Deep Generative Models for hydrological time-series simulations

BHAVSAR Ferdinand[1], Lionel Benoit[1], Edith Gabriel[1]

SWGEN 2025 - Conference on Stochastic Weather Generators

03 December 2025

[1]INRAE, Biostatistics and Spatial Processes (Biosp) team

Application: Water quality monitoring around the Cigéo site

Source: Andra

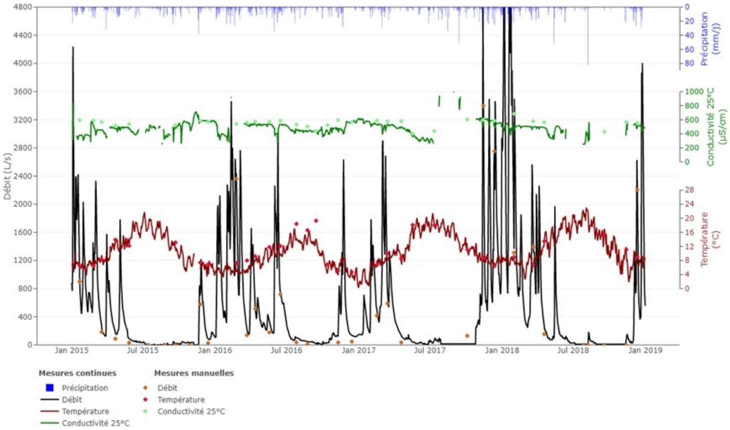
Application: Continuous monitoring of hydrological variables

1. Water level
2. Temperature
3. pH
4. Conductivity at 25°C
5. Dissolved O2
6. O2 saturation
7. Nitrates concentration
8. Turbidites
9. FDom (Fluorescent Dissolved Organic Matter) / Organical Carbon
10. PAH (Polycyclic Aromatic Hydrocarbon)

Source: Andra

Application: Continuous monitoring of hydrological variables

The data are **multivariate time-series** with many missing values (from 2012 to 2025, 4h between each observations).

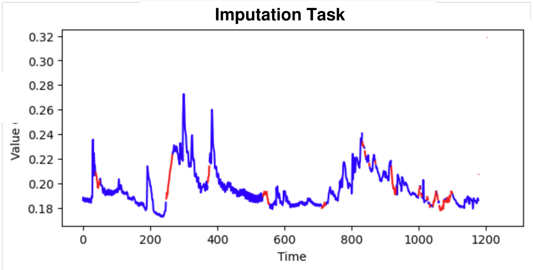


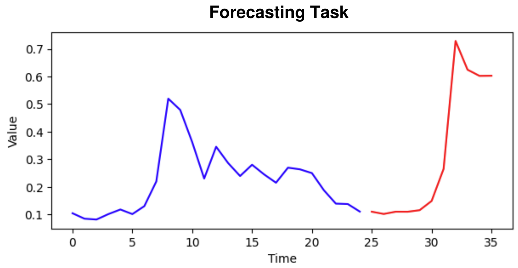
Source: Andra

Modeling multivariate hydrological variables with missing values

Postdoc objective: Develop a simulation method to simulate the target hydrological variables

We have two tasks to tackle:





How do we find the red part ? ("Fill the gaps")

Deep generative learning: a transformation problem

Problem

Given a dataset, we want to sample new, never-seen before, convincing simulations with the same properties as the data from the dataset

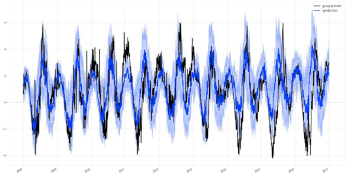
Deep generative learning: a transformation problem

Problem

Given a dataset, we want to sample new, never-seen before, convincing simulations with the same properties as the data from the dataset

Easy to sample Random Variable

Complex and unknown Random Variable



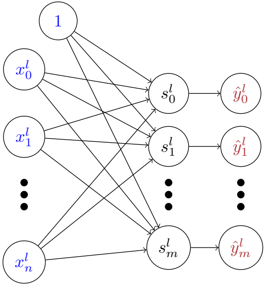
Example of complex RV: time-series data

Generative models: Deep learning

We approximate the transformation G with a neural network G_θ.

