + 3). \quad (38)$$

544

545

546

547

548

549

550

551

552

At the same location \mathbf{x} , the correlation coefficient between two variables U and V are defined as, 553
554
555

$$\rho(U, V) = \frac{\text{cov}(U(\mathbf{x}, t), V(\mathbf{x}, t))}{\text{cov}(U(\mathbf{x}, t), U(\mathbf{x}, t)) \text{cov}(V(\mathbf{x}, t), V(\mathbf{x}, t))}, \quad (39)$$

where $\text{cov}(U(\mathbf{x}, t), V(\mathbf{x}, t))$ is the time-averaged covariance between U and V . 556
557
558

The two-point correlation coefficient of U is defined as, 559
560
561

$$\rho(U(\mathbf{x}_0), U(\mathbf{x})) = \frac{\text{cov}(U(\mathbf{x}_0, t), U(\mathbf{x}, t))}{\text{cov}(U(\mathbf{x}_0, t), U(\mathbf{x}_0, t)) \text{cov}(U(\mathbf{x}, t), U(\mathbf{x}, t))}. \quad (40)$$

The anchor point \mathbf{x}_0 is selected as major cities (e.g. Boston, Hong Kong) in figure 3, 564
565
566
567
568
569
570
571
572

The global root-mean-square error (RMSE) of an arbitrary statistic \mathcal{Q} (e.g. in figure 5 and table 3,4) is defined as, 573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598

$$\text{RMSE}(\mathcal{Q}) = \left[\frac{1}{S} \int_S \left(\hat{\mathcal{Q}}(\theta, \varphi, t) - \mathcal{Q}(\theta, \varphi, t) \right)^2 \cos \theta d\theta d\varphi \right]^{1/2}, \quad (41)$$

where $\hat{\mathcal{Q}}$ is the GEN² prediction and \mathcal{Q} is the reference statistics. 573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598

3.2 Wheeler-Kiladis spectrum

The Wheeler-Kiladis spectrum is computed following the procedure described in Wheeler and Kiladis [19], Kiladis et al. [20]. Given data as a function of longitude, latitude, and time $[(\theta, \phi, t)]$ the latitude range is first truncated to $\phi \in [-15^\circ, 15^\circ]$. Then for each latitude ϕ_j , and time t_j the Fourier spectrum is computed in the azimuthal direction θ giving rise to a azimuthal wavenumber m . Then for each ϕ_j and m_j the time series of data is split into a series of overlapping segments. Following [19, 20] we set the length of each segment to 96 days and the overlap to 65 days. Each segment is then detrended using a linear fit and Fourier transformed in time - giving a temporal frequency f . After averaging over all latitudes and all temporal segments, the raw Wheeler-Kiladis spectra are shown in figure 2. 573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598

Due to the strong “redness” of the spectra, detailed features corresponding to the equatorial waves are obscured. To better identify the ridges of the spectra, we first apply a 1-2-1 filter ten times to obtain a much smoother “background” spectra. Then the raw spectra in figure 2 are divided by the background [19]. The results, as shown in figure 2(c) of the main manuscript, more clearly show the spectral peaks that correspond to different types of equatorial waves. Note that the spectra in our results should not be directly compared against the plots in [19]. The reason is that their analysis was based on the long-wave radiation data, which are proxy for cloudiness, whereas our analysis focuses on near-surface zonal wind speed. 573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598

599 **Supplementary Notes**

600

601 To illustrate the debiasing capabilities of the ML correction step, we show in figure 3
602 the bias in the 97.5% quantile, predicted with or without ML correction. As explained
603 at the beginning of Appendix 3, the statistics are evaluated for the fluctuation fields.
604 Before applying ML correction (middle column), the error of the conditional Gaussian
605 emulator is already moderately accurate. For example, the error of T at most locations
606 is within $3K$, and the highest error is within $3K$. The ML model (right column)
607 consistently reduces the error of all the state variables at almost all the locations. A
608 more quantitative comparison is provided in table 1,2. The bias in standard deviation,
609 quantile, skewness, and kurtosis are all significantly reduced by ML correction.

610 The bias reduction in two-point correlations are shown in figure 4. We select Lagos
611 and Tehran for visualization, because the bias of the conditional Gaussian emulator
612 is more pronounced at these two locations. Top panels in figure 4 are the bias of the
613 conditional Gaussian emulator, without ML correction. Bottom panels are the bias of
614 GEN² prediction.

