Regional Climate Model Emulation Using Diffusion models

Auteur : Temoschenko, Maxime
Promoteur(s) : Louppe, Gilles
Faculté : Faculté des Sciences appliquées
Diplôme : Master : ingénieur civil électricien, à finalité spécialisée en "electronic systems and devices"
Année académique : 2024-2025
URI/URL : http://hdl.handle.net/2268.2/24763

University of Liège
School of Engineering and Computer Science

Regional Climate Model Emulation Using Diffusion models

Master's Thesis Conducted by

Maxime Temoschenko

With the aim of obtaining the degree of
Master of Science in Electrical Engineering

Under the supervision of
Pr. Gilles Louppe

Academic Year 2024-2025

Acknowledgments

I would like to thank my supervisor, Prof. Gilles Louppe, for his guidance and valuable advice throughout the completion of this thesis.
I would also like to thank Prof. Xavier Fettweis and the members of the SAIL team for their insightful discussions and support with the model implementation.
Finally, I am grateful to my family and friends for their unwavering support.

Contents

Abstract

Regional Climate Models (RCMs) are essential for producing high-resolution climate simulations tailored to specific regions. However, their reliance on discretized physical laws and boundary conditions from Global Climate Models (GCMs) limits flexibility for assimilating new observations and makes long-term simulations computationally expensive, reducing the ability to explore multiple possible climate states from the same GCM.

This thesis investigates the application of *Score-Based Data Assimilation* (SDA) ,reyling on diffusion models, in the context of the *Modèle Atmosphérique Régional* (MAR) over Belgium. The problem is framed from a stratistical perspective as an inverse problem.

The approach consists of (1) Training a diffusion model to approximate the prior distribution of MAR outputs, (2) Exploiting zero-shot conditional generation to incorportate synthetic and real-world observations.

A proof-of-concept is demonstrated training on two variables : the temperature and wind speed, presenting the capabilities of the model to generate samples aligned with the observations in various contexts including incorporating IRM weather stations and generate samples from the GCM ERA-5 land-surface variables given at the boundaries, replicating the simulation of MAR but for two variables.

Several limitations are acknowledged: the learned prior contains artefacts, posteriors conditioned on part of the observations fail to predict other unassimilated parts, the model has not yet been successfully scaled to all MAR variables or to a full simulation with all prognostic variables across pressure levels, the generated ensembles are not calibrated, and computational gains over traditional simulations are limited.

Introduction

Context

Regional Climate Models (RCMs) play an essential role in climate science by providing high-resolution simulations (on the order of a few kilometers) of climate processes over specific domains. To run these simulations, RCMs are typically forced at their boundaries by Global Climate Models (GCMs), which capture large-scale atmospheric circulation but operate at much coarser resolutions (tens to hundreds of kilometers). Through dynamic downscaling and regionally tailored land-surface parameterizations, RCMs refine GCM outputs to finer spatial scales, enabling the study of regional phenomena such as topography-driven precipitation, local extreme weather events, or snowpack evolution. These fine-scale simulations are crucial for impact studies in sectors such as agriculture, water management, and renewable energy production, where local climate conditions directly determine outcomes.

Problem Statement

Despite their importance, RCMs present significant limitations. Simulating discretized physical laws on fine spatial grids with small timesteps is computationally expensive, requiring substantial time and resources to produce long climatological runs. Furthermore, RCMs are inherently constrained and biased by their boundary conditions: incorporating new observations (e.g., from weather stations) within the domain, applying the model to new regions, or forcing it with another GCM requires additional costly simulations. This lack of flexibility contrasts with the growing need for adaptive, data-driven approaches that can assimilate diverse and evolving sources of information.

In addition, the high computational cost of physics-based simulations restricts the size of ensembles that can be produced. Consequently, only a limited number of trajectories are typically available, which fails to capture the full diversity of climate states consistent with a given set of observations.

Recent advances in deep learning, and in particular generative diffusion models, open the door to a new paradigm for climate simulation. Instead of explicitly solving physical equations, these models can learn the prior distribution of climate states directly from RCM outputs. Once such a prior is learned, it can be leveraged in zero-shot settings, enabling tasks on unseen scenarios without the need for retraining.

Objectives

This thesis investigates the potential of generative diffusion models as a flexible and efficient alternative to traditional physics-based RCM simulations. The specific objectives are:

1. Train a diffusion model to approximate the prior distribution of RCM outputs.
2. Investigate the ability of model to incorporate alternative forcings and real-world observations in a zero-shot manner from the learned prior.

3. Demonstrate, as a proof of concept, the emulation of RCM simulations from coarser GCM boundary conditions in a simplified setting.

4. Assess the ability of diffusion models to exploit their stochastic nature in order to generate large, diverse ensembles at a fraction of the computational cost of traditional RCMs.

Chapter 1

Diffusion Model

1.1 DDPM : Denoising Diffusion Probabilistic Models

The core idea behind Denoising Diffusion Probabilistic Models (DDPMs) is to gradually add Gaussian white noise to an image over a fixed number of steps, until the image becomes indistinguishable from pure noise (forward process). This process establishes pairs of (noisy image, original image) that are used to train a neural network to reverse the forward process.That is, to predict and recover the original image from its noisy counterpart (backward process). Once trained, the model can generate new samples by starting from random Gaussian noise and iteratively applying the learned denoising steps in reverse.

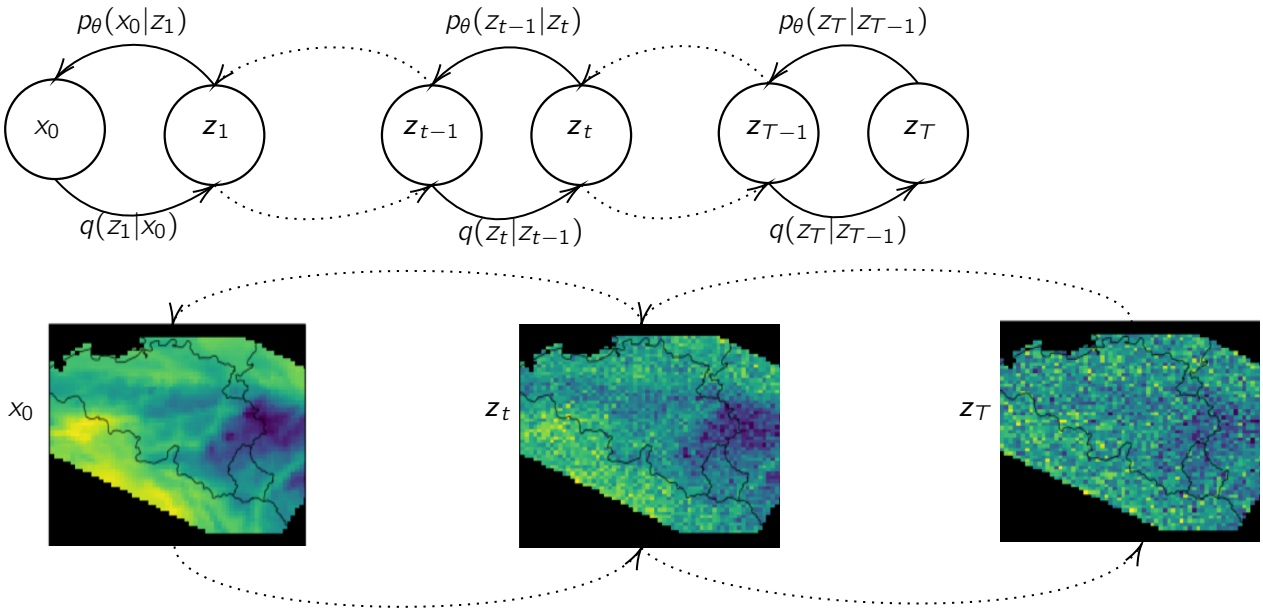


Figure 1.1: Illustration of the DDPM forward and backward processes. In the forward process, data is progressively corrupted by adding Gaussian noise at each step. In the backward process, the model learns to reconstruct the original data from the noisy version.

1.1.1 Forward Process

In the forward process, the latent variables scale down the previous variable and add progressively Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$, with $\{\beta_i\}_{i=1}^{T} \in [0, 1]$ as hyperparameters

$$\begin{cases} z_1 = \sqrt{1-\beta_1}x_0 + \sqrt{\beta_1}\epsilon_1, \text{if} t = 1 \\ z_t = \sqrt{1-\beta_t}z_{t-1} + \sqrt{\beta_t}\epsilon_t, \forall t \in \{2, ..., T\} \end{cases} \tag{1.1}$$

Therefore, the forward transition is defined as :

$$q(z_t|z_{t-1}) \sim \mathcal{N}(\sqrt{1-\beta_t}z_{t-1}, \beta_t I_t)$$

Since the diffusion process is defined as a first-order Markov chain, one obtains :

$$q(z_{1...T}|x) = q(z_1|x)\prod_{t=2}^{T} q(z_t|z_{t-1})$$

By chain substitution and defining $\alpha_t \doteq \prod_{i=1}^{T} 1-\beta_i$, the transition from the original distribution to the latent variable t is derived :

$$z_T = \sqrt{\alpha_t}x_0 + (1-\alpha_t)\epsilon$$

$$q(z_t|x_0) \sim \mathcal{N}(\sqrt{\alpha_t}x_0, (1-\alpha_t)I)$$

1.1.2 Backward process

The reconstruction of the original data from the Gaussian noise would be possible if $q(z_{t-1}|z_t)$ was known. However, this distribution is often not tractable. Therefore, the conditional backward distributions are learned to approximate $q(z_{t-1}|z_t)$.
Assuming the distribution at the end of the diffusion process is a standard normal distribution (pure noise) and that the learnable reverse distribution between following latent variables is approximated by a normal distribution :

$$\begin{cases} p_\theta(z_T) \sim \mathcal{N}(0, I) \\ p_\theta(z_{t-1}|z_t) \sim \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \end{cases}$$

However, the backward process conditioned on the original distribution is tractable :

$$\begin{aligned} q(z_{t-1}|z_t, x_0) &= \frac{q(z_t|z_{t-1}, x_0)q(z_{t-1}|x_0)}{q(z_t|x_0)} \quad \text{(Baye's rule)} \\ &\sim \mathcal{N}(\tilde{\mu}(z_t|x_0), \tilde{\beta}_t I) \quad \text{(Product of Gaussian random variables)} \end{aligned} \tag{1.2}$$

$$\begin{cases} \tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}\left(z_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\epsilon_t}\right) \quad \text{(Derivation from (1.2))} \\ \mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(z_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t, t)}\right) \quad \text{(Reparametrization of the target)} \end{cases} \tag{1.3}$$

Maximum Likelihood Estimation Given a training dataset $\{x^i\}_{i=1}^N$, the optimal parameters could be approximated by maximizing the probability of the dataset given a parameter combination. The problem can be equally formulated as the the search of the parameter in the parameter space maximizing the log-likelihood of the dataset.

$$\hat{\theta} = \text{argmax}_{\theta\in\Theta} \sum_{i=1}^{N} log(p_\theta({x_0}^i)|\theta)$$

The original distribution is marginalized over the latent space as:

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}, \boldsymbol{z}_{1\dots T}) d\boldsymbol{z}_{1\dots T} = \int p_\theta(\boldsymbol{z}_T) p_\theta(\boldsymbol{x}_0|\boldsymbol{z}_1) \prod_{t=2}^{T} p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t) d\boldsymbol{z}_{1\dots T}$$

In practice, this integral is intractable since it requires to integrate over a high-dimensional latent steps across all diffusion steps $t = 1, ..., T$. In fact, the number of possible trajectories grows exponentially as the number of steps increases. In conclusion, training a diffusion model based on the maximum likelihood estimation is infeasible.

Tractable approximation Defining a loss function as minimizing the opposite of the log probability of the dataset,

$$\begin{aligned}
-\log(p_\theta(\boldsymbol{x}_0)) &= -\log\left(\int p_\theta(\boldsymbol{x}, \boldsymbol{z}_{1:T})\, d\boldsymbol{z}_{1:T}\right) \quad \text{(marginalization over latent variables)} \\
&= -\log\left[\int q(\boldsymbol{z}_{1:T}|\boldsymbol{x}) \frac{p_\theta(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q(\boldsymbol{z}_{1:T}|\boldsymbol{x})}\, d\boldsymbol{z}_{1:T}\right] \quad \text{(artifice)} \\
&\geq -\int q(\boldsymbol{z}_{1:T}|\boldsymbol{x}) \log\left[\frac{p_\theta(\boldsymbol{x}, \boldsymbol{z}_{1:T})}{q(\boldsymbol{z}_{1:T}|\boldsymbol{x})}\right] d\boldsymbol{z}_{1:T} \quad \text{(Jensen's inequality)} \\
&:= \mathcal{L}_{VLB} \qquad (1.4)
\end{aligned}$$

Instead of minimizing the opposite of the log-likelihood, one searches to minimize a lower bound of the opposite of the log-likelihood.
At the end, the loss is decomposed as :

$$\mathcal{L}_{\text{VLB}} = \mathbb{E}_{q(\boldsymbol{z}_{1\dots T})}\left[-\log p_\theta(\boldsymbol{x}_0|\boldsymbol{z}_1) + \sum_{t=2}^{T} \mathrm{D}_{\text{KL}}\left(q(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t, \boldsymbol{x}_0) \,\|\, p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)\right) + \mathrm{D}_{\text{KL}}\left(q(\boldsymbol{z}_T|\boldsymbol{x}_0) \,\|\, p_\theta(\boldsymbol{z}_T)\right)\right] \tag{1.5}$$

$$= \mathbb{E}_{q(\boldsymbol{z}_{1\dots T})}\left[L_0 + \sum_{t=2}^{T} L_{t-1} + L_T\right] \tag{1.6}$$

Noting that L_T does not rely on the parameters (both distributions of the KL-divergence are parameter independent), the latter is ignored since it is a constant term.

Loss Function For L_t, the KL-divergence of two normal distributions is known and by reparametrization (1.3) with the noise, the equation simplifies as :

$$L_t = \mathbb{E}_{\boldsymbol{x}_0, \epsilon}\left[\frac{1}{2||\Sigma_{\boldsymbol{\theta}}||_2^2} ||\boldsymbol{\mu}_t(\boldsymbol{x}_t, \boldsymbol{x}_0) - \boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)||^2\right] \tag{1.7}$$

$$= \mathbb{E}_{\boldsymbol{x}_0, \epsilon}\left[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t||\Sigma_{\boldsymbol{\theta}}||_2^2)} ||\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{(}\bar{\alpha}_t \boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)||^2\right] \tag{1.8}$$

Using a simplified loss function has shown improved results in experiments. In this approach, the timestep t is sampled uniformly from the interval $[1, T]$, and the loss simplifies to the mean squared error between the predicted and true noise at that specific timestep t.

$$\mathcal{L}_{DDPM} := \mathbb{E}_{t\sim[1,T], \boldsymbol{x}_0, \boldsymbol{\epsilon}_t}\left[||\boldsymbol{\epsilon}_t - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}_t, t)||^2\right] \tag{1.9}$$

Training and Sampling Algorithms

Algorithm 1 Training $\epsilon_\theta(\mathbf{x}, t)$

Input: Data distribution $q(\mathbf{x}_0)$, network ϵ_θ

for $i = 1$ to N **do**
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
  $t \sim \mathcal{U}(\{0, 1, ..., T\})$
  $\epsilon \sim \mathcal{N}(0, I)$
  $\mathbf{x}_t \leftarrow \mu(t)\mathbf{x}_0 + \sigma(t)\epsilon$ {Diffusion process}
  $\ell \leftarrow \|\epsilon - \epsilon_\theta(\mathbf{x}_t, t)\|_2^2$
  $\theta \leftarrow \text{GradientDescent}(\theta, \nabla_\theta \ell)$
end for

Algorithm 2 Sampling $s_\theta(\mathbf{x}, t)$ [2]

$\mathbf{x} \sim \mathcal{N}(0, I)$ {Initialize noise}
$T \leftarrow$ total steps
for $t = T$ **downto** 1 **do**
  $\mathbf{z} \sim \mathcal{N}(0, I)$ **if** $k > 1$ **else** 0
  $\mathbf{x}_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_k}}\left(\mathbf{x}_t - \frac{1-\alpha_k}{\sqrt{1-\bar{\alpha}_k}}\epsilon_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
end for
return $\mathbf{x}$ {Generated sample}

1.2 Score-based generative model

Another approach to diffusion model is based on score-matching.

Problem Let the data distribution $p(x)$, an unknown distribution and some available samples $\{\boldsymbol{x}\}_{i=1}^{N}$, the goal of generative model is to compute the best approximation data distribution $p_\theta(x)$ of the original data distribution.
The approximation probability density function is defined as :

$$\begin{cases} p_\theta(x) := \dfrac{e^{-f_\theta(\boldsymbol{x})}}{Z_\theta} \\ Z_\theta = \int_{x\in\mathcal{X}} e^{-f_\theta(\boldsymbol{x})} d\boldsymbol{x} \end{cases} \tag{1.10}$$

The normalizing denominator term Z_θ computation requires to integrate over the entire space of $\boldsymbol{x}$ which is intractable.

Score Function One way to get rid of this normalizing constant is to introduce the score function. Intuitively, the score function defines the direction in which the data point has to be pushed in order to increase the data point log probability.
Figure 1.2 illustrates the contour lines of the probability density for a mixture of two Gaussian distributions. Lighter colors indicate regions of higher density. The overlaid arrows represent the score function, which points in the direction of the gradient of the log-density. The length of each arrow corresponds to the magnitude of the score, with longer arrows indicating regions of steeper gradients.

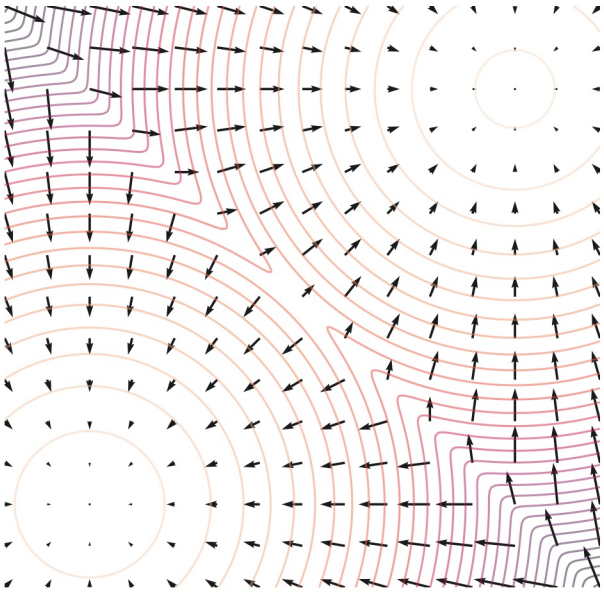


Figure 1.2: Illustration of the score function over the density of a mixture of two Gaussians [3]

$$\begin{aligned} s_\theta(\boldsymbol{x}) &:= \nabla_{\boldsymbol{x}} \log p_\theta(\boldsymbol{x}) \\ &= \nabla_{\boldsymbol{x}} \log \left(\frac{e^{-f_\theta(\boldsymbol{x})}}{Z_\theta} \right) \\ &= \nabla_{\boldsymbol{x}} \left(-f_\theta(\boldsymbol{x}) \right) \end{aligned} \tag{1.11}$$

The idea is that instead of maximizing the likelihood, one searches to minimize the fisher divergence between the score function of the learnable distribution and the original data distribution :

$$\hat{\theta}_{\text{fisher}} = \arg\min_{\theta\in\Theta} \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x})}\left[\|\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) - \boldsymbol{s}_\theta(\boldsymbol{x})\|_2^2\right] \tag{1.12}$$

$$= \arg\min_{\theta\in\Theta} \mathcal{L}_{\text{fisher}}(\theta) \tag{1.13}$$

However, the score function of the original distribution is still unknown since it relies on the original distribution itself.
After some derivations [4], the search of optimal parameter is rewritten as

$$\hat{\theta} = \arg\min_{\theta\in\Theta} \frac{1}{2}\mathbb{E}_{p(\boldsymbol{x})}\left[\text{tr}(\nabla_{\boldsymbol{x}} s_\theta(\boldsymbol{x})) + \frac{1}{2}\|s_\theta(\boldsymbol{x})\|_2^2\right] \tag{1.14}$$

Assuming $\boldsymbol{x} \in \mathbb{R}^d$ and $s_\theta(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}^d$, The first term $tr(\nabla_{\boldsymbol{x}} s_\theta(\boldsymbol{x})) = \sum_{i=1}^d \frac{\partial s_{\theta,i}}{\partial x_i}$ is evaluated using a backward pass through the score function estimator $s_\theta(\boldsymbol{x})$ (typically a neural network) for each component of the input vector $\boldsymbol{x}$. This process is computationally expensive when the input vector is high-dimensional (such as climatology variables).

Sampling The primary motivation for employing the score function lies in its role within a sampling strategy known as Langevin Dynamics. This method begins by drawing an initial sample from a prior distribution and then iteratively refines this sample toward regions of higher data likelihood. The refinement is guided by an estimate of the score function, which approximates the gradient of the data log-probability. To maintain stochasticity and encourage exploration, random noise is injected at each iteration

$$\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \epsilon \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) + \sqrt{2\epsilon}\boldsymbol{z}_i$$

Here, $\boldsymbol{z}_i \sim \mathcal{N}(0, \mathbb{I})$ introduces Gaussian noise at each iteration, while ϵ is a hyperparameter that controls the step size magnitude.
In our case, $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$ is approximated by the neural network $\boldsymbol{s}_\theta(\boldsymbol{x})$.

Challenge of score-based generative models There are two main limitations associated with Equation (1.14). The first, previously discussed, concerns the high computational cost of one of its terms. The second limitation arises in low-density regions of the data distribution. In these areas, the scarcity of data points hinders the accurate approximation of the score function. Consequently, if the initial sample in the Langevin Dynamics procedure falls within such a low-density region, which is likely when sampling from the prior, the poorly estimated score may misguide the sampling trajectory. This can cause the Langevin Dynamics to deviate significantly from the true data manifold, ultimately degrading the quality of the generated samples.

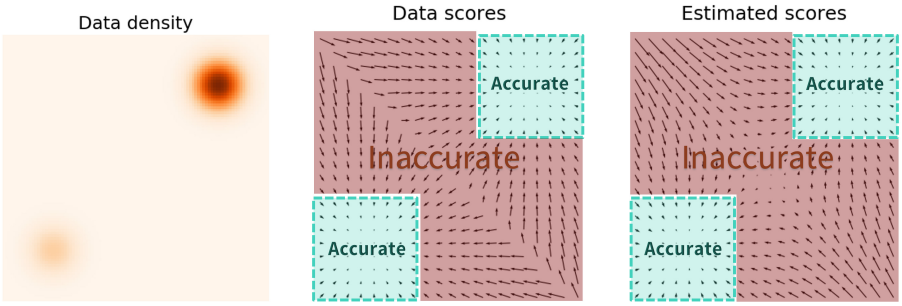


Figure 1.3: Illustration of the limitations of score estimation in low-density regions. [3]**Left:** Ground-truth data density, where darker regions indicate higher density. **Center:** True score function (i.e., the gradient of the log-density). **Right:** Estimated score function obtained from a learned model. The estimated score is only reliable near the modes of the data distribution, where sufficient training samples are available. In low-density regions, the approximation is poor and can mislead sampling algorithms such as Langevin Dynamics.

1.3 Noise conditional Score Network

To address the challenges posed by low-density regions, one strategy is to perturb the original data distribution by adding noise. This has the effect of populating low-density areas with more samples, making it easier to estimate the score function. This approach is effective because score matching losses are weighted by the data distribution, which down-weights errors in regions with few samples.

However, this introduces a trade-off: using a low noise level preserves the structure of the original distribution but results in poor score estimation in sparse regions. Conversely, high noise levels enable more accurate score estimation in these regions, but at the cost of distorting the underlying data distribution.

To balance these extremes, a multi-scale approach is adopted, in which the data is perturbed using several noise levels.

Discretized noise level scale The data distribution is perturbed using a set of N increasing Gaussian noise levels $\{\sigma_i\}_{i=1}^{N}$. These noise levels are selected such that the smallest one mitigates issues in low-density regions, while the largest remains sufficiently close to the original distribution. For each noise level σ_i, the perturbed distribution is defined as:

$$p_{\sigma_i}(\tilde{\boldsymbol{x}}) = \int p(\boldsymbol{x})\, \mathcal{N}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}, \sigma_i^2 \mathbb{I})\, d\boldsymbol{x}.$$

In practice, samples from this distribution are obtained using:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} + \sigma_i \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbb{I}) \tag{1.15}$$

with $\boldsymbol{x}$ drawn from the original data distribution. The goal is to train a neural network $\boldsymbol{s}_\theta(\tilde{\boldsymbol{x}}, \sigma_i)$ to approximate the score function of each perturbed distribution:

$$\boldsymbol{s}_\theta(\tilde{\boldsymbol{x}}, \sigma_i) \approx \nabla_{\tilde{\boldsymbol{x}}} \log p_{\sigma_i}(\tilde{\boldsymbol{x}}).$$

The previous objective (1.14) is equivalent to the **Denoising Score Matching (DSM)** objective [5], provided that the noise levels σ_i are sufficiently small.

$$\mathcal{L}_{\text{DSM}}(\theta, q_{\sigma_i}) = \frac{1}{2}\mathbb{E}_{p(\mathrm{x})}\mathbb{E}_{q_\sigma(\tilde{\mathrm{x}}|\mathrm{x})}\left[\left\| s_\theta(\tilde{\mathbf{x}}, \sigma_i) + \frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma_i^2}\right\|_2^2\right],$$

where the perturbation kernel is Gaussian:

$$q_{\sigma_i}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) = \mathcal{N}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}, \sigma_i^2 \mathbb{I}), \quad \text{and thus} \quad \nabla_{\tilde{\mathrm{x}}} \log q_{\sigma_i}(\tilde{\mathbf{x}} \mid \mathbf{x}) = -\frac{\tilde{\mathbf{x}} - \mathbf{x}}{\sigma_i^2}.$$

To integrate information from all noise scales, the total loss is computed as a weighted sum:

$$\mathcal{L}(\theta; \{\sigma_i\}_{i=1}^N) := \frac{1}{N}\sum_{i=1}^{N} \gamma(\sigma_i)\, \mathcal{L}_{\text{DSM}}(\theta, q_{\sigma_i}) \tag{1.16}$$

where $\gamma(\sigma_i)$ denotes the weight associated with each noise level [3].

1.4 Score-Based Generative Modeling via SDEs [1]

Perturbing the original data distribution using a continuous spectrum of noise scales gives rise to a stochastic process, i.e., a system that evolves over time with inherent randomness.
As in previous settings, samples $\mathbf{x}_0$ are drawn from the unknown data distribution $p(\mathbf{x}_0)$. A diffusion process is then defined over time: for all $t \in [0, T]$, the variable $\mathbf{x}_t$ evolves such that at the final time T, $\mathbf{x}_T$ follows a tractable prior distribution (e.g., a Gaussian), from which sampling becomes straightforward [1].
This time-dependent evolution is described by a Stochastic Differential Equation (SDE), commonly formulated as:

$$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\,\mathrm{d}t + g(t)\,\mathrm{d}\mathbf{w} \tag{1.17}$$

where:

1. $\mathbf{f}(\mathbf{x}, t)$ is the *drift coefficient*, representing the deterministic part of the system's evolution over time.
2. $g(t)$ is the *diffusion coefficient*, which controls the intensity of the randomness at time t.
3. $\mathrm{d}\mathbf{w}$ denotes an infinitesimal increment of a Wiener process (or Brownian motion), introducing stochasticity into the evolution of $\mathbf{x}_t$.