Input

$$\begin{bmatrix} x_0^l \\ x_1^l \\ \vdots \\ x_n^l \end{bmatrix}$$



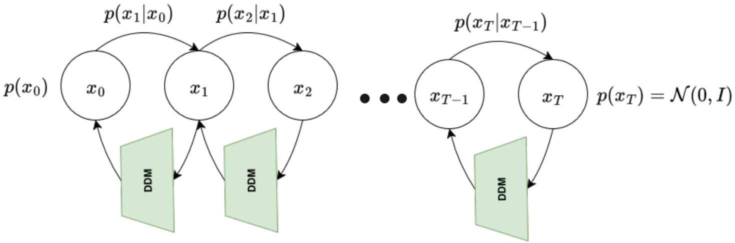
Output

$$\begin{bmatrix} \hat{y}_0^l \\ \hat{y}_1^l \\ \vdots \\ \hat{y}_m^l \end{bmatrix} = \begin{bmatrix} \nu(s_0^l) \\ \nu(s_1^l) \\ \vdots \\ \nu(s_m^l) \end{bmatrix}$$

$$s_i^l = b_i^l + \sum_{j=0}^{n} W_{i,j}^l x_j^l$$

$$\begin{bmatrix} s_0^l \\ s_1^l \\ \vdots \\ s_m^l \end{bmatrix} = \begin{bmatrix} b_0^l & W_{0,0}^l & W_{0,2}^l & \dots & W_{0,n}^l \\ b_1^l & W_{1,0}^l & W_{1,2}^l & \dots & W_{1,n}^l \\ \vdots & \vdots & \vdots & & \\ b_m^l & W_{m,0}^l & W_{m,2}^l & \dots & W_{m,n}^l \end{bmatrix} \begin{bmatrix} 1 \\ x_0^l \\ x_1^l \\ \vdots \\ x_n^l \end{bmatrix}$$

Denoising Diffusion Probabilistic Models (Sohl-Dickstein et al., 2015; Ho et al., 2020)



$$\theta^* = \arg\min_\theta D_{\mathrm{KL}}\big(p_\theta(x_{t-h} \mid x_t) \,\|\, p(x_{t-h} \mid x_t, x_0)\big) \tag{1}$$

CSDI: Conditional Score-based Diffusion Models for Irregular Time Series Imputation

We want a **state of the art** diffusion generative model, designed for both forecasting and imputation: **CSDI** (Tashiro et al., 2021).

Recent, good documentation, code available, good reputation in the community.

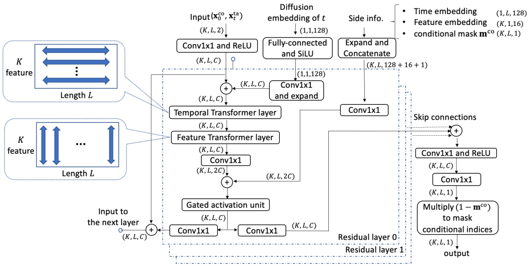
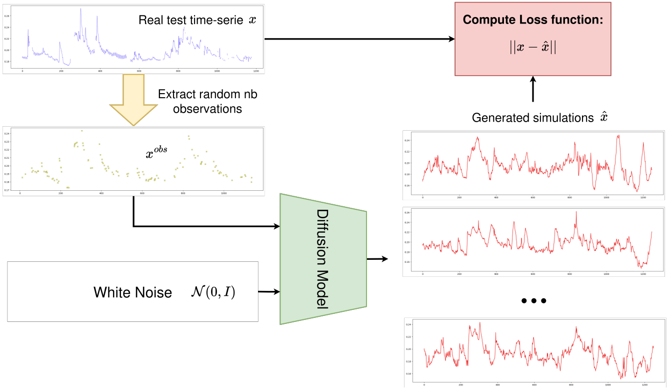


Figure 1: CSDI architecture overview (Tashiro et al., 2021)

CSDI training: Simplified overview



where x is the training time-serie, $\hat{x} = D_\theta(\epsilon, t, x^{obs})$ is the predicted time-serie, t is the diffusion time step, ϵ is Gaussian noise, $m(x)$ is the mask and x^{obs} are the observed values.

