Advanced Machine Learning (ML) Techniques for Land Surface Modeling

Dan Lu (lud1@ornl.gov)

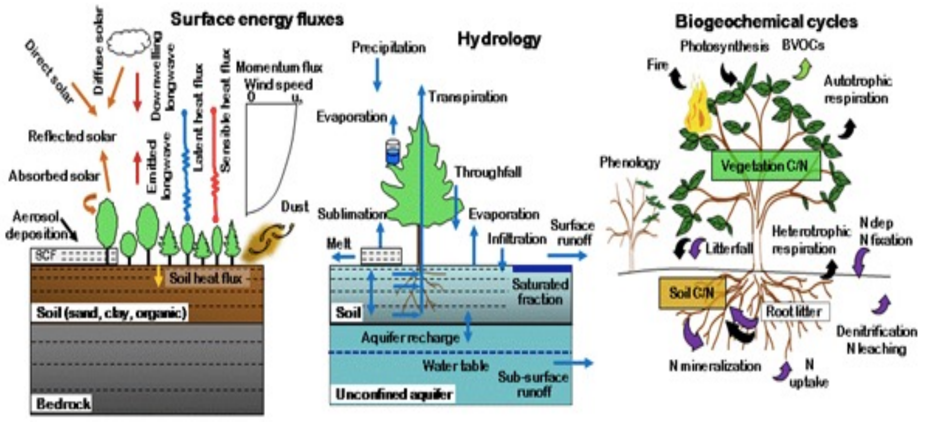
Yanfang Liu; Zezhong Zhang; Feng Bao; Guannan Zhang; Daniel Ricciuto

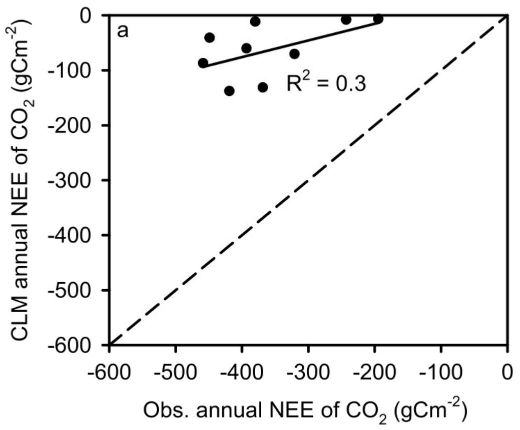
ORNL is managed by UT-Battelle, LLC for the US Department of Energy

June 10, 2024

Land surface models need calibration to improve the prediction accuracy

- Energy Exascale Earth System Model (E3SM), land model (ELM) simulates terrestrial water, energy, and biogeochemical processes in terrestrial surfaces.
- It is an important tool for improving our predictive understanding of ecosystem responses to climate change.





There are significant discrepancies between ELM model simulation with default parameter values and observed NEE at MOFLUX forest site. Model calibration is needed for improving prediction.

Gu et al., JGR, 2016

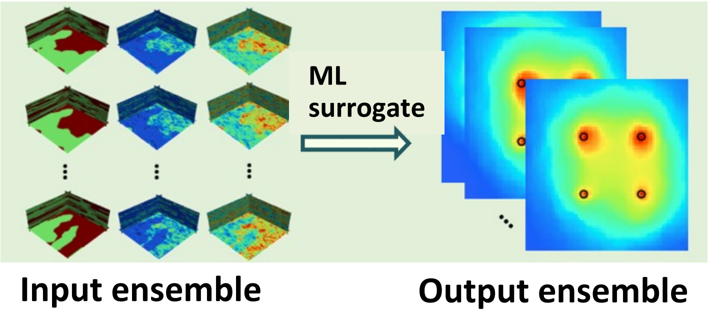
- ELM involves 65+ unknown parameters, and the use of default parameter values has shown large model discrepancy from site observations.
- Thus, ELM calibration is required at every site for improving the prediction accuracy globally.

Model calibration and uncertainty quantification (UQ) are computationally expensive

- Calibrating the ELM model is challenging because of its strong nonlinearity and unconstrained parameters, which requires UQ of parameter estimates.
- Estimating uncertainty in nonlinear inverse problems is computationally demanding.
- Several methods have been proposed to reduce the computational cost.

