Supporting Information for "Learning to infer weather states using partial observations"

Jie Chao[1,2], Baoxiang Pan[2], Quanliang Chen[1], Shangshang Yang[2,3], Jingnan Wang[2,4], Congyi Nai[2,5], Yue Zheng[6], Xichen Li[2], Huiling Yuan[3], Xi Chen[2], Bo Lu[7], Ziniu Xiao[2]

[1]School of Atmospheric Sciences, Chengdu University of Information Technology, Sichuan, China

[2]Institute of Atmospheric Physics, Chinese Academy of Science, Beijing, China

[3]Key Laboratory of Mesoscale Severe Weather, Ministry of Education, and School of Atmospheric Sciences, Nanjing University, Jiangsu, China

[4]College of Computer, National University of Defense Technology, Hunan, China

[5]Institute of Geographic Sciences and Natural Resources Research, Chinese Academy of Sciences, Beijing, China

[6]Clustertech LTD, Hong Kong, China

[7]National Climate Center, China Meteorological Administration, Beijing, China

Contents of this file

6. Figures S1 Network architecture of diffusion model

7. Figures S2 The first three EOF modes.

S1. Details of probabilistic diffusion model

Here, we provide detailed mathematical formulations and implementation specifics of the deployed probabilistic diffusion model. For more information and useful learning materials, refer to the works of Sohl-Dickstein et al.(2015), Ho et al. (2020), Song et al. (2020) , Kingma et al. (2021), Ho & Salimans (2022) and Luo (2022).

Diffusion models are probabilistic models that describe the evolution of a stochastic process over time. In the context of deep learning diffusion models, the diffusion process and its reverse process are fundamental concepts.

The diffusion process is the forward process through which a model generates data, typically images, from a simple noise distribution (often Gaussian noise) to the target distribution. A step-by-step derivation is provided below.

First, we define the following Gaussian process to transform the target distribution $p(\mathbf{x}_0)$ to a prior distribution $p(\mathbf{x}_T)$:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{1}$$

Here $\mathbf{x}_{t\in[1,T]}$ are latent variables with increasing noise level; $\mathcal{N}$ is Gaussian distribution; $\mathbf{I}$ is identity matrix; β_t is diffusion coefficient, which is pre-defined so that, give large enough T, $p(\mathbf{x}_T|\mathbf{x}_0)$ is drawn close to $p(\mathbf{x}_T)$, which is $\mathbf{x}_0$ agnostic.

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{2}$$

We parameterize the Gaussian encoder with mean $\mu_t(x_t) = \sqrt{\alpha_t}x_{t-1}$, and variance $\mathbf{\Sigma}_t(\boldsymbol{x}_t) = (1-\alpha_t)\mathbf{I}$, Here $\alpha_t = 1-\beta_t$. Mathematically, encoder transitions are denoted as:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \tag{3}$$

$$= \sqrt{\alpha_t\alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2} \tag{4}$$

$$= \dots \tag{5}$$

$$= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon} \tag{6}$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{7}$$

These assumptions depict a systematic process of adding Gaussian noise to the data input over time. As we continue to corrupt the data, it gradually transitions until it is entirely characterized by pure Gaussian noise.

In essence, the reverse process aims to infer the noise distribution that could have generated the observed data. Similar to the diffusion process, the reverse process is represented as:

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \mathbf{\Sigma}_\theta(\mathbf{x}_t, t)) \quad p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T)\prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \tag{8}$$

Here, $\mathbf{\Sigma}_\theta$ is parameterized as an interpolation between its analytical lower and upper bounds (Dhariwal & Nichol, 2021). The optimization of μ_θ involves maximizing the variational lower bound (ELBO) on the log-likelihood of the training samples (Sohl-Dickstein et al., 2015; Kingma et al., 2021).

