Supporting Information for "Reliable precipitation nowcasting using probabilistic diffusion model"

Congyi Nai[1,3], Baoxiang Pan[2], Jiarui Hai[4], Xi Chen[2], Qiuhong Tang[1,3],Guangheng Ni[4], Qingyun Duan[5], Bo Lu[6], Ziniu Xiao[2], Xingcai Liu[1,3,*]

[1] Key Laboratory of Water Cycle and Related Land Surface Processes, Institute of Geographic Sciences and Natural Resources Research, Chinese Academy of Sciences, Beijing, China

[2] Institute of Atmospheric physics, Chinese Academy of Sciences, Beijing, China

[3] University of Chinese Academy of Sciences, Beijing, China

[4] State Key Laboratory of Hydro-science and Engineering, Department of Hydraulic Engineering, Tsinghua University, Beijing 100084, China

[5]The National Key Laboratory of Water Disaster Prevention, Hohai University, Nanjing, China

[6]Laboratory for Climate Studies and CMA-NJU Joint Laboratory for Climate Prediction Studies, National Climate Center, China Meteorological Administration, Beijing, China

*Corresponding author. E-mail address: xingcailiu@igsnrr.ac.cn

1 Details of diffusion model

1.1 Basic diffusion

Let X_0 be a sample from the data distribution $q(X_0)$, and defines a sequence of increasingly noisy versions of X which we call the latent variables X_t $(t = 1 \ldots T)$ through the forward diffusion process, described by

$$q(X_t|X_{t-1}) = N(X_t; \sqrt{1-\beta_t}X_{t-1}, \beta_t \boldsymbol{I}) \tag{1}$$

Then, the form of $q(X_t|X_0)$ can be recursively derived through repeated applications of the reparameterization trick, suppose we have $\{\epsilon_t, \bar{\epsilon}_t\}_{t=0}^{T} \sim N(0, I)$, Then, for an arbitrary sample $X_t \sim q(X_t|X_0)$, we can rewrite it as:

$$\begin{aligned}
x_t &= \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_{t-1} \\
&= \sqrt{\alpha_t}(\sqrt{\alpha_{t-1}}x_{t-2} + \sqrt{1-\alpha_{t-1}}\epsilon_{t-2}) + \sqrt{1-\alpha_t}\epsilon_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{\alpha_t - \alpha_t\alpha_{t-1}}\epsilon_{t-2} + \sqrt{1-\alpha_t}\epsilon_{t-1} \\
&= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\epsilon}_{t-2} \\
&= \cdots \\
&= \sqrt{\prod_{i=1}^{t}\alpha_i}\, x_0 + \sqrt{1-\prod_{i=1}^{t}\alpha_i}\, \bar{\epsilon}_0 \\
&= \sqrt{\bar{\alpha}_t}X_0 + \sqrt{1-\bar{\alpha}_t}\bar{\epsilon}_0, \ where\ \bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i
\end{aligned} \tag{2}$$

In equation 3, we have leveraged the property that the sum of two independent Gaussian random variables retains a Gaussian distribution, with the mean being the sum of the two individual means and the variance being the sum of their variances. $\sqrt{\alpha_t - \alpha_t\alpha_{t-1}}\epsilon_{t-2}$ is a sample from Gaussian $N(0, (\alpha_t - \alpha_t\alpha_{t-1})\boldsymbol{I})$, $\sqrt{1-\alpha_t}\epsilon_{t-1}$ is a sample from Gaussian $N(0, (1-\alpha_t)\boldsymbol{I})$, we can then treat their sum as a random variable sampled from Gaussian $N(0, (1-\alpha_t+\alpha_t-\alpha_t\alpha_{t-1})\boldsymbol{I})$. Hence, the X_t can be sampled directly from X_0, the transition kernel is

$$q(X_t|X_0) = N\left(X_t; \sqrt{\bar{\alpha}_t}X_0, \sqrt{1-\bar{\alpha}_t}\boldsymbol{I}\right), where\ \alpha_t = 1-\beta_t, \bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i. \tag{3}$$

Given X_0 and a Gaussian vector $\epsilon \sim N(0, I)$ and applying the transformation

$$X_t = \sqrt{\bar{\alpha}_t} X_0 + \sqrt{1-\bar{\alpha}_t}\epsilon. \tag{4}$$

When the $\bar{\alpha}_T \to 0,\ X_T$ is well approximated by Gaussian distribution. During the forward process, noise is gradually added to the data until it loses its original spatial structure characteristics and becomes pure noise. If we can solve the reverse process $P(X_{t-1}|X_t)$, we can sample $X_T \sim N(0, I)$ then a sequence of neural networks is employed to gradually reduce the noise in a series of steps $X_T, X_{T-1} \dots X_0$. These properties suggest learning a learnable Markov chain model $P_\theta(X_{t-1}|X_t)$ to approximate the true reverse process:

$$P_\theta(X_{t-1}|X_t) = N(X_{t-1}; \mu_\theta(X_t), \Sigma_\theta(X_t)), \tag{5}$$