Surrogate Modeling

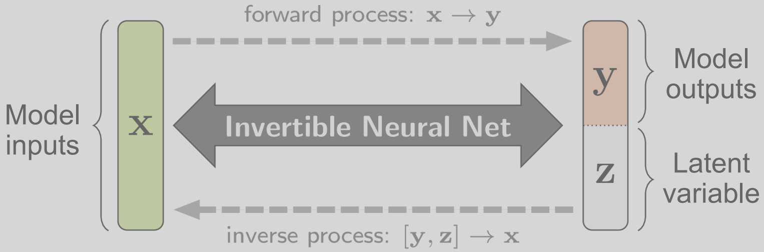
Build a fast surrogate of the ELM, and then evaluate the surrogate in the standard UQ process



- Surrogate modeling reduces time of a single model run.

Invertible Neural Network

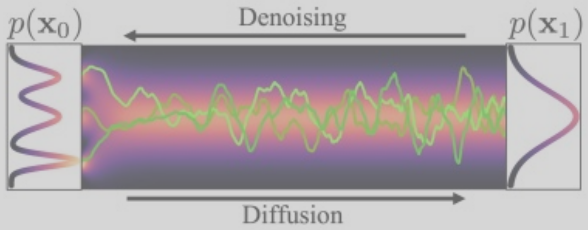
INN solves UQ problem directly based on ELM model simulation samples



- INN learns bijective mapping between inputs and outputs.
- Evaluating the trained INN backwards produces parameter posterior samples.

Diffusion-based UQ

Diffusion models are generative ML method.

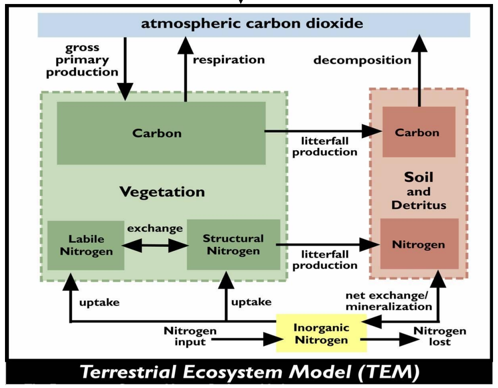


- It includes two processes.
- The reverse denoising process transforms standard Gaussian samples to target parameter posterior samples.

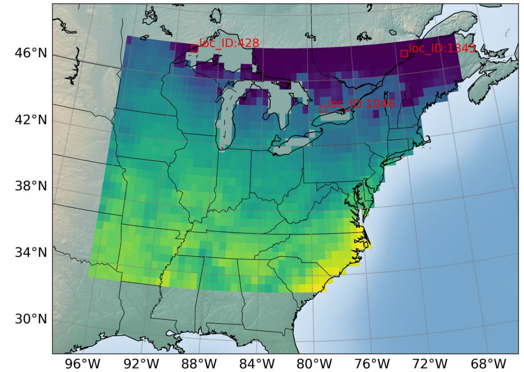
Surrogate modeling to reduce computational time of UQ

Two strategies: build an accurate surrogate model using limited ELM simulation samples