Then, diffusion model can be optimized by maximizing the ELBO, which can be derived as follows:

$$\log p(\boldsymbol{x}) = \log \int p(x_{0:T}) dx_{1:T} \tag{9}$$

$$= \log \int \frac{p(x_{0:T}) q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)} dx_{1:T} \tag{10}$$

$$= \log \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] \tag{11}$$

$$\geq \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_{0:T})}{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \right] \tag{12}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_1|\boldsymbol{x}_0) \prod_{t=2}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{x}_0)} \right] \tag{13}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)}{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} + \log \prod_{t=2}^{T} \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{\frac{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) q(\boldsymbol{x}_t|\boldsymbol{x}_0)}{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_0)}]} \right] \tag{14}$$

$$= \mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_{\mathbf{0}}(x_0|\boldsymbol{x}_1) \right] + \mathbb{E}_{q(\boldsymbol{x}_T|\boldsymbol{x}_0)} \left[\log \frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T|\boldsymbol{x}_0)} \right]$$
$$+ \sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}|\boldsymbol{x}_0)} \left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)} \right] \tag{15}$$

$$= \mathbb{E}_{q(x_1|x_0)} \left[\log p_\theta(x_0|x_1) \right] - D_{\mathrm{KL}}(q(x_T|x_0) \parallel p(x_T))$$
$$- \sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(x_{t-1}|x_t, x_0) \parallel p_{\boldsymbol{\theta}}(x_{t-1}|x_t)) \right] \tag{16}$$

We now explain the three terms on the right-hand side of the Eq. 16:

• $\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1) \right]$ represents the expected log-likelihood of the initial data $\boldsymbol{x}_0$ given the sampled intermediate data $\boldsymbol{x}_1$. For the first step, we have $\mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)} \left[\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1) \right] = 0$.

• $D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_0) \parallel p(\boldsymbol{x}_T))$ denotes the KL divergence between the approximate posterior distribution $q(\boldsymbol{x}_T|\boldsymbol{x}_0)$ and the prior distribution $p(\boldsymbol{x}_T)$ at the final time step $\mathbf{T}$. Where $p(\boldsymbol{x}_T) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, it implies that $\mathbb{E}_{q(\boldsymbol{x}_{T-1}|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_{T-1}) \parallel p(\boldsymbol{x}_T)) \right] = 0$

• $\sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)) \right]$ represents the sum of the expected KL divergences between the approximate posterior distributions $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ and the

conditional distributions $p_\theta(\boldsymbol{x}_t|\boldsymbol{x}_{t+1})$ for each intermediate time step $\boldsymbol{t}$ in the reverse diffusion process.

Given the analysis above, maximizing $\log p(\boldsymbol{x})$ can be approximately achieved by minimizing the third term. While minimizing each KL Divergence term individually can be challenging for arbitrary posteriors, we can leverage Bayes' rule to simplify the process:

$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1}, x_0)q(x_{t-1}|x_0)}{q(x_t|x_0)} \tag{17}$$

$$= \frac{\mathcal{N}(x_t; \sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)\mathbf{I})\mathcal{N}(x_{t-1}; \sqrt{\alpha_{t-1}}x_0, (1-\bar{\alpha}_{t-1})\mathbf{I})}{\mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I})} \tag{18}$$

$$\propto \exp\left\{-\left[\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{2(1-\alpha_t)} + \frac{(x_{t-1} - \sqrt{\alpha_{t-1}}x_0)^2}{2(1-\bar{\alpha}_{t-1})} - \frac{(x_t - \sqrt{\alpha_t}x_0)^2}{2(1-\bar{\alpha}_t)}\right]\right\} \tag{19}$$

$$= \exp\left\{-\frac{1}{2}\left[\frac{(x_t - \sqrt{\alpha_t}x_{t-1})^2}{1-\alpha_t} + \frac{(x_{t-1} - \sqrt{\alpha_{t-1}}x_0)^2}{1-\bar{\alpha}_{t-1}} - \frac{(x_t - \sqrt{\alpha_t}x_0)^2}{1-\bar{\alpha}_t}\right]\right\} \tag{20}$$