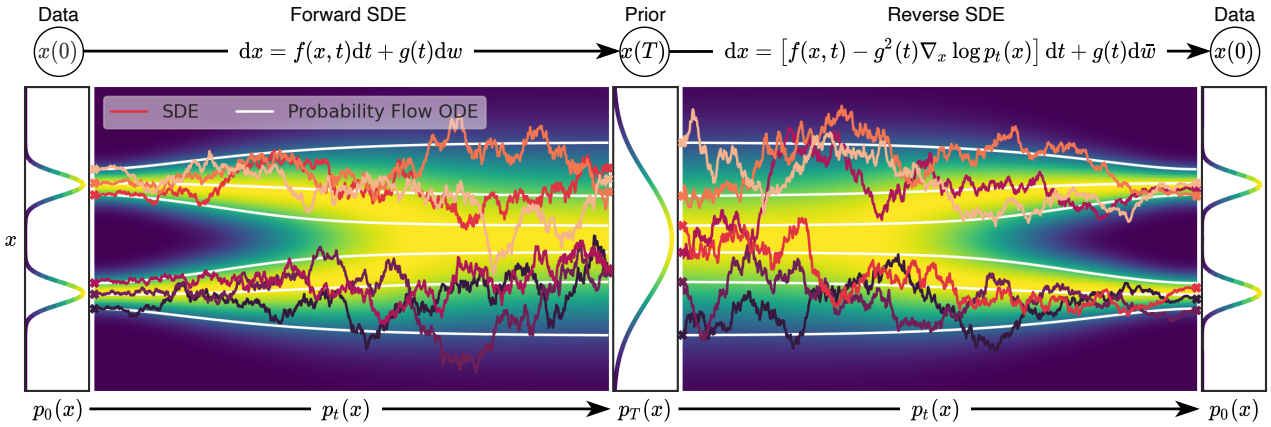


Figure 1.4: **Left**: original data distribution . A stochastic process (forward SDE) maps this distribution to a prior(**middle**), typically Gaussian noise. The reverse SDE then maps this prior back to the data distribution (**right**), enabling generation of new samples. [1]

Reverse SDE To generate new samples, one starts by sampling from the prior distribution and simulates the time-reversed SDE. This is possible because the reverse-time SDE admits a closed-form expression [6]:

$$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathrm{x}} \log p_t(\mathbf{x})\right] \mathrm{d}t + g(t)\, \mathrm{d}\mathbf{w} \tag{1.18}$$

In practice, the true score function $\nabla_{\mathrm{x}} \log p_t(\mathbf{x})$ is not accessible, so a neural network is trained to approximate it : $\mathbf{s}_\theta(\boldsymbol{x}, t) \approx \nabla_{\mathrm{x}} \log p_t(\mathbf{x})$, similarly to the previous score-based formulations. Unlike the discretized setting, the noise level here varies continuously , and the model must therefore learn to estimate the score function for all $t \in [0, T]$.

Estimation of the score function is based on a continuous generalization of (1.16), it is a weighted sum of Fisher loss divergence (1.12).

$$\mathcal{L}_{score} = \mathbb{E}_t \left\{\lambda(t)\mathbb{E}_{\mathrm{x}(0)}\mathbb{E}_{\mathrm{x}(t)|\mathrm{x}(0)} \left[\|\mathbf{s}_\theta(\mathbf{x}(t), t) - \nabla_{\mathrm{x}(t)} \log p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))\|_2^2\right]\right\} \tag{1.19}$$

This objective is a continuous weighted combination of Fisher divergences, where $t \sim \mathcal{U}(0, T)$ is sampled uniformly over time, and $\lambda(t) > 0$ is a positive weighting function. In practice, $\lambda(t)$ is often set inversely proportional to the expected norm of the score, i.e.,

$$\lambda(t) \propto \left(\mathbb{E}\left\|\nabla_{\mathrm{x}(t)} \log p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))\right\|_2^2\right)^{-1},$$

to ensure balanced contributions from different noise levels.

To train this objective, we require access to the transition kernel $p_{0t}(\mathbf{x}(t)|\mathbf{x}(0))$. When the forward SDE has affine drift and additive noise, this kernel is Gaussian and has closed-form expressions for its mean and variance. In more general settings, this kernel can be estimated either by solving the Kolmogorov forward equation or by simulating the forward SDE and applying alternative score matching techniques, such as sliced score matching or finite-difference score matching, that do not require computing the exact gradient of the log transition density [1].

Once the score-based model $\mathbf{s}_\theta(\mathbf{x}(t), t)$ is trained to approximate $\nabla_{\mathrm{x}(t)} \log p_t(\mathbf{x}(t))$, it can be directly plugged into the reverse-time SDE to simulate data samples. These samples approximate the original data distribution when the model is well-trained.

Connection with DDPM and NCSN Yang Song, the father of score-based generative modeling through SDE [1] , demonstrates that its framework is a generalization of DDPM (1.1) [2] and NCNS (1.3) [3] in the continuous case.

1. Variance Exploding (VE) SDE
   From NCNS, There are N discretized noise scales $\{\sigma_i\}_{i=1}^N$ (1.3) is rewritten as :

   $$\forall i \in \{1, ..., N\} : \mathbf{x}_i = \mathbf{x}_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2}\epsilon \tag{1.20}$$

   As N converges to the infinity, previous equations becomes an SDE :

   $$d\mathbf{x} = \sqrt{\frac{d\sigma^2(t)}{dt}} d\mathbf{w} \tag{1.21}$$

2. Variance Preserving (VP) SDE
   For DDPM 1.1, As N converges to the infinity. (1.1) becomes :

   $$d\mathbf{z} = -\frac{1}{2}\beta(t)\mathbf{z}dt + \sqrt{\beta(t)}d\mathbf{w} \tag{1.22}$$

Secondly, in a recent work [7], the authors emphasize that, as previously mentioned, diffusion models currently represent the state-of-the-art in generative artificial intelligence. They outperform alternative approaches such as Generative Adversarial Networks (GANs) and Normalizing Flows, not only in generation quality but also in training stability, as they are generally less sensitive to hyperparameter choices.

However, this performance comes at a cost: sampling speed. As discussed in the previous frameworks, generating a single sample typically requires hundreds or even thousands of neural network evaluations. For instance, in DDPM, up to 1000 steps may be required [2], which significantly slows down the generation process. Consequently, a major line of current research focuses on accelerating the sampling process.

In their work, Zhang et al. [7] identify a key limitation of previous approaches: approximating the score function directly can become unstable in regions where the score exhibits high variance, making training and inference less reliable. To address this, they reformulate the SDE loss as a function of the predicted noise rather than the predicted score, offering a more stable and practical training objective. $\epsilon_\phi(x(t), t) = -\sigma(t)\, s_\phi(x(t), t)$

This reformulation leads to a final loss of the form:

$$\mathcal{L}_{\text{score}} = \mathbb{E}_{t\sim\mathcal{U}(0,T)}\, \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\mathcal{I})}\, \mathbb{E}_{\mathbf{x}(0)\sim p(\mathbf{x}(0))} \left[\|\boldsymbol{\epsilon}_\theta(\mathbf{x}(t), t) - \boldsymbol{\epsilon}\|_2^2 \right], \tag{1.23}$$

where $\mathbf{x}(t) = \mathbf{x}(0) + \sigma(t)\boldsymbol{\epsilon}$, and the neural network $\boldsymbol{\epsilon}_\theta$ is trained to recover the noise directly.

This noise prediction loss, closely related to the DDPM objective introduced in (1.9), is adopted as the final training objective in [7] and will also serve as the foundation for the experiments conducted in this thesis.

1.5 Diffusion Posterior Sampling for General Noisy Inverse Problems (DPS)

1.5.1 Context of Inverse Problems

In inverse problems, the objective is not only to sample from the prior distribution $p(\mathbf{x})$, but from the posterior $p(\mathbf{x}|\mathbf{y})$, where $\mathbf{y}$ is an observation. Typically, the forward mapping $\mathbf{x} \to \mathbf{y}$, or equivalently $p(\mathbf{y}|\mathbf{x})$, is known, whereas the inverse mapping is ill-posed or non-injective, hence the term inverse problem.

1.5.2 Direct Approach

A straightforward approach consists in training a neural network $s_\theta(\mathbf{x}(t), t \mid \mathbf{y})$ to approximate the conditional score function $\nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t)|\mathbf{y})$, by directly conditioning on the observation $\mathbf{y}$. However, this method requires retraining the network for each new observation $\mathbf{y}$, or each distinct observation process, which is computationally expensive and inflexible.

1.5.3 Problem Statement

To overcome this, we leverage Bayes' rule:

$$\nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t)|\mathbf{y}) = \nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t)) + \nabla_{\mathbf{x}(t)} \log p(\mathbf{y}|\mathbf{x}(t)). \tag{1.24}$$

The first term is approximated by a pre-trained unconditional score model $s_\theta(\mathbf{x}(t), t) \approx \nabla_{\mathbf{x}(t)} \log p(\mathbf{x}(t))$. The challenge lies in approximating the second term $\nabla_{\mathbf{x}(t)} \log p(\mathbf{y}|\mathbf{x}(t))$, since there is no direct relationship between the noised variable $\mathbf{x}(t)$ and the observation $\mathbf{y}$. This issue is illustrated in Figure 1.5.

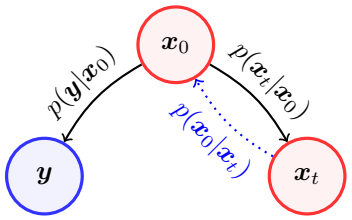


Figure 1.5: Black line: tractable; blue dotted line: intractable . Figure from [8]

Observation Model We assume the following observation model:

$$\mathbf{y} = \mathcal{A}(\mathbf{x}) + \epsilon,$$

where $\mathcal{A}$ is a known measurement function and $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbf{I})$ models Gaussian measurement noise. This yields the tractable likelihood:

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathcal{A}(\mathbf{x}), \sigma^2\mathbf{I}).$$

Approximation Strategy We aim to approximate $\log p(\mathbf{y}|\mathbf{x}(t))$. Applying the law of total probability:

$$p(\mathbf{y}|\mathbf{x}(t)) = \int p(\mathbf{y}|\mathbf{x})\, p(\mathbf{x}|\mathbf{x}(t))\, d\mathbf{x}.$$

This integral is intractable, as it involves marginalizing over all possible clean signals $\mathbf{x}$ that could have generated the noised input $\mathbf{x}(t)$.

Assuming a Gaussian observation model, we use Tweedie's formula to approximate the conditional mean of the posterior $p(\mathbf{x}|\mathbf{x}(t))$:

$$\begin{aligned}\hat{\mathbf{x}}_0(\mathbf{x}(t)) &= \mathbb{E}_{\mathrm{x}\sim p(\mathrm{x}|\mathrm{x}(t))}[\mathbf{x}] \\ &= \frac{\mathbf{x}(t) + \sigma(t)^2 \nabla_{\mathrm{x}(t)} \log p(\mathbf{x}(t))}{\mu(t)} \\ &\approx \frac{\mathbf{x}(t) + \sigma(t)^2 s_\theta(\mathbf{x}(t), t)}{\mu(t)}.\end{aligned}$$

The key approximation made in DPS is to replace the intractable marginal likelihood gradient $\nabla_{\mathrm{x}(t)} \log p(\mathbf{y}|\mathbf{x}(t))$ by the following:

$$\begin{aligned}\nabla_{\mathrm{x}(t)} \log p(\mathbf{y}|\mathbf{x}(t)) &\approx \nabla_{\mathrm{x}(t)} \log p(\mathbf{y}|\hat{\mathbf{x}}_0(\mathbf{x}(t))) \\ &\approx -\frac{1}{\sigma^2} \nabla_{\mathrm{x}(t)} \|\mathbf{y} - \mathcal{A}(\hat{\mathbf{x}}_0(\mathbf{x}(t)))\|_2^2,\end{aligned}$$

where the second line follows from the log-likelihood of a Gaussian.

Final Score Estimate Substituting this approximation back into Equation (1.24), we obtain the final expression for the approximate posterior score:

$$\nabla_{\mathrm{x}(t)} \log p(\mathbf{x}(t)|\mathbf{y}) \approx s_\theta(\mathbf{x}(t), t) - \frac{1}{\sigma^2} \nabla_{\mathrm{x}(t)} \|\mathbf{y} - \mathcal{A}(\hat{\mathbf{x}}_0(\mathbf{x}(t)))\|_2^2. \tag{1.25}$$

This provides a closed-form approximation to the posterior score, allowing sampling from the posterior $p(\mathbf{x}|\mathbf{y})$ without retraining the score model. Instead, inference is conditioned on new observations $\mathbf{y}$ via the second term in Equation (1.25), avoiding the need to retrain a conditional model for each inverse problem instance [8, 9].

Chapter 2

Score-based Data Assimilation

2.1 Score-based Assimilation

Data assimilation is a technique aimed at recovering the trajectory of a process's state, which is often intractable from noisy or partial observations. In fact, recovering the full state trajectory solely from limited and imperfect observations is impossible. Therefore, data assimilation incorporates the transition dynamics between successive states in addition to the observations to better estimate the state. Traditional data assimilation methods are computationally expensive.

On the other hand, although sampling time in diffusion models is not optimal compared to other generative AI methods, their inference time is almost negligible compared to traditional data assimilation methods for estimating and predicting the state of a system (e.g., Kalman Filter, 4D-VAR).

This motivates François Rozet and Gilles Louppe to introduce Score-based data assimilation [9].

2.1.1 Statement of the problem

Let $\mathbf{u}_{1:L}$ be the trajectory of states of a discretized stochastic process. In data assimilation, one researches to estimate these states. This is achieved by incorporating a noisy and imperfect observation source $y = \mathcal{A}(u_{1:L}) + \eta$ along with the physical model : transition dynamics from successive states $p(u_{i+1}|u_i)$.
Assuming $\mathbf{u}_{1:L} \in \mathbb{R}^{DXL}$ and $y \in \mathbb{R}^M$, $\mathcal{A} : \mathbb{R}^{D\times L} \to \mathbb{R}^M$ is the measurement function linking the states to the observation. The imperfection of the observational instruments are modelized by the

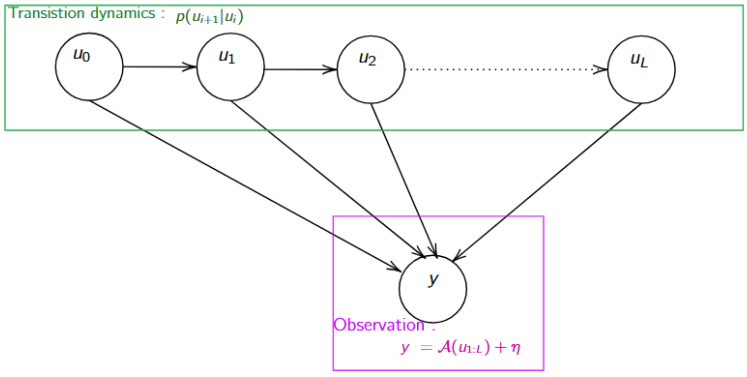


Figure 2.1: Visualization of the evolution of the discretized stochastic process together with the observation source. The dependencies from one state to the next, as well as from all states to the observations, are explicitly modeled, while the mapping from observations back to the full set of states is referred to as the inverse problem.

gaussian noise η [9].
The Bayesian inverse problem is formulated as :

$$\begin{aligned} p(\mathbf{u}_{1:L}|\mathbf{y}) &= \frac{p(\mathbf{y}|\mathbf{u}_{1:L})p(\mathbf{u}_{1:L})}{p(\mathbf{y})} \quad \text{(Bayes' Formula)} \\ &= \frac{p(\mathbf{y}|\mathbf{u}_{1:L})}{p(\mathbf{y})} p(\mathbf{u}_1) \prod_{i=1}^{L-1} p(\mathbf{u}_{i+1}|\mathbf{u}_i) \quad \text{(first order Markov chain)} \end{aligned} \tag{2.1}$$

In some cases as in climatology, the states of the dynamical can be high-dimensional and the trajectories arbitrary long (hourly data of dozens of years). Therefore, training a neural network $\nabla_{\mathrm{u}_{1:L}} p(\mathbf{u}_{1:L}(t))$ would be challenging.

Markov Blanket In a set of random variables, the conditional distribution of a variable given all other variables can sometimes be reduced to conditioning on only a subset of the variables. This subset is called the *Markov blanket* of the variable. Formally, for a set of random variables $\mathbf{u}_{1:L} = \{u_1, \ldots, u_L\}$, the Markov blanket of x_i is a subset $\mathbf{u}_{b_i} \subset \mathbf{u}_{\setminus i}$ such that

$$p(u_i \mid \mathbf{u}_{\setminus i}) = p(u_i \mid \mathbf{u}_{b_i}).$$

That is, knowing the values of the variables in the Markov blanket is sufficient to infer x_i, and adding information from the other variables does not change its conditional distribution.

For a dynamical system modeled as a first-order Markov chain, the minimal Markov blanket of a state u_i is simply its immediate predecessor and successor:

$$\mathbf{u}_{b_i} = \{u_{i-1}, u_{i+1}\}.$$

This reflects the dynamical structure: each state depends directly only on the previous state, and it directly influences only the next state. Therefore, for an exact first-order Markov chain, the conditional distribution of u_i given all states can be exactly reduced to its Markov blanket:

$$p(u_i \mid u_{1:L\setminus i}) = p(u_i \mid u_{i-1}, u_{i+1}).$$

When diffusion noise is added to the states, as in score-based diffusion models, this property no longer holds exactly. Rozet and Louppe introduce the notion of a *pseudo-Markov blanket* $\bar{\mathbf{u}}_{b_i}(t)$ for the perturbed states $u_i(t)$, such that

$$\nabla_{u_i(t)} \log p(\mathbf{u}_{1:L}(t)) \approx \nabla_{u_i(t)} \log p(u_i(t), \bar{\mathbf{u}}_{b_i}(t)).$$

The intuition behind the pseudo-Markov blanket is that, at early diffusion times $t \approx 0$, the states are almost unperturbed and $\bar{\mathbf{u}}_{b_i}(t)$ coincides with the true Markov blanket, while at late diffusion times $t \approx 1$, the states are dominated by noise and conditioning on other variables adds little information. For intermediate diffusion times, it is hypothesized that the local dynamical structure can still be exploited by defining the pseudo-blanket as a window of $2k+1$ states around $u_i(t)$:

$$\bar{\mathbf{u}}_{b_i}(t) = \{u_{i-k}(t), \ldots, u_{i-1}(t), u_{i+1}(t), \ldots, u_{i+k}(t)\},$$

with $k \ll L$. Using this local window, instead of modeling the score for the full trajectory, one can train a neural network to approximate the score only for the subsequence:

$$s_\theta(\mathbf{u}_{i-k:i+k}(t), t) \approx \nabla_{\mathrm{u}_{i-k:i+k}} \log p(\mathbf{u}_{i-k:i+k}(t)).$$

This reduces the computational cost while leveraging the system's local Markovian structure, under the hypothesis that distant states provide negligible additional information.

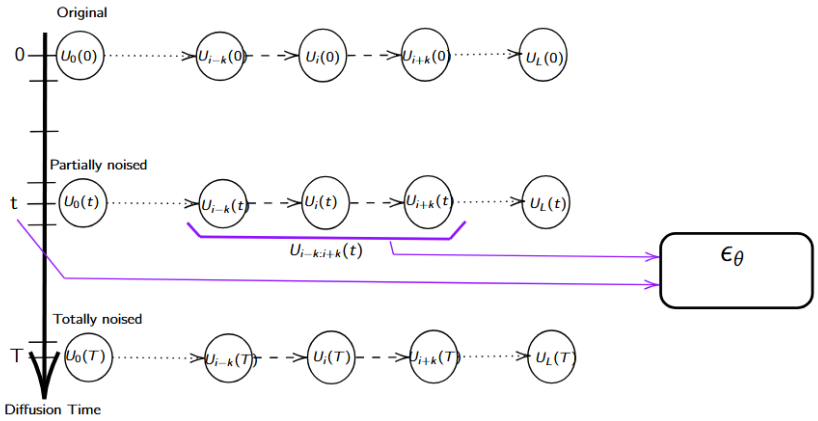


Figure 2.2: Representation of the training procedure: a random diffusion time is sampled, along with a random pseudo-blanket at that diffusion time (a subset of successive states noised at level t). This pseudo-blanket is fed to the denoiser, which is tasked with predicting the corresponding unnoised ground truth pseudo-blanket

Algorithm 3 Training $\epsilon_\theta(\mathbf{u}_{i-k:i+k}(t), t)$ [9]

for $i = 1$ to N **do**
  $\mathbf{u}_{1:L} \sim p(\mathbf{u}_{1:L})$
  $i \sim \mathcal{U}(\{k+1, \dots, L-k\})$
  $t \sim \mathcal{U}(0,1),\ \epsilon \sim \mathcal{N}(0, I)$
  $\mathbf{u}_{i-k:i+k}(t) \leftarrow \mu(t)\,\mathbf{u}_{i-k:i+k} + \sigma(t)\,\epsilon$
  $\ell \leftarrow \|\epsilon_\theta(\mathbf{u}_{i-k:i+k}(t), t) - \epsilon\|_2^2$
  $\theta \leftarrow \text{GradientDescent}(\theta, \nabla_\theta \ell)$
end for

Algorithm 4 Composing $s_\theta(\mathbf{u}_{i-k:i+k}(t), t)$ [9]

$s_{1:k+1} \leftarrow s_\theta(\mathbf{u}_{1:2k+1}(t), t)[:k+1]$
for $i = k+2$ to $L-k-1$ **do**
  $s_i \leftarrow s_\theta(\mathbf{u}_{i-k:i+k}(t), t)[k+1]$
end for
$s_{L-k:L} \leftarrow s_\theta(\mathbf{u}_{L-2k:L}(t), t)[k+1:]$
return $s_{1:L}$

2.1.2 Likelihood

Problem of Score Instability

As t increases, the noise distorts the data distribution and the learned score exhibits errors compared to the score of the true data distribution. Moreover, replacing the score in Tweedie's formula by the learned score and amplifying it by the coefficient $\frac{\sigma(t)}{\mu(t)}$ results in the approximation of $\mathbf{x}(t)$ by $\hat{\mathbf{x}}_0(\mathbf{x}(t))$ that does not hold as t increases (wrong score and amplification).

In addition, if we consider a precise instrument, i.e : a small measurement noise covariance Σ_y, the division by Σ_y further amplifies the error in the approximated conditional likelihood score of the observation given $\mathbf{x}(t)$. This creates an instability: both the growing factor $\frac{\sigma(t)}{\mu(t)}$ at large t and the inverse covariance Σ_y^{-1} may lead to arbitrarily large corrections dominated by unreliable score estimates.

This motivates a naive but practical approximation often adopted in the literature: replacing Σ_y by the identity matrix $\mathcal{I}$. Doing so removes the strong amplification effect due to small measurement noise variance, and prevents the large-t regime from dominating the likelihood term. However, this comes at the cost of ignoring the actual scale of the measurement noise, since all observation errors are treated as if they were isotropic and of equal variance.

Mitigation via Measurement Noise Accounting

Rozet and Louppe [9] mitigate the instability by explicitly accounting for the measurement noise in the perturbed likelihood. Assuming a Gaussian prior, they approximate the covariance of $p(\mathbf{x}|\mathbf{x}(t))$ as

$$\hat{\Sigma} \approx \frac{\sigma(t)^2}{\mu(t)^2}\,\Gamma,$$

where Γ is a positive semi-definite matrix capturing the prior covariance structure. Then, the perturbed likelihood becomes

$$p(\mathbf{y}|\mathbf{x}(t)) \approx \mathcal{N}\Big(\mathbf{y} \,\Big|\, A(\hat{\mathbf{x}}(\mathbf{x}(t))),\, \Sigma_y + \frac{\sigma(t)^2}{\mu(t)^2} A\Gamma A^\top\Big),$$

with A the Jacobian of the measurement function. The corresponding likelihood score is

$$\nabla_{x(t)} \log p(\mathbf{y}|\mathbf{x}(t)) \sim \Big(\Sigma_y + \frac{\sigma(t)^2}{\mu(t)^2} A\Gamma A^\top\Big)^{-1} \big(\mathbf{y} - A(\hat{\mathbf{x}}(\mathbf{x}(t)))\big).$$

Because the covariance grows with $\sigma(t)^2/\mu(t)^2$, its inverse damps the amplification of errors in the score. The large factor is effectively cancelled, preventing variance explosion, similar to how a Kalman filter down-weights measurements when uncertainty is high.

Chapter 3

MAR : "Modèle Atmosphérique Régional"

3.1 Context

The available dataset consists of high-resolution numerical climate variables, simulated by the regional climate model MAR (*Modèle Atmosphérique Régional*).
A Regional Climate Model (RCM) simulates climate processes over a limited region, as opposed to Earth System Models(ESM).
This spatial restriction enables RCMs to produce higher-resolution outputs for a given computational cost, allowing for more detailed simulations of regional climate patterns. Moreover, certain fine-scale processes may not be well captured by GCMs, but can be better resolved by RCMs.
Typically, a RCM consists of at least a mesoscale atmospheric model coupled with a land-atmosphere interaction model. Boundary conditions forced by a coarser-resolution GCM drive the numerical simulations. This process is known as downscaling [10].

MAR was initially developed to study the climate of Groenland [11]. Over time, the model has been adapted to study different types of regional climate, including those of West Africa [12] and more temperate climate in Europe [13], and has more recently been applied to Belgium [14] [15].

3.2 Description

3.2.1 Overview

MAR is based on a mesoscale atmospheric core that solves the three-dimensional hydrostatic primitive equations of the atmospheric motion [16] coupled with a one-dimensional surface-atmosphere (Energy and mass exchange at the surface) transfer scheme : SSIVAT (Sea Ice Soil Vegetation Atmosphere Transfer) [17]. The latter allows for interaction between land and various surface types, including snow, ice, vegetation, and soil.
MAR uses a common atmospheric core for all configurations; however, surface-atmosphere coupling and transfer parameterizations are configurable so the model can be tailored to represent region-specific processes and different regional climates.