CSDI: Loss function

Training the model to learn the **reverse distribution**:

$$\theta^* = \arg\min_\theta D_{\mathrm{KL}}\big(p_\theta(x_{t-h} \mid x_t) \,\|\, p(x_{t-h} \mid x_t, x_0)\big) \tag{2}$$

It's the same as training the model to **denoise** x_t into $\hat{x}_0 = D_\theta(x_t, t)$ (Ho et al., 2020):

$$\mathcal{L}(\theta) = \mathbb{E}\left[\|x_0 - \hat{x}_0\|^2\right] \tag{3}$$

For CSDI, the loss is computed only on the masked values (Tashiro et al., 2021), i.e.:

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left\|x_0^{miss} - \hat{x}_0^{miss}\right\|^2\right] \tag{4}$$

where x_0^{miss} are the masked values, i.e. $x_{0_{/x_0^{obs}}}$, and $\hat{x}_0^{miss}$ are the corresponding predicted values.

CSDI: Loss function

Training the model to learn the **reverse distribution**:

$$\theta^* = \arg\min_{\theta} D_{\mathrm{KL}}\big(p_\theta(x_{t-h} \mid x_t) \,\|\, p(x_{t-h} \mid x_t, x_0)\big) \tag{2}$$

It's the same as training the model to **denoise** x_t into $\hat{x}_0 = D_\theta(x_t, t)$ (Ho et al., 2020):

$$\mathcal{L}(\theta) = \mathbb{E}\left[\|x_0 - \hat{x}_0\|^2\right] \tag{3}$$

For CSDI, the loss is computed only on the masked values (Tashiro et al., 2021), i.e.:

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left\|x_0^{miss} - \hat{x}_0^{miss}\right\|^2\right] \tag{4}$$

where x_0^{miss} are the masked values, i.e. $x_{0_{/x_0^{obs}}}$, and $\hat{x}_0^{miss}$ are the corresponding predicted values.

CSDI: Loss function

Training the model to learn the **reverse distribution**:

$$\theta^* = \arg\min_{\theta} D_{\mathrm{KL}}\left(p_\theta(x_{t-h} \mid x_t) \,\|\, p(x_{t-h} \mid x_t, x_0)\right) \tag{2}$$

It's the same as training the model to **denoise** x_t into $\hat{x}_0 = D_\theta(x_t, t)$ (Ho et al., 2020):

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left\|x_0 - \hat{x}_0\right\|^2\right] \tag{3}$$

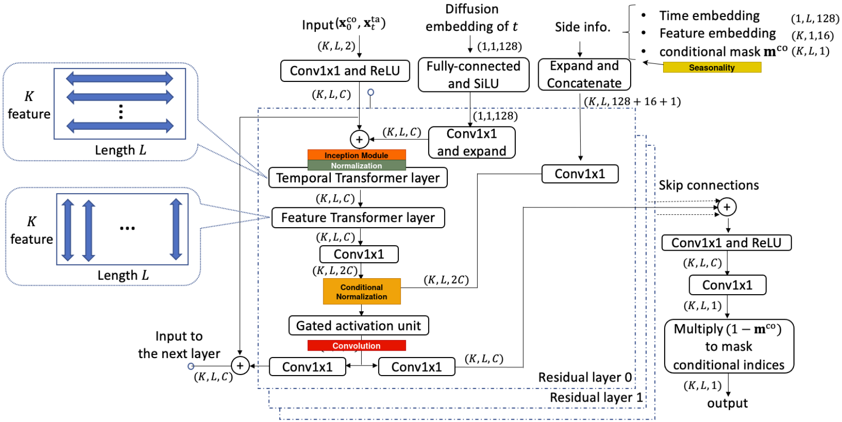
For CSDI, the loss is computed only on the masked values (Tashiro et al., 2021), i.e.:

$$\mathcal{L}(\theta) = \mathbb{E}\left[\left\|x_0^{miss} - \hat{x}_0^{miss}\right\|^2\right] \tag{4}$$

where x_0^{miss} are the masked values, i.e. $x_{0_{/x_0^{obs}}}$, and $\hat{x}_0^{miss}$ are the corresponding predicted values.