$$= \exp\left\{-\frac{1}{2}\left(\frac{1}{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}}\right)\left[x_{t-1}^2 - 2\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)x_0}{1-\bar{\alpha}_t}x_{t-1}\right]\right\} \tag{21}$$

$$\propto \mathcal{N}(x_{t-1}; \underbrace{\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\alpha_{t-1}}(1-\alpha_t)x_0}{1-\bar{\alpha}_t}}_{\mu_q(\boldsymbol{x}_t, \boldsymbol{x}_0)}, \underbrace{\frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{I}}_{\boldsymbol{\Sigma}_q(t)}) \tag{22}$$

Hence, it is demonstrated that at each step $\boldsymbol{x}_{t-1} \sim q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ follows a normal distribution. We use the KL Divergence between two Gaussian distributions for calculation.

$$\arg\min_{\boldsymbol{\theta}} D_{\mathrm{KL}}\left(q\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0\right) \| p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right)\right) \tag{23}$$

$$= \arg\min_{\boldsymbol{\theta}} D_{\mathrm{KL}}\left(\mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q(t)\right) \| \mathcal{N}\left(\boldsymbol{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_q(t)\right)\right) \tag{24}$$

$$= \arg\min_{\boldsymbol{\theta}} \frac{1}{2}\left[\log \frac{|\boldsymbol{\Sigma}_q(t)|}{|\boldsymbol{\Sigma}_q(t)|} - d + \operatorname{tr}\left(\boldsymbol{\Sigma}_q(t)^{-1}\boldsymbol{\Sigma}_q(t)\right) + \left(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right)^T \boldsymbol{\Sigma}_q(t)^{-1}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right)\right] \tag{25}$$

$$= \arg\min_{\boldsymbol{\theta}} \frac{1}{2}\left[\left(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right)^T \left(\sigma_q^2(t)\mathbf{I}\right)^{-1}\left(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right)\right] \tag{26}$$

$$= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)}\left[\left\|\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right\|_2^2\right] \tag{27}$$

After optimizing the Diffusion Model, the sampling procedure simplifies to sampling Gaussian noise from $p(\boldsymbol{x}_T)$ and iteratively running the denoising transitions $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ for T steps to generate a novel $\boldsymbol{x}_0$. In practice, we denote $\boldsymbol{\mu}_{\boldsymbol{\theta}}$ as function of neural network parameterization for $\nabla p(\mathbf{x}_t|\mathbf{x}_0)$, which is commonly known as the *score function* (Y. Song et al., 2020).

S2. CLIN

In our approach, we merge the acquired climatology prior with station observations to deduce the posterior probability distribution of the target variable. This allows us to jointly modify both observed and unobserved regions throughout the denoising steps, yielding generated samples that are spatially coherent, faithful and adaptive to observation constraints, and uncertainty-aware. The specific parameters of CLIN are presented in the following table. S1.

S3. Model parameters schedule

We trained the neural network on the NVIDIA Tesla V100 32GB GPU using CUDA version 12.3. The neural network architecture details of the diffusion model are illustrated

in Fig. S1. Typical hyperparameter configurations for diffusion models are often derived from the(Ho & Salimans, 2022).

The specific hyperparameters of the model are presented in the following table. S1.

we embed the time information, and stack the time embedding as an additional channel to all UNet blocks. Each contracting block consists of a long sequence of $\{C_{3*3} + N + ReLU\}_3$ operations and a short sequence of $\{C_{1*1}\}_1$ operations, concatenated as a residual block. Here, C_{n*n} is convolution layer with kernel receptive field of size $n * n$. N is group normalization, ReLU is rectified linear unit function. Each expand block consists of a long sequence of $\{R_2 + C_{3*3} + N + ReLU\}_3$ operations and a short sequence of $\{R_2, C_{1*1}\}_1$ operations, concatenated as a residual block. Here, R_n resizes the data by n times using linear interpolation. We begin with a channel size of 64 and double/shrink the channel size by 2 along each contracting/expanding block.