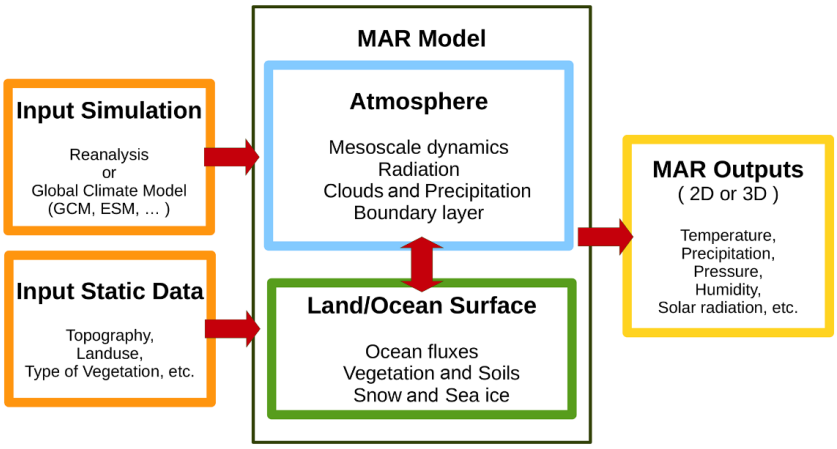


Figure 3.1: Approach of the MAR model: The MAR model (black box) is driven by a reanalysis or global model (upper orange box) and static data (lower orange box), producing 2D or 3D meteorological variables (yellow box). [18]

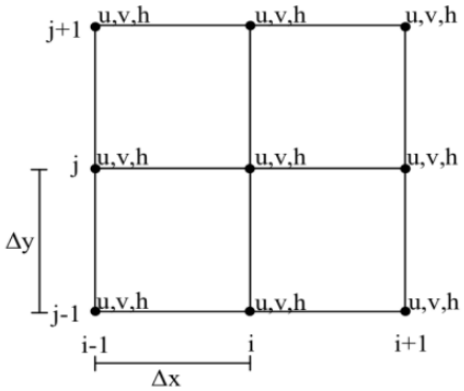


Figure 3.2: Arakawa A-grid : Spatial Discretization on a grid where the climate variables are defined on the corners. Figure from [19]

3.2.2 Atmosphere core

3.2.3 MAR Physical Processes

The primitive equations in MAR form a system of eight coupled equations with eight unknowns [16]: the horizontal wind components u and v, the vertical velocity $\dot{\sigma}$ in the terrain-following σ coordinate, the pressure difference p^* between surface pressure and pressure at the top of the model, the specific humidity q, the air density ρ, and the potential temperature θ. These variables interact through partial differential equations governing horizontal and vertical momentum, the thermodynamic state of humid air, continuity, energy, and humidity evolution. The system thus describes the full dynamics of the atmosphere in continuous time and space.

Numerical scheme

To solve these continuous equations numerically, MAR employs a spatial discretization using the Arakawa A-grid [19], where all variables, wind components, temperature, and humidity, are defined at the corners of each grid cell. This collocation simplifies the representation of the fields on the computational grid.

The numerical scheme relies on a splitting technique, decomposing the evolution over a small

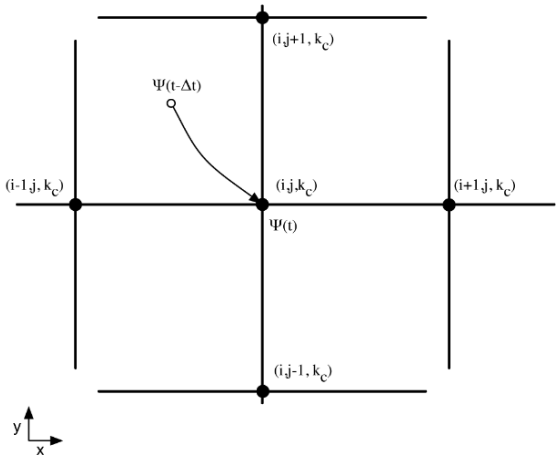


Figure 3.3: Principle of the semi-Lagrangian method: To compute the transported quantity Ψ at the current grid point (solid circle) at time t, trace back the trajectory to find its departure point (open circle) at time $t-\Delta t$. Since the departure point typically lies between grid nodes (triangles), cubic-spline interpolation is employed to determine $\Psi(t-\Delta t)$. Figure adapted from [21].

time step into three successive processes [20]. First, the transport step moves meteorological quantities along air parcel trajectories, following a semi-Lagrangian approach: trajectories are traced backward like in a Lagrangian method, but values are stored on the fixed Eulerian grid. Second, an adjustment step accounts for the rapid effects of gravity and sound waves after the transport along trajectories, modifying the fields accordingly. Third, turbulent exchange processes act to smooth fluctuations produced by the previous steps, representing subgrid-scale mixing. By treating these processes separately and then combining their effects additively, the method reduces the computational complexity while maintaining physical fidelity at small interval of time [20].

Regarding advection, the semi-Lagrangian approach : The Eulerian method calculates the quantities at fixed grid points but requires very small time steps to remain stable. The fully Lagrangian method tracks individual air parcels but is computationally prohibitive. The semi-Lagrangian method traces parcel trajectories to determine transport but interpolates back onto the fixed gridy.

3.2.4 Boundary conditions

Concerning lateral boundary conditions, MAR is typically forced with the global reanalysis dataset ERA-5, which provides atmospheric fields every six hours. The forcing variables include temperature, wind speed components, pressure at each vertical level of MAR, as well as sea surface temperature [14].

3.2.5 Relaxation procedure

The lateral relaxation procedure forces the regional climate model (RCM) toward the large-scale state provided by a global model (GCM or reanalysis) only within a lateral buffer (or relaxation) zone that surrounds the integration domain. Progressively stronger constraints are applied near the domain edge, where the model values are ultimately fixed to the large-scale values, moving inward through the buffer the constraints weaken and, in the interior, the RCM is free to evolve according to its dynamical core and physical parameterisations (3.4). Forcing fields are provided at regular intervals (typically every 6 h) and interpolated in time at each model step to avoid temporal discontinuities. [22]

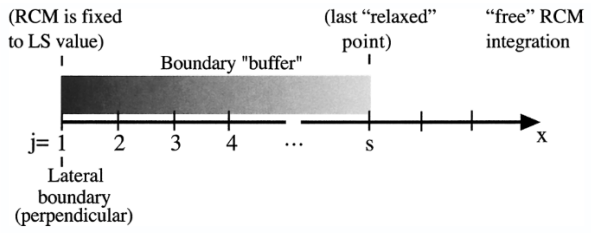


Figure 3.4: Schematic of the lateral relaxation zone and domain partitioning. (from [22].)

The relaxation tendency applied to any prognostic variable φ (for example T, u, v, q, p_s) can be written as

$$\left.\frac{\partial \varphi}{\partial t}\right|_{\text{relax}} = -\frac{\varphi - \varphi_{\text{LS}}}{\tau(\mathbf{x})} + K_h(\mathbf{x})\,\nabla_h^2(\varphi - \varphi_{\text{LS}}) \tag{3.1}$$

where φ_{LS} denotes the corresponding large-scale value from the GCM/reanalysis, $\tau(\mathbf{x})$ is the relaxation timescale (a function of horizontal position that is small close to the lateral boundary and increases toward the interior), and $K_h(\mathbf{x})$ is the horizontal diffusivity (typically largest in the buffer and reduced inward).

The two terms in (3.1) have distinct roles. The first term is a Newtonian (or nudging) term that relaxes the RCM state toward the large-scale forcing on a timescale $\tau(\mathbf{x})$. The second term is a diffusion operator acting on the difference $\varphi - \varphi_{\text{LS}}$; it spreads and damps spatial gradients of that difference so the transition from forced boundary to free interior is smooth and numerically stable.

3.2.6 Land Surface Model Integration

The land surface component is governed by the SISVAT (Soil Ice Snow Vegetation Atmosphere transfer) scheme, simulating energy and mass exchanges between the surface and atmosphere. This is a vertical model 1D with one vegetation layer along, one layer for the soil and 4 subsurface layers [23].This physically based model represents surface heterogeneity through a mosaic approach, partitioning grid cells into multiple tiles (e.g., bare soil, vegetation, snow) to account for subgrid variability. For each tile independently, SISVAT solves coupled energy and water budgets, simulating processes such as radiative transfer (shortwave and longwave), turbulent heat fluxes (sensible and latent), soil moisture diffusion, vegetation transpiration, snowpack evolution, and runoff generation. Final grid-scale outputs are area-weighted averages of tile-specific fluxes. The model interacts bidirectionally with MAR's atmospheric core: downward radiation, precipitation, and near-surface meteorological fields force the land surface processes, while SISVAT returns upward fluxes of sensible heat, latent heat, and momentum, along with skin temperature and albedo [23].

Soil-Vegetation System

The land surface module simulates energy and water exchanges using a resistance network that governs flows between soil, vegetation, and the atmosphere 3.5. This framework partitions fluxes into parallel pathways:

1. **Resistances**:
    - r_{av}: *Aerodynamic resistance* for vegetation, impeding turbulent heat/moisture exchange between the canopy and atmosphere.
    - r_{ag}: *Aerodynamic resistance* for soil, restricting turbulent fluxes from bare ground.
    - r_c: *Canopy resistance* to transpiration, modulated by plant water stress.

2. **Interaction with the atmospheric core**
    - Meteorological: Temperature (T_a), specific humidity (q_a), wind speed, pressure.
    - Radiative: Downward shortwave/longwave radiation.
    - Hydrological: Precipitation.
3. **Parameters** (Characterization of the environment): 13 parameters to characterize the vegetation and soil profile (Displacement height, albedo, Soil parameters, ...)
4. **Derived Variables** (Outputs):
    - *Fluxes*: Sensible heat (SHF), latent heat (LHF), evapotranspiration (ET).
    - *Surface states*: Vegetation temperature (T_v), ground temperature (T_g).
    - *Hydrological*: Runoff components, soil moisture, sublimation.

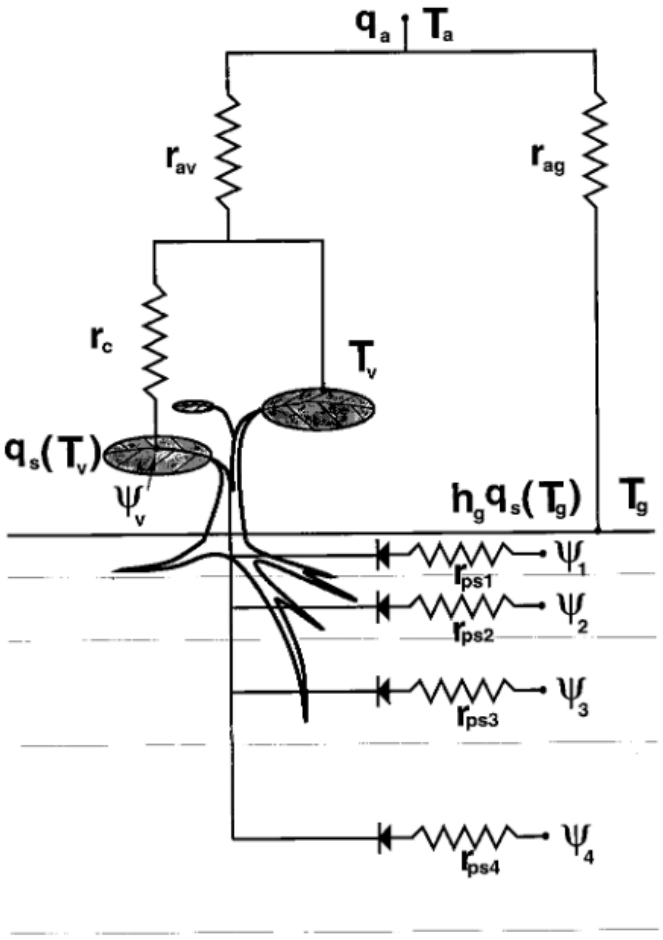


Figure 3.5: Resistance network in SISVAT's soil-vegetation system. The vegetation pathway (top) combines r_{av} and r_c; the soil pathway (bottom) uses r_{ag}.

Snow Model

The snow model in SISVAT is based on the CROCUS model [25,26]. It represents a broad range of snowpack processes, including radiative exchanges at the snow surface, turbulent fluxes and heat

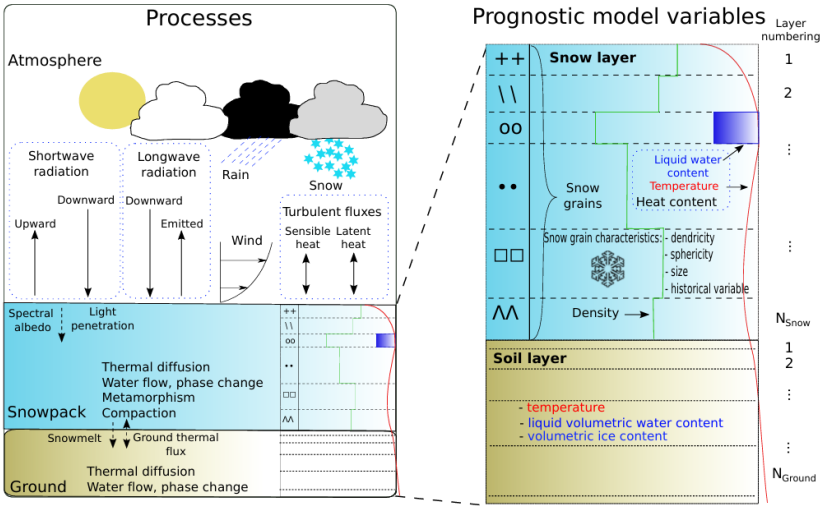


Figure 3.6: Representation of CROCUS variables, parameters. Figure from [24]

exchange with the atmosphere, internal processes such as heat conduction and water percolation within the snowpack, as well as snow compaction.

The interaction with the atmospheric core is similar to that of the soil-vegetation system. The snowpack is represented as a stack of snow layers. The model describes snow properties such as dendricity (an indicator of snow freshness), sphericity (degree to which snow crystals are rounded), grain size, density, and snow-surface age, Thickness, heat content for each snow layer.

Its outputs include simulated vertical profiles of temperature, density, and liquid water content in the snowpack, snow depth and the compaction of individual layers, snow surface temperature, water runoff at the base of the snowpack, as well as a detailed energy balance, which accounts for sensible and latent heat fluxes, shortwave and longwave radiation, and the energy associated with snowfall and rainfall.

Probabilistic Perspective From a probabilistic perspective, MAR is seen as simulator generating climate trajectories from an unknown, complex ,high-dimensional distribution:

$$\mathbf{x}_{1:L} \sim p(\mathbf{x}_{1:L})$$

MAR operates as black-box mapping initial lateral boundary conditions to climate variable trajectories. The objective is not to study or reproduce the internal physical mechanisms of the black-box, but rather to approximate its output statistically using generative AI (model approximating $p(\mathbf{x}_{1:L})$)

Chapter 4

Datasets

4.1 MAR Inputs

The inputs are the boundary conditions given by a General Circulation Model (GCM) along with the soil-vegetation profile of the region (topography of the region, density of vegetation(NDVI), Vegetation type, Surface roughness, ...). In the context of the dataset used in this work, Xavier Fettweis performed climate simulations using the MAR model, forced at the lateral boundaries with the global reanalysis dataset ERA-5 [27].

ERA-5 combines historical observations with climatological models using conventional data assimilation (4D-VAR). The observations come from various sources, such as ground-level SYNOP stations and satellites.

At its outputs, ERA-5 provides slightly fewer than 300 variables across 37 pressure levels, ranging from 1000 hPa to 1 hPa, with an hourly temporal resolution starting from 1940.

For MAR, the inputs are the temperature T, the wind components (U and V), the specific humidity q across all pressure levels, and the surface pressure SP. Some of the prognostic variables in the system of equations of the physical core of MAR are not directly provided but derived from other physical laws . ρ and θ are derived from the inputs through the equation of state and thermodynamic relations, while $\dot{\sigma}$ is determined via the continuity equation. In this way, MAR uses the prescribed forcing fields to integrate the full dynamical system consistently.

4.2 MAR Outputs / MAR Dataset

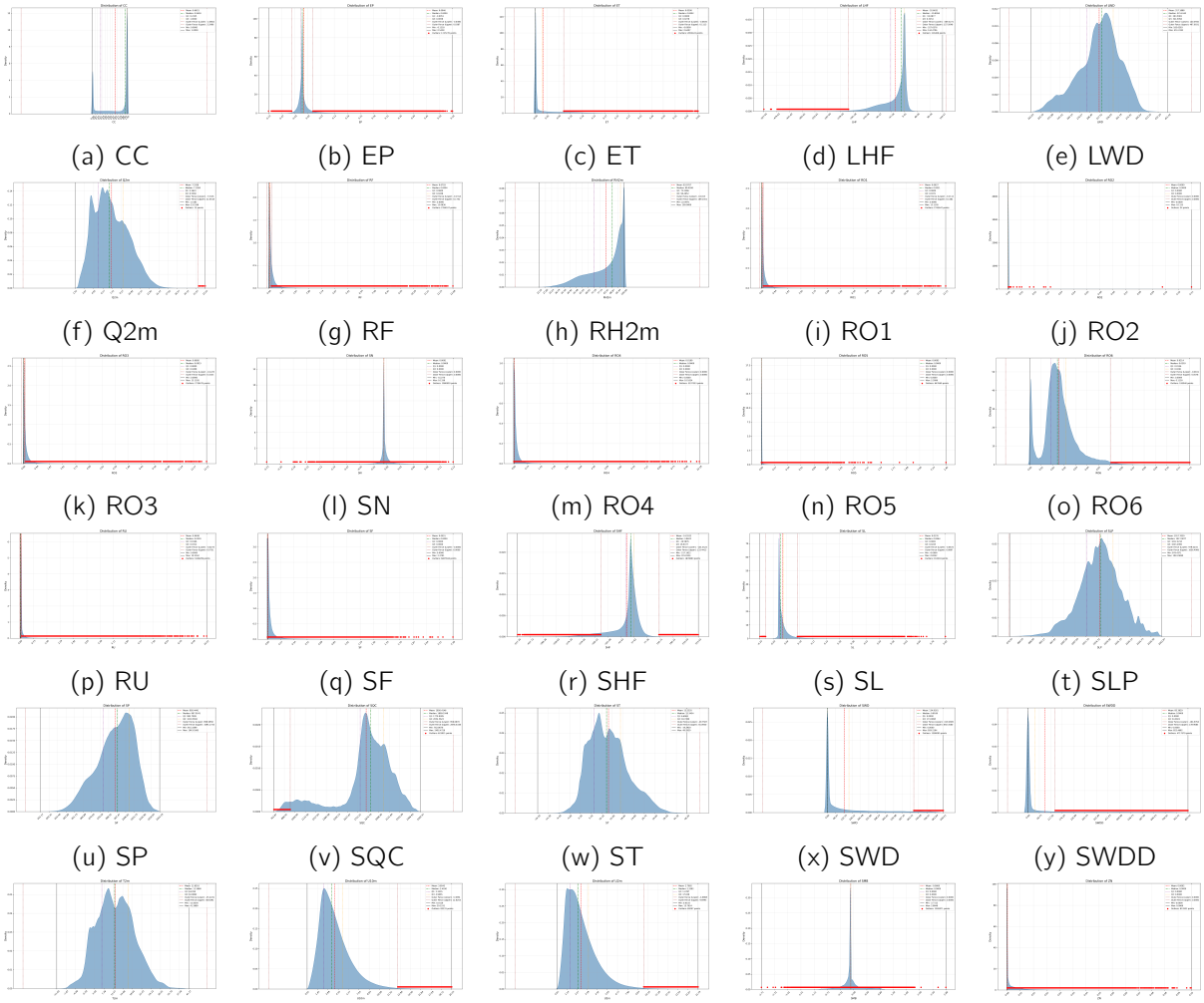
The output includes 30 meteorological variables available at hourly resolution derived from the various variables in the physical generator.Here, we have surface variables (2D) so the variables are not defined at pressure levels. Later in this work, "MAR dataset","MAR Trajectory" or simply "MAR" is considered as the actual outputs of the physical generator. A complete description of these variables :

- Cloud Cover (**CC** [%])
  Represents cloudiness: 0% indicates clear skies, 100% corresponds to fully overcast conditions.
- Evaporation (**EP** [mm/h])
  Amount of water evaporating from the surface per hour.
- Evapotranspiration (**ET** [mm/h])
  Combined loss of water by evaporation and transpiration from vegetation.

- Latent Heat Flux (**LHF** [W/m^2])
  Energy exchanged between the surface and atmosphere during phase changes of water (e.g., evaporation or condensation).
- Long Wave Downward Radiation (**LWD** [W/m^2])
  Thermal infrared radiation emitted downward from the atmosphere to the surface.
- 2m Specific Humidity (**Q2m** [g/kg])
  Amount of water vapor in the air measured 2 meters above the surface.
- Rainfall (**RF** [mm/h])
  Precipitation amount per hour.
- 2m Relative Humidity (**RH2m** [%])
  Ratio of actual to maximum possible water vapor at 2 meters above the surface.
- Rainfall (after canopy) (**RO1** [mm/h])
  Rain that reaches the ground after interaction with vegetation.
- Runoff from Snowmelt (**RO2** [mm/h])
  Water generated from melting snow.
- Rainfall interacting with Snow Layers (**RO3** [mm/h])
  Rain infiltrating or interacting with snowpack.
- Runoff from Surface Permeability (**RO4** [mm/h])
  Water running off the surface when soil permeability limits infiltration.
- Runoff due to Soil Saturation (**RO5** [mm/h])
  Excess water flowing after the soil becomes saturated.
- Runoff from Percolation (**RO6** [mm/h])
  Water percolating through the soil contributing to runoff.
- Sublimation from Snow (**SN** [mm/h])
  Amount of snow directly converting to vapor (solid to gas).
- Total Runoff (**RU** [mm/h])
  Sum of Snowfall and rainfall
- Snowfall (**SF** [mm/h])
  Hourly water-equivalent of falling snow.
- Sensible Heat Flux (**SHF** [W/m^2])
  Heat exchanged between surface and atmosphere via conduction and convection.
- Sublimation from Soil (**SL** [mm/h])
  Water vapor flux from soil moisture via direct vaporization.
- Sea Level Pressure (**SLP** [hPa])
  Atmospheric pressure adjusted to sea level.
- Mass Balance (**SMB** [mm/h])
  Net accumulation or loss of mass (typically snow/ice) over time.
- Sublimation from Snow (**SN** [mm/h])
  Loss of water from the snowpack directly to the atmosphere by sublimation.

- Surface Pressure (**SP** [hPa])
  Atmospheric pressure measured at ground level.
- Soil Humidity Content (**SQC** [kg])
  Average water content in the soil.
- Surface Temperature (**ST** [°C])
  Temperature measured at the surface of the ground.
- Shortwave Radiation Downward (**SWD** [W/m^2])
  Incoming shortwave radiation at the surface.
- Direct Shortwave Radiation Downward (**SWDD** [W/m^2])
  Direct component of incoming shortwave radiation.
- 2m Air Temperature (**T2m** [°C])
  Air temperature measured at 2 meters above ground.
- 10m Wind Speed (**U10m** [m/s])
  Wind speed measured 10 meters above the ground.
- 2m Wind Speed (**U2m** [m/s])
  Wind speed measured 2 meters above the ground.

Figure [4.1] shows the distribution and related statistical information for the trajectories simulated by MAR in 2022.



(a) CC (b) EP (c) ET (d) LHF (e) LWD

(f) Q2m (g) RF (h) RH2m (i) RO1 (j) RO2

(k) RO3 (l) SN (m) RO4 (n) RO5 (o) RO6

(p) RU (q) SF (r) SHF (s) SL (t) SLP

(u) SP (v) SQC (w) ST (x) SWD (y) SWDD

(z) T2m (aa) U10m (ab) U2m (ac) SMB (ad) ZN

Figure 4.1: Distributions of MAR outputs

4.3 IRM Dataset

4.3.1 Dataset Overview

The IRM ("Institut Royal Météorologique de Belgique") provides synoptic-style observations from 29 automatic weather stations. Each station reports hourly values, computed as a short-term average within the hour. The variables used in this study are listed in Table 4.1.

Variable	Height	Averaging Window
TEMP	1.5 m	Mean between T-11 and T-10 (1 minute)
WIND SPEED	10 m	Mean between T-20 and T-10 (10 minutes)

Table 4.1: IRM SYNOP station variables and their measurement protocols

Observation Process

We extract corresponding MAR grid-point values at each station location, taking the nearest point based on the Haversine distance. Let

$$H\colon \mathbb{R}^{n_x \times n_y} \to \mathbb{R}^N$$

be the operator that maps a gridded MAR field $\mathbf{x}(t)$ to its values at the N station coordinates:

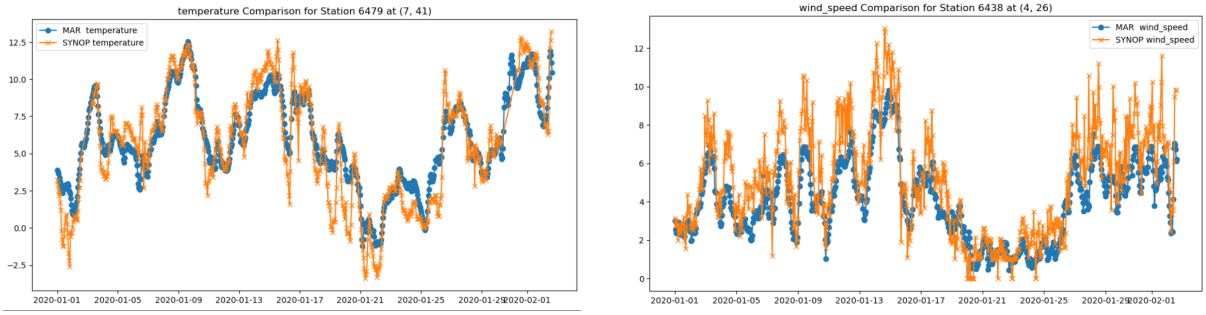
$$\mathbf{y}_{\text{MAR}}(t) = H\big(\mathbf{x}(t)\big), \quad \mathbf{y}_{\text{IRM}}(t) = \big[y_i(t)\big]_{i=1}^{N}.$$

Table 4.2 summarizes the direct correspondence between IRM station variables and MAR output fields.

IRM Variable	**MAR Field**
Temperature (TEMP)	T2m
10 m Wind Speed (WIND)	U10m

Table 4.2: Mapping of IRM SYNOP variables to MAR reanalysis fields.

Because IRM values are averaged over a short window (1min and 10 min) and MAR fields are hourly means, a mismatch arises. Moreover, MAR does not explicitly assimilate these sparse point observations, so a non-linear bias is expected. Figure 4.2 illustrates this mismatch over the year 2020 for one weather station.



(a) Temperature (T2m vs. TEMP)

(b) Wind speed (U10m vs. WIND)

Figure 4.2: Year 2020 comparison of IRM synoptic station observations and corresponding MAR grid-point values.

4.4 ERA-5-Land Reanalysis

4.4.1 Dataset Conditioning

Description

We select three variables from the ERA-5 Land hourly dataset for January 2020 (a period not seen by the MAR denoiser during training) over a region centered on Belgium: the 2m air temperature ($T2M_{ERA}$) and the two horizontal wind components ($W_{U10M,ERA}$, $W_{V10M,ERA}$). From the wind components, we compute the wind speed norm as

$$U10m_{ERA} = \sqrt{(W_{U10M,ERA})^2 + (W_{V10M,ERA})^2}.$$

The resulting observation y^* is defined on a grid of size $[Y_{ERA}, X_{ERA}]$, where each grid cell stores two variables: $\{T2M_{ERA}, U10m_{ERA}\}$.

Observation Process

The observation process is similar to the artificial coarsening experiment, except that the grid is no longer divided into non-overlapping square patches, but each MAR grid point is instead linked to

a corresponding region in the ERA-5 domain(Minimum Haversine length), because the two grids use different coordinate systems and resolutions, their points do not align directly. ERA-5 relies on a regular 0.25° × 0.25° latitude-longitude grid (approximately 25 km at the equator), whereas MAR uses a 5 km × 5 km grid defined in the Belgian Lambert 72 projection. The observational operator maps the MAR domain onto the ERA-5 grid (observational space). For each ERA-5 grid cell, we identify the corresponding MAR pixels, and compute the average value of each variable over those pixels. This produces the aggregated fields $\widehat{T2M_{ERA}}$ and $\widehat{U10m_{ERA}}$.

Figure 4.3 illustrates the downscaling process from the ERA-5 grid to the MAR grid.