Therefore, in a diffusion model, we are only interested in learning conditionals $P_\theta(X_{t-1}|X_t)$, the diffusion model can be optimized by maximizing the variational lower bound (VLB) of the log-likelihood of the data X_0,

$$\begin{aligned}
\mathbb{E}_{q(X_0)}(-\log P_\theta(X_0)) &\le \mathbb{E}_{q(X_0)}[-\log P_\theta(X_0) + D_{KL}\,(q(X_{1:T}|X_0)||P_\theta(X_{1:T}|X_0))] \\
&= \mathbb{E}_{q(X_0)}\left[-\log P_\theta(X_0) + \int q(X_{1:T}|X_0) \log \frac{q(X_{1:T}|X_0)}{P_\theta(X_{0:T})/P_\theta(X_0)} d\,X_{1:T}\right] \\
&= \mathbb{E}_{q(X_0)}\left[-\log P_\theta(X_0) + \int q(X_{1:T}|X_0) \log \frac{q(X_{1:T}|X_0)}{P_\theta(X_{0:T})} d\,X_{1:T} + \log P_\theta(X_0)\right] \\
&= \mathbb{E}_{q(X_{0:T})} \log \frac{q(X_{1:T}|X_0)}{P_\theta(X_{0:T})} = L_{VLB}
\end{aligned} \tag{6}$$

We can rewrite variational lower bound (VLB) as,

$$\begin{aligned}
L_{VLB} &= \mathbb{E}_{q(x_0 T)}[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})}] \\
&= \mathbb{E}_q[\log \frac{\prod_{t=1}^{T} q(x_t|x_{t-1})}{p_\theta(x_T)\prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)}] \\
&= \mathbb{E}_q[-\log\ p_\theta(\boldsymbol{x}_T) + \sum_{t=1}^{T} \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)}] \\
&= \mathbb{E}_q[-\log\ p_\theta(\boldsymbol{x}_T) + \sum_{t=2}^{T} \log \frac{q(x_t|x_{t-1})}{p_\theta(x_{t-1}|x_t)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}] \\
&= \mathbb{E}_q[-\log\ p_\theta(\boldsymbol{x}_T) + \sum_{t=2}^{T} \log (\frac{q(x_{t-1}|x_t,x_0)}{p_\theta(x_{t-1}|x_t)} \cdot \frac{q(x_t|x_0)}{q(x_{t-1}|x_0)}) + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}] \\
&= \mathbb{E}_q[-\log\ p_\theta(\boldsymbol{x}_T) + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t,x_0)}{p_\theta(x_{t-1}|x_t)} + \sum_{t=2}^{T} \log \frac{q(x_t|x_0)}{q(x_{t-1}|x_0)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}] \\
&= \mathbb{E}_q[-\log\ p_\theta(\boldsymbol{x}_T) + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t,x_0)}{p_\theta(x_{t-1}|x_t)} + \log \frac{q(x_T|x_0)}{q(x_1|x_0)} + \log \frac{q(x_1|x_0)}{p_\theta(x_0|x_1)}] \\
&= \mathbb{E}_q[\log \frac{q(x_T|x_0)}{p_\theta(x_T)} + \sum_{t=2}^{T} \log \frac{q(x_{t-1}|x_t,x_0)}{p_\theta(x_{t-1}|x_t)} - \log p_\theta(X_0|X_1)] \\
&= \mathbb{E}_q[D_{KL}(q(X_T|X_0)||p_\theta(X_T)) + \sum_{t=2}^{T} D_{KL}(q(X_{t-1}|X_t,X_0)||p_\theta(X_{t-1}|X_t) - \log p_\theta(X_0|X_1)]
\end{aligned} \tag{7}$$

This formulation also has an elegant interpretation, which is revealed when inspecting each individual term:

1. $L_0 = \mathbb{E}_q[\log p_\theta(X_0|X_1)]$ can be interpreted as a reconstruction term.
2. $L_T = \mathbb{E}_q[D_{KL}(q(X_T|X_0)||p_\theta(X_T))]$ represents how close the distribution of the final noisified input is to the standard Gaussian prior, is equal to zero under our

assumptions.

3. $L_t = \mathbb{E}_q[\sum_{t=2}^{T} D_{KL}(q(X_{t-1}|X_t, X_0)||p_\theta(X_{t-1}|X_t)]$ is a denoising matching term. The $q(X_{t-1}|X_t, X_0)$ acts as a ground-truth signal and $p_\theta(X_{t-1}|X_t)$ is our desired denoising transition step. This term is therefore minimized when the two denoising steps match as closely as possible. It is the primary optimization objective.