TEM parameters

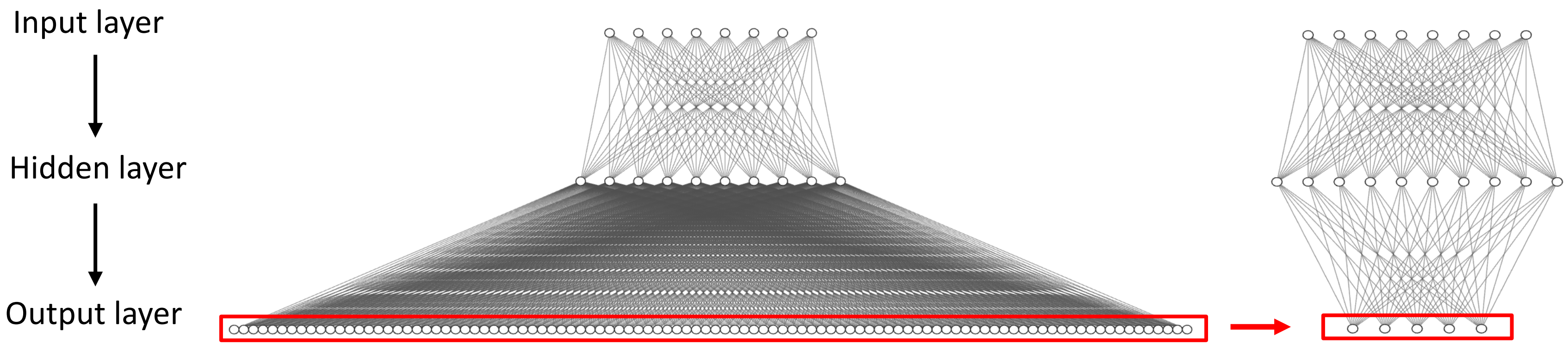


TEM outputs



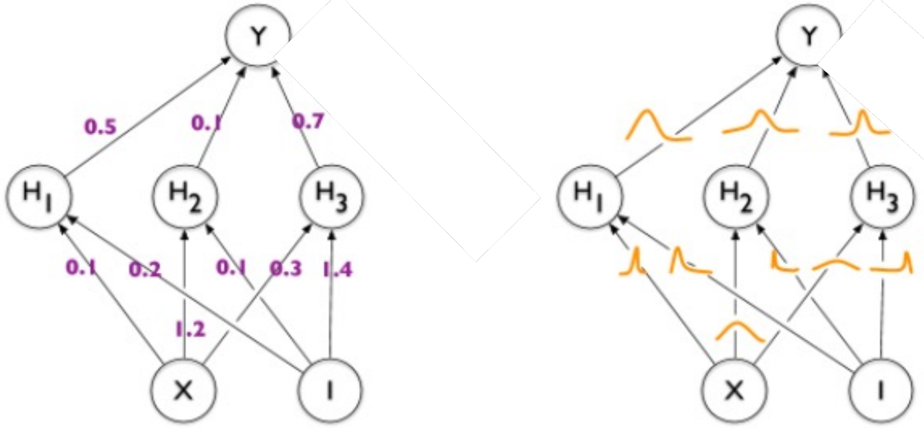
Large number of spatiotemporal model outputs

Strategy I: Dimension reduction to simplify the NN structure



Dimension reduction on output layer reduces NN parameters from 1000 to 100

Strategy II: BNN to improve training reliability for small data

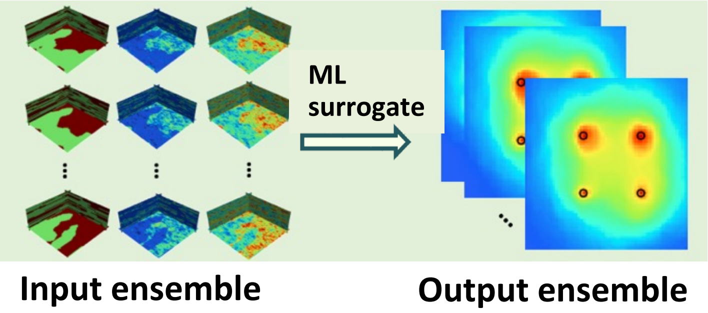


Bayesian neural network (BNN) can use a small training data set to produce an accurate surrogate by avoiding overfitting and provide UQ.

Methods to improve computational efficiency of ELM calibration

Surrogate Modeling

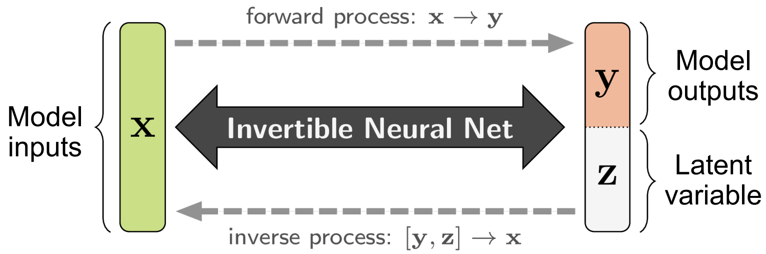
Build a fast surrogate of the ELM, and then evaluate the surrogate in the standard UQ process



- Key is to build an accurate surrogate using limited ELM simulation samples.
- Dimension reduction, BNN, physics-informed ML, etc.
- Evaluating the surrogate model in traditional UQ process to reduce costs.