615 Table 3 and 4 summarize the error of two-point correlations at more locations,
616 selected from different regions and climate over the world. At most locations, the
617 conditional Gaussian emulator already achieves an accurate prediction, with RMSE
618 lower than 0.03. After applying ML correction, the errors are significantly reduced
619 at all the locations considered. These results demonstrate the robustness of the ML
620 correction.

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

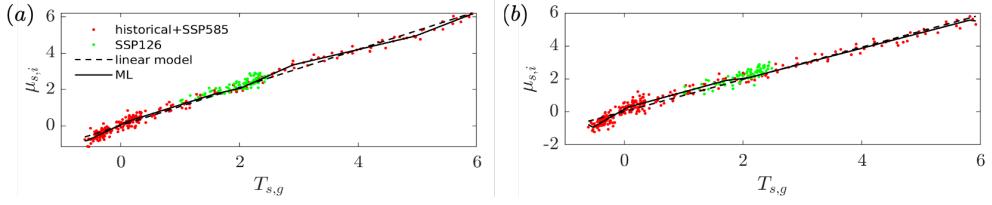
641

642

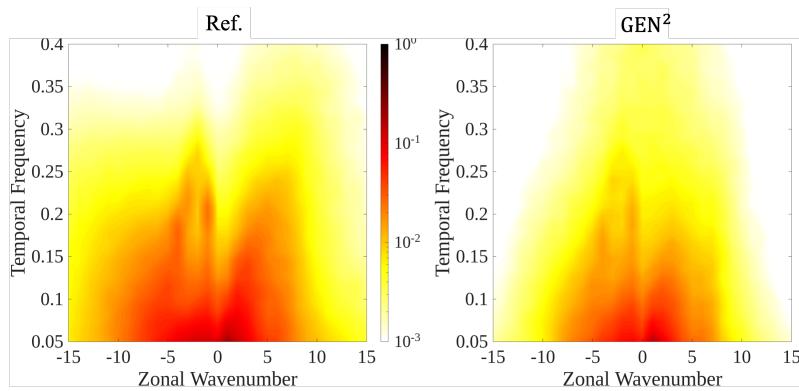
643

644

Supplementary Figures



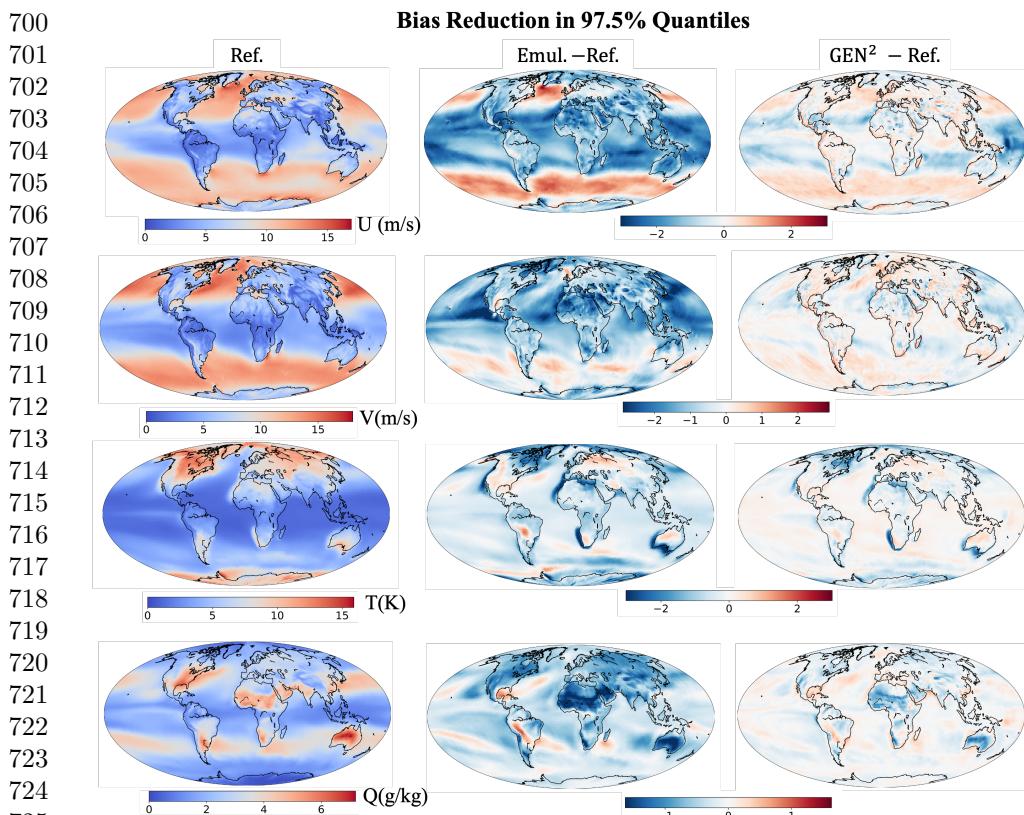
Supplementary Fig 1 Linearity of PCA coefficients as functions of the global mean temperature. Jun-Aug mean of (a) the first and (b) second PCA coefficients in each year of MPI dataset, from 1950 to 2100, plotted versus the global mean temperature. Red dots: true seasonal mean obtained from the historical and SSP5-8.5 scenario. Green dots: SSP1-2.6 scenario. Black dashed line: linear regression; Solid line: machine-learned function.