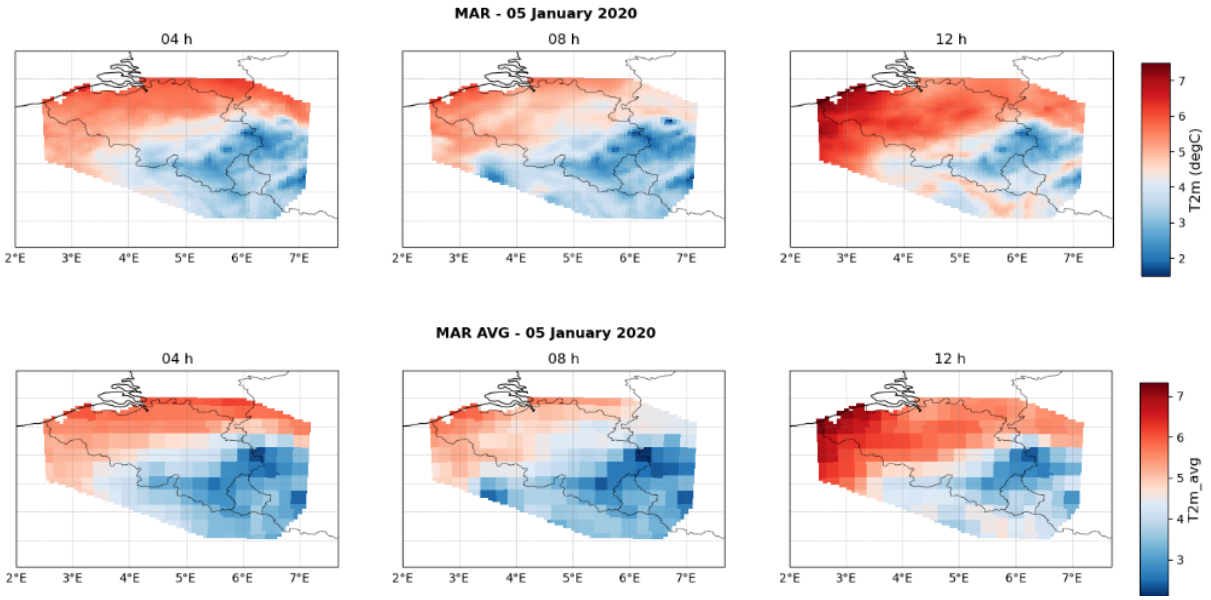
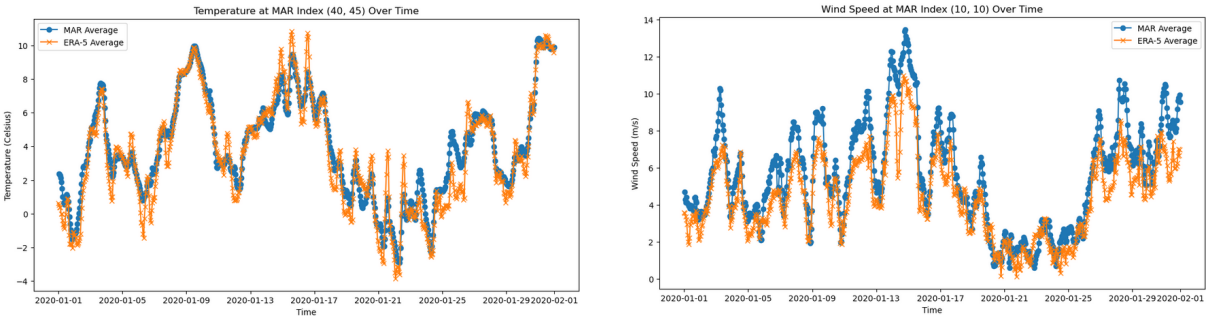


Figure 4.3: (a) T2M MAR trajectory shown every 4 hours, (b) Trajectory downscaled (Application of the os-bservator)

Discrepancy between single-level ERA-5 and MAR

Comparing a MAR grid cell average (Observation Process) with the corresponding ERA-5 grid at the same date will inevitably reveal discrepancies. First, the ERA-5 data we use here is not the exact dataset employed to drive MAR's boundary conditions in the physical generator. Second, the comparison is made for interior points, well inside the domain, rather than along the lateral boundaries. Consequently, when you spatially average the high-resolution MAR output back to the coarser ERA-5 grid, you should not expect those upscaled values to coincide with the original GCM fields in the interior region. MAR relies solely on the GCM fields at its boundaries and then applies its own physical parametrizations to evolve conditions inward, whereas a GCM like ERA-5 performs global data assimilation using observations from many sources.

Figure 4.4 demonstrates the temporal difference at a single grid cell between single-level ERA-5 data and the corresponding averaged MAR patch over the same period (i.e. after applying the observational sampling to MAR). Figure 4.5 shows the temporal and spatial differences across the entire domain at three timestamps, each separated by four hours. The results indicate the presence of a bias that is neither constant in time nor uniform in space.



(a) Temperature (T2m vs. TEMP) (b) Wind speed (U10m vs. WIND)

Figure 4.4: Comparison for January 2020 at a single grid cell between the ERA-5 single-level reanalysis and the MAR average patch for the same period

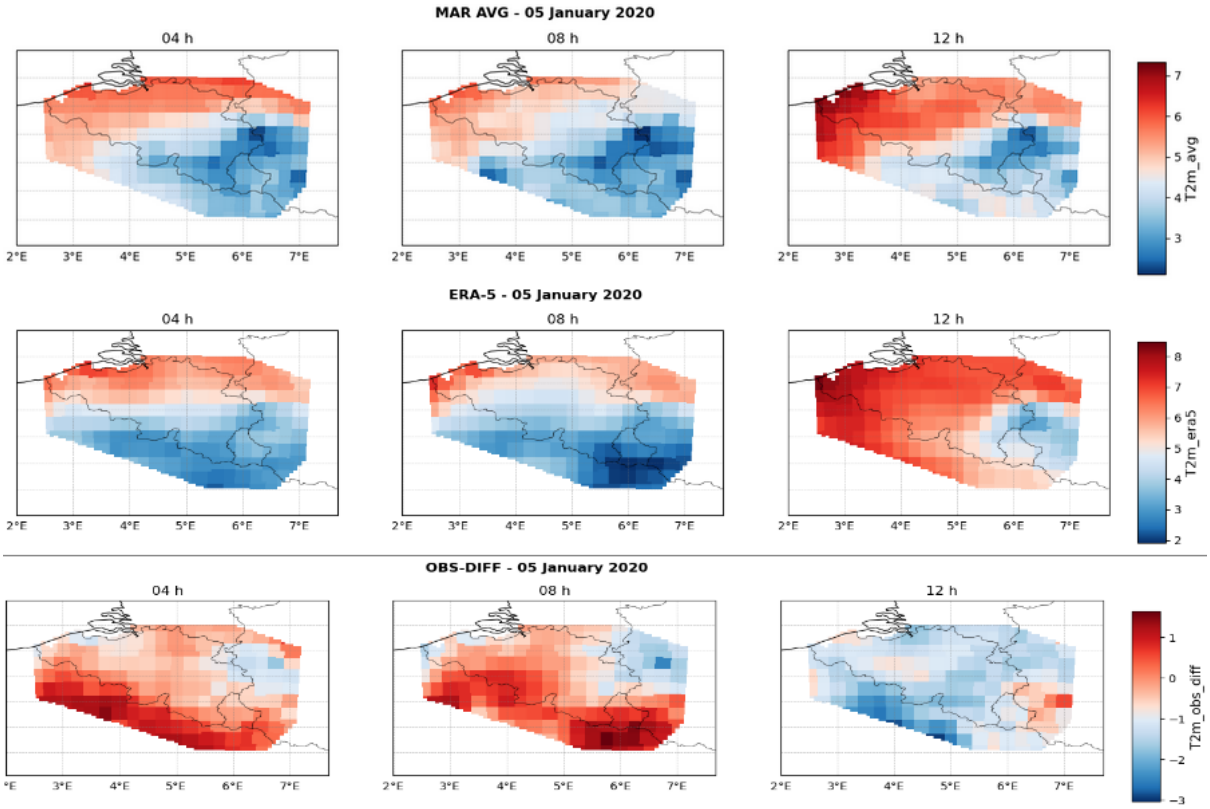


Figure 4.5: a) MAR average patches b) reanalysis single-level ERA-5 c) difference between a) and b)

Chapter 5

Experiments

5.1 Implementation

5.1.1 Model Architecture

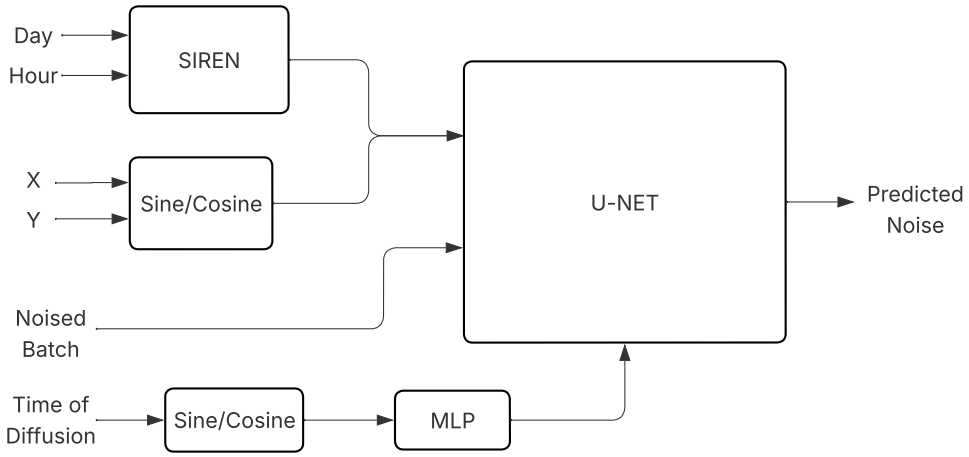


Figure 5.1: Conceptural representation of the architecture. The date is embedded before being processed by the Denoiser along with the noised batch. The backbone U-NET predicts the added noise on the noised batch

Figure [5.2] illustrates the different components of the model. The denoiser receives as input the noised batch, the context, and the embeddings corresponding to the diffusion time. Its output is the predicted noise for the noised batch. The context consists of both spatial and temporal components.

Backbone

Adaptation The U-Net undergoes several minor adaptations, as described in [2] to make it suitable as a backbone for diffusion models. In fact, for (1.23), the purpose of the model is to predict the noise added to the original sample at a given timestep.
The first one is to replace convolution blocks at a given level with Res-Net blocks [28]. The main advantage is to avoid vanishing gradient, thus enabling the training of deeper architectures, achieving better results.
Moreover, in diffusion models, the neural network needs to know the diffusion time or step (see (1.23) , (1.14), (1.9)). This is done by adding the sinusoidal embedding of the timestep at the end of the ResNet block.

Finally, adding self-attention blocks [29] helps the model to capture long-range dependencies, since convolutions are limited to capturing neighboring dependencies (locality principle).

Implementation A Description of the model : Table [5.1] presents the U-NET architecture

Path	Level	Input Size	# Blocks	Hidden Channels	Self-Attention
Encoder	0	$64 \times 64 \times 29$	2	64	No
Encoder	1	$32 \times 32 \times 64$	3	128	No
Encoder	2	$16 \times 16 \times 128$	4	256	No
Encoder	3	$8 \times 8 \times 256$	3	512	No
Bottleneck	4	$4 \times 4 \times 512$	2	768	Yes
Decoder	3	$8 \times 8 \times 768$	3	512	No
Decoder	2	$16 \times 16 \times 512$	4	256	No
Decoder	1	$32 \times 32 \times 256$	3	128	No
Decoder	0	$64 \times 64 \times 128$	2	64	No
Output	-	$64 \times 64 \times 64$	-	$\rightarrow$ 24 (final data channels)	No

Table 5.1: Full U-Net architecture derived from the implementation: encoder, bottleneck, and decoder with block counts, hidden channels, and attention. Input includes 24 data + 5 context channels, output reconstructs only the 24 data channels.

used for the two-variable case ($T2m$ and $U10m$). Experimentally, we found that adding self-attention significantly improves the backbone's ability to denoise images, but it comes at the cost of higher computational resources; therefore, it is applied only at the bottleneck. The 2D spatial region is discretized on a 64×64 grid, and we consider local windows of 12 consecutive hours. The timestamp and variables are concatenated along a single channel dimension to make the input compatible with the U-NET architecture (2D data with channels). Additionally, contextual information can be incorporated: in this model, we add 5 layers of 64×64 features concatenated along the input channels, resulting in an input tensor of size $[Y, X, (W \times \text{VAR} + \text{CONTEXT})]$, which is $[64, 64, 29]$ in this case. The input corresponds to a batch of images noised at level t, and the output predicts the noise at each pixel, noting there is not the context appended at the output $[Y, X, (W \times \text{VAR})]$.

30 variables The model is designed for 30 variables using an assimilation window of only 3 consecutive hours, since larger windows would significantly increase the number of channels and make training more difficult (but not impossible !).

5.1.2 Conditioning

Temporal Conditioning

Meteorological variables show temporal fluctuations, on a daily and seasonal scale. To take advantage of this knowledge, a dedicated neural network learns a contextual representation of time, passed to the main U-Net model later on. This avoids explicitly encoding any prior relationship or knowledge, which could be a challenging task. This network receives as input the fraction of the hour within a year (to account for diurnal patterns, such as solar radiation) and the fraction of the day within a year to account for seasonal fluctuations.
The chosen architecture for this module is SIREN (Sinusoidal Representation Network) [30], which is well suited for learning high-frequency patterns through the use of sinusoidal activations. Specifically, the fractional day of the year and fractional hour of the day are converted into sine and cosine components, resulting in a 4-dimensional input vector. This vector is then passed through a SIREN

Path	Level	Input Size	# Blocks	Hidden Channels	Self-Attention
Encoder	0	$64 \times 64 \times 95$	3	128	No
Encoder	1	$32 \times 32 \times 128$	3	128	No
Encoder	2	$16 \times 16 \times 128$	3	256	No
Encoder	3	$8 \times 8 \times 256$	3	384	No
Bottleneck	4	$4 \times 4 \times 384$	3	768	Yes
Decoder	3	$8 \times 8 \times 768$	3	384	No
Decoder	2	$16 \times 16 \times 384$	3	256	No
Decoder	1	$32 \times 32 \times 256$	3	128	No
Decoder	0	$64 \times 64 \times 128$	3	128	No
Output	-	$64 \times 64 \times 128$	-	$\rightarrow$ 90 (final data channels)	No

Table 5.2: Full U-Net architecture for the VP-SDE model. Input includes 90 data channels (30 variables $\times$ 3 timesteps) plus 5 context channels, output reconstructs only the 90 data channels. Each level has 3 residual blocks, and self-attention is applied only at the bottleneck.

network consisting of 6 fully connected layers, each with 256 hidden units and sinusoidal activations (with $w_0 = 30$ for the first layer). The network output is reshaped to $[1, Y, X]$ to match the spatial resolution of the main input.

Spatial Conditioning

As we work with climate variables, the local spatial relationship provided by the convolutions in the U-NET could not be sufficient enough to provide precise positional information. Therefore, we encode the position of the grid with sine and cosine as additional context channels in the backbone U-NET. Spatial context is encoded via a fixed 2D sinusoidal grid over the 64×64 region. The grid is constructed by taking sine and cosine functions of normalized spatial coordinates along both axes, producing 4 channels that are constant across time but vary across space. Moreover, the U-NET is initially designed for squared images and the region of interest is not symmetrical, therefore a mask is used in order to make the difference between the region of interest and no man's land. The network is therefore allowed to predict the noise outside the limits of the mask, but these values are meaningless and are not taken into account in future computation (such as the loss).

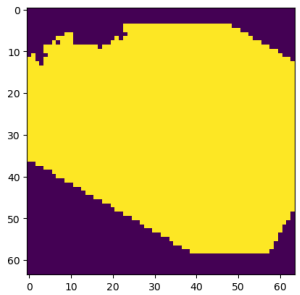


Figure 5.2: 64x64 spatial grid of the batch. The network is allowed to predict the noise on the whole grid but region in purple is ignored in further computation. Yellow region represents the spatial MAR values of a given variable at a given timestep

Normalization

The variables are normalized based on z-score.

$$z = \frac{x - \mu}{\sigma} \tag{5.1}$$

There is a normalization per variable and per split of dataset (training, validation, and testing dataset)

5.1.3 Training Setup

Dataset splitting

Model	Training	Test	Validation
T2m, U10m	1940-2005	2014-2022	2006-2013
all variables	2000-2016	2017-2020	2021-2022

Table 5.3: Dataset splitting for both models

Loss Function

For the training phase, the model predicts the noise added to the batch in the forward diffusion process. The loss function is the root mean square error between the predicted noise and the true noise (noise added in the forward process), ignoring pixels outside the defined region by multiplying with the mask w.

$$L = \mathbb{E}_{x,c}\left[\left\| w \odot \left(s_\theta(x, c) - \nabla_x \log p(x \mid c)\right)\right\|_2^2\right]$$

Noise Scheduler

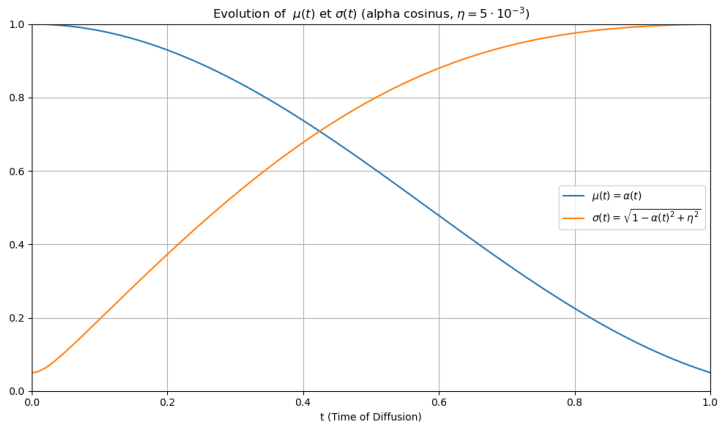


Figure 5.3: Figure shows the evolution of variance and mean over diffusion time.

The noise scheduler is the cosine motivated because it preserves details at the beginning and at the end of the diffusion process allowing for smoother transitions. The latter has improved quality samples and more stability than linear scheduler [31]. In the original code, numerical instabilities are addressed by not going exactly towards 0 at the end of the mean of the diffusion process but rather to $\eta = 0.0001$. Reminding the forward process, application of noise to the batch :

$$\mathbf{x}(t) = \mu(t)\mathbf{x}(0) + \sigma(t)\epsilon$$

$$\sigma(t) = \sqrt{1 - \alpha(t)^2 + \eta} \quad ; \mu(t) = \alpha(t)$$
$$\alpha(t) = \cos^2(\arccos(\sqrt{\eta})t)$$

Optimizer

The model is optimized using the AdamW optimizer, with a cosine-scheduled learning rate up to 2×10^{-4} and a weight decay of 10^{-4}

5.2 Metrics

5.2.1 Metrics

Qualitative Metrics

Radially averaged power spectrum The *Radially Averaged Power Spectral Density (RAPSD)* [32] characterizes how the variance of a two-dimensional field is distributed across spatial scales, providing insight into the distribution of energy over those scales. To obtain the RAPSD, the full 2-D spectrum is reduced to a 1-D, direction-independent curve: we compute the 2-D Fourier transform

$$F(k_x, k_y),$$

form the power spectral density

$$S(k_x, k_y) = |F(k_x, k_y)|^2,$$

convert each point to its radial wavenumber

$$k = \sqrt{k_x^2 + k_y^2},$$

and average the power within radial bins to obtain

$$P(k).$$

For visualization, the wavenumber is converted to wavelength, and P is plotted against wavelength on log–log axes.

Comparing overlaid RAPSD curves for real and generated data reveals whether the model has successfully captured the energy distribution across spatial scales.

Rank Histograms *Rank histograms* [33] are constructed as following . For each pixel, the N ensemble members are sorted from lowest to highest, forming an ordered array. The ground-truth value is then positioned within this array: if it is smaller than all ensemble members, it receives rank 0, whereas if it falls between the fifth and sixth members it is assigned rank 5. Repeating this procedure for all pixels yields a histogram of ranks. A flat histogram indicates that the ensemble is well calibrated, a U-shaped histogram reveals underdispersion where the ensemble spread is too small and observations frequently fall outside the ensemble, and a hump-shaped histogram indicates overdispersion where the spread is too large and the ensemble predictions are too extreme. Any other systematic deviation suggests a mismatch between the ensemble and the ground truth.

Quantitative Metrics

Root Mean Squared Error *Root Mean Squared Error (RMSE)* of the ensemble mean [34] is used to compare observations with the posterior, either directly on the MAR grid (5.2) or after projection onto the observational space (5.3) (e.g., a coarser grid or sparse measurement locations). To enable comparison across variables, the RMSE can be normalized (5.2.1).

$$\text{RMSE} = \sqrt{\frac{1}{N_x N_y N_t} \sum_{t=1}^{N_t} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left(\hat{X}_{i,j,t} - X_{i,j,t}\right)^2} \tag{5.2}$$

$$\text{RMSE}^{\text{OBS}} = \sqrt{\frac{1}{N_t N_\Gamma} \sum_{t=1}^{N_t} \sum_{\gamma=1}^{N_\Gamma} \left(A(\hat{X})_{\gamma,t} - A(X)_{\gamma,t}\right)^2} \tag{5.3}$$

$$\text{NRMSE} = \frac{\text{RMSE}}{\sigma_{\text{z-score}}}$$

Here, N_t is the number of time steps, N_x and N_y denote the spatial dimensions of the MAR grid, and N_Γ is the number of elements in the observational space.

Average Ensemble Spread *Average Ensemble variability* [34] measures the average variability of the ensemble members around their mean.

$$s_t^2 = \frac{1}{N_\Gamma (M-1)} \sum_{\gamma=1}^{N_\gamma} (\overline{A(X)}_{t,\gamma} - A(X)_{t,\gamma})^2 \tag{5.4}$$

where M is the number of members of the ensemble. For the samples on the MAR grid, A is the identity function, otherwise it is the observer mapping from the MAR grid to another space with the corresponding N_γ elements. $\overline{A(X)}_{t,\gamma}$ denotes the average over members of the element γ at timestep t. Finally, following the paper [34], the spread is estimated as :

$$\sigma_{\text{SPREAD}} = \sqrt{\left(\frac{M+1}{M}\right) \frac{1}{T} \sum_{t=1}^{T} s_t^2} \tag{5.5}$$

Observation Consistency Metric (OCR) We introduce a metric to quickly assess whether the produced posterior samples projected in the observation space are consistent with the ground-truth observations.

Formally, let the states be $\mathbf{x} \in \mathcal{X}$ and the observations $\mathbf{y} \in \mathcal{Y}$ wher $\mathcal{Y}$ is the observation space. The measurement function is $\mathcal{A} : \mathcal{X} \to \mathcal{Y}$. We assume imperfect measurements modeled as a centered Gaussian:

$$\mathbf{y} = \mathcal{M}(\mathbf{x}, \sigma_y) = \mathcal{A}(\mathbf{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_y^2),$$

where σ_y characterizes the standard deviation of the measurement device. The projection of the posterior $\widehat{\mathbf{x}}$ is written :

$$\widehat{\mathbf{y}} = \mathcal{A}(\widehat{\mathbf{x}})$$

Following the bias-variance decomposition [35]:

$$\text{(N)MSE}\big(\mathcal{A}(\hat{\mathbf{x}}),\, y = \mathcal{M}(\mathbf{x}^*)\big) = \mathbb{E}\Big[\big(\mathcal{A}(\hat{\mathbf{x}}) - y\big)^2\Big] = \underbrace{\sigma_y^2}_{\text{measurement noise}} + \underbrace{\text{Bias}^2}_{\text{systematic error}} + \underbrace{\text{Variance}}_{\text{posterior fluctuations}}.$$

If NMSE $\approx \sigma_y^2$, or equivalently $\frac{\text{RMSE}}{\sigma_y} \approx 1$, this indicates that the posterior samples, when projected into the observational space, are unbiased and stable, limited primarily by the irreducible measurement noise σ_y^2.

This motivates the definition of the **Observation Consistency Ratio**:

$$\text{OCR} = \frac{\text{NRMSE}}{\sigma_y}.$$

Interpretation:

1. OCR ≈ 1: the prediction error is essentially limited by measurement noise, the posteriors in the observation space do not exhibit significant bias and variance compared to the observations.
2. OCR < 1: likely due to undersampling, with few posterior samples, the empirical MSE can occasionally fall below the measurement noise.
3. OCR $\gg 1$: the total error exceeds the measurement noise, reflecting additional variability or bias in the posterior projections ; the posteriors do not align with the observations.

C2ST : Classifier two-samples Test The *C2ST* paper formalizes the method as follows [36]:

Method A practical way to assess whether two datasets, denoted D_{real} and D_{gen}, originate from the same underlying distribution is to frame the problem as a binary classification task. Each real sample is assigned the label 0 and each generated sample the label 1, and the two sets are merged into a single dataset.

$$D = \{(x_i^{real}, 0)\}_{i=1}^{n} \cup \{(x_j^{gen}, 1)\}_{j=1}^{m},$$

where $x_i \in D_{\text{real}}$ and $x_j^{gen} \in D_{\text{gen}}$. In the present case, the sample sizes satisfy $n \approx m$. The dataset D is randomly shuffled and split into two disjoint subsets: a training set D_{train} and a test set D_{test}. A binary classifier $f : X \to [0, 1]$ is then trained on D_{train}, where $f(z)$ outputs the estimated probability that z belongs to the generated class. The performance of the classifier is evaluated on D_{test} using metrics such as classification accuracy or ROC AUC. When the two datasets are drawn from the same distribution, the accuracy of a well-calibrated classifier should remain close to the chance level of 50% for a balanced dataset. Values significantly higher than this threshold indicate detectable statistical differences between the two distributions.

5.3 Unconditional generation

In the unconditional generation setting the objective is to draw samples from the model's learned data distribution without providing any explicit conditioning information. In this section we evaluate the fidelity of these unconditional samples by comparing them to the MAR dataset, examining both large-scale structures and fine-grained patterns. This comparison is fundamental: if the model fails to produce realistic unconditional samples, subsequent tasks such as guidance over real-world observations would not be possible.

5.3.1 Visual Evaluation

5.3.2 Statistical Comparison

Spectral Decomposition

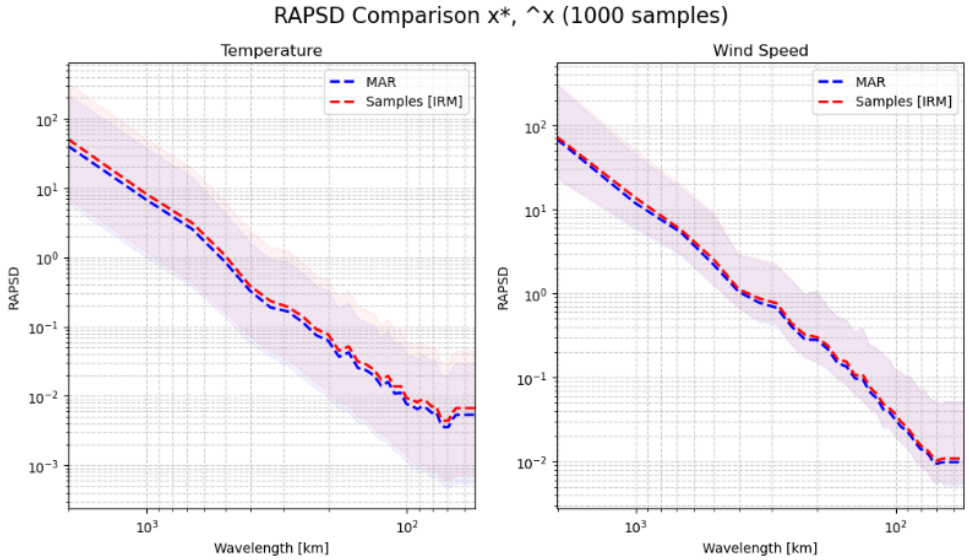


Figure 5.4: Spectral decomposition of 1000 real samples from the MAR dataset and 1000 unconditional samples generated by the model. Dashed lines indicate the median; shaded regions correspond to the 5th-95th percentile range.