Invertible Neural Network

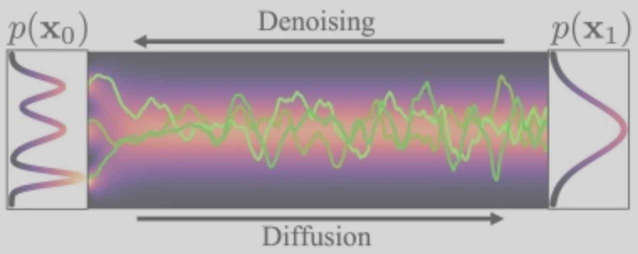
INN solves UQ problem directly based on ELM model simulation samples



- INN learns bijective mapping between inputs and outputs.
- Evaluating the trained INN backwards produces parameter posterior samples.

Diffusion-based UQ

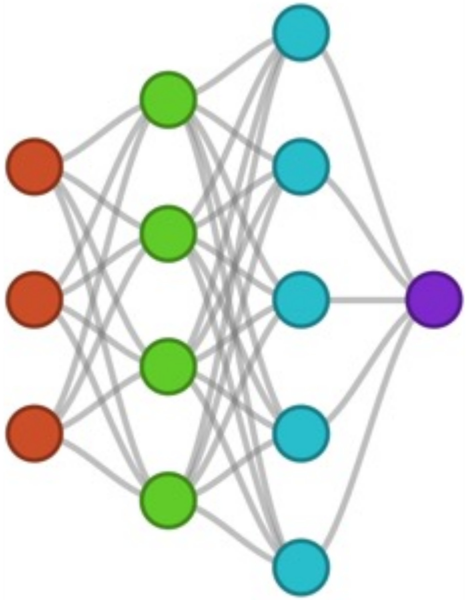
Diffusion models are generative ML method.



- It includes two processes.
- The forward diffusion process adds noise to the target distribution and transform it to a standard Gaussian.
- The reverse denoising process transforms the standard Gaussian samples to the target parameter posterior samples.

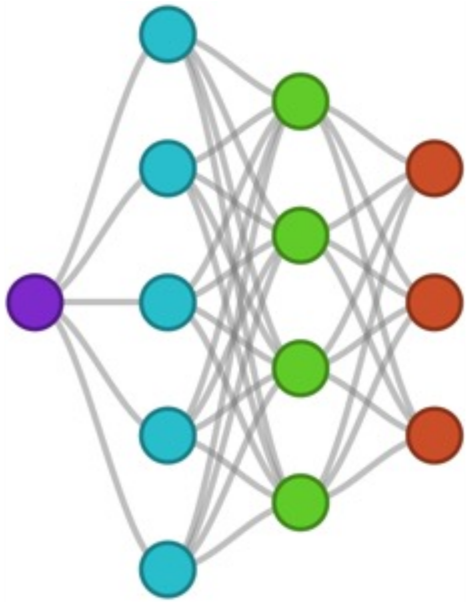
Invertible neural networks (INN) to efficiently solve UQ

- NN for forward model approximation, learns x → y mapping.



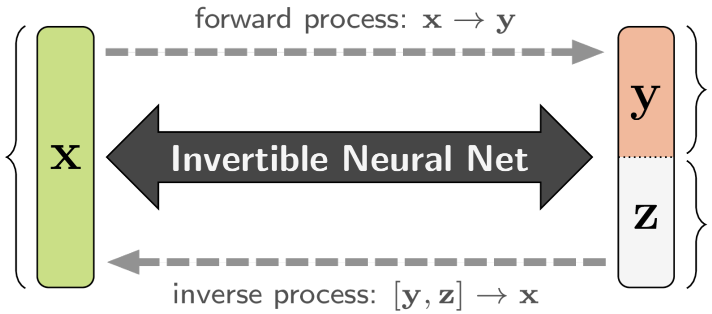
Input x → Output y

- NN can be problematic to learn y → x mapping due to nonunique solutions of x for a given y.



Output y ~~→~~ Input x

- We developed INN by introducing a latent variable z such that the mapping between x and [y, z] is bijective.
- INN learns the bijective mapping.