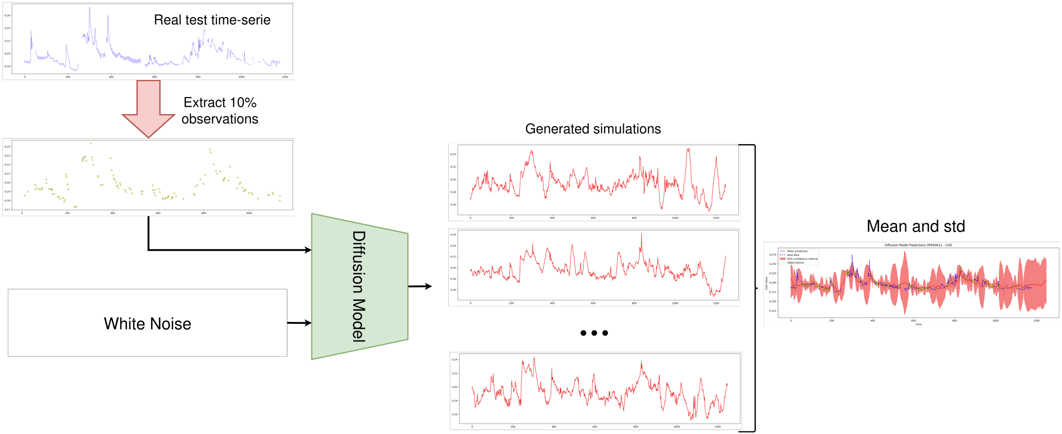
Adapting CSDI to our case

Results with the base CSDI model were disappointing.. So we made some modifications to the architecture:



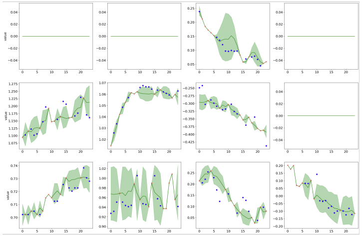
Evaluation methodology

Metrics used: RMSE, MAE, CRPS. 100 generated tests simulations.

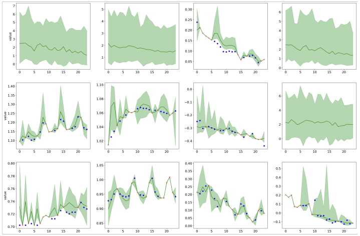


Imputation: some outputs vizualed

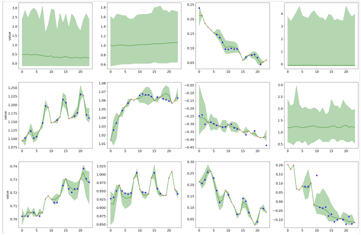
Gaussian Processes



Original CSDI



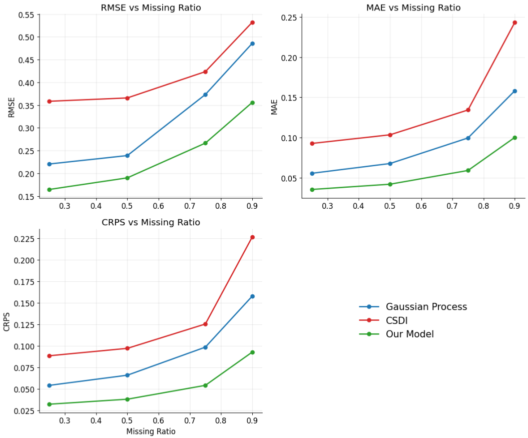
Our CSDI



Imputation: metrics

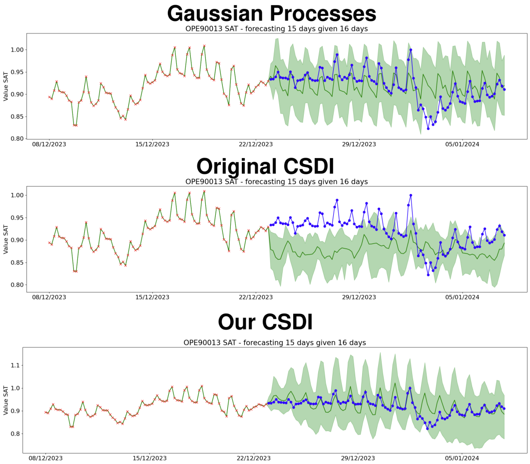
Imputation performance across models and missing ratios. Best value per metric highlighted in green.