Method

Interpretation The spectral decomposition 5.4 shows that the generated samples reproduce the real data's frequency content over a wide range of scales: both large-scale structure and fine-scale details are present in the synthetic fields. Small systematic differences between the median spectra and the shown quantiles indicate residual discrepancies : these may reflect either remaining model limitations at particular wavelengths or sampling variability from a limited ensemble.

C2ST : Classifier two-samples Test

Experiment We used $n = m = 1000$. The classifier is an EfficientNet [37] pretrained on ImageNet [38] and fine-tuned to tell real from generated samples. The network's first convolutional layer was changed to accept 24 input channels (two variables $\times$ 12 timesteps) by copying the pretrained 3-channel kernels 8 times. Training ran for 40 iterations and performance was measured on a held-out test set. The classifier reached about 77% accuracy and a ROC AUC of 0.85.

Table 5.4: Classification Metrics Summary on the test set

	Precision	Recall	F1-score	Support
Generated (0)	0.73	0.82	0.77	196
Real (1)	0.80	0.72	0.76	204
Accuracy		0.77		400
Macro Avg	0.77	0.77	0.76	400
Weighted Avg	0.77	0.77	0.76	400

Confusion Matrix: $\begin{bmatrix} 160 & 36 \\ 58 & 146 \end{bmatrix}$

TP = 146, TN = 160, FP = 36, FN = 58

ROC AUC = 0.8515

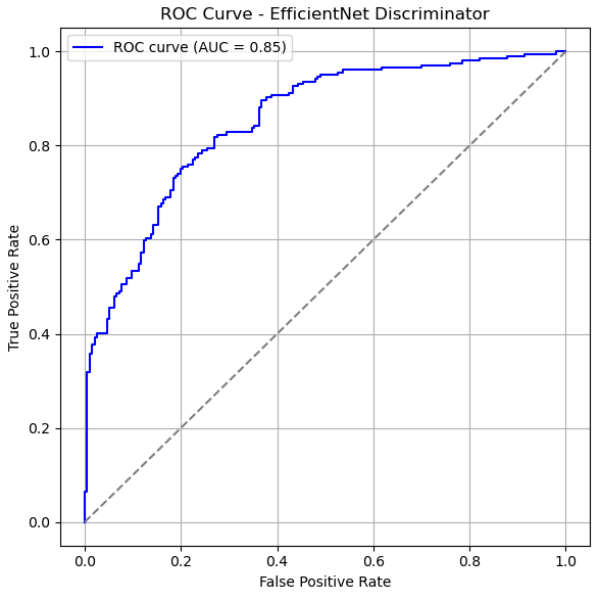


Figure 5.5: ROC Curve of the binary classifier (EfficientNET) on the test set

Limitations This test only tells us whether a difference between the two datasets is detectable by the chosen classifier, not why the difference exists. The result depends on the model and how it was trained: a weak model may miss differences, while a model that overfits can give misleadingly high scores. The features the classifier uses to separate the datasets might be harmless (small values outside the region of interest, slight edge effects) or important (different temporal patterns across the 12 timesteps).

5.3.3 Observations

Although the classifier can distinguish real from generated samples better than chance (approximately 77% accuracy, ROC AUC ≈ 0.85), the spectral decomposition indicates that the generator reproduces the data's scale-dependent energy across a wide range of wavelengths. This suggests that the model captures both large-scale structures and fine-scale variability well enough for the unconditional samples to be credible and usable for further exploration. Nonetheless, the classifier's performance indicates that some discrepancies persist between the generated and real datasets, and that the model likely produces samples with artifacts.

5.3.4 To go further

There is still room for improvement and for a deeper evaluation of the results. A natural extension would be to compare the generator with alternative generative models, such as GANs, VAEs, or other diffusion models on metrics such as Continuous Ranked probability Score (CRPS) [39] ... . In addition, a thorough climatological variable analysis could be conducted.

5.4 Artificial Coarsen Experiment

5.4.1 Motivation

As discussed previously, the physical generator MAR is driven by lateral boundary conditions provided by a Global Climate Model (GCM): multi-level climate variables (temperature, humidity, winds). Here, to demonstrate our diffusion framework's downscaling capability in a simplified proof-of-concept setting, we emulate GCM input by coarsening MAR's surface-level fields.

5.4.2 Observer

Concretely, let $\mathbf{x}(t) \in \mathbb{R}^{n_x \times n_y}$ denote a high-resolution MAR snapshot at time t. We partition the fine grid into non-overlapping square patches of size $p \times p$. The "coarse" input is

$$\textbf{DOWNSCALE}(\mathbf{x}(t)) = \bigcup_{\mathcal{P}_{i,j} \in \mathcal{P}} [A(\mathbf{x}, \mathbf{t})]_{i,j} \quad ; [A(\mathbf{x}, \mathbf{t})]_{i,j} = \frac{1}{p^2} \sum_{(u,v) \in \mathcal{P}_{i,j}} \mathbf{x}_{u,v}(t),$$

$$\mathbf{y}(\mathbf{x}(t)) = \textbf{DOWNSCALE}(\mathbf{x}(t)) + \epsilon_y \quad ; \epsilon_y \sim \mathcal{N}(0, \sigma_y^2)$$

where $\mathcal{P}_{i,j}$ is the (i,j)th patch. In our experiments, $p = 16$ (so the coarse grid is $n_x/16 \times n_y/16$). The projection of the generated samples on the observation space.

$$\widehat{\mathbf{y}}(\widehat{\mathbf{x}}(t)) = \textbf{DOWNSCALE}(\widehat{\mathbf{x}}(t))$$

5.4.3 Setup

We take 100 trajectories of 24 hours and apply the observation operator to the corresponding reanalysis data. Gaussian noise ϵ is then added to the normalized data, after which **a single posterior** is inferred. The model was trained with a 12-hour window, and to generate the full trajectory we follow Algorithm 2.1.1: for each 24-hour trajectory, we extract the 6th hour within each sample (to align roughly with the center of the local receptive window) and slide the local window 24 times across the trajectory.

The measurement noise is assumed to match the noise level used during the sampling procedure.

N	C	τ	γ	Σ_y
256	2	0.5	0.01	σ_y

Table 5.5: Parameters used in the experiment for the sampling.

5.4.4 Visual Evaluation

In the following figures [5.6, 5.7], we show the ground truth, which is a 24 hour trajectory from the MAR dataset, along with the observation y obtained by averaging over patches to simulate the coarse input from a GCM, and the corresponding conditional sample. Overall, the model appears to capture and reconstruct both the structure and fine grained details accurately, although some variability remains due to the inherent stochasticity of diffusion models

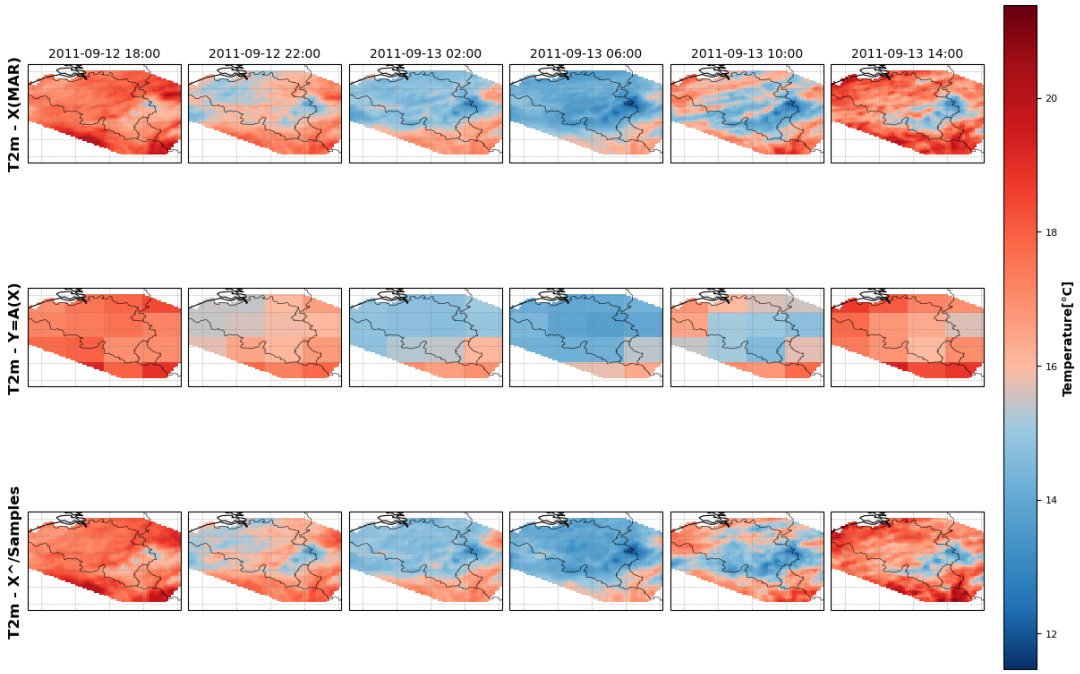


Figure 5.6: 24h MAR Temperature trajectory shown every 4 hours
a) MAR Sample b) a) upscale c) Conditional generation (downscaling) given b)

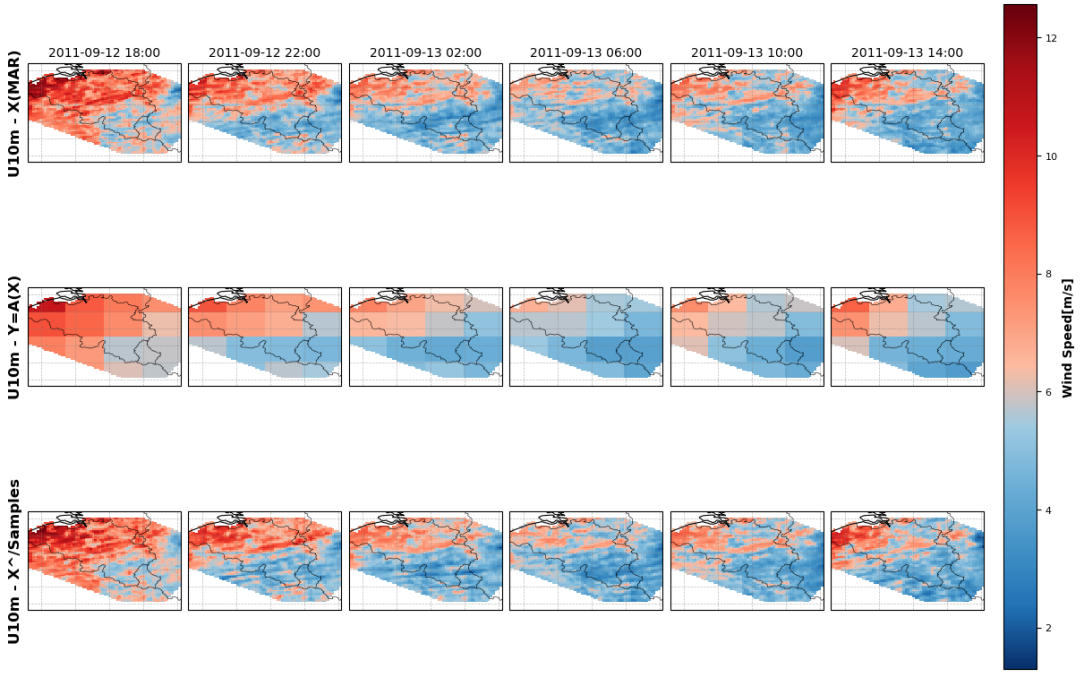


Figure 5.7: 24h MAR Wind speed trajectory shown every 4 hours
a) MAR Sample b) a) upscale c) Conditional generation (downscaling) given b)

Measurement Noise σ_y	$\widehat{\mathbf{x}}$ vs X^*				$\widehat{\mathbf{y}}$ vs y			
	NRMSE		RMSE		NRMSE		RMSE	
	T2M	U10m	T2M	U10m	T2M	U10m	T2M	U10m
0.001	0.0681	0.2728	0.5080	0.5794	0.00127	0.00112	0.00925	0.002497
0.01	0.0711	0.2807	0.5298	0.5962	0.01170	0.01194	0.08724	0.02537
0.1	0.0994	0.3218	0.7412	0.6836	0.1086	0.1170	0.8095	0.2485
1.0	0.2780	0.6608	2.0729	1.4036	1.0258	1.0601	7.6493	2.2519

Table 5.6: NRMSE and RMSE for $X^\wedge$ vs X^* and $A(X^\wedge)$ vs X^* at different observational noise levels, separated by variables T2M and U10m.

5.4.5 Quantitative Evaluation

Metrics

Interpretation

Table[5.6] confirms the expected trend that total error grows with increasing observational noise. The (N)RMSE between a single generated sample $\hat{X}$ and the MAR ground truth X^* is not zero. This is natural: a coarse-scale input (GCM) corresponds to an ensemble of plausible fine-scale realizations, so one generated draw should be interpreted as a single member of that conditional ensemble rather than as an exact reconstruction of the held-out MAR field.

The relative error between the NRMSE evaluated in the observation space (comparing $A(\hat{X})$ to y with noise applied) and the declared observational noise is at most 27% (T2m at 0.001). This indicates that, after projection into observation space, differences between generated and true fields are largely compatible with measurement uncertainty: the model produces samples that reflect the observational uncertainty. For every observational noise, $OCR \leq 1.27$ shows that the posteriors are aligned with the observations, this comparison is made when projecting the posteriors in the observation space.

5.5 Artificial Simulation Settings

5.5.1 Motivation

We extend the coarsening experiment by generating conditional samples in which only averaged patches along the boundaries are assimilated, while the core of the region evolves according to the learned dynamics of the prior (MAR). This allows us to assess whether these dynamics remain consistent with the prescribed boundaries. Note that this does not correspond to the same lateral boundary region used in MAR, rather, it is a proof of concept in which the free-running RCM domain corresponds to the area of Wallonia and Brussels.

5.5.2 Observer

Let's write :

$$DOWNSCALE(\mathbf{x}(t)) = \bigcup_{\mathcal{P}_{i,j} \in \mathcal{P}} [A(\mathbf{x}, \mathbf{t})]_{i,j} \quad ; [A(\mathbf{x}, \mathbf{t})]_{i,j} = \frac{1}{p^2} \sum_{(u,v) \in \mathcal{P}_{i,j}} \mathbf{x}_{u,v}(t),$$

$$BOUNDARY(\mathbf{x}(t)) = \bigcup_{\mathcal{P}_{i,j} \in \partial\mathcal{P}} [A(\mathbf{x}, \mathbf{t})]_{i,j} \quad ; CORE(\mathbf{x}(t)) = \bigcup_{\mathcal{P}_{i,j} \in \mathcal{P} \setminus \partial\mathcal{P}} [A(\mathbf{x}, \mathbf{t})]_{i,j}$$

where $\mathcal{P}$ denotes the set of all patches and $\partial\mathcal{P}$ the patches at the boundaries of the region. In this context, the observation is made on the boundary so :

$$\mathbf{y}(\mathbf{x}(\mathbf{t})) = BOUNDARY(\mathbf{x}(\mathbf{t})) + \epsilon \quad ; \epsilon \sim \mathcal{N}(0, \sigma_y^2)$$

and the projection of the posteriors on the observation space :

$$\widehat{\mathbf{y}}(\widehat{\mathbf{x}}(t)) = BOUNDARY(\widehat{\mathbf{x}}(t))$$

5.5.3 Setup

See Artificial Coarsen Experiment setup

5.5.4 Visual Evaluation

Figures [5.6, 5.7] present the reconstruction of the assimilation over a 1-day period. For each timestamp, we display: (1) the MAR simulation, (2) the corresponding observation obtained by applying $BOUNDARY()$ to MAR, and (3) a conditional sample generated given this observation. As will be emphasized later, the objective is not to achieve an exact match between the generated and MAR fields. Visually, the conditional sample appears to exhibit physically consistent dynamics when compared to MAR (although this requires verification). Interpreting the first and third rows as two possible realizations from the conditional distribution, given the observations in the second row, is therefore reasonable.
Figures[5.10, 5.11] show the patches at the core, applying $CORE()$ on the conditional sampling and on MAR trajectory. There is a difference. Differences are more flagrant on the wind variable. However, core patch averages are used only as a coarse sanity check. A more thorough physical plausibility check has to be assessed. Therefore, a mismatch in CORE patch means does not imply physical inconsistency.

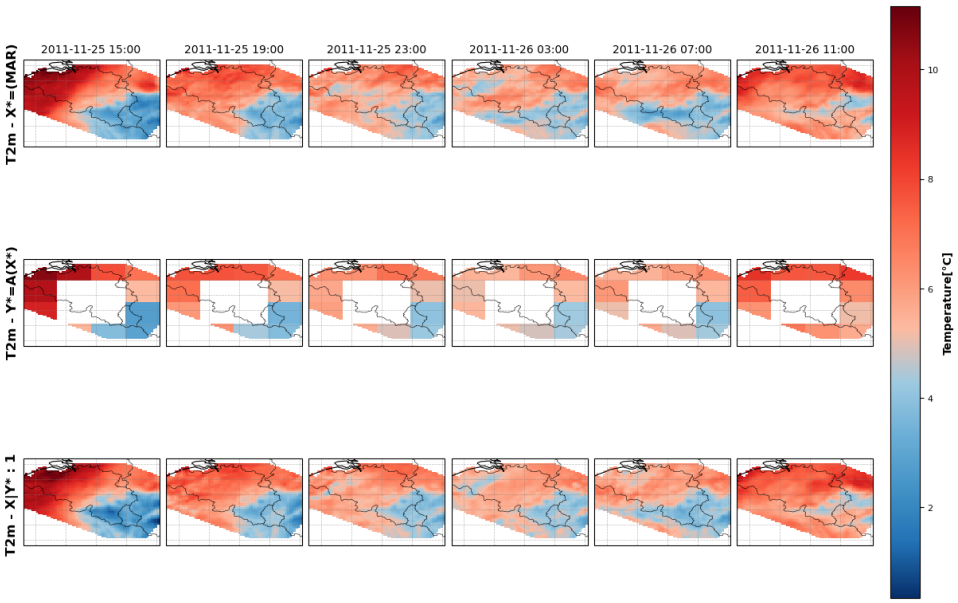


Figure 5.8: a) $\mathbf{x}$*24h MAR Temperature trajectory shown every 3 hours b) Observation : $\mathbf{y}(\mathbf{x}^*)$ c) $\hat{\mathbf{x}} \sim \hat{p}(\mathbf{x}|\mathbf{y}(\mathbf{x}^*))$

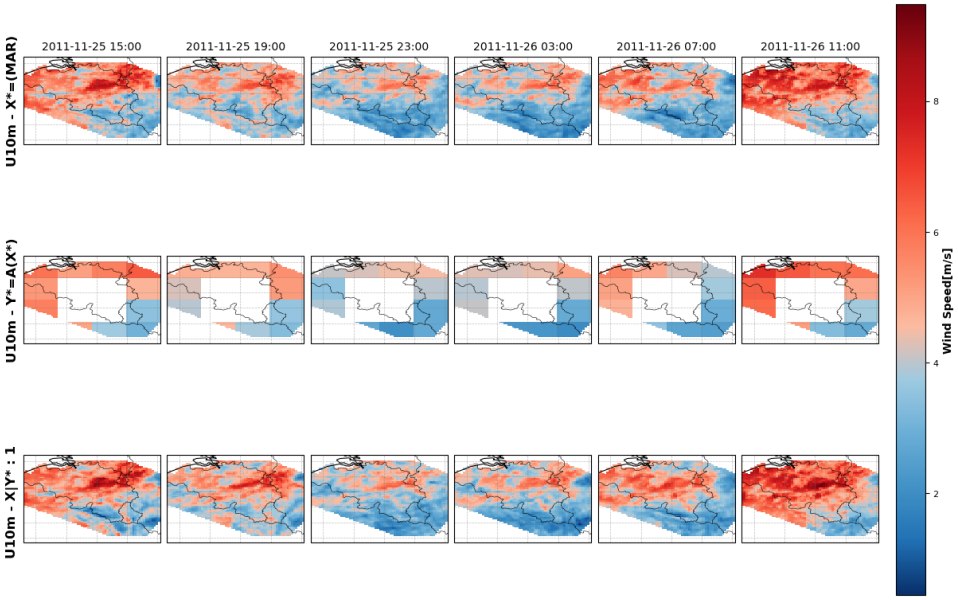


Figure 5.9: a) $\mathbf{x}^*$:24h MAR Wind speed trajectory shown every 3 hours b) Observation : $\mathbf{y}(\mathbf{x}^*)$ c) $\hat{\mathbf{x}} \sim \hat{p}(\mathbf{x}|\mathbf{y}(\mathbf{x}^*))$

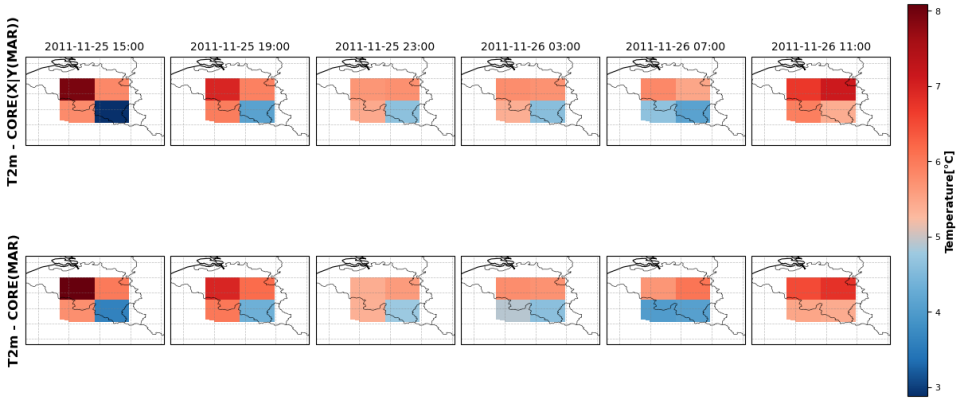


Figure 5.10: Temperature : a) *CORE*() applied to MAR b) *CORE*() applied to generated sample figure [5.6]

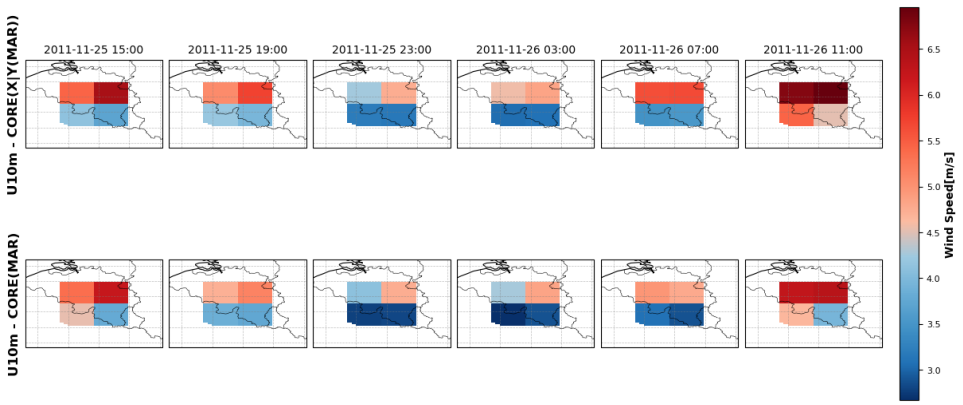


Figure 5.11: Wind speed : a) *CORE*() applied to MAR b) *CORE*() applied to generated sample figure [5.7]

Experiment	NRMSE		RMSE	
	T2m	U10m	T2m	U10m
MAR vs X|Y	0.0806	0.3294	0.6011	0.6997
CORE(MAR) vs CORE(X|Y)	0.0622	0.2808	0.4636	0.5964
$\mathbf{y}$ vs $\widehat{\mathbf{y}}$	0.0113	0.0117	0.0841	0.0248

Table 5.7: NRMSE and RMSE for T2m and U10m across a) all samples b) patches at the boundary c) patches at the core

5.5.5 Quantitative Evaluation

RMSE between 1) MAR and conditional generated sample 2) patches (after application of **BOUNDARY**() on MAR trajectory and conditional generated sample) at the boundaries and 3) patches (after application of **CORE**() on MAR trajectory and conditional generated sample) at the core. The results show that boundary patches in the conditional generation are consistent with the noisy observations, with NRMSE values close to the observational measurement noise level ($\sigma_y = 0.01$). In particular, the OCR indicates posterior captures the observations uncertainty ($OCR \leq 1.17$) and that the posteriors show neither a significant bias nor a significant variance relative to the observations This suggests that the model can effectively assimilate coarse-grained boundary information, analogous to GCM-provided lateral forcings in MAR.
In the unobserved core, RMSE values are higher, which reflects a greater posterior uncertainty due to the lack of direct observations. The latter is consistent with MAR's domain decomposition, where interior evolution is driven by free-running RCM dynamics. While visual inspection suggests physically plausible trajectories, further physical diagnostics are required to confirm dynamical consistency in the posterior samples.
The primary aim of this experiment is not to reproduce exactly the average patches of the unobserved core, but to assess whether the generated fields are consistent with the assimilated boundary conditions. Since the core region is not part of the observational input, deviations from the MAR average patches in this area are not inherently problematic. At present, we cannot determine whether a closer or further match in the core should be expected without further investigation of the posterior's physical consistency. Such an analysis would be necessary to assess whether the average patches in the core should match more closely or whether the observed differences are entirely compatible with the underlying dynamics. The observed deviations may simply reflect an alternative plausible realization of the conditional distribution given the available boundary observations, one that is as valid, from a probabilistic standpoint, as the MAR trajectory.

Furthermore, unlike traditional numerical RCMs that are predominantly deterministic, the diffusion-based approach explicitly aims to enable sampling of multiple diverse realizations conditioned on the same input. In this context, variability in the unobserved core is not necessarily a flaw but rather an intended feature, reflecting the model's capacity to represent a range of physically plausible scenarios consistent with the boundary forcings, though confirming their physical plausibility remains an essential step.

5.6 Artificial Sparse Experiment

5.6.1 Motivation

In this section, we investigate the assimilation of sparse observations over the Belgian region, aiming to replicate the data assimilation process used with weather stations. In meteorology, when only a limited number of observations, such as those from sparse weather stations, are available, the objective is to estimate the atmospheric state across the entire region.

5.6.2 Observer

To simulate this scenario, we use the locations of 29 weather stations as the observational points. At these locations, we retain the corresponding MAR values while discarding all others

$$H : \mathbb{R}^{n_x \times n_y} \to \mathbb{R}^N$$

$$\mathbf{y}_{sparse} = H(\mathbf{x}(t)) + \epsilon$$

$$\widehat{\mathbf{y}}_{sparse} = H(\widehat{\mathbf{x}}(t))$$

5.6.3 Setup

See Artificial Coarsen Experiment

5.6.4 Visual Evaluation

From a qualitative perspective, figures [5.12, 5.13] show that the reconstructions closely match the observations and appear to originate from the same distribution as MAR (ground truth). Some variability is noticeable outside the Belgian territory, which can be attributed to the lack of weather stations and corresponding observations in those regions, leading to a less constrained posterior. In contrast, in figures [5.15, 5.14], the observations contain higher noise levels ($\sigma = 1$), resulting in samples that deviate more noticeably from the corresponding MAR fields.