If we have knowledge of X_0, we can obtain $q(X_{t-1}|X_t, X_0)$ through the Bayes' theorem,

$$
\begin{aligned}
q(X_{t-1}|X_t, X_0) &= q(X_t|X_{t-1}, X_0)\frac{q(X_{t-1}|X_0)}{q(X_t|X_0)} \\
&\propto exp\,(-\frac{1}{2}(\frac{\left(X_t-\sqrt{\alpha_t}X_{t-1}\right)^2}{\beta_t} + \frac{\left(X_{t-1}-\sqrt{\bar{\alpha}_{t-1}}X_0\right)^2}{1-\bar{\alpha}_{t-1}} - \frac{\left(X_t-\sqrt{\bar{\alpha}_t}X_0\right)^2}{1-\bar{\alpha}_t})) \\
&= exp\,(-\frac{1}{2}((\frac{\alpha_t}{\beta_t} + \frac{1}{1-\bar{\alpha}_{t-1}})X_{t-1}^2 + \left(\frac{2\sqrt{\alpha_t}}{\beta_t}X_t + \frac{2\sqrt{\bar{\alpha}_{t-1}}}{1-\bar{\alpha}_{t-1}}X_0\right)X_{t-1} + C(X_t, X_0))) \\
&= N(X_{t-1}; \tilde{\mu}(X_t, X_0), \tilde{\beta}(t)I) \qquad (8)
\end{aligned}
$$

Recall equation 8 and equation 5, we can obtain,

$$\tilde{\mu}_\theta(X_t, X_0) = \frac{1}{\sqrt{\alpha_t}}(X_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(X_t, t)) \qquad (9)$$

Let us consider the $L_t = D_{KL}(q(X_{t-1}|X_t, X_0)||p_\theta(X_{t-1}|X_t))$, given equation 6 and equation 8, we can get the loss function,

$$
\begin{aligned}
L_t &= \mathbb{E}_{x_0,\epsilon}[\frac{1}{2\|\Sigma_\theta(x_t,t)\|_2^2} \| \tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t) \|^2] \\
&= \mathbb{E}_{x_0,\epsilon}[\frac{1}{2\|\Sigma_\theta\|_2^2} \| \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_t) - \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)) \|^2] \\
&= \mathbb{E}_{\mathbf{x}_0,\epsilon}[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\mathbf{\Sigma}_\theta\|_2^2} \| \boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \|^2] \\
&= \mathbb{E}_{\mathbf{x}_0,\epsilon}[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\|\mathbf{\Sigma}_\theta\|_2^2} \| \boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t) \|^2] \qquad (10)
\end{aligned}
$$

Ho et al. (2020) propose to reweight various terms in L_{VLB} for better sample quality, to compute this objective, we generate samples $X_t \sim q(X_t|X_0)$, then train a model ϵ_θ to predict the added noise using a standard mean-squared error loss:

$$L_{simple} = \mathbb{E}_{t\sim[1,T], X_0\sim q(X_0), \epsilon\sim N(0,I)}[||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathrm{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t)||^2]. \qquad (11)$$

2.3 Conditional diffusion

So far, we have focused on modeling the data distribution p(x). However, we are often also interested in the conditional distribution of $\mathrm{P}(X_t|\mathrm{y})$, as it enables us to better investigate how different conditional information influences the generation of variable X. Begin with the score-based formulation of a diffusion model, the goal is to learn $\nabla \log \mathrm{P}(X_t|\mathrm{y})$, by Bayes rules, we can get the equivalent:

$$
\begin{aligned}
\nabla \log \mathrm{P}(X_t|\mathrm{y}) &= \nabla\, log(\frac{P(y|X_t)\,P(X_t)}{P(y)}) \qquad (12) \\
&= \nabla\, log\, P(X_t) + \nabla\, log\, P(y|X_t) - \nabla\, log\, P(y) \qquad (13) \\
&= \underbrace{\nabla \log P(X_t)}_{unconditional\ score} + \underbrace{\nabla \log \mathrm{P}(\mathrm{y}|X_t)}_{conditional\ score} \qquad (14)
\end{aligned}
$$

To better control the conditional information, a hyperparameter γ is introduced to scale the gradient of the conditioning information. The score function can then be summarized as:

$$\nabla \log P(X_t|y) = \nabla \log P(X_t) + \gamma \nabla \log P(y|X_t). \tag{15}$$

Intuitively speaking, the $\gamma = 0$ the diffusion model can ignore the conditional information entirely, while a large γ value would cause the model to heavily incorporate the conditional information during sampling. In order to implement effective control over the conditional information, we use classifier-free guidance (Ho & Salimans, 2021). To get the score function under Classifier-Free Guidance, we can rearrange:

$$\nabla \log P(y|X_t) = \nabla \log P(X_t|y) - \nabla \log P(X_t). \tag{16}$$

Substituting equation (16) into equation (15) then we get:

$$\nabla \log P(X_t|y) = \nabla \log P(X_t) + \gamma(\nabla \log P(X_t|y) - \nabla \log P(X_t)). \tag{17}$$

$$= \underbrace{(1-\gamma)\nabla \log P(X_t)}_{unconditional\ score} + \underbrace{\gamma \nabla \log P(X_t|y)}_{conditional\ score} \tag{18}$$