Supplementary Fig 2 Raw Wheeler-Kiladis spectrum of zonal wind. Spectra are computed using 1979-2018 ERA5 reference data and GEN² prediction. These spectra are normalized by the “background power” to obtain the spectra in figure 2(c) of the main manuscript.

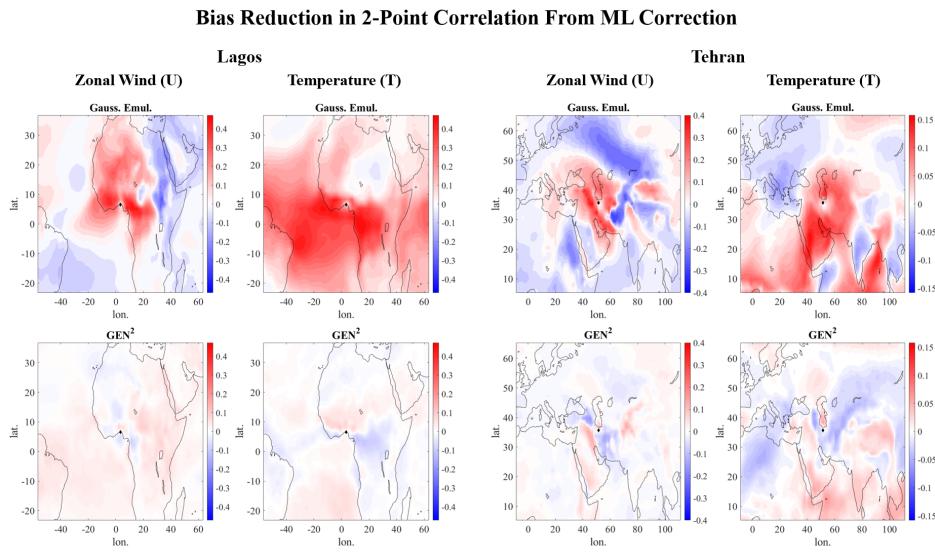
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690

691
692
693
694
695
696
697
698
699



726 **Supplementary Fig 3 Machine-learning correction of the bias in quantiles.** Left column:
727 97.5% quantile of zonal wind, meridional wind, temperature, computed from reference data. Middle
728 column: bias of conditional Gaussian emulation. Right column: bias of “GEN²” approach.
729

730
731
732
733
734
735
736



Supplementary Fig 4 Machine-learning correction of the bias in two-point correlations.
 Bias in two-point correlations of zonal wind and temperature, centered at Lagos and Tehran. Contour plots represent bias relative to reference ERA5 data. Panels labeled “Gauss. Emul.” correspond to predictions of the conditional Gaussian emulator *only* (no ML correction) and “GEN²” represents the full model prediction (with ML correction) .