S4. Evaluation metrics

S4.1 Pearson correlation coefficient (corr)

The Pearson correlation coefficient (*corr*) between prediction $\hat{x}$ and observation x is calculated as follows:

$$corr = \frac{\sum_{i=1}^{n}(\hat{x}_i - \bar{\hat{x}})(x_i - \bar{x})}{\sqrt{\sum_{i=1}^{n}(\hat{x}_i - \bar{\hat{x}})^2 \cdot \sum_{i=1}^{n}(x_i - \bar{x})^2}} \tag{28}$$

S4.2 Root mean square error(RMSE)

The root mean square error (RMSE) between prediction $\hat{x}$ and observation x is calculated as follows:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{29}$$

S4.3 Empirical Orthogonal Function

Empirical Orthogonal Function (EOF) analysis, also known as Principal Component Analysis (PCA) in some contexts, is a widely used statistical method in various fields, including meteorology, oceanography, climatology, and geophysics.

It is employed to analyze and extract the dominant patterns of variability present in a multivariate dataset, such as spatial patterns in climate data or in oceanographic data. The detailed calculation method for EOF is based on the PrincipalComponents function in Mathematica.

S4.4 Kolmogorov-Smirnov test

The Kolmogorov-Smirnov test (KS test) is a statistical method used to compare the empirical cumulative distribution function (CDF) of a sample dataset with a reference probability distribution or another sample dataset. It is particularly useful for assessing whether the two datasets are drawn from the same underlying distribution or if they differ significantly.

The KS test operates by computing the maximum difference (or maximum deviation) between the two cumulative distribution functions. This maximum difference, often denoted as the KS statistic (D), represents the largest vertical distance between the empirical CDF and the theoretical (or reference) CDF. The KS statistic is then compared against critical values from the Kolmogorov-Smirnov distribution, which depends on the sample size and the significance level chosen for the test.

The specific computation method for the Kolmogorov-Smirnov test is derived from Mathematica's KolmogorovSmirnovTest function.

S4.5 Power spectrum density

The radial averaged power spectrum density (PSD) is a quantitative measure used in various fields of science and engineering, including signal processing, optics, and geophysics. It provides valuable insights into the distribution of power across different spatial frequencies in a given signal or image. In this paper, the PSD is calculated by first computing the Fourier transform of the signal or image to obtain its frequency domain representation. The power spectrum density is then computed as the squared magnitude of the Fourier transform. The PSD further averages the power spectrum density over concentric circles or spherical shells centered at the origin, hence the term "radial averaged." This averaging process is performed to capture the isotropic characteristics of the signal or image, ensuring that contributions from all directions are considered equally.

The PSD is particularly useful for analyzing signals or images with rotational symmetry or spatial periodicity. By averaging the power spectrum density radially, it becomes possible to discern patterns or structures that are not readily apparent in the original signal or image. Additionally, the PSD can be used to quantify the dominant spatial frequencies present in the signal or image, providing valuable information for further analysis or interpretation.

In short, the radial averaged power spectrum density offers a comprehensive view of the spatial frequency content of a signal or image, facilitating insights into its underlying structure and characteristics.

The specific calculation method for the PSD is derived from the pySTEPS library in Python.

References

Dhariwal, P., & Nichol, A. (2021). Diffusion models beat gans on image synthesis.

Advances in neural information processing systems, *34*, 8780–8794.

Ho, J., Jain, A., & Abbeel, P. (2020). Denoising diffusion probabilistic models. *Advances in neural information processing systems*, *33*, 6840–6851.

Ho, J., & Salimans, T. (2022). Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*.

Kingma, D., Salimans, T., Poole, B., & Ho, J. (2021). Variational diffusion models. *Advances in neural information processing systems*, *34*, 21696–21707.