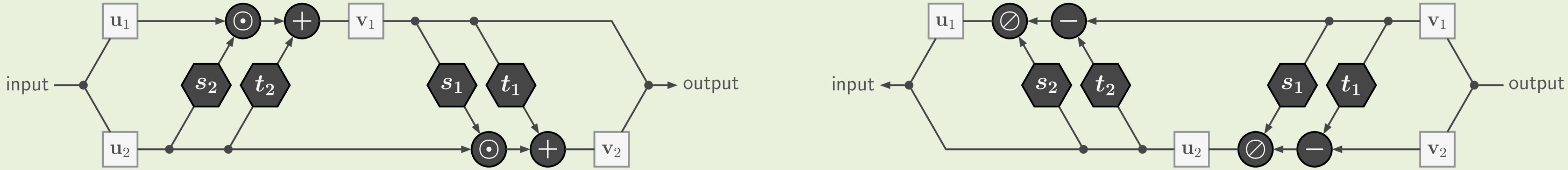
Input x ⇄ {Output y, Latent z}

INN solves both forward and inverse UQ efficiently

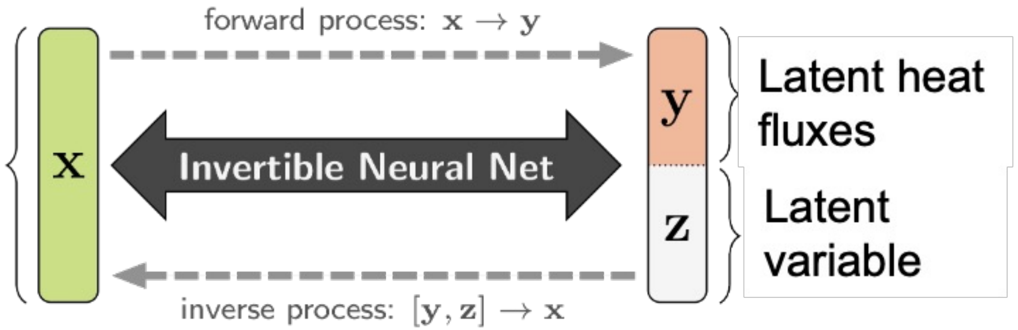
- INN is a class of networks that provide bijective mappings between inputs and outputs.
- INN solves both probabilistic inverse problems and forward approximations efficiently.

INN

- Building block of INN is the affine coupling layer;
- INN learns bijective mapping between inputs [u1, u2] and outputs [v1, v2];
- These affine coupling layers are chained to construct deep INNs.



Parameter
rootb_par
slatop
flnr
frootcn
froot_leaf
br_mr
crit_dayl
crit_onset_gdd



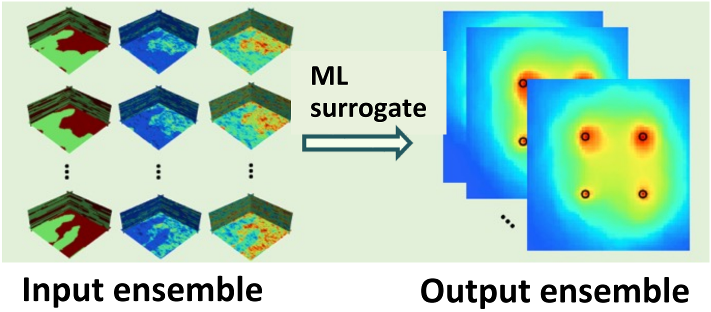
Lu et al., ICLR 2022.

- We train INN on the forward process.
- Evaluate the trained INN backwards for parameter uncertainty quantification.
- In application to ELM model calibration, INN produced parameter posterior distributions like those produced by MCMC but with significantly enhanced computational efficiency.

Methods to improve computational efficiency of ELM calibration

Surrogate Modeling

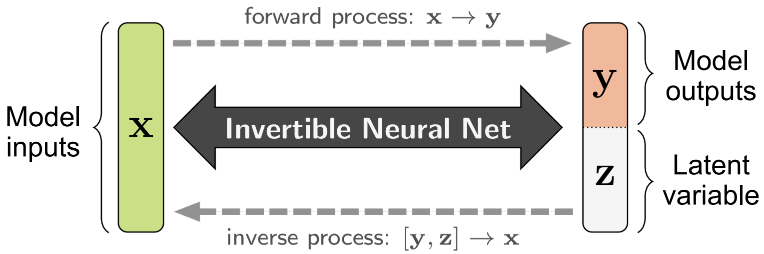
Build a fast surrogate of the ELM, and then evaluate the surrogate in the standard UQ process



- Surrogate modeling demands an accurate surrogate across the entire parameter space.
- It requires a new MCMC simulation whenever likelihood functions vary.