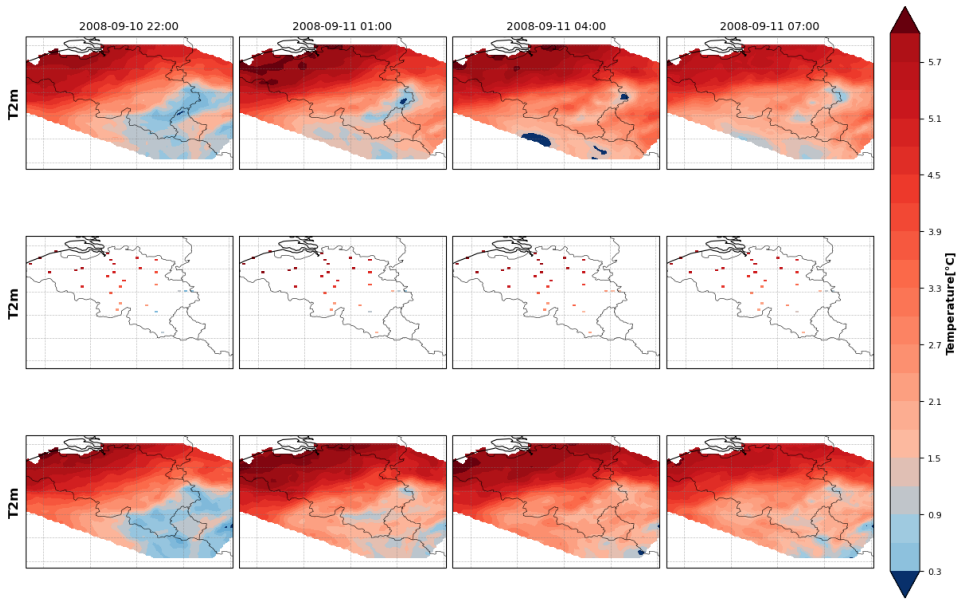


Figure 5.12: 12h MAR Temperature trajectory shown every 3 hoiurs a) MAR Sample b) a) at weather station positions with noise $\sigma = 0.01$ c) Conditional Generation given b)

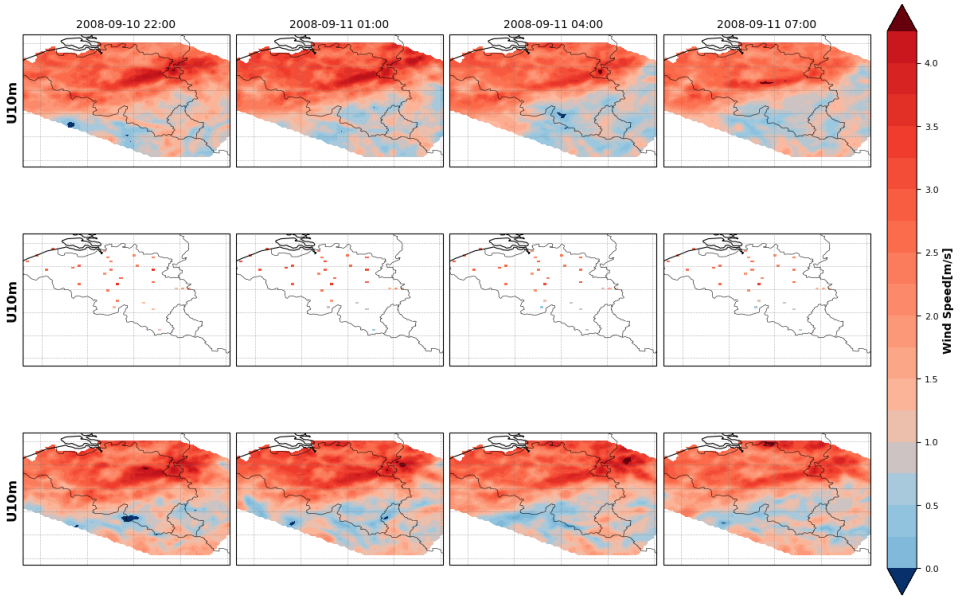


Figure 5.13: 12h MAR Wind speed trajectory shown every 3 hours a) MAR Sample b) a) at weather station positions with noise $\sigma = 0.01$ c) Conditional Generation given b)

5.6.5 Quantitative Evaluation

Measurement noise σ_y	$X^\wedge$ vs X^*				$\mathbf{y}$ vs $\widehat{\mathbf{y}}$			
	NRMSE		RMSE		NRMSE		RMSE	
	T2M	U10m	T2M	U10m	T2M	U10m	T2M	U10m
0.001	0.1239	0.3825	0.9239	0.8124	0.0076	0.0063	0.05645	0.01339
0.01	0.1337	0.5456	0.9966	1.1589	0.0086	0.0068	0.06421	0.0145
0.1	0.1645	0.5367	1.2113	1.1401	0.1097	0.1134	0.8178	0.2408
1.0	0.2696	0.6375	2.0101	1.3541	0.9975	1.0757	7.4379	2.2850

Table 5.8: NRMSE and RMSE for $X^\wedge$ vs X^* and $A(X^\wedge)$ vs X^* at different observational noise levels, separated by variables T2M and U10m.

5.6.6 Interpretation

Only 29 out of 2846 grid points ($\approx 1\%$) are directly observed.

In the **sample space** ($\hat{X}$ vs. X^*), the NRMSE for T2M increases from 0.124 to 0.270 as σ_y rises from 0.001 to 1.0 (a factor of ≈ 2.18), while for U10m it grows from 0.383 to 0.638 ($\approx 1.67\times$). These increases are monotonic but modest, indicating that higher observational noise is only partly reflected in the full-field reconstruction errors.

In the **observation space** ($A(\hat{X})$ vs. $A(X^*)$), the OCR values often fall below 1, which suggests that the number of observations may be insufficient since values below 1 would imply surpassing the measurement noise. However, the OCR values generally range around 1 ($0.63 < OCR < 1.08$), indicating that for these 100 observations, the posteriors align well with the observations. This implies that the posteriors are neither significantly biased nor exhibit excessive variance at the weather station locations relative to the corresponding observations.

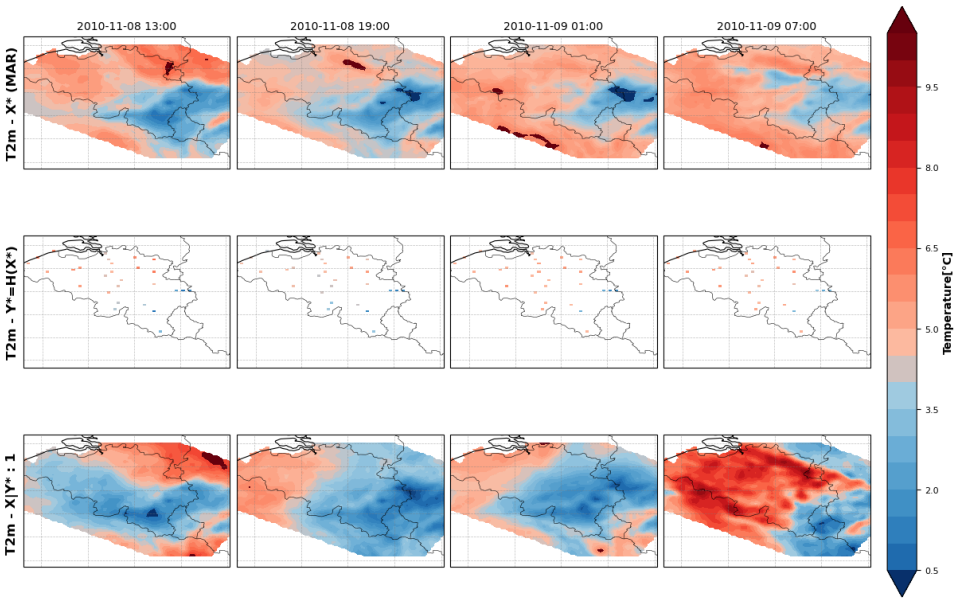


Figure 5.14: **Noisy Observations :** 24h MAR Temperature trajectory shown every 6 hours a) MAR Sample b) a) at weather station positions with noise $\sigma = 1$ c) Conditional Generation given b)

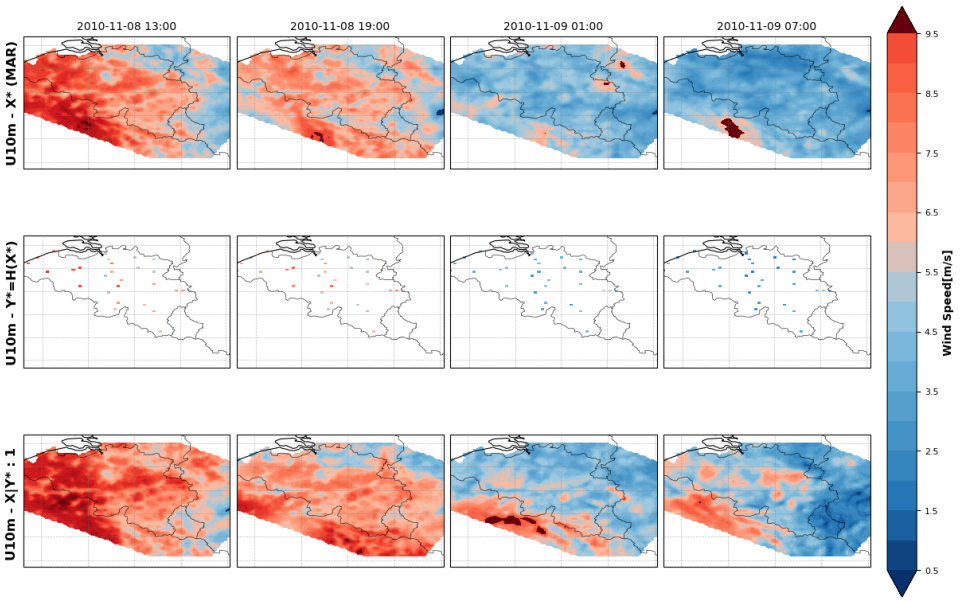


Figure 5.15: **Noisy Observations :** 24h MAR Wind speed trajectory shown every 6 hours a) MAR Sample b) a) at weather station positions with noise $\sigma = 1$ c) Conditional Generation given b)

At $\sigma_y = 1.0$, the observation-space NRMSE (≈ 1) exceeds the sample-space NRMSE (T2M: 0.270, U10m: 0.638). A possible, but unsupported, interpretation is that with such sparse coverage ($\sim 1\%$) and noisy measurements, the reconstruction is guided more by the learned prior than by the observations outside the measurement locations.

5.7 Guidance on IRM Dataset

5.7.1 Observer

See Artificial Sparse Experiment, noting that there is only projection of samples is noted $\widehat{y} = H(\widehat{\mathbf{x}}(\mathbf{t}))$ and that $\mathbf{y}$ correspond to the IRM data

N	C	τ	γ	Σ_y
206	3	0.107	0.0376	0.006

Table 5.9: Parameters used in the experiment for the sampling.

5.7.2 Qualitative Results

Samples are generated from observations recorded at weather stations. Figures 5.16 and 5.17 illustrate, for a 12-hour window (with timestamps shown every 3 hours on a specific date), the following: the corresponding MAR output variable, the MAR values at the weather station locations, the IRM values at the same date, the reconstruction (or conditional sampling) obtained from the sparse IRM observations, and finally the reconstructed values at the weather station locations.

The observation model is the same as in the previous subsection, namely a Linear Gaussian model.

Figures 5.18 and 5.19 present the differences, expressed in z-score normalization units (where 1.0 corresponds to the variable's variance), between (i) the MAR values at the weather station locations and the IRM values at the same locations, and (ii) the conditional sample values at the weather station locations and the IRM values. As expected, the first difference confirms that MAR and IRM are misaligned both spatially and temporally, as discussed earlier. More importantly, the second difference is close to zero for both variables, details on these values are provided later, indicating that the generated samples effectively incorporate the IRM observations.

5.7.3 Quantitative Results

Assimilating all IRM Weather stations

We quantify both difference using NRMSE across all values and timesteps at weather station locations (figures [5.18,5.19]).

$A(MAR)$ vs IRM				$A(X\|IRM)$ vs IRM / $\widehat{\mathbf{y}}$ vs $\mathbf{y}$			
NRMSE		RMSE		NRMSE		RMSE	
T2M	U10m	T2M	U10m	T2M	U10m	T2M	U10m
0.3304	0.4748	2.463	1.0086	**0.0075**	**0.0056**	0.0562	0.0118

Table 5.10: NRMSE and RMSE for $A(MAR)$ vs IRM and $A(X|IRM)$ vs IRM considering a gaussian observer with the estimated observation process noise, separated by variables T2M and U10m.

For A(MAR) vs. IRM, the NRMSE values are 0.330 (T2M) and 0.475 (U10m). These relatively high errors indicate a marked discrepancy between MAR and IRM at station locations, consistent with the spatial and temporal misalignment discussed earlier. The corresponding RMSE values (2.463 for T2M, 1.009 for U10m) underline the same mismatch in absolute units.

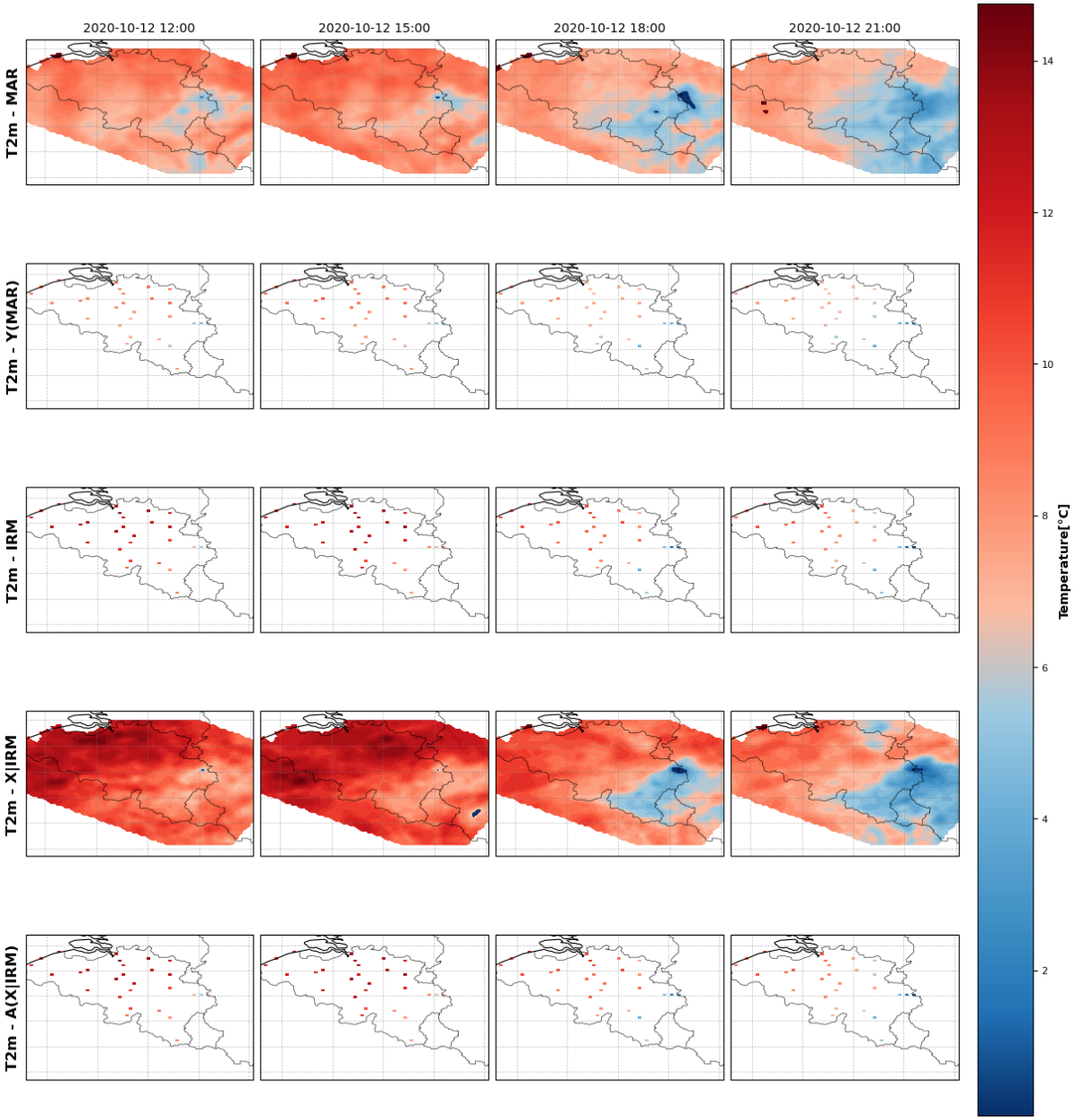


Figure 5.16: a) T2M MAR trajectory ; b) observer applied to a ; c) IRM data at the same timestamp (TEMP) ; d) Conditional Sampling given c; e) observer applied to d

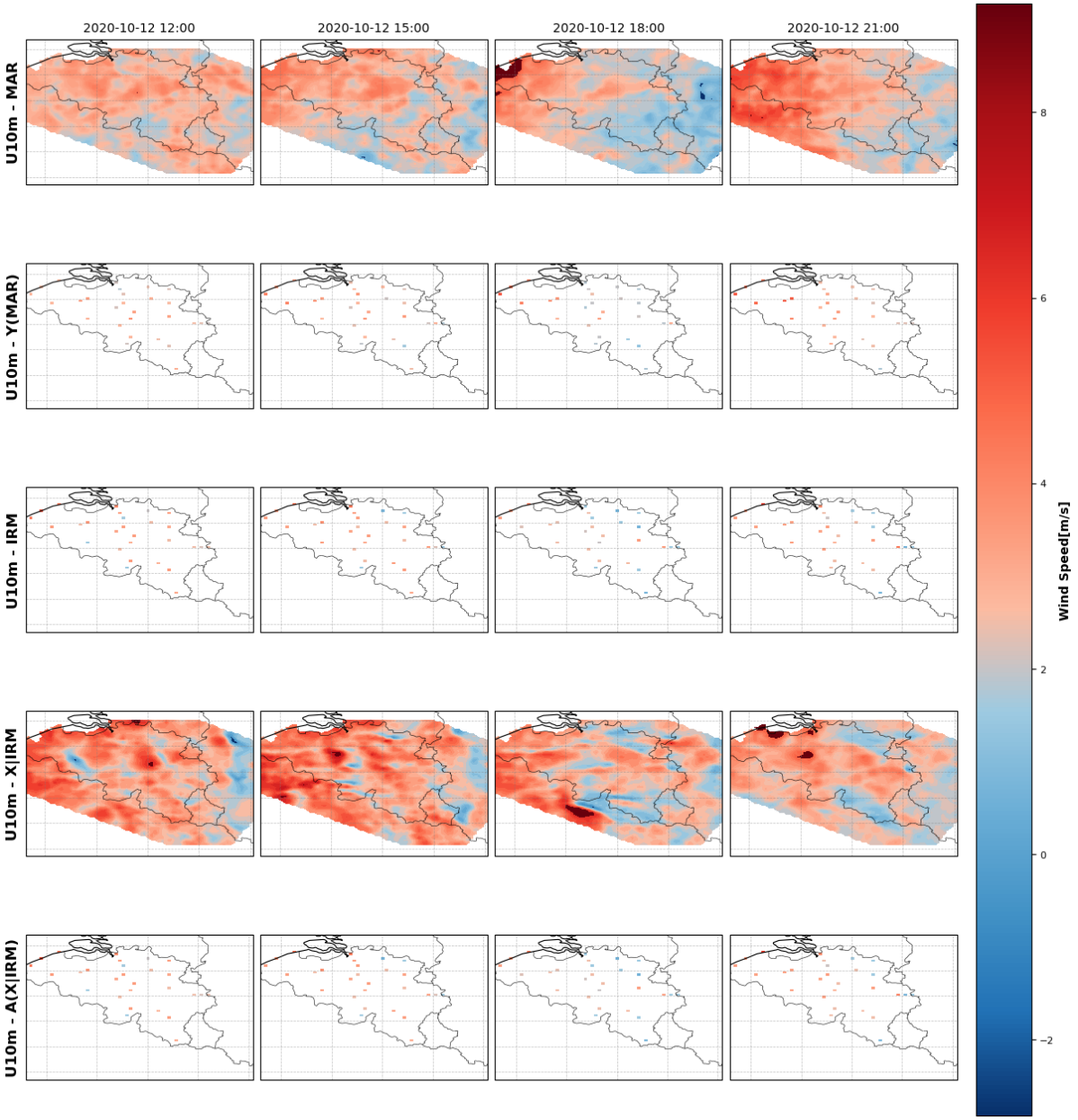


Figure 5.17: a) U10m MAR trajectory ; b) observer applied to a ; c) IRM data at the same timestamp (WIND SPEED) ; d) Conditional Sampling given c; e) observer applied to d

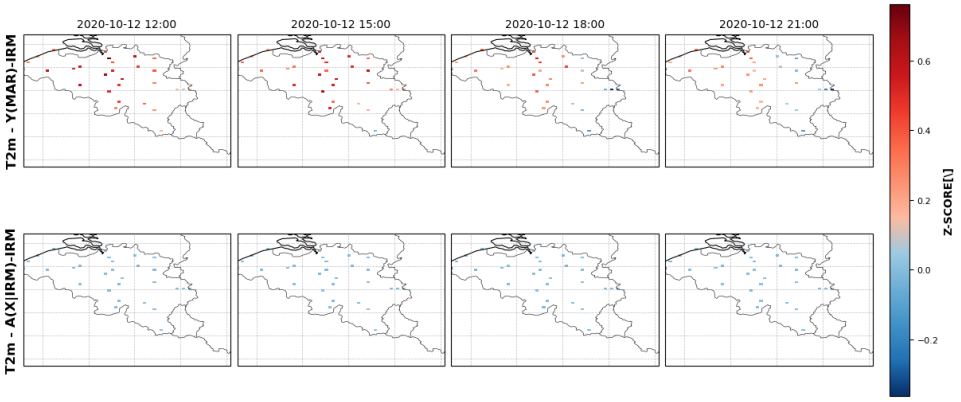


Figure 5.18: a) Difference between Y(MAR) and IRM for the Temperature normalized b) Difference between IRM and A(X|IRM)

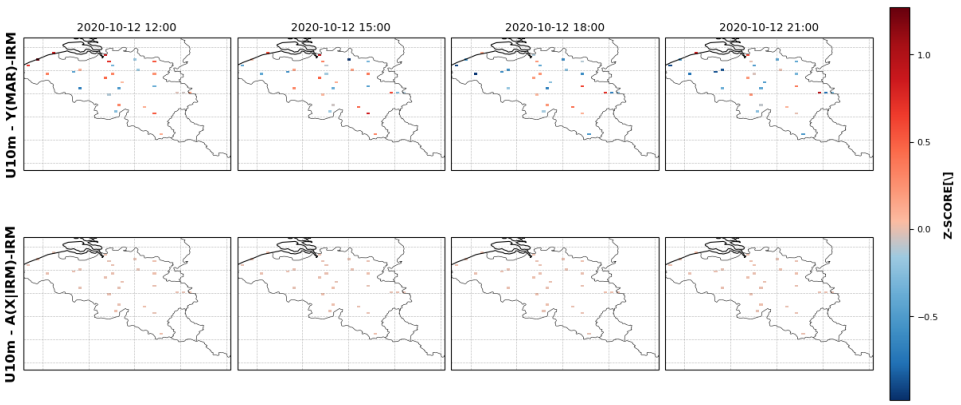


Figure 5.19: a) Difference between Y(MAR) and IRM for the Temperature normalized b) Difference between IRM and A(X|IRM)

In contrast, for $A(X \mid \text{IRM})$ vs. IRM, the NRMSE values decrease substantially to 0.0075 (T2M) and 0.0056 (U10m). The reduced NRMSE posterior in the reconstructions indicates align far better with the IRM observations than the original MAR fields.

Assimilating increasing number of IRM Weather stations

Procedure The goal of this experiment is to assess whether, given a sufficient number of weather stations, we can generate MAR trajectories that better align with real-world observations. Following the methodology of [40], we progressively increase the number of weather stations used for assimilation. For each number of assimilation stations, we perform 10 evaluation runs, each with a random split of the available stations into an observed set (assimilation/training) and a left-out set (test points). The number of assimilation stations is fixed within each run, while the split changes.

The observation process is applied only to the selected assimilation stations in each split. These are used to generate conditional samples, and performance is evaluated on the left-out stations. We use the root mean squared error (RMSE) as the evaluation metric, with conditional generation performed on IRM data as described below.

Two RMSE values are computed: (i) the RMSE between the generated samples and IRM at the test points, and (ii) the RMSE between MAR and IRM at the test points. The latter is expected to remain relatively constant regardless of the number of assimilation stations, as it mainly reflects the inherent differences between MAR outputs and IRM values at weather stations.

If $RMSE_{\text{SAMPLE-IRM}}$ falls below $RMSE_{\text{MAR-IRM}}$, it would indicate that the generated samples are better aligned with IRM observations on the non-assimilated (test) points than the baseline MAR outputs.

Algorithm 5 Evaluation Protocol for Conditional MAR Trajectories

Input:

- Total stations: NB_IRM (e.g., 29)
- Batch repetitions: NB_BATCH
- Time window size: Δt

Initialize: Results = {}
for train_size = 1 **to** NB_IRM **do**
  for $n = 1$ **to** NB_BATCH **do**
    date $\sim \mathcal{U}$(available dates) {Random timestamp}
    IRM $\leftarrow$ getIRM(date, date + Δt) {Real observations}
    BATCH $\leftarrow$ getBatch(date, date + Δt) {MAR model output}
    train_mask, test_mask $\leftarrow$ RandomSplitMask(train_size, NB_IRM)
    $y_{\text{BATCH\_train}} \leftarrow A(\text{BATCH}, \text{train\_mask})$ {Retained stations}
    $y_{\text{BATCH\_test}} \leftarrow A(\text{BATCH}, \text{test\_mask})$ {Left-over stations}
    $y_{\text{IRM\_test}} \leftarrow$ IRM[test_mask]
    $x_{\text{sample}} \leftarrow$ ConditionalSampling($y = y_{\text{BATCH\_train}}, A = A(\cdot, \text{train\_mask})$)
    $y_{\text{sample\_test}} \leftarrow A(x_{\text{sample}}, \text{test\_mask})$
    RMSE_irm_sample $\leftarrow RMSE(y_{\text{IRM\_test}}, y_{\text{sample\_test}})$
    RMSE_mar_sample $\leftarrow RMSE(y_{\text{BATCH\_test}}, y_{\text{sample\_test}})$
    Store Results(train_size, n, RMSE_irm_sample, RMSE_mar_sample)
  end for
end for

The results of this experiment are shown in the figure below:

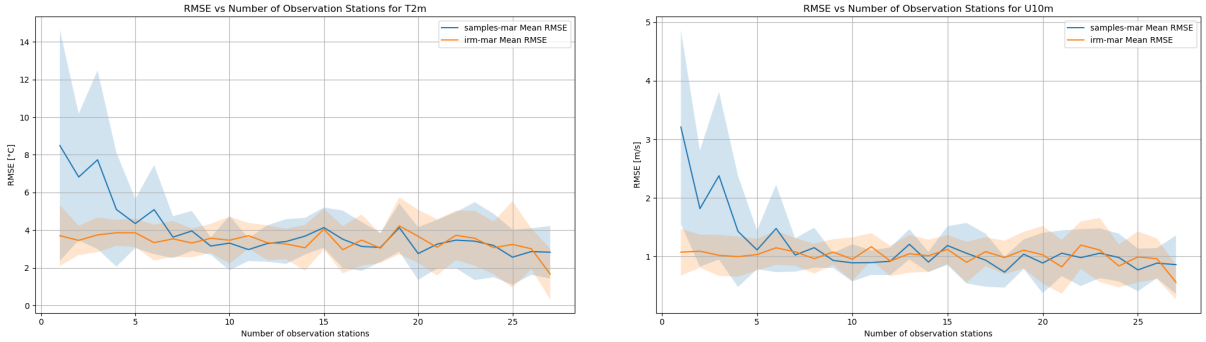


Figure 5.20: Average RMSE on the left-over weather station locations for varying numbers of assimilated weather stations.