Luo, C. (2022). Understanding diffusion models: A unified perspective. *arXiv preprint arXiv:2208.11970*.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., & Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning* (pp. 2256–2265).

Song, J., Meng, C., & Ermon, S. (2020). Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., & Poole, B. (2020). Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Table S1. Hyperparameters of Diffusion model

Hyperparameter	**Setting**	**Parameter**
Learning Rate	10^{-4}	$\alpha_t^2 = 1 - \sigma_t^2 = \frac{1}{1+e^{-\lambda_t}}$
Batch Size	64	$\lambda_t = -2\log\tan(at+b)$
Channel	64	$b = \arctan(e^{-\frac{\lambda\max}{2}})$
Optimizer	Adam	$a = \arctan(e^{-\frac{\lambda\min}{2}}) - b$
Number of Iterations	1000	$t = \frac{i}{1000}$, Where $i = 0, 1, 2, \ldots, 1000$
$\lambda\min$	-20	$\text{embedding}(t) = [\sin(2\pi\omega t); \cos(2\pi\omega t)]$
$\lambda\max$	20	$\omega \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

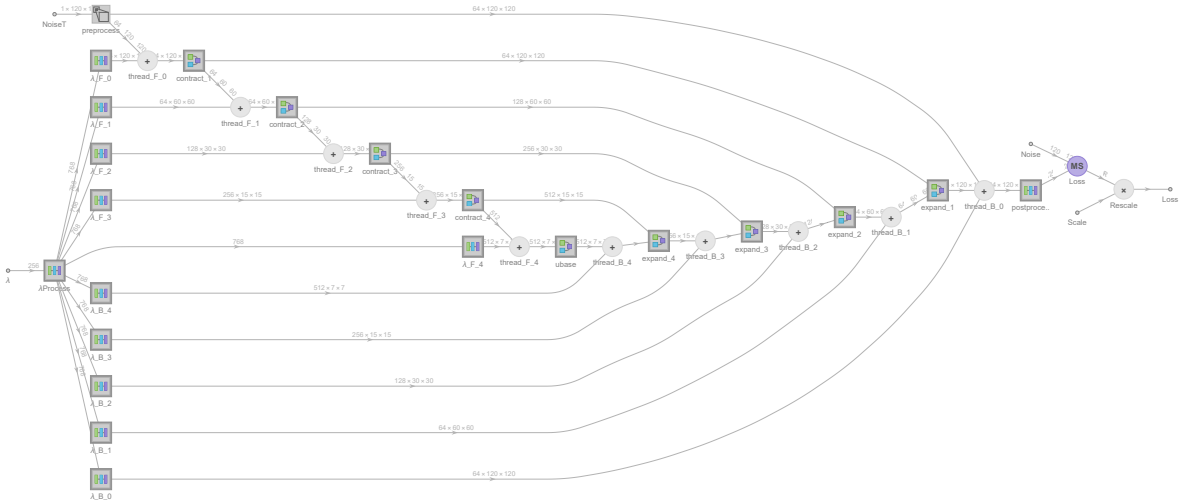

Figure S1. Network Architecture of diffusion model. Each contracting block consists of a long sequence of $\{C_{3*3} + N + ReLU\}_3$ operations and a short sequence of $\{C_{1*1}\}_1$ operations, concatenated as a residual block. Here, C_{n*n} is convolution layer with kernel receptive field of size $n*n$. N is group normalization, ReLU is rectified linear unit function. Each expand block consists of a long sequence of $\{R_2 + C_{3*3} + N + ReLU\}_3$ operations and a short sequence of $\{R_2, C_{1*1}\}_1$ operations, concatenated as a residual block.