Miss. Ratio	Model	RMSE	MAE	CRPS
0.25	Gaussian Proc.	0.2205	0.0556	0.0542
	CSDI	0.3584	0.0927	0.0886
	Our Model	0.1645	0.0355	0.0323
0.5	Gaussian Proc.	0.2390	0.0677	0.0660
	CSDI	0.3658	0.1034	0.0972
	Our Model	0.1900	0.0420	0.0381
0.75	Gaussian Proc.	0.3733	0.0995	0.0986
	CSDI	0.4236	0.1343	0.1255
	Our Model	0.2664	0.0591	0.0542
0.9	Gaussian Proc.	0.4857	0.1582	0.1578
	CSDI	0.5320	0.2432	0.2265
	Our Model	0.3558	0.1002	0.0928



Forecasting: some outputs vizualed

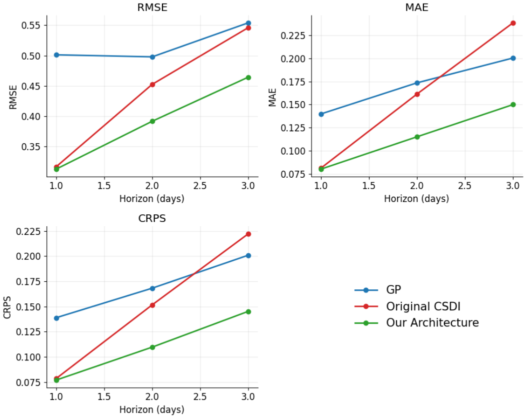
16 days seen, 15 days to predict



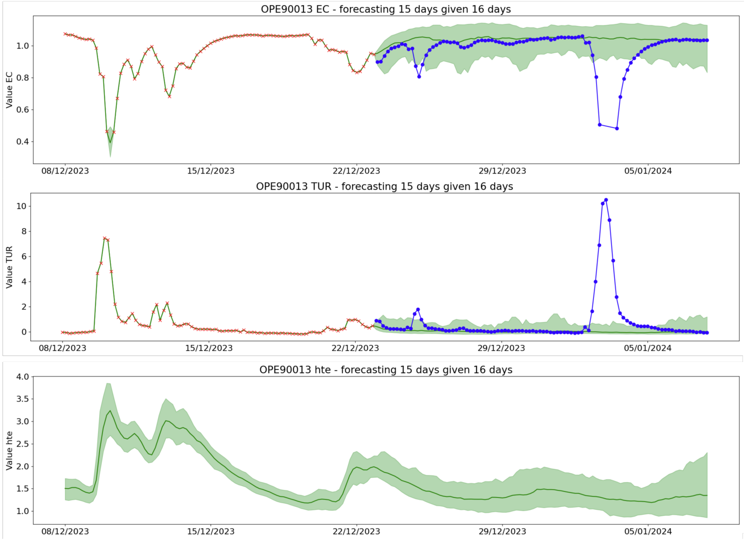
Forecasting: Metrics

Forecasting results for horizons 1, 2 and 3 days. Best values highlighted in green.