Even when assimilating a large number of weather stations, the RMSE for the generated samples remains almost identical to the MAR-IRM RMSE. This suggests that the assimilation did not lead to a closer alignment between the generated samples and the IRM station observations. The likely reason is threefold.

First, limitations in the representativeness of the learned prior prevent the posterior from producing spatial patterns that both reproduce the assimilated station values and remain accurate at unobserved locations.

Second, while MAR and IRM both aim to represent the same physical quantity (e.g., temperature at 2 m), they measure it differently.For example, MAR provides hourly means, whereas IRM uses a 10-minute (or even 1-minute) mean within the previous hour. Combined with differences in spatial representativeness (a pointwise station measurement vs. a 5 km $\times$ 5 km grid cell), this creates a non-linear mismatch in both time and space. A naive, centered Gaussian observation model is too limited to capture such mismatches.

Third, unlike in the Oklahoma experiment [40], the dataset used to train our diffusion model (MAR) never assimilated IRM station data. The prior therefore has no built-in knowledge of IRM-specific measurement characteristics. As a result, even when many IRM stations are provided during assimilation, the model cannot reproduce the dynamics of left-out stations because of the combination of the naive observation model and the complete lack of IRM-specific information in MAR.

5.7.4 Summary

Conditioning on IRM values produces posteriors that are more consistent with the values of the IRM weather stations whenever they are assimilated. However, the generated conditional sample trajectory is unable to predict IRM station values that are not assimilated. This is likely due to the lack of representativeness of the prior and the simplicity of the observation process relative to the complex, non-linear mismatch between the two datasets. This mismatch stems from factors such as differing measurement procedures, the absence of IRM assimilation in MAR's physical generator, and spatial discrepancies.

5.7.5 To Go Further

As discussed earlier, a fundamental issue is the mismatch between the MAR and IRM datasets. One source of this discrepancy lies in the differences in how the variables are defined and measured. For instance, the averaging periods and measurement protocols differ between the two sources.

A possible approach to mitigate this mismatch could be to average the hourly data over a full day. This would help align the MAR data's averages with more temporally aggregated IRM observations. However, this would also entail transitioning to daily data, which introduces new challenges. Training a model with a time step of one day over multi-day windows becomes more difficult due to the significant variability between consecutive days, despite some variables exhibiting daily cycles (e.g., solar radiation patterns).

This remains an open direction for future research.

5.8 Guideance on reanalysis single levels ERA-5

5.8.1 Motivation

MAR trajectories are generated by constraining the domain boundaries using input from a coarser-resolution General Circulation Model (GCM). For instance, when MAR is applied over Belgium, the boundary conditions are derived from ERA5 reanalysis data, including several variables across multiple pressure levels.

In these experiments, we rely on single-level ERA5 data, which differs from the full set of physical variables across several pressure levels typically used to generate MAR trajectories. While this setup does not fully replicate the physical simulation pipeline behind MAR, it offers an interesting approximation: by taking only the GCM boundary inputs, we attempt to reconstruct fine-grained MAR-like outputs across the entire region. This approach can be seen as a simplified, one-step abstraction of the complete MAR simulation process. The two experiments are the following :

1. Downscaling Following experiments using artificially coarsened data, we aim to perform downscaling over the full region using actual GCM data.

2. Basic Simulator We condition on the ERA-5 data but only at the boundaries rather than the full region.

Setup

Parameters used in these experiments

N	C	τ	γ	Σ_y
256	2	0.3	0.1	0.01

Table 5.11: Parameters for the sampling

5.8.2 Downscaling

Visualization

Figure[5.21, 5.22] demonstrates the downscaling of ERA-5 ($T2M_{ERA} \rightarrow T2M$, $U10M_{ERA} \rightarrow U10M$) on the first two lines on a timescale of 24 hours, here beginning the third January 2020. The last line shows the sample in the observational space after application of the observer ($T2M \rightarrow \widehat{T2M_{ERA}}$, $U10M \rightarrow \widehat{U10M_{ERA}}$) allowing for a comparison with the observation (ERA-5). The projection of the posterior in the observational space is consistent with the observation.

Quantitative Results

Experiment	NRMSE		RMSE	
	T2m	U10m	T2m	U10m
$\mathbf{y}$ vs $\widehat{\mathbf{y}}$	0.0037	0.0042	0.0277	0.0090

Table 5.12: NRMSE and RMSE for T2m and U10m

Table[5.12] shows that the NRMSE between the observation and projection of the posterior in the observational space(Coarsen/ERA-5 Grid) is small (< 0.01 in normalized units so less than 1% of the variance of the variables) , indicating that the posteriors are highly consistent with the observations.

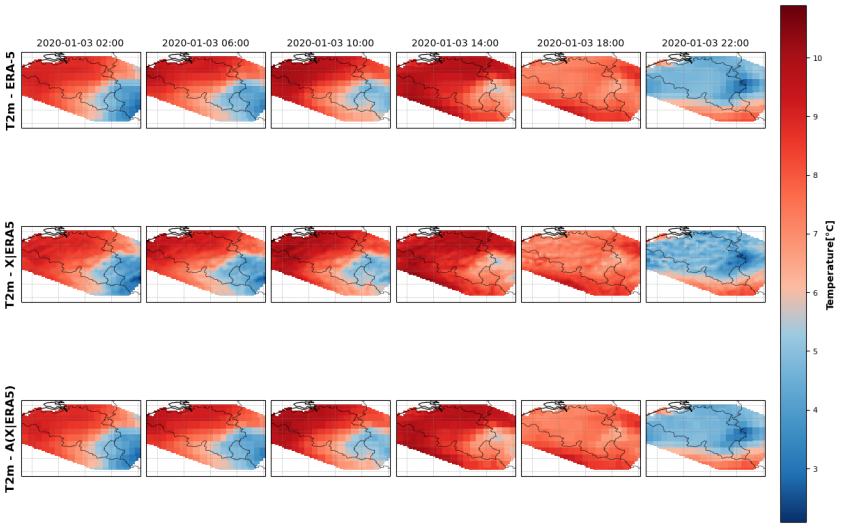


Figure 5.21: Illustration of the downscaling ERA-5 on a trajectory of 24 hours (shown every 4 hours) of Temperature.
a) $\mathbf{y}$: ERA-5 Observation b) $\hat{\mathbf{x}} \sim \hat{p}(\mathbf{x}|\mathbf{y})$ Sample conditioned on the observation c) $\hat{\mathbf{y}} = \mathcal{A}(\hat{\mathbf{x}})$ Application of the observer on the posterior.
The posterior is consistent with the observation.

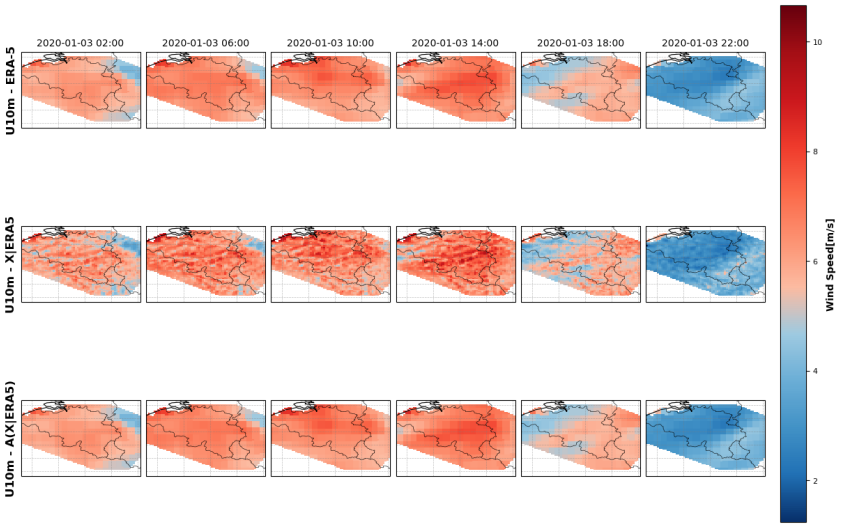


Figure 5.22: Illustration of the downscaling ERA-5 on a trajectory of 24 hours (shown every 4 hours) of wind speed.
a) $\mathbf{y}$: ERA-5 Observation b) $\hat{\mathbf{x}} \sim \hat{p}(\mathbf{x}|\mathbf{y})$ Sample conditioned on the observation c) $\hat{\mathbf{y}} = \mathcal{A}(\hat{\mathbf{x}})$ Application of the observer on the posterior.
the posterior is consitent with the observation.

5.8.3 Elementary MAR Simulator

The ERA-5 values at the region boundaries are used to condition the model. The observer is adapted to account only for pixels located along the boundaries of the MAR domain. Boundary data from the GCM (ERA-5) for single-level variables (temperature and wind speed) at an hourly resolution serve as input to generate conditional samples.

Visual Evaluation

Figures 5.23 and 5.24 show a posterior sample conditioned on observations of the coarse lateral boundary conditions. The posterior is projected onto both the observational space and the ERA-5 grid. Overall, the sample appears consistent, which is encouraging for future research on MAR simulations.
Figures 5.25 and 5.26 illustrate an ensemble of posterior samples generated from the same initial conditions. Subtle differences can be observed within the domain, while near the boundaries, the samples become more homogeneous.

Quantitative metrics

Basic Simulator Table 5.13 compares the RMSE between the observations (domain boundaries) and a posterior sample, projected (1) at the boundaries of the domain and (2) onto the ERA-5 grid. Since the coarsen boundaries of the posterior have less than or equal to 1% derivation of the corresponding standard derivation of the variable with respect to the observation, we conclude that the posterior aligns with the lateral boundary conditions. Meanwhile, in the core of the region, a higher standard deviation is expected.

Diverse Ensemble Table 5.14 shows the average normalized standard deviation of the ensemble, which is smaller at the coarsened boundaries than over the entire region. This indicates that the posteriors exhibit less variability at the boundaries (below 1%) compared to the whole region.

$BOUNDARY(ERA-5)$ vs $BOUNDARY(\widehat{\mathbf{X}})$				$ERA-5$ vs $UPSCALE(\widehat{\mathbf{X}})$			
NRMSE		RMSE		NRMSE		RMSE	
T2M	U10m	T2M	U10m	T2M	U10m	T2M	U10m
0.010	0.0093	0.0788[°C]	0.0198[m/s]	0.1362	0.5013	1.0161[°C]	1.0648 [m/s]

Table 5.13: NRMSE and RMSE are computed between observations and samples projected onto the observational space, as well as between ERA-5 (used for boundary construction) and samples projected onto the ERA-5 grid.

Experiment	Standard Deviation of the ensemble	
	T2m	U10m
$\sigma(BOUNDARY(\widehat{\mathbf{x}}))$	0.00632	0.00771
$\sigma(UPSCALE(\widehat{\mathbf{x}}))$	0.03725	0.16324

Table 5.14: **Normalized** Standard deviation of 10 ensemble members given one observation, averaged over 100 observations, for the elementary MAR simulator experiment.
First line shows the standard deviation on the observational space.
Second line shows the standard deviation when the ensemble samples are projected on the ERA grid.

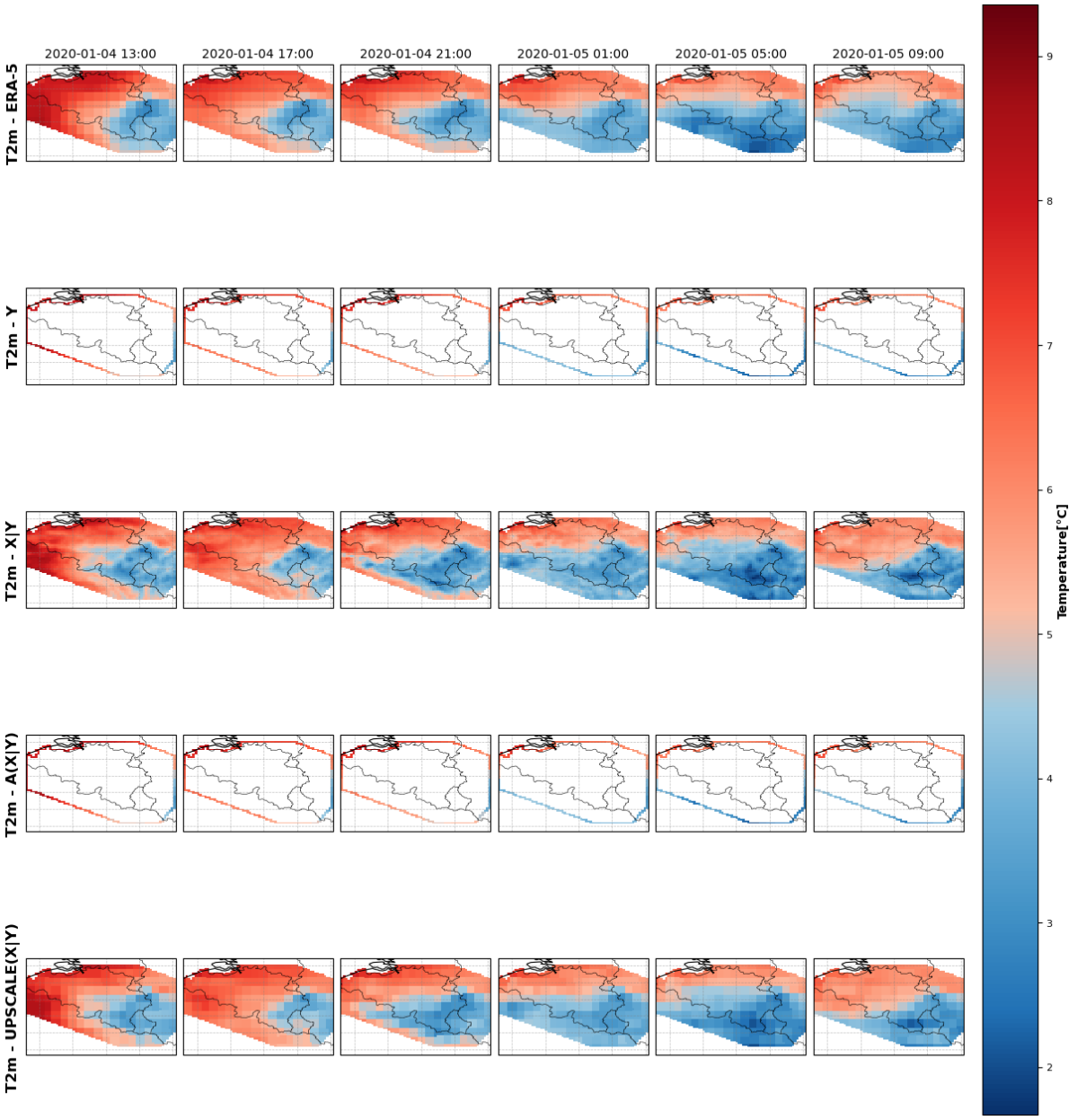


Figure 5.23: Temperature : Trajectory of 24 hours (shown every 4 hours) a) ERA-5 Trajectory b) $\mathbf{y}$: Observation ERA-5 at the lateral boundaries of the domain c) $\hat{\mathbf{x}} \sim \hat{p}(\mathbf{x}|\mathbf{y})$: Conditional sample given the observation d) $\hat{\mathbf{y}} = \mathcal{A}(\hat{\mathbf{x}})$ Projection of the sample on the observational space (Downscaling on the lateral boundaries) e) Projection of the sample on the ERA-GRID (Downscale on the whole region)

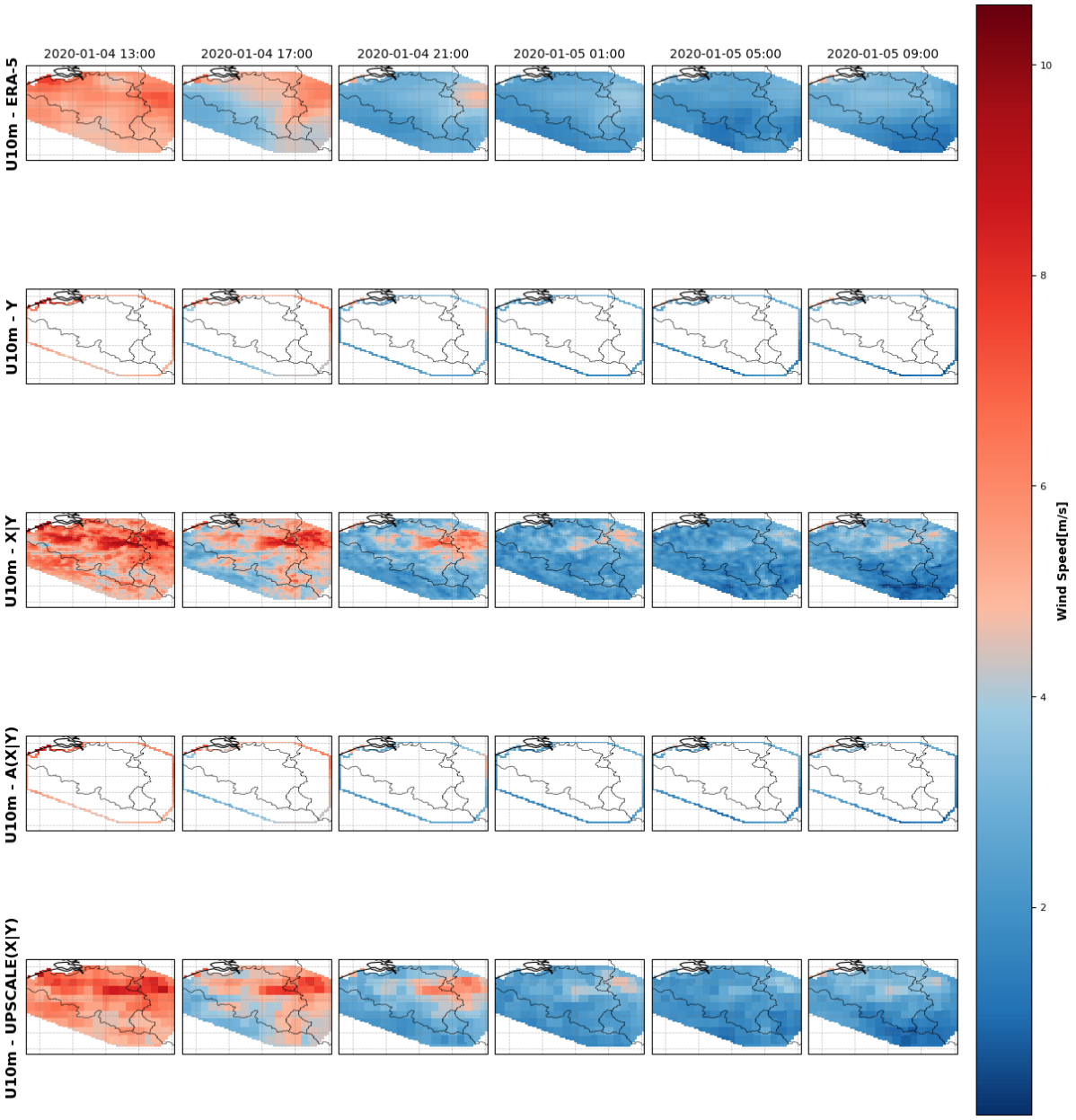


Figure 5.24: Wind speed : Trajectory of 24 hours (shown every 4 hours) a) ERA-5 Trajectory b) $\mathbf{y}$: Observation ERA-5 at the lateral boundaries of the domain c) $\hat{\mathbf{x}} \sim \hat{p}(\mathbf{x}|\mathbf{y})$: Conditional sample given the observation d) $\hat{\mathbf{y}} = \mathcal{A}(\hat{\mathbf{x}})$ Projection of the sample on the observational space (Downscaling on the lateral boundaries) e) Projection of the sample on the ERA-GRID (Downscale on the whole region)

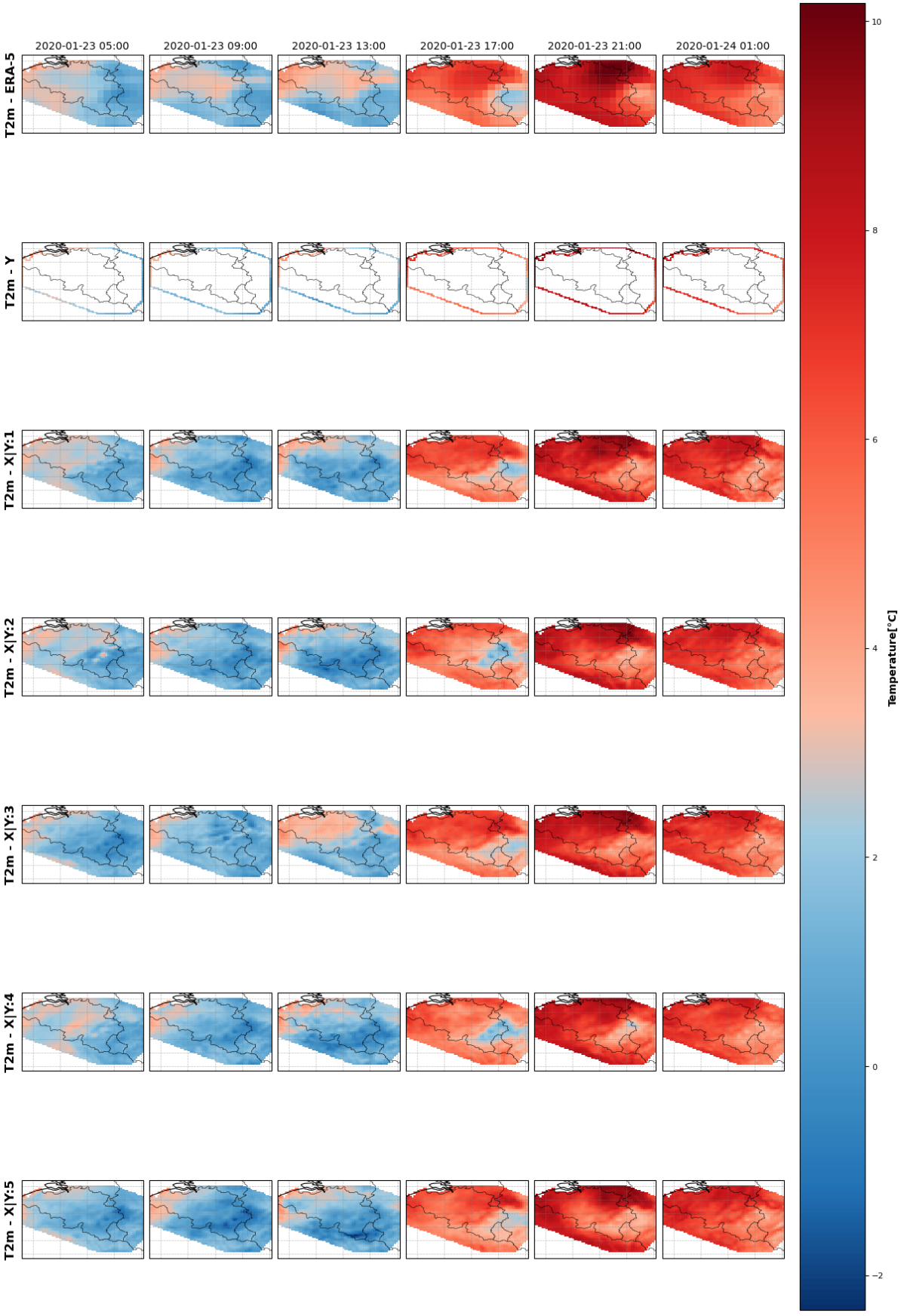


Figure 5.25: Visualization of the Temperature an ensemble of posteriors conditioned with the same lateral boundaries on a timescale of 24 hours shown every 4 hours.
a) ERA-5 trajectory b) $\mathbf{y}$: Observation : ERA-5 at the lateral boundaries (e-g) $\hat{\mathbf{x}} \sim p(\mathbf{x}|\mathbf{y})$ Ensemble of conditional samples.

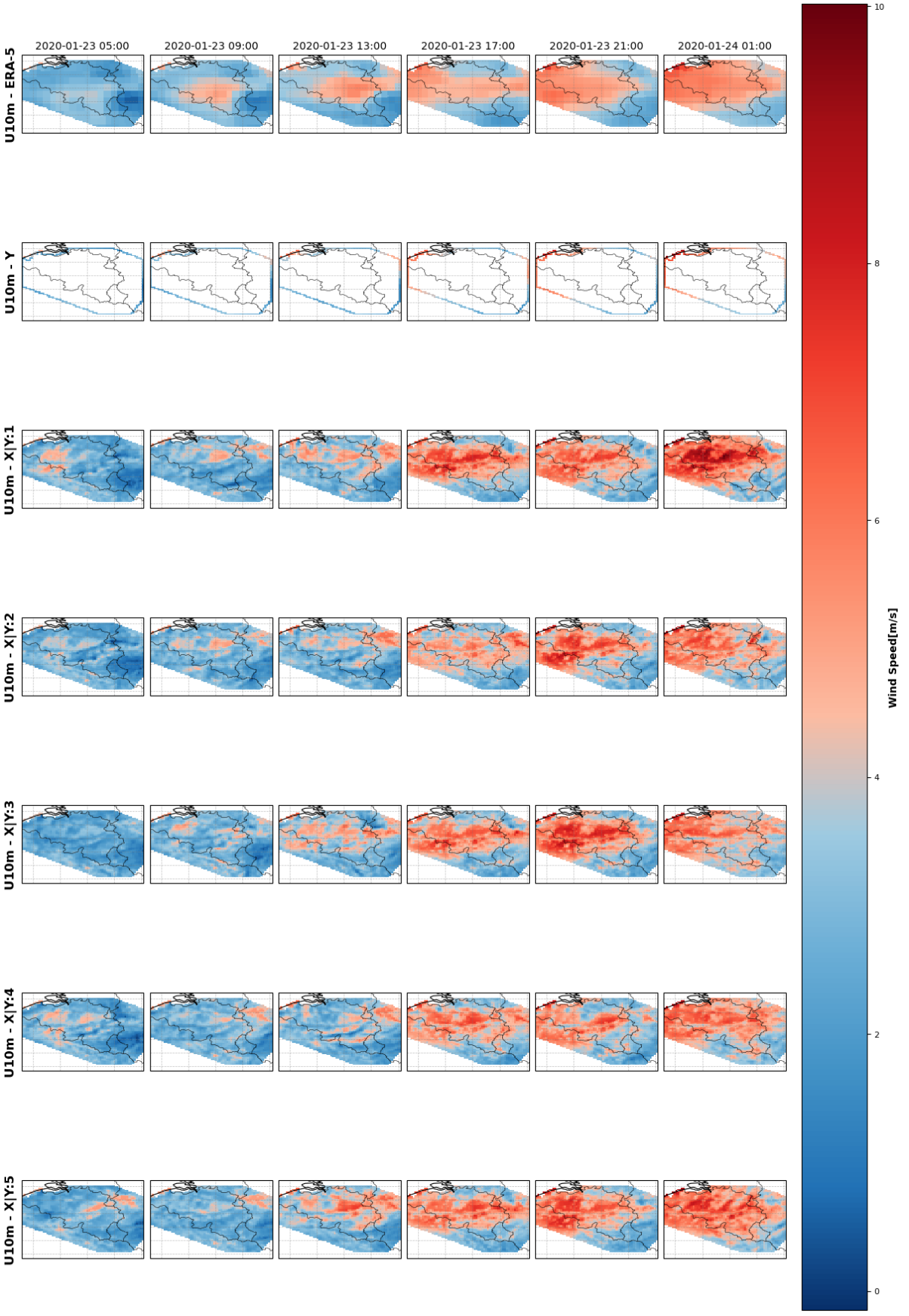


Figure 5.26: Visualization of the Wind speed an ensemble of posteriors conditioned with the same lateral boundaries on a timescale of 24 hours shown every 4 hours.
a) ERA-5 trajectory b) $\mathbf{y}$: Observation : ERA-5 at the lateral boundaries (e-g) $\hat{\mathbf{x}} \sim p(\mathbf{x}|\mathbf{y})$ Ensemble of conditional samples.