Invertible Neural Network

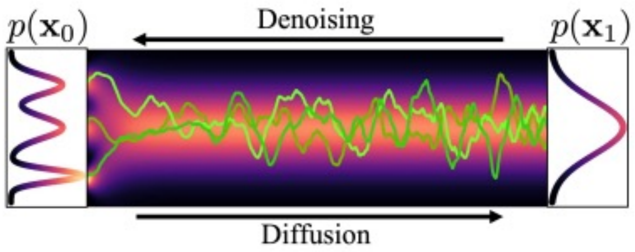
INN solves UQ problem directly based on ELM model simulation samples



- It is limited to INN structure.
- The dimension of [x] and [y, z] should be the same.
- The training of INN is unstable, heavily dependent on the hyperparameters.

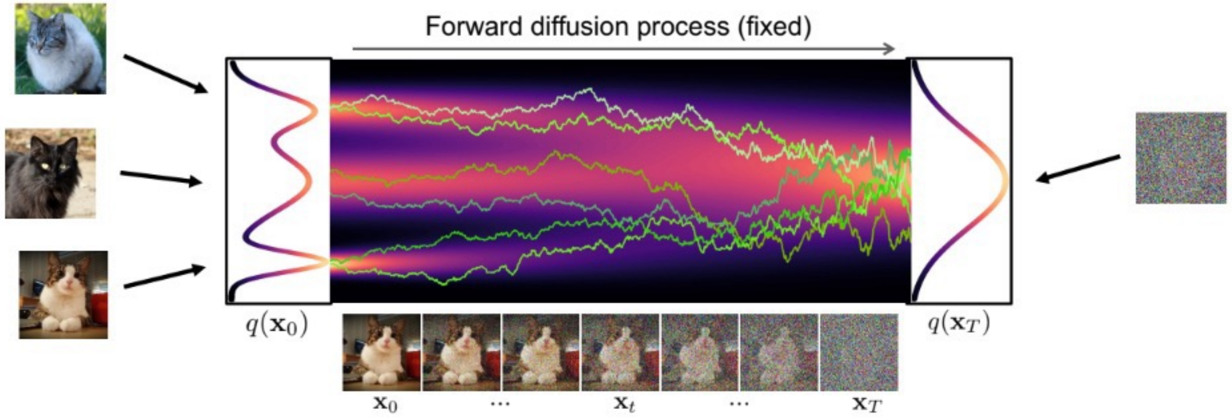
Diffusion-based UQ

Diffusion models are generative ML method.



- It includes two processes.
- The forward diffusion process adds noise to the target distribution and transform it to a standard Gaussian.
- The reverse denoising process transforms the standard Gaussian samples to the target parameter posterior samples.

Score-based diffusion models to estimate $p(X|Y = y)$ for UQ



The forward process is given by a forward stochastic differential equation (SDE):

$$dZ_t = b(t)Z_t dt + \sigma(t)dW_t \quad \text{with } Z_0 = X|Y \text{ and } Z_1 = Z, \tag{5}$$

where Z_0 is the initial state and Z_1 is the terminal state, W_t is a standard d-dimensional Brownian motion, $b(t)$ is the drift coefficient, and $\sigma(t)$ is the diffusion coefficient. The backward process is given by an associated reverse-time SDE:

$$dZ_t = \left[b(t)Z_t - \sigma^2(t)S(Z_t, t)\right] dt + \sigma(t)dB_t \quad \text{with } Z_0 = X|Y \text{ and } Z_1 = Z, \tag{6}$$

where B_t is the backward Brownian motion and $S(Z_t, t)$ is the score function.

The score function in Eq. (6) is defined by

$$S(Z_t, t) := \nabla_z \log p(Z_t),$$

which is uniquely determined by the initial distribution $p(Z_0)$ and the coefficients $b(t)$ and $\sigma(t)$ in the forward SDE of Eq. (5). Substituting

$$p(Z_t) = \int p(Z_t, Z_0)dZ_0 = \int p(Z_t|Z_0)p(Z_0)dZ_0$$

- ❖ Traditional score-based diffusion model uses a NN to learn the score function;
  - The training data are generated by solving the forward process;
  - It requires storing many stochastic paths of the forward SDE;
  - It is computationally expensive and memory intensive;
- ❖ With the learned score function, it solves a reverse SDE repeatedly to generate targe samples X|Y;
  - For each sample generation, it requires solving the reverse SDE for many time steps.