From Tweedie's formula and equation 5, we can get,

$$\nabla \log p(x_t) = -\frac{1}{\sqrt{1-\bar{\alpha}_t}}\epsilon \tag{19}$$

The equation 19 means that estimating ϵ is equivalent to estimating a scaled version of the score function. So, in this paper, we model the conditional distribution of precipitation frames in the future given the past precipitation frames $\boldsymbol{P} = [p_1, p_2, \ldots p_M]$, we learn two sets of neural networks, $\epsilon_\theta(X_t, t)$ and $\epsilon_\theta(X_t, t, P)$, to approximate the unconditional and conditional score functions $\nabla \log P(X_t)$ and $\nabla \log P(X_t|y)$, our conditional diffusion loss function is:

$$L_{condition} = \mathbb{E}_{t\sim[1,T], X_0 \sim q(X_0), \epsilon \sim N(0,I)}[||\epsilon - \epsilon_\theta(X_t, t, P)||^2]. \tag{19}$$

2 Details of baseline model

2.1 Generative models of radar

DGMR holds the current state of the art in precipitation nowcasting, the generator is built with convolutional and convolutional GRU layers and it was trained with two adversarial loss and a regularization loss. The first loss is defined by a spital discriminator, which ensures spital consistency. The second loss is defined by a temporal discriminator, which is a 3D convolutional neural network that aims to impose temporal consistency. The regularization term encourages the prediction's mean precipitation fields to match the mean of past precipitation amount.

We utilized Google-Colab to load the saved DGMR model and pconducted inference on our test dataset, see https://github.com/deepmind/deepmind-research/tree/master/nowcasting. DGMR exhibits the capability to generate forecasts up to 90 minutes. However, for the purpose of comparison, we only evaluated its performance using the first 30 minutes of forecasted results, calculating relevant metrics.

2.2 U-Net

We use a U-Net encoder–decoder model as baseline similarly to how it was used in related studies (Ayzel et al., 2020). This type of model first employs an encoder that reduces the spatial resolution using pooling and convolutional layers, while the decoder then increases the resolution by applying up-sampling and convolutional layers to the learned patterns. To prevent gradient vanishing and share the low-level patterns of the precipitation fields, skip connections are used from the encoder to the decoder (Srivastava et al., 2015). In this paper, U-Net serves as the baseline for deterministic forecasting using deep learning.

2.3 PySTEPS

PySTEPS is an open-source Python library designed for radar precipitation forecasting and analysis, it is available at https://github.com/pySTEPS/pysteps. It offers a comprehensive range of algorithms, among which STEPS is a widely used precipitation nowcasting system based on ensembles, considered to be state-of-the-art of non-ML-based method. In this study, we adopt PySTEPS as a non-machine learning baseline.

3 Details of metrics

we use the M to denote number of the ensemble members, and f_m to denote the ensemble member, so the ensemble mean can be written as,

$$\bar{f} = \frac{1}{M}\sum_{m=1}^{M} f_m \tag{20}$$

3.1 MAE

The (spatial) mean-absolute-error (MAE) at forecast time step t between ensemble means $\bar{f}$ and observation f_{obser} is defined as,

$$MAE_t(\bar{f}, f_{obser}) = \frac{1}{P}\sum_{p=1}^{P} \left| \bar{f} - f_{obser} \right| \tag{21}$$

where p indexes all the geospatial locations. And we can consider extreme value prediction accuracy under different precipitation intensities, we use an intensity mask $[f_{obser} > 4]$ and $[f_{obser} > 8]$ to get the masked prediction and observation $\bar{f}_m, f_{m\_obser}$

$$MAE_{t,mid}(\bar{f}_m, f_{m\_obser}) = \frac{1}{P}\sum_{p=1}^{P} \left| \bar{f}_m - f_{m\_obser} \right| \tag{22}$$

3.2 Correlation

The spital correlation between ensemble mean and observation is defined as.

$$Corr_t(\bar{f}, f_{obser}) = \frac{\sum_p (\bar{f}_p - \overline{\bar{f}_p})(f_{obser,p} - \overline{f_{obser,p}})}{\sqrt{\sum_p (\bar{f}_p - \overline{\bar{f}_p})^2}\sqrt{\sum_p (f_{obser,p} - \overline{f_{obser,p}})^2}} \tag{23}$$

where $\overline{\bar{f}_p}$ means to average in space. In deployment, we flatten the prediction and observation then use the *corrcoef function* from the *NumPy* library.

3.3 Critical Success Index

The Critical Success Index (CSI) is a statistical measure that quantifies the accuracy of spatial prediction by evaluating the correct identification of specific events or outcomes.