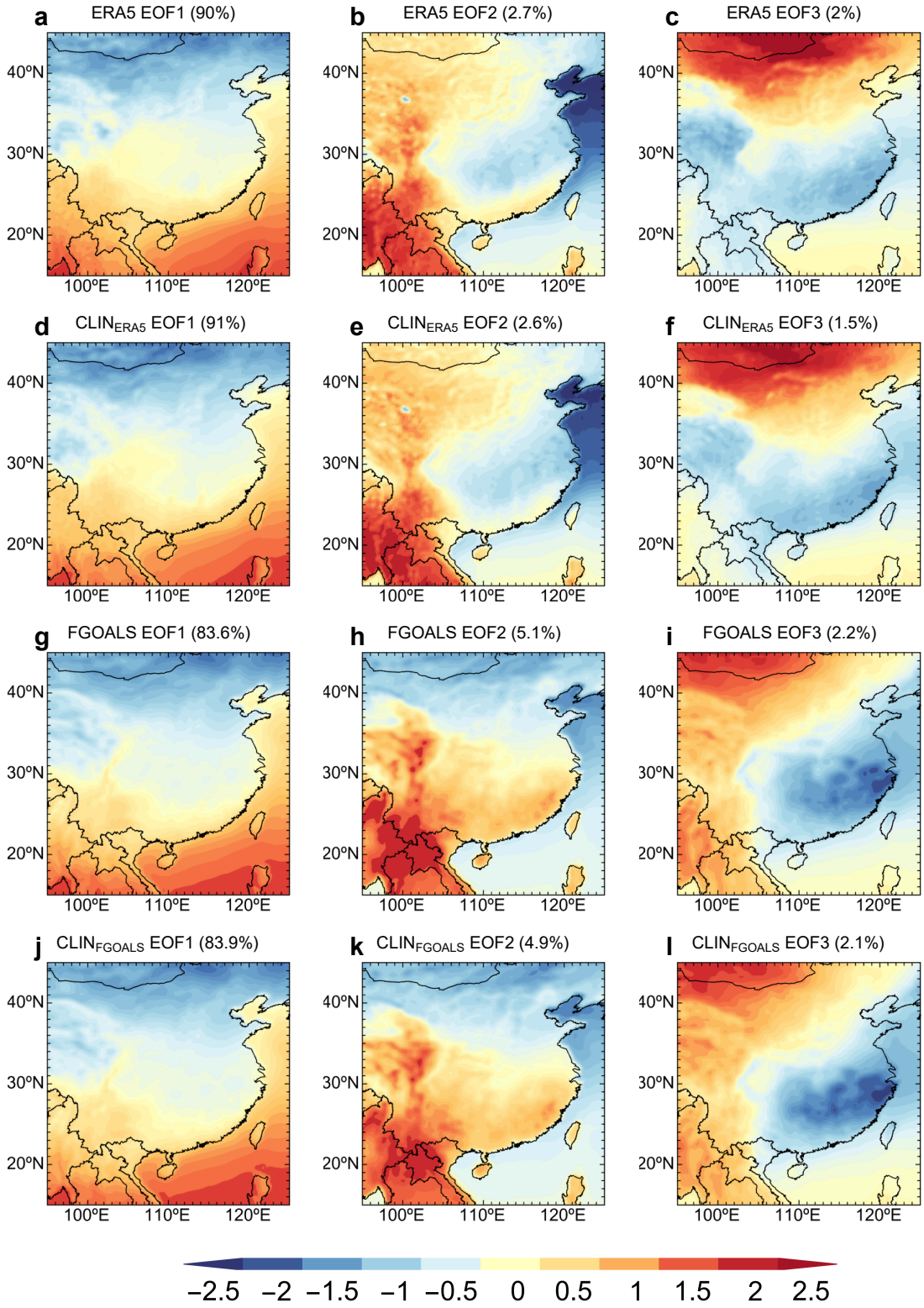

Figure S2. The first three EOF modes. ERA5 (a-c), *Clin*_ERA5 (d-f), FGOALS (g-i) and CLIN_FGOALS (j-l).