737
 738
 739
 740
 741
 742
 743
 744
 745
 746
 747
 748
 749
 750
 751
 752
 753
 754
 755
 756
 757
 758
 759
 760
 761
 762
 763
 764
 765
 766
 767
 768
 769
 770
 771
 772
 773
 774
 775
 776
 777
 778
 779
 780
 781
 782

783 **Supplementary Tables**

784

785

786

Statistics	RMSE of U stats			RMSE of V stats		
	Em.	GEN ²	Change	Em.	GEN ²	Change
Std	0.43	0.15	-65%	0.47	0.13	-73%
97.5% quantile	1.14	0.46	-60%	0.99	0.33	-66%
Skewness	0.41	0.19	-53%	0.27	0.14	-48%
Kurtosis	0.84	0.57	-33%	0.68	0.46	-33%

787 **Supplementary Table 1** RMSE of single-point statistics of U ,
788 V . RMSE is defined as equation (41). Columns labeled as “Em.” are the
789 RMSE of the prediction of conditional Gaussian emulator, and “GEN²”
790 is the full-model prediction. “Change” columns are the relative error
791 change from conditional Gassuain emulator to GEN², more precisely,
792 $(\text{RMSE}(\text{GEN}^2) - \text{RMSE}(\text{Em.})) / \text{RMSE}(\text{Em.})$.

793

794

795

796

797

798

799

800

Statistics	RMSE of T stats			RMSE of Q stats		
	Em.	GEN ²	Change	Em.	GEN ²	Change
Std	0.23	0.10	-56%	0.19	0.05	-72%
97.5% quantile	0.67	0.35	-48%	0.46	0.17	-63%
Skewness	0.34	0.20	-42%	0.50	0.27	-47%
Kurtosis	0.89	0.68	-24%	1.78	1.38	-23%

801 **Supplementary Table 2** Same as table 1, but for temperature T and
802 Q .

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

City	Anchor point x_0		RMSE of $\rho(U(x_0), U(x))$			RMSE of $\rho(V(x_0), V(x))$		
	Em.	Lat(N)	GEN ²	Change	Em.	GEN ²	Change	
Boston	-71.1	42.4	0.016	0.009	-43%	0.018	0.008	-53%
Los Angeles	-118.2	34.1	0.028	0.011	-62%	0.026	0.009	-67%
Chicago	-87.6	41.9	0.020	0.008	-59%	0.017	0.008	-54%
Houston	-95.4	29.8	0.020	0.009	-57%	0.017	0.009	-48%
Kansas City	-94.6	39.1	0.022	0.008	-64%	0.017	0.008	-51%
London	-0.1	51.5	0.018	0.009	-50%	0.019	0.008	-60%
Anchorage	-149.9	61.2	0.019	0.012	-36%	0.021	0.009	-58%
Paris	2.4	48.9	0.019	0.009	-53%	0.019	0.008	-57%
Athens	23.7	38.0	0.024	0.010	-57%	0.022	0.009	-60%
Moscow	37.6	55.8	0.024	0.010	-58%	0.022	0.008	-66%
Stockholm	18.1	59.3	0.022	0.010	-57%	0.020	0.008	-61%
Tokyo	139.7	35.7	0.017	0.008	-51%	0.017	0.009	-48%
Hong Kong	114.2	22.3	0.026	0.009	-64%	0.025	0.009	-65%
New Delhi	77.1	28.6	0.028	0.010	-64%	0.028	0.010	-65%
Tehran	51.4	35.7	0.030	0.011	-63%	0.034	0.010	-70%
Astana	71.5	51.2	0.022	0.009	-60%	0.022	0.009	-60%
Cairo	31.2	30.0	0.029	0.009	-70%	0.027	0.008	-69%
Cape Town	18.4	-33.9	0.018	0.009	-50%	0.022	0.009	-60%
Lagos	3.4	6.5	0.043	0.015	-65%	0.040	0.009	-78%
Kisangani	25.2	0.1	0.047	0.015	-69%	0.036	0.009	-76%
Mombasa	39.7	-4.0	0.041	0.018	-55%	0.038	0.010	-74%
Sydney	151.2	-33.9	0.020	0.009	-56%	0.020	0.008	-58%
Brasília	-47.9	-15.8	0.029	0.012	-58%	0.044	0.010	-78%
Bogota	-74.1	4.7	0.054	0.023	-57%	0.034	0.017	-49%
Buenos Aires	-58.4	-34.6	0.020	0.009	-55%	0.019	0.009	-54%

Supplementary Table 3 RMSE of two-point correlation of U , V . Columns labeled as “Em.” are the RMSE of the prediction of conditional Gaussian emulator, and “GEN²” is the full-model prediction. “Change” columns are the relative error change from conditional Gaussian emulator to GEN², more precisely, $(\text{RMSE}(\text{GEN}^2) - \text{RMSE}(\text{Em.})) / \text{RMSE}(\text{Em.})$.