The CSI is defined as the ratio of true positives (TP) to the sum of true positives, false positives (FP), and false negatives (FN). Mathematically, it is expressed as,

$$CSI = \frac{TP}{TP+FP+FN} \tag{24}$$

- TP represents the number of true positive outcomes, which signifies the accurate prediction of events or occurrences.
- FP corresponds to false positives, indicating instances where the event was predicted, but did not materialize.
- FN denotes false negatives, signifying cases where the event occurred but was not correctly predicted.

The CSI values range between 0 and 1, where a CSI of 1 indicates perfect spatial accuracy in prediction, implying that all positive outcomes were correctly forecasted without any false alarms. Conversely, a CSI of 0 suggests that none of the events were accurately predicted.

3.4 Continuous Ranked Probability Score

CRPS is used to evaluate the calibration and sharpness. It quantifies the discrepancy between the forecasted cumulative distribution function (CDF) and the observed CDF, defined as,

$$CRPS = \int_{-\infty}^{+\infty} [F(f_m) - 1(t \le z)]^2 dz \tag{25}$$

where F denotes the CDF of the prediction distribution and $1(t \le z)$ is an indicator function that is 1 if $t \le z$ and 0 otherwise. In the case of a deterministic forecast (like Unet) the CRPS reduces to the mean absolute error (MAE).

3.5 Spread-skill ratio

The SSR evaluates the reliability of the ensemble. It is a ratio that quantifies the balance between calibration and sharpness, providing insights into the trade-off between these two critical aspects of predictive modeling.

$$SSR = \frac{Spread}{RMSE} \tag{26}$$

where the spread is defined as,

$$Spread = \sqrt{\frac{1}{P}\sum_{p=1}^{P} Var(f_{m,p})} \tag{27}$$

and the RMSE is defined as,

$$RMSE = \sqrt{\frac{1}{P}\sum_{p=1}^{P} (\bar{f} - f_{obser})^2} \tag{28}$$

4 Additional results

4.1 Skill evaluation

Figure S1 includes PySTEPS metrics calculated over the entire test dataset. Due to UNet's blurred predictions, it falls short of PySTEPS in terms of CSI8.

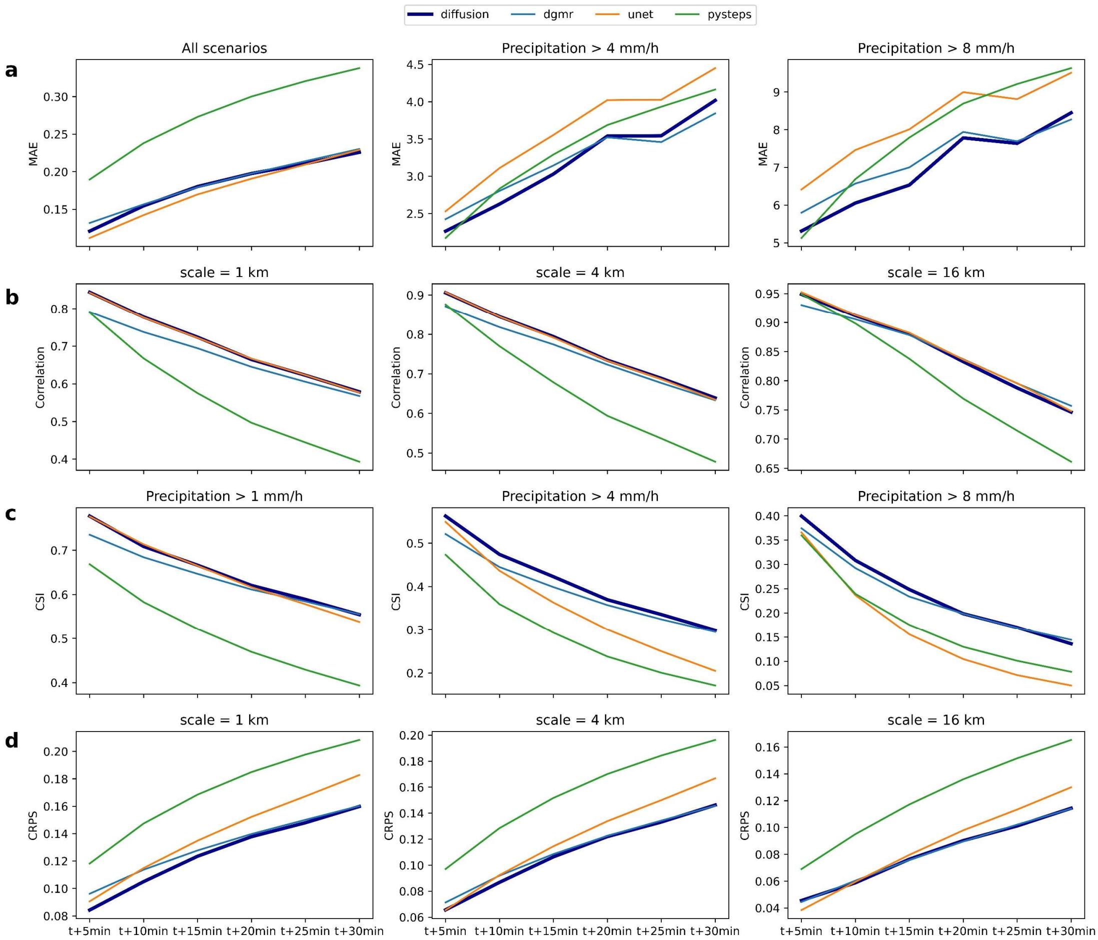

Figure S1. The full version of Figure 2, incorporating metrics for PySTEPS.