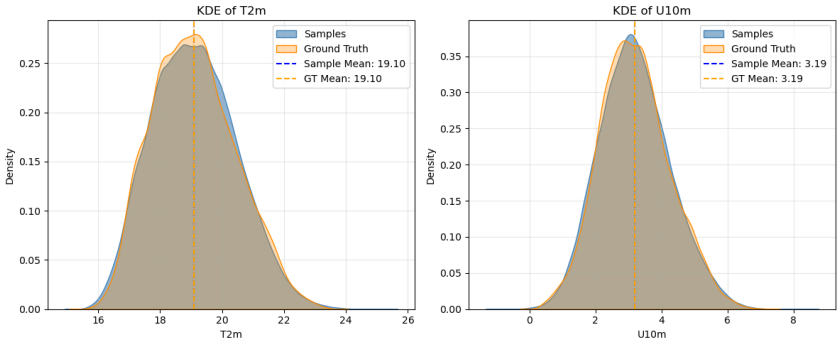


Figure 5.27: Kernel Density Estimation of the MAR trajectory and the 50 posterior samples. The means are aligned, but there is a noticeable mismatch in variability.

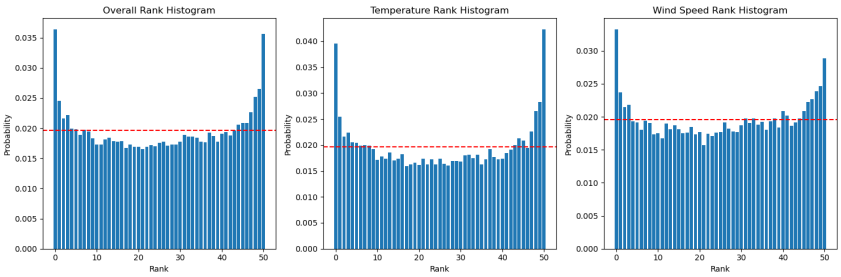


Figure 5.28: Rank histograms for both variables. The U-shape suggests that the ensemble is underdispersive, with the generated posteriors underestimating the uncertainty relative to the ground truth.

5.9 Ensemble Posterior and Model Calibration

5.9.1 Method

Following the approach in [41], we generate posterior samples conditioned on a single observation, namely a coarsened MAR trajectory over 12 hours (13th February 2011, from 9:00 to 21:00). The objective is to compare the posterior distribution with the prior. For each observation, 50 samples are produced within the 12-hour assimilation window.
The observer and experimental setup are the same as those defined in the Artificial Coarsening Experiment. Two diagnostics are considered. First, we compare the kernel density estimations (KDE) [42] of both the "ground truth" and the ensemble samples to identify potential biases or large discrepancies. Second, we analyze rank histograms. While conclusions are limited due to the single observation, these diagnostics provide a first indication of the ensemble sampling performance.

5.9.2 Visualization

5.9.3 Interpretation

Figure 5.27 shows the kernel density estimations of the MAR trajectory and the 50 posterior samples. While the ensemble mean is well aligned with the observation, clear discrepancies remain in the variability of the two distributions. Figure 5.28 presents the rank histograms of the ground truth relative to the posterior ensemble. The U-shape indicates that the ensemble is underdispersive: the observed values frequently fall outside the range spanned by the ensemble. Although extrapolation from a single observation is delicate, these diagnostics suggest that the sampling strategy

is suboptimal and/or that the prior has not sufficiently captured the dynamics of MAR.Further investigation with additional observations would be needed to confirm this behavior.

5.10 30 variables

5.10.1 Setup

We consider a MAR trajectory spanning three consecutive hours, downscale it using the observer defined in the artificial coarsening experiment, and generate 50 posterior samples.

N	C	τ	γ	Σ_y
256	2	0.3	0.01	0.001

Table 5.15: Parameters used in the experiment.

5.10.2 Qualitative Evaluation

On the experiment with all the variables , the following figure shows the decomposition of all variables where for the generated samples, the median of the spectal decomposition is shown : As shown on the figure 5.29, the model struggles to capture how "threshold" variables work. In fact, referring to the figure[4.1], the spectral decomposition do not align when the variable has a lot of constant values and sometimes some other real values (the model has to predict whether the variable is on/off (constant) and when it is on what is the exact value of this on). These variables are often related to runoff, rainfall, snowfall when often for example there is no rainfoll nor snowfall and if there is one, the model should predict the spike value correctly.

5.11 Computation Time Comparison

Xavier Fettweis stated that outputting from Reanalysis ERA-5, hourly MAR all variables, from 1940 to 2022 takes approximately seven days using 100 parallelized tasks on NIC-5 (MPI). If we approximate 1 task $\approx$ 1 core, then he used 100 cores of CPUs EPYC 7542.

Concerning the Elementary MAR Simulator forced at the boundaries by two variables derived from the ERA-5 land , it 56 seconds to generate and compose **one** posterior (not an ensemble) over a timer period of 24 hours (composing 24 consecutive sliding windows of 12 hours, each time taking the fifth timestamp to recompose) on GPU RTX A5000. Therefore, by naively scaling sequentially we could expect that the simulation from lateral boundary conditions takes approximately a little less than 20 days. Moreover, it outpus only 2 variables compared to the list of 30 variables. In fact, diffusion models present the disadvantage of slower sample generation compared to other generative AI models such as GANs.

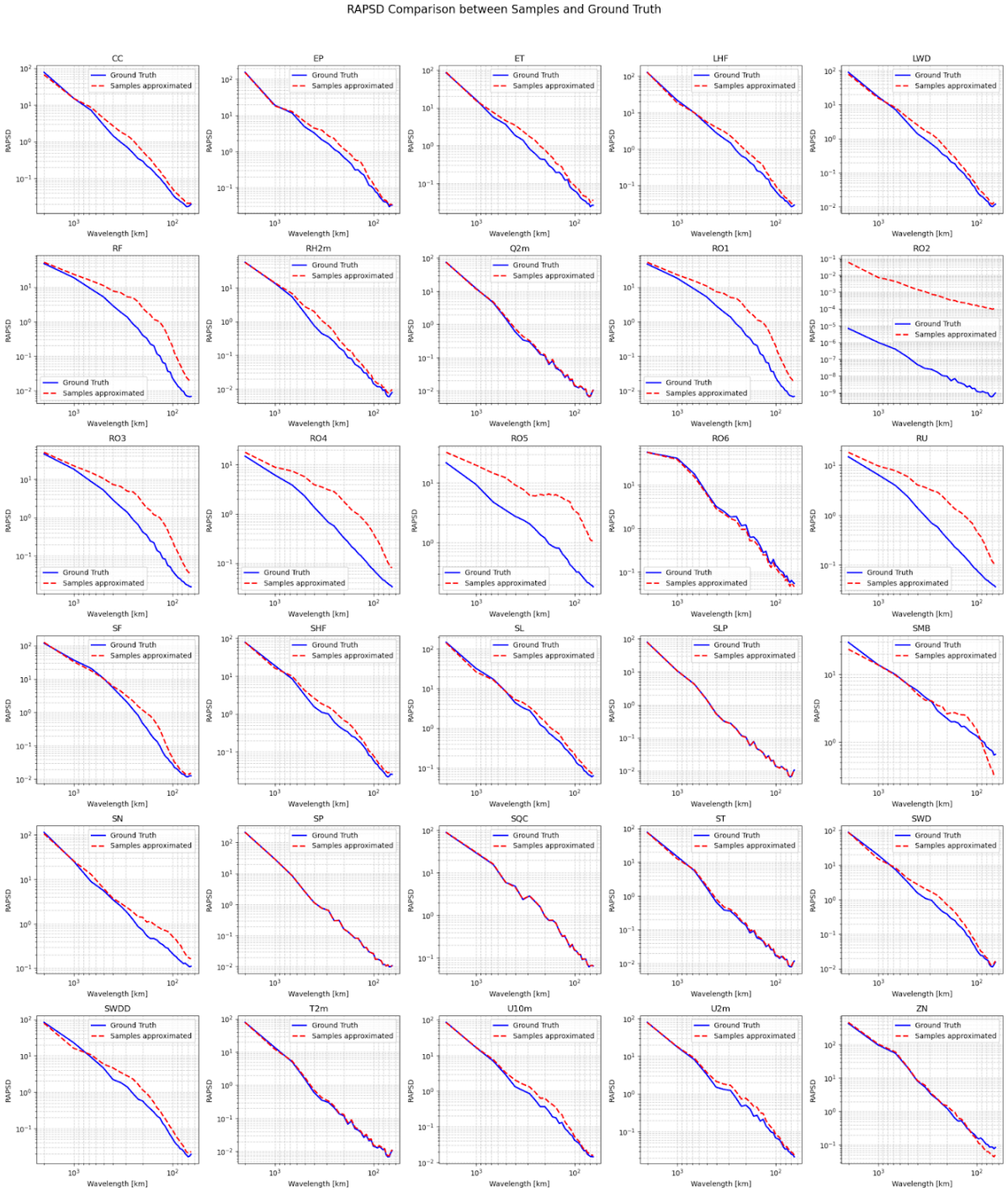


Figure 5.29: Spectral decomposition of 50 samples for all variables .

Chapter 6

Conclusions and perspectives

6.1 Conclusions

In this work, we've studied the application of the Score-based data assimilation [9] in the context of the *Modèle Atmosphérique Régional* (MAR) over the Belgian region. The problem was approached from a statistical perspective, framing it as an inverse problem.
The contributions are the following :

1. **Unconditional generation**
   Train a diffusion model to learn the distribution of MAR outputs and assess the quality of the samples drawn from the learned distribution.
   At the end, for two variables : the temperature and the wind speed, the samples capture the distribution of energy across scales but still present artefacts.

2. **Zero-shot conditional generation**
   Assess the performance of MAR in controlled measurement-noise settings on various scenarios such as artificially coarsened data as well as sparse noisy observations.
   The posterior samples are aligned with the observations.

3. **Integration of real-world measurements**
   Study the flexibility of SDA to integrate heterogeneous data sources: IRM sparse weather stations and ERA-5 land surface data, allowing the assessment of the performance of real-world assimilation and a simplified simulator setting of MAR.
   The posterior samples are more aligned with the observations compared to the MAR outputs at the same period and several different samples can be drawn from the same observation.
   However, if one part of the observation is assimilated, the posterior cannot predict the other part of the non-assimilated. The reason for this remains unclear. Sometimes this is a desired feature (e.g , the observation represents the boundary while the core may differ), but sometimes not (e.g , conditioning on part of the IRM stations yields a posterior that predicts values inconsistent with the non-assimilated weather stations).

4. **Ensemble variability**
   Ensemble variability was analyzed using rank histograms to study the spread of ensembles generated by the model. The test on a single observation showed that the produced ensemble was under-dispersive, and the KDEs of both the ground truth and the generated samples did not align perfectly.

5. **Generalization to multiple variables**
   Assess how the current model behaves on all MAR outputs. The model fails to capture the energy distribution across spatial scales for some variables.

6. **Computational considerations**
   Assess the computational cost of the approach to identify bottlenecks and trade-offs between performance and efficiency. No significant improvement was observed when approximating the time required to condition the model over an extended period. The main advantage remains that, once trained, the model can be directly applied to any zero-shot scenario.

6.2 Limitations

6.2.1 Learned prior

While the learned prior successfully captures the distribution of energy across spatial scales for the simplified case with two variables (T2m and U10m), it struggles significantly when extended to 30 variables. In particular, threshold-type variables, which exhibit mostly constant values with occasional sharp spikes, are poorly represented.
Even in the reduced two-variable setting, a trained classifier can still distinguish between generated samples and ground truth data, indicating the presence of persistent artefacts in the generated distribution. This shows that the learned prior does not fully reproduce the complexity of the reference model.

6.2.2 Alternative forcings and real world-observations

- **Controlled noise settings (Artificial Coarsen, Sparse, Simulator):** The resulting posteriors generally align with the observations. However, posterior quality in unobserved regions (particularly in the sparse setting) and variability across samples require further quantification.
- **IRM dataset:** A mismatch exists between IRM measurement definitions and MAR output variables, complicating the definition of the observation operator. Assimilation of weather stations reduces the NRMSE at observed points, but posteriors fail to generalize to unassimilated stations. The source of this failure cannot be clearly attributed to either (1) insufficient representativeness of the learned prior, or (2) a strong non-linear mismatch between IRM and MAR datasets, including possible issues such as: (a) differences in variable measurements, and (b) the absence of IRM assimilation within MAR.
- **ERA5 dataset:** Posterior variability has not been studied with sufficiently detailed metrics, leaving uncertainty about the robustness of results. We restricted only to land-surface variables for the proof of concept meanwhile considering prognostic variables across pressure levels would be more meaningful to represent the simulation of MAR.

6.2.3 Ensemble evalutation and computation power

Ensemble evaluation was limited to a single downscaling experiment, which strongly restricts the generality of conclusions. In this case, the ensemble mean was unbiased relative to the reference, but the ensemble spread was overconfident, and kernel density estimates revealed noticeable distributional discrepancies. More systematic evaluation across different forcings and multiple samples is needed, as well as investigation of alternative sampling strategies.
Regarding computational efficiency, the comparison between physical RCM simulations and diffusion-based generative models is not strictly one-to-one, as different hardware was used. Nevertheless, the proposed framework does not currently exhibit a clear computational advantage due to (1) the high training cost of diffusion models and (2) their relatively slow sampling speed.

6.2.4 Procedure

Our primary objective has been to present a proof-of-concept rather than conducting an exhaustive performance comparison with multiple baselines, though such evaluation remains important. More specifically, we propose comparing the ensemble variability against the physical generator MAR within the same experimental setup and measuring computational efficiency and ensemble calibration and diversity using climatology-oriented metrics such as the Continuous Ranked Probability Score (CRPS) and the Spread-Skill Ratio (SSR). In addition, comparisons with other diffusion model baselines, alternative generative frameworks, or even traditional climate techniques are valuable. These can be evaluated using quantitative metrics designed to capture discrepancies between data distributions in a single value, such as the Wasserstein distance [43] or the Structural Similarity Index Measure (SSIM) [44].

6.3 Perspectives

6.3.1 Learnable prior

Despite studying an RCM model initially developed for the Groenland, we didn't even cover a tip of the iceberg.
A first avenue is to improve the quality of generated samples. This could involve testing alternative backbones such as Vision Transformers [45], which have shown promising ability to capture long-range dependencies. The exploration of different sampling strategies is important, as the choice of sampling directly impacts the quality and reliability of the generated samples.

Another direction is to train models using the full set of variables, while addressing the issue of threshold-type variables. One possible strategy would be to train the network to predict a mask on these critical variables to tell whether this is a constant value or a spike.

6.3.2 Dataset discrepancy

A second perspective concerns the discrepancy between datasets. Reducing or at least quantifying such mismatches could substantially improve results. In statistical downscaling, quantile mapping [46] is a widely used technique to mitigate the gap between coarse GCM outputs and fine-scale RCM simulations. For the IRM stations, try to find a way to mitigate the way the values are defined with respect to MAR.

6.3.3 Simulator

Finally, the most important point: study a simulator with real plausible GCM inputs to MAR, namely: temperature, surface pressure, specific humidity, and the two wind speed components across several pressure levels.

Naive idea

A first idea is to continue training a diffusion model solely on the RCM outputs,

$$\mathbf{x} = [\mathbf{x}_{MAR}] \,.$$

However, as in this work, if the RCM outputs are mostly land and soil variables, it is difficult to define an observer

$$\mathcal{A} : \mathcal{X}_{MAR} \to \mathcal{X}_{GCM}, \quad \mathbf{y} \in \mathcal{X}_{GCM}.$$

For example, how can one relate the absolute wind speed defined at 2 meters above the ground to its decomposition into two perpendicular components at the boundaries of the domain across several pressure levels?

To explore

Another idea is to define the states as

$$\mathbf{x} = [\mathbf{x}_{MAR}, \mathbf{x}_{GCM}],$$

where $\mathbf{x}_{MAR}$ corresponds to the output variables and $\mathbf{x}_{GCM}$ corresponds to the coarse GCM inputs, ERA-5 in this work. First, train a diffusion model to learn to denoise the added noise to such states with land characteristics, such as elevation or soil properties, embedded as context channels in order to take advantage of all MAR inputs. Afterwards, defining the observations as

$$\mathbf{y} \in \mathcal{X}_{GCM},$$

with a GCM providing prognostic variables at the boundaries, which may differ from the one used during training. Defining an operator

$$\mathcal{A} : [\mathcal{X}_{MAR} \cup \mathcal{X}_{GCM}] \to \mathcal{X}_{GCM}$$

is then straightforward, in order to generate conditional samples given the observations : GCM inputs.

Bibliography

[1] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon et Ben Poole : Score-based generative modeling through stochastic differential equations, 2020.

[2] Jonathan Ho, Ajay Jain et Pieter Abbeel : Denoising diffusion probabilistic models, 2020.

[3] Yang Song et Stefano Ermon : Generative modeling by estimating gradients of the data distribution. *CoRR*, abs/1907.05600, 2019.

[4] Aapo Hyvärinen et Peter Dayan : Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

[5] Pascal Vincent : A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

[6] Brian D.O. Anderson : Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[7] Qinsheng Zhang et Yongxin Chen : Fast sampling of diffusion models with exponential integrator, 2023.

[8] Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky et Jong Chul Ye : Diffusion posterior sampling for general noisy inverse problems. *In The Eleventh International Conference on Learning Representations*, 2023.

[9] François Rozet et Gilles Louppe : Score-based data assimilation. *In Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[10] L. Ruby Leung : *Regional Climate Modelsregional climate model*, pages 8902–8919. Springer New York, New York, NY, 2012.

[11] Jason E Xavier, Cécile Box, Charles Agosta, Christoph Amory, Charlotte Kittel, Dirk Lang, Horst Van As et Hubert Machguth : Reconstructions of the 1900-2015 greenland ice sheet surface mass balance using the regional climate MAR model. *The Cryosphere*, 11(2):1015–1033, 2017.

[12] Guillaume Chagnaud, Hubert Gallée, Thierry Lebel, Gérémy Panthou et Théo Vischel : A boundary forcing sensitivity analysis of the west african monsoon simulated by the modèle atmosphérique régional. *Atmosphere*, 11(2), 2020.

[13] Coraline Wyard, Xavier Fettweis et Michel Erpicum : Etude de l'évolution de l'enneigement dans les hautes fagnes (belgique) au cours des cinquante dernières années à l'aide du modèle climatique régional MAR. *In XXVIIIe colloque annuel de l'Association Internationale de Climatologie*. ULg, Liège, Belgium, 2015.

[14] Coraline Wyard, Chloé Scholzen, Xavier Fettweis, Jean Van Campenhout et Louis François : Decrease in climatic conditions favouring floods in the south-east of belgium over 1959-2010 using the regional climate model mar. *International Journal of Climatology*, 37:2782–2796, 09 2016.

[15] Coraline Wyard, Sébastien Doutreloup, Alexandre Belleflamme, Martin Wild et Xavier Fettweis : Global radiative flux and cloudiness variability for the period 1959–2010 in belgium: A comparison between reanalyses and the regional climate model mar. *Atmosphere*, 9:262, 07 2018.

[16] Hubert Gallée et Guy Schayes : Development of a three-dimensional meso-γ primitive equation model: katabatic winds simulation in the area of terra nova bay, antarctica. *Monthly Weather Review*, 122(4):671–685, 1994.

[17] Koen De Ridder et Hubert Gallée : Land surface–induced regional climate change in southern israel. *Journal of Applied Meteorology*, 37(11):1470 – 1485, 1998.

[18] S. Doutreloup, X. Fettweis, R. Rahif, E. Elnagar, M. S. Pourkiaei, D. Amaripadath et S. Attia : Historical and future weather data for dynamic building simulations in belgium using the regional climate model mar: typical and extreme meteorological year and heatwaves. *Earth System Science Data*, 14(7):3039–3051, 2022.

[19] Sarah N Collins, Robert S James, Pallav Ray, Katherine Chen, Angie Lassman et James Brownlee : Grids in numerical weather and climate models. *In* Yuanzhi Zhang et Pallav Ray, éditeurs : *Climate Change and Regional/Local Responses*, chapitre 4. IntechOpen, Rijeka, 2013.

[20] G. I. Marchuk : Numerical methods in weather prediction, 1974.

[21] Thomas Goelles, Klaus Grosfeld et G. Lohmann : Semi-lagrangian transport of oxygen isotopes in polythermal ice sheets: Implementation and first results. *Geoscientific Model Development*, 7:1395–1408, 07 2014.

[22] Philippe Marbaix, Hubert Gallée, Olivier Brasseur et Jean-Pascal van Ypersele : Lateral boundary conditions in regional climate models: A detailed study of the relaxation procedure. *Monthly Weather Review - MON WEATHER REV*, 131, 03 2003.

[23] Koen De Ridder et Guy Schayes : The iagl land surface model. *Journal of Applied Meteorology - J APPL METEOROL*, 36:167–182, 02 1997.

[24] V. Vionnet, E. Brun, S. Morin, A. Boone, S. Faroux, P. Le Moigne, E. Martin et J.-M. Willemet : The detailed snowpack scheme crocus and its implementation in surfex v7.2. *Geoscientific Model Development*, 5(3):773–791, 2012.

[25] E Brun, Eric Martin, V Simon, C Gendre et Cécile Coléou : An energy and mass model of snow cover suitable for operational avalanche forecasting. *J. Glaciol.*, 35:333–, 01 1989.

[26] E. Brun, P. David, M. Sudul et G. Brunot : A numerical model to simulate snow-cover stratigraphy for operational avalanche forecasting. *Journal of Glaciology*, 38(128):13–22, 1992.

[27] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, Adrian Simmons, Cornel Soci, Saleh Abdalla, Xavier Abellan, Gianpaolo Balsamo, Peter Bechtold, Gionata Biavati, Jean Bidlot, Massimo Bonavita, Giovanna De Chiara, Per Dahlgren, Dick Dee, Michail Diamantakis, Rossana Dragani, Johannes Flemming, Richard Forbes, Manuel Fuentes, Alan

Geer, Leo Haimberger, Sean Healy, Robin J. Hogan, Elías Hólm, Marta Janisková, Sarah Keeley, Patrick Laloyaux, Philippe Lopez, Cristina Lupu, Gabor Radnoti, Patricia de Rosnay, Iryna Rozum, Freja Vamborg, Sebastien Villaume et Jean-Noël Thépaut : The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020.

[28] Sergey Zagoruyko et Nikos Komodakis : Wide residual networks, 2017.

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser et Illia Polosukhin : Attention is all you need, 2023.

[30] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell et Gordon Wetzstein : Implicit neural representations with periodic activation functions, 2020.

[31] Alex Nichol et Prafulla Dhariwal : Improved denoising diffusion probabilistic models. *CoRR*, abs/2102.09672, 2021.

[32] Evan Ruzanski et V. Chandrasekar : Scale filtering for improved nowcasting performance in a high-resolution x-band radar network. *IEEE Transactions on Geoscience and Remote Sensing*, 49(6):2296–2307, 2011.

[33] Thomas M Hamill : Interpretation of rank histograms for verifying ensemble forecasts. *Monthly Weather Review*, 129(3):550–560, 2001.

[34] V. Fortin, M. Abaza, F. Anctil et R. Turcotte : Why should ensemble spread match the rmse of the ensemble mean? *Journal of Hydrometeorology*, 15(4):1708–1713, 2014.

[35] Giorgio Valentini et Thomas Dietterich : Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. volume 5, pages 725–775, 07 2004.

[36] David Lopez-Paz et Maxime Oquab : Revisiting classifier two-sample tests, 2018.

[37] Mingxing Tan et Quoc V. Le : Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

[38] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li et Li Fei-Fei : Imagenet: A large-scale hierarchical image database. *In 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[39] James E Matheson et Robert L Winkler : Scoring rules for continuous probability distributions. *Management science*, 22(10):1087–1096, 1976.

[40] Peter Manshausen, Yair Cohen, Peter Harrington, Jaideep Pathak, Mike Pritchard, Piyush Garg, Morteza Mardani, Karthik Kashinath, Simon Byrne et Noah Brenowitz : Generative data assimilation of sparse weather station observations at kilometer scales, 2025.

[41] Jonathan Schmidt, Luca Schmidt, Felix Strnad, Nicole Ludwig et Philipp Hennig : A generative framework for probabilistic, spatiotemporally coherent downscaling of climate simulation, 2025.

[42] Yen-Chi Chen : A tutorial on kernel density estimation and recent advances, 2017.

[43] Sebastian Bischoff, Alana Darcher, Michael Deistler, Richard Gao, Franziska Gerken, Manuel Gloeckler, Lisa Haxel, Jaivardhan Kapoor, Janne K Lappalainen, Jakob H Macke, Guy Moss, Matthijs Pals, Felix Pei, Rachel Rapp, A Erdem Sağtekin, Cornelius Schröder, Auguste Schulz, Zinovia Stefanidi, Shoji Toyota, Linda Ulmer et Julius Vetter : A practical guide to sample-based statistical distances for evaluating generative models in science, 2024.

[44] Z. Wang, E.P. Simoncelli et A.C. Bovik : Multiscale structural similarity for image quality assessment. *In The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, volume 2, pages 1398–1402 Vol.2, 2003.

[45] Yanghao Li, Hanzi Mao, Ross Girshick et Kaiming He : Exploring plain vision transformer backbones for object detection, 2022.

[46] Santiago Mendoza Paz et Patrick Willems : Uncovering the strengths and weaknesses of an ensemble of quantile mapping methods for downscaling precipitation change in southern africa. *Journal of Hydrology: Regional Studies*, 41:101104, 2022.

Declaration on the use of automatic tools for writing the manuscript

I hereby certify that I have not used any generative intelligence tool in the writing of text, graphics, images, or data reproduced in this manuscript. The tools used for writing this document include:

1. DeepL : all along the project
2. Grammarly : all along the project