875

876

877

878

879

880

881

882

883

884

885

886	Anchor point \mathbf{x}_0			RMSE of $\rho(T(\mathbf{x}_0), T(\mathbf{x}))$			RMSE of $\rho(Q(\mathbf{x}_0), Q(\mathbf{x}))$		
	City	Lon(E)	Lat(N)	Em.	GEN ²	Change	Em.	GEN ²	Change
887	Boston	-71.1	42.4	0.017	0.012	-30%	0.016	0.009	-43%
888	Los Angeles	-118.2	34.1	0.021	0.011	-48%	0.022	0.010	-54%
889	Chicago	-87.6	41.9	0.014	0.010	-29%	0.017	0.009	-46%
890	Houston	-95.4	29.8	0.019	0.014	-28%	0.015	0.008	-47%
891	Kansas City	-94.6	39.1	0.014	0.010	-30%	0.016	0.009	-45%
892	London	-0.1	51.5	0.023	0.010	-55%	0.022	0.011	-49%
893	Anchorage	-149.9	61.2	0.026	0.013	-48%	0.026	0.014	-48%
894	Paris	2.4	48.9	0.023	0.010	-54%	0.023	0.012	-47%
895	Athens	23.7	38.0	0.018	0.012	-35%	0.023	0.009	-60%
896	Moscow	37.6	55.8	0.017	0.011	-36%	0.021	0.010	-51%
897	Stockholm	18.1	59.3	0.021	0.010	-53%	0.021	0.011	-49%
898	Tokyo	139.7	35.7	0.019	0.011	-40%	0.015	0.009	-39%
899	Hong Kong	114.2	22.3	0.023	0.014	-41%	0.020	0.011	-46%
900	New Delhi	77.1	28.6	0.031	0.018	-43%	0.022	0.011	-47%
901	Tehran	51.4	35.7	0.030	0.018	-42%	0.033	0.010	-68%
902	Astana	71.5	51.2	0.017	0.011	-32%	0.026	0.011	-59%
903	Cairo	31.2	30.0	0.031	0.014	-55%	0.027	0.008	-69%
904	Cape Town	18.4	-33.9	0.022	0.012	-45%	0.022	0.010	-55%
905	Lagos	3.4	6.5	0.094	0.018	-81%	0.034	0.013	-63%
906	Kisangani	25.2	0.1	0.072	0.015	-79%	0.046	0.011	-77%
907	Mombasa	39.7	-4.0	0.100	0.023	-77%	0.059	0.015	-75%
908	Sydney	151.2	-33.9	0.025	0.012	-52%	0.021	0.010	-53%
909	Brasilia	-47.9	-15.8	0.042	0.013	-68%	0.030	0.012	-62%
910	Bogota	-74.1	4.7	0.092	0.026	-71%	0.054	0.025	-53%
911	Buenos Aires	-58.4	-34.6	0.019	0.012	-38%	0.017	0.010	-42%

Supplementary Table 4 Same as table 3, but for temperature T and humidity Q .