4.2 Additional case

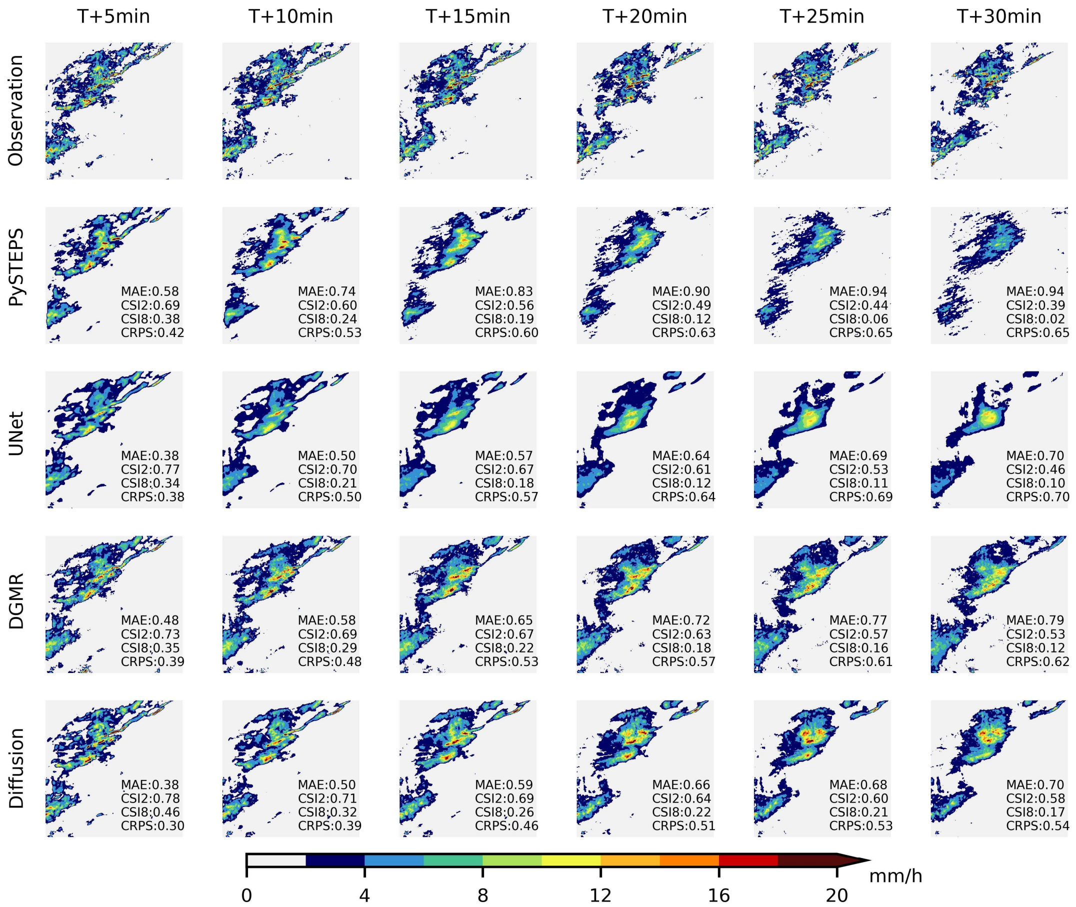

Figure S2. An additional case in Section 5.1

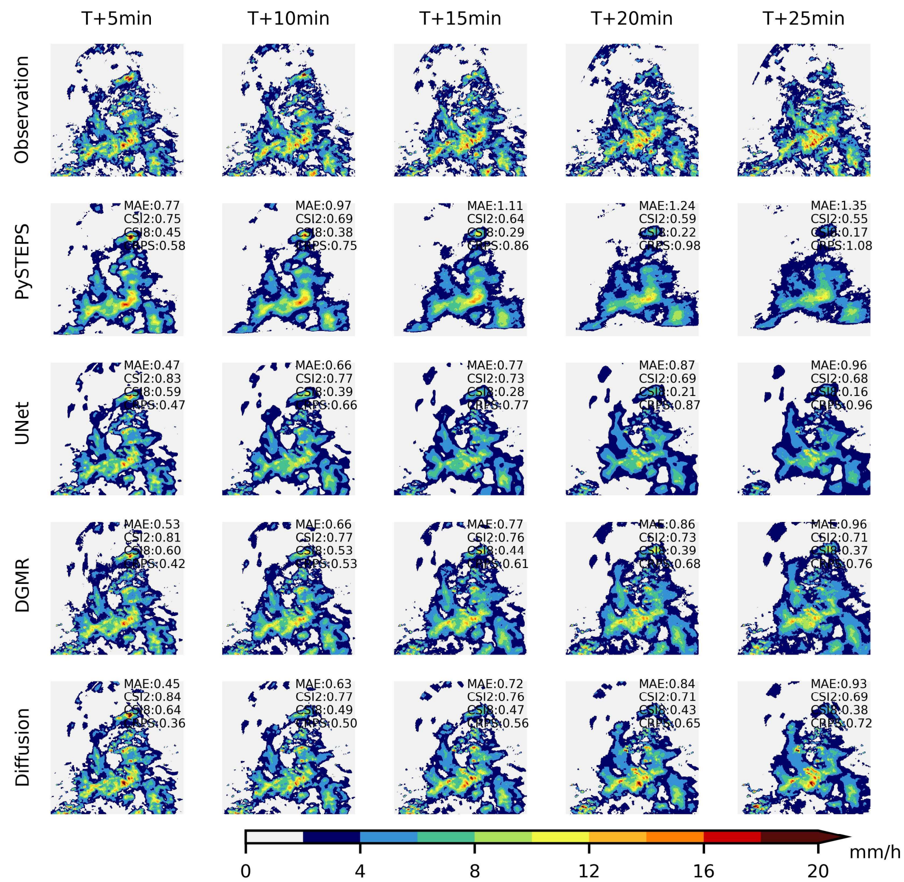

Figure S3. An additional case in Section 5.1

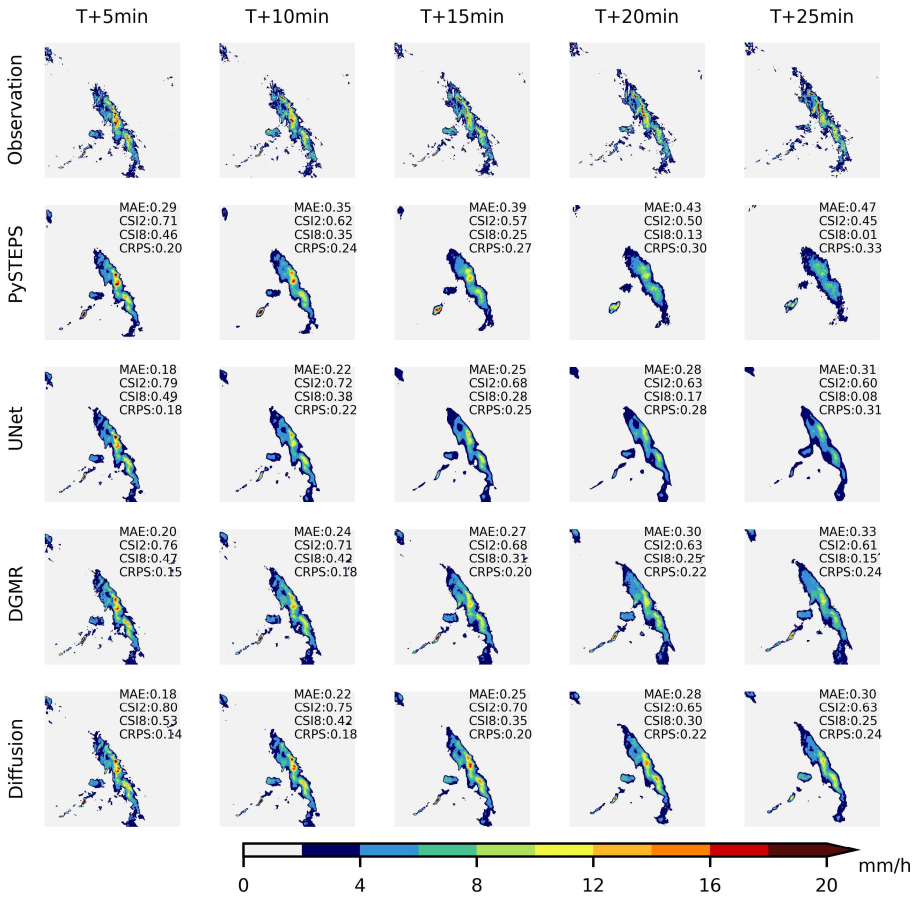

Figure S4. An additional case in Section 5.1

4.3 Reliability cases

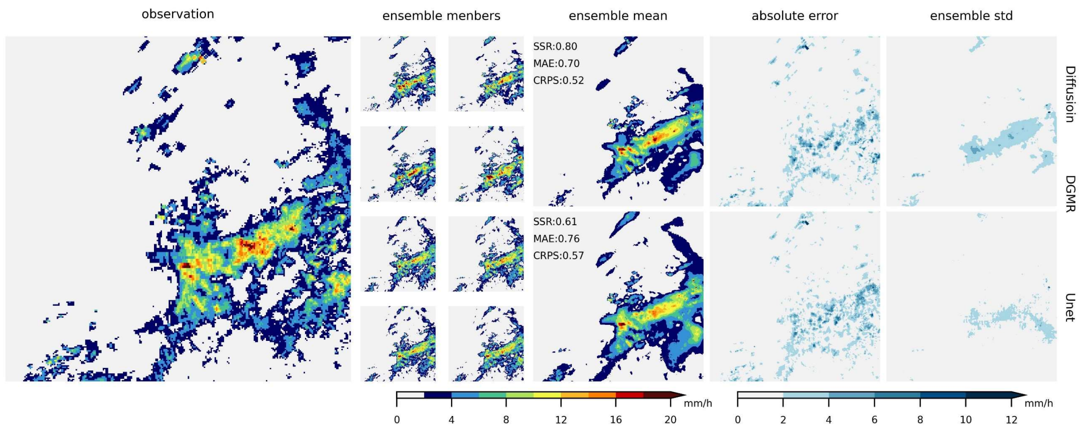

Figure S5. An additional case in Section 5.3