Horizon	Model	RMSE	MAE	CRPS
1 day	GP	0.5013	0.1398	0.1388
	Original CSDI	0.3167	0.0809	0.0783
	Our Architecture	0.3129	0.0799	0.0768
2 days	GP	0.4979	0.1736	0.1682
	Original CSDI	0.4526	0.1614	0.1516
	Our Architecture	0.3918	0.1150	0.1096
3 days	GP	0.5540	0.2006	0.2010
	Original CSDI	0.5461	0.2388	0.2223
	Our Architecture	0.4645	0.1500	0.1452



A problem with the method: long term forecasting and unforeseen events

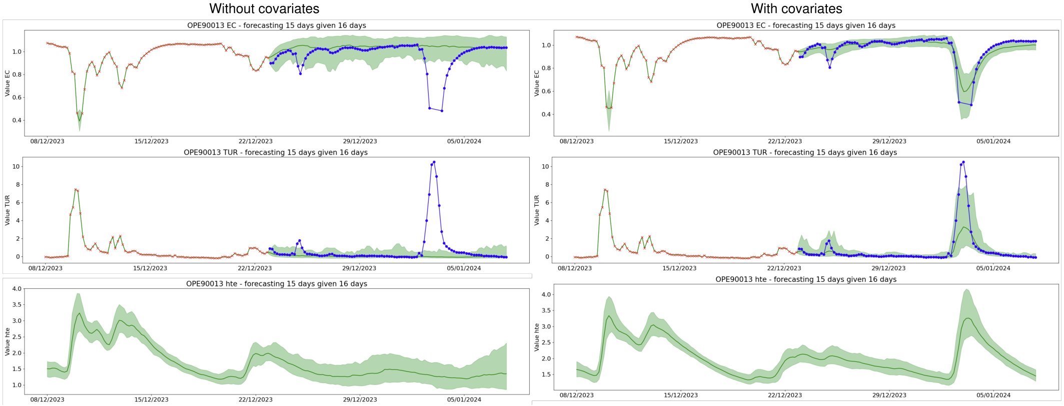


Daily SIM / SAFRAN-grid Data (weather)

- Data source: SAFRAN reanalysis from Météo-France (Vidal et al., 2010)

- Precipitation (liquid and solid) - daily totals
- Air temperature (min, max, or mean) at 2m
- Wind speed (e.g. 10m)
- Specific humidity or relative humidity at 2m
- Global / direct / diffuse solar radiation
- Snow- and soil-related variables: soil wetness index, soil water content, snow water equivalent, evapotranspiration

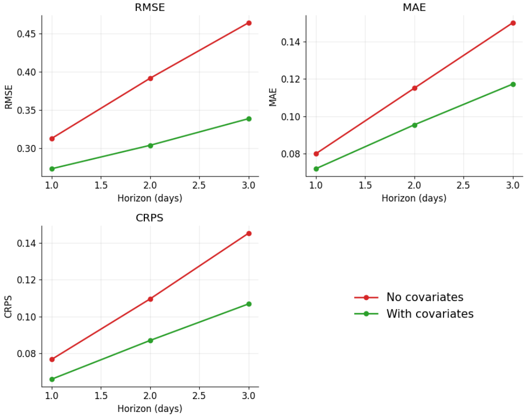
Forecasting using covariates: some outputs vizualed



Forecasting using covariates: metrics

Forecasting results for horizons 1, 2 and 3 days. Best values highlighted in green.

Horizon	Setting	RMSE	MAE	CRPS
6	w/o Cov.	0.3129	0.0799	0.0768
	with Cov.	0.2733	0.0719	0.0661
12	w/o Cov.	0.3918	0.1150	0.1096
	with Cov.	0.3040	0.0954	0.0871
18	w/o Cov.	0.4645	0.1500	0.1452
	with Cov.	0.3389	0.1172	0.1069



Conclusion

We need to adapt the architecture of CSDI to our specific case study to obtain good results.

Our model seems to be performing better than the baselines for both imputation and forecasting.

Using relevant covariates (weather data) improves forecasting results.

Next steps:

- Take into account dry periods: regime-switching diffusion models
- Take into account the spatial component of the data
- Sensibility Analysys for covariates selection (Yachouti et al., 2025)
- Interpolation on the river network
- Impact of climate change and anthropic factors

Conclusion

We need to adapt the architecture of CSDI to our specific case study to obtain good results.
Our model seems to be performing better than the baselines for both imputation and forecasting.
Using relevant covariates (weather data) improves forecasting results.
Next steps:

- Take into account dry periods: regime-switching diffusion models
- Take into account the spatial component of the data
- Sensibility Analysys for covariates selection (Yachouti et al., 2025)
- Interpolation on the river network
- Impact of climate change and anthropic factors

Conclusion

We need to adapt the architecture of CSDI to our specific case study to obtain good results.
Our model seems to be performing better than the baselines for both imputation and forecasting.
Using relevant covariates (weather data) improves forecasting results.
Next steps:

- Take into account dry periods: regime-switching diffusion models
- Take into account the spatial component of the data
- Sensibility Analysys for covariates selection (Yachouti et al., 2025)
- Interpolation on the river network
- Impact of climate change and anthropic factors

Ho, J., A. Jain, and P. Abbeel (2020). Denoising diffusion probabilistic models.