908

909

910

911

912

913

914

915

916

917

918

919

920

References	921
[1] Tebaldi, C., Arblaster, J.M.: Pattern scaling: Its strengths and limitations, and an update on the latest model simulations. <i>Climatic Change</i> 122 (3), 459–471 (2014) https://doi.org/10.1007/s10584-013-1032-9	922
[2] Osborn, T.J., Wallace, C.J., Lowe, J.A., Bernie, D.: Performance of Pattern-Scaled Climate Projections under High-End Warming. Part I: Surface Air Temperature over Land (2018) https://doi.org/10.1175/JCLI-D-17-0780.1	923
[3] Box, G.E.P., Jenkins, G.M., Reinsel, G.C.: Time Series Analysis: Forecasting and Control, Third edition edn. Prentice Hall, Englewood Cliffs, N.J. (1994). http://www.gbv.de/dms/bowker/toc/9780130607744.pdf	924
[4] Wang, M., Souza, A., Ferrari, R., Sapsis, T.: Spatially-resolved emulation of climate extremes via machine learning stochastic models (2023)	925
[5] Storch, H.v., Langenberg, H., Feser, F.: A Spectral Nudging Technique for Dynamical Downscaling Purposes. <i>Monthly Weather Review</i> 128 (10), 3664–3673 (2000) <a href="https://doi.org/10.1175/1520-0493(2000)128<3664:ASNTFD>2.0.CO;2">https://doi.org/10.1175/1520-0493(2000)128<3664:ASNTFD>2.0.CO;2	926
[6] Miguez-Macho, G., Stenchikov, G.L., Robock, A.: Regional Climate Simulations over North America: Interaction of Local Processes with Improved Large-Scale Flow. <i>Journal of Climate</i> 18 (8), 1227–1246 (2005) https://doi.org/10.1175/JCLI3369.1	927
[7] Sun, J., Zhang, K., Wan, H., Ma, P.-L., Tang, Q., ZHANG, S.: Impact of Nudging Strategy on the Climate Representativeness and Hindcast Skill of Constrained EAMv1 Simulations. <i>Journal of Advances in Modeling Earth Systems</i> 11 (2019) https://doi.org/10.1029/2019MS001831	928
[8] Huang, Z., Zhong, L., Ma, Y., Fu, Y.: Development and evaluation of spectral nudging strategy for the simulation of summer precipitation over the Tibetan Plateau using WRF (v4.0). <i>Geoscientific Model Development</i> 14 (5), 2827–2841 (2021) https://doi.org/10.5194/gmd-14-2827-2021	929
[9] Barthel Sorensen, B., Zepeda-Núñez, L., Lopez-Gomez, I., Wan, Z.Y., Carver, R., Sha, F., Sapsis, T.: A probabilistic framework for learning non-intrusive corrections to long-time climate simulations from short-time training data. <i>arXiv</i> (2024). https://doi.org/10.48550/arXiv.2408.02688 . http://arxiv.org/abs/2408.02688	930
[10] Barthel Sorensen, B., Charalampopoulos, A., Zhang, S., Harrop, B.E., Leung, L.R., Sapsis, T.P.: A Non-Intrusive Machine Learning Framework for Debiasing Long-Time Coarse Resolution Climate Simulations and Quantifying Rare Events Statistics. <i>Journal of Advances in Modeling Earth Systems</i> 16 (3), 2023–004122 (2024) https://doi.org/10.1029/2023MS004122	931
	932
	933
	934
	935
	936
	937
	938
	939
	940
	941
	942
	943
	944
	945
	946
	947
	948
	949
	950
	951
	952
	953
	954
	955
	956
	957
	958
	959
	960
	961
	962
	963
	964
	965
	966

967 [11] Song, Y., Sohl-Dickstein, J., Kingma, D.P., Kumar, A., Ermon, S., Poole, B.:
 968 Score-Based Generative Modeling through Stochastic Differential Equations.
 969 arXiv (2021). <https://doi.org/10.48550/arXiv.2011.13456> . <http://arxiv.org/abs/2011.13456>

971

972 [12] Batzolis, G., Stanczuk, J., Schönlieb, C.-B., Etmann, C.: Conditional image
 973 generation with score-based diffusion models. arXiv preprint arXiv:2111.13606
 974 (2021)

975

976 [13] Bischoff, T., Deck, K.: Unpaired Downscaling of Fluid Flows with Diffusion
 977 Bridges. arXiv (2023). <https://doi.org/10.48550/arXiv.2305.01822> . <http://arxiv.org/abs/2305.01822>

978

979 [14] Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for
 980 generative modeling. arXiv preprint arXiv:2210.02747 (2022)

981

982 [15] Albergo, M.S., Boffi, N.M., Vanden-Eijnden, E.: Stochastic interpolants: A
 983 unifying framework for flows and diffusions. arXiv preprint arXiv:2303.08797
 984 (2023)

985

986 [16] Anderson, B.D.: Reverse-time diffusion equation models. Stochastic Processes and
 987 their Applications **12**(3), 313–326 (1982)

988

989 [17] Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional Networks for
 990 Biomedical Image Segmentation. arXiv (2015). <https://doi.org/10.48550/arXiv.1505.04597> . <http://arxiv.org/abs/1505.04597>

991

992 [18] He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recog-
 993 nition. arXiv (2015). <https://doi.org/10.48550/arXiv.1512.03385> . <http://arxiv.org/abs/1512.03385>

994

995 [19] Wheeler, M., Kiladis, G.N.: Convectively Coupled Equatorial Waves: Analysis of
 996 Clouds and Temperature in the Wavenumber–Frequency Domain (1999)

997

998 [20] Kiladis, G.N., Wheeler, M.C., Haertel, P.T., Straub, K.H., Roundy, P.E.: Con-
 999 vectively coupled equatorial waves. Reviews of Geophysics **47**(2) (2009) <https://doi.org/10.1029/2008RG000266>

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012