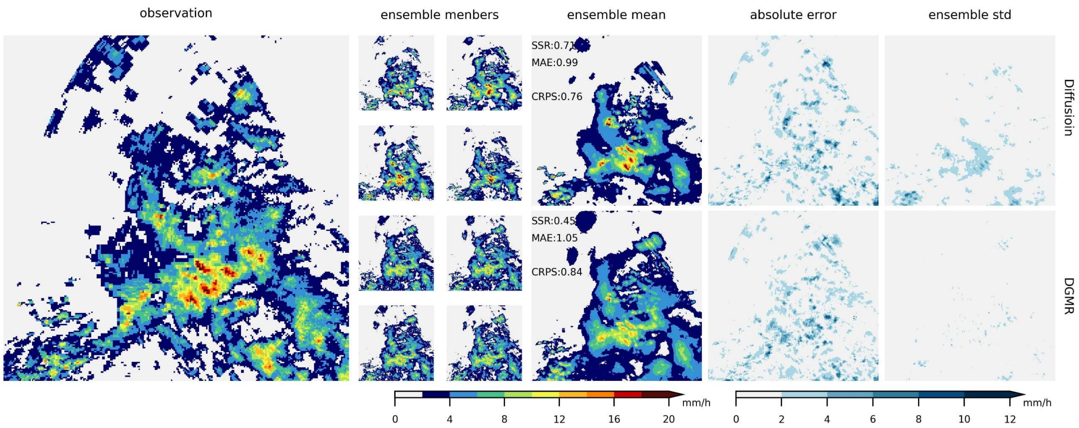

Figure S6. An additional case in Section 5.3

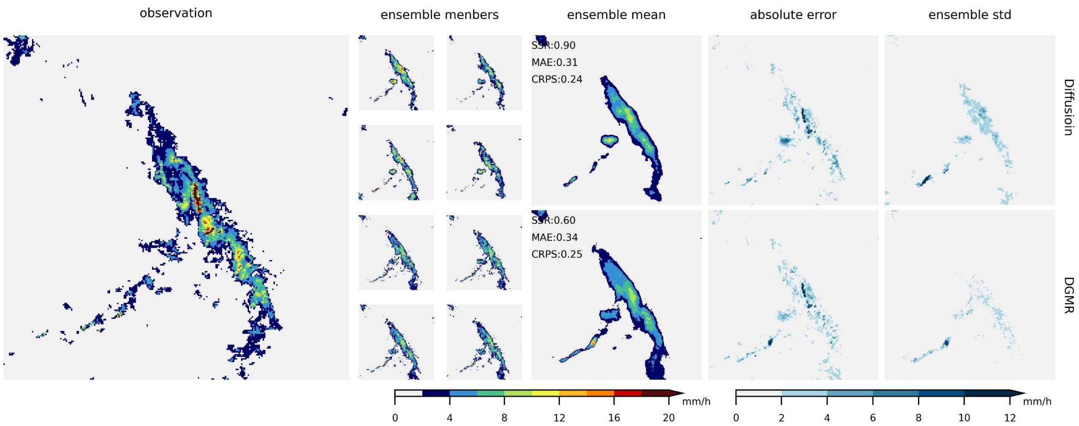

Figure S7. An additional case in Section 5.3

Reference

Vikram Voleti, Alexia Jolicoeur-Martineau and Christopher Pal (2022). MCVD: Masked Conditional Video Diffusion for Prediction, Generation, and Interpolation. https://doi.org/10.48550/arXiv.2205.09853

Understanding diffusion models: A unified perspective." arXiv preprint arXiv:2208.11970 (2022). https://doi.org/10.48550/arXiv.2208.11970

Yang, L., Zhang, Z., Song, Y., Hong, S., Xu, R., Zhao, Y., ... & Yang, M. H. (2022). Diffusion models: A comprehensive survey of methods and applications. ACM Computing Surveys. https://doi.org/10.48550/arXiv.2209.00796