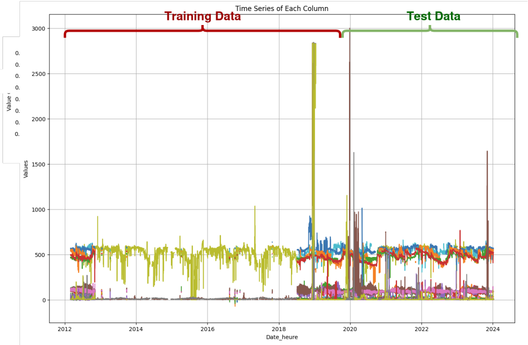
Sohl-Dickstein, J., E. A. Weiss, N. Maheswaranathan, and S. Ganguli (2015). Deep unsupervised learning using nonequilibrium thermodynamics. *CoRR abs/1503.03585*.

Tashiro, Y., J. Song, Y. Song, and S. Ermon (2021). Csdi: Conditional score-based diffusion models for probabilistic time series imputation.

Vidal, J.-P., E. Martin, L. Franchistéguy, M. Baillon, and J.-M. Soubeyroux (2010, September). A 50-year high-resolution atmospheric reanalysis over France with the Safran system. *International Journal of Climatology 30*(11), P. 1627–1644. DOI: 10.1002/joc.2003. Publié en ligne dans Wiley InterScience (www.interscience.wiley.com). Version auteur dans fichier pdf attaché.

Yachouti, M., G. Perrin, and J. Garnier (2025, April). Towards History-aware Sensitivity Analysis For Time Series. working paper or preprint.

Adapting the data to deep learning



Deep learning does not like unormalized data. We do min-max normalization on the time-series along the time axis:

$$x^c_{norm} = \frac{x^c - x^c_{min}}{x^c_{max} - x^c_{min}} \quad (5)$$

where $c = \{0, C\}$, C is the number of variables.

Distribution realism (Cigéo data)

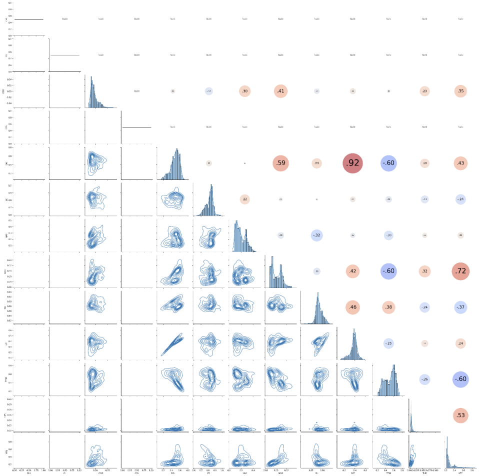


Figure 2: Real data linear correlations

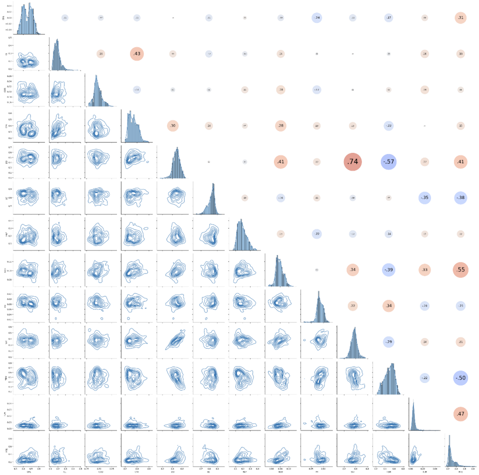


Figure 3: Our model linear correlations