Our Diffusion-Based Uncertainty Quantification (DBUQ) method

- <u>Objective:</u> draws samples to approximate posterior distribution of parameter X given observed y,

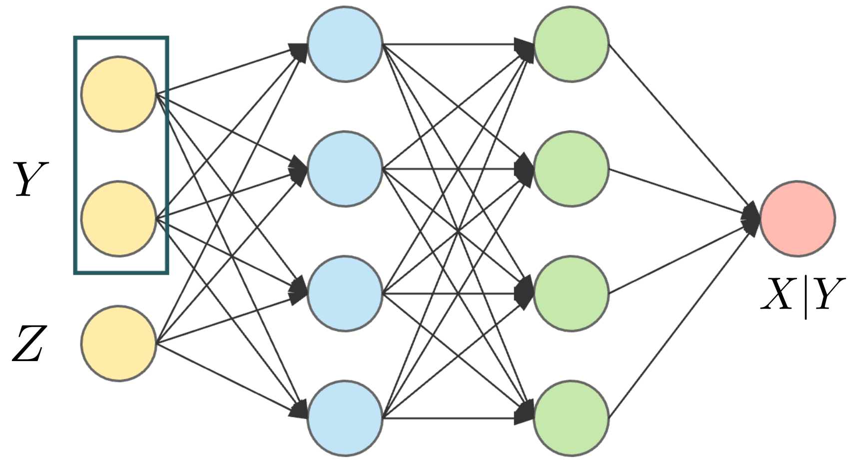
$$p(X|Y=y) \propto p(Y=y|X)p(X)$$

- DBUQ formulates a generative model *F* to draw the target samples ,

$$X|Y \approx F(Y, Z; \theta)$$

- DBUQ uses a NN to estimate *F* using training samples;

- After training, this NN can evaluate standard Gaussian samples, Z, to quickly generate the desired parameter posterior samples $X|Y$ at $Y=y$



❖ Use a NN to learn the relationship between [Y, Z] and X|Y;
  - X|Y is the parameter of interest;
  - Y is the observation variable;
  - Z is the standard Gaussian variable.

❖ The generation of target samples of X|Y is computationally and memory efficient;

❖ For any given observation data, the NN can generate corresponding parameter posterior samples without the need for re-training.

Our Diffusion-Based Uncertainty Quantification (DBUQ) method

- Objective: Generate training samples to train the NN and learn the relationship between Z and X|Y.
- In the reverse SDE, the relationship between Z and X|Y is not deterministic, which means it cannot be learned.
- Through some derivation, we found the listed ODE solves the same distribution of X|Y, and it also provides a unique mapping between Z and X|Y.
- Thus, we solve the ODE to generate the training data.
- To solve the ODE, we need to estimate the score function S, which involves an integral.

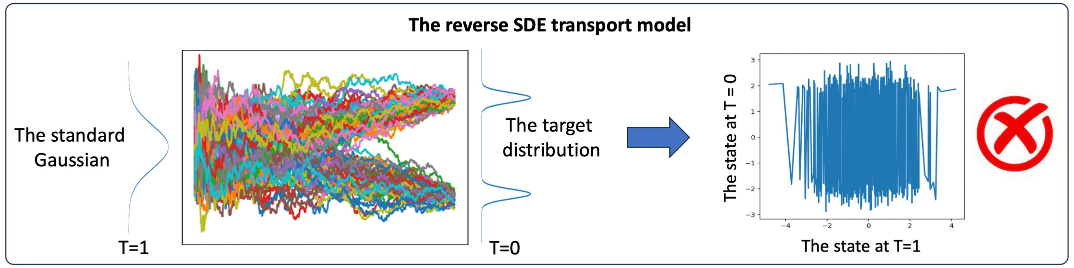
$$S(Z_t, t) := \nabla_z \log p(Z_t).$$

$$p(Z_t) = \int p(Z_t, Z_0) dZ_0 = \int p(Z_t | Z_0) p(Z_0) dZ_0$$

- We use Monte Carlo (MC) method to estimate the integral based on prior samples $\mathcal{D}_{\text{prior}} = \{(x_j, y_j)\}_{j=1}^{J}$ generated from the ELM model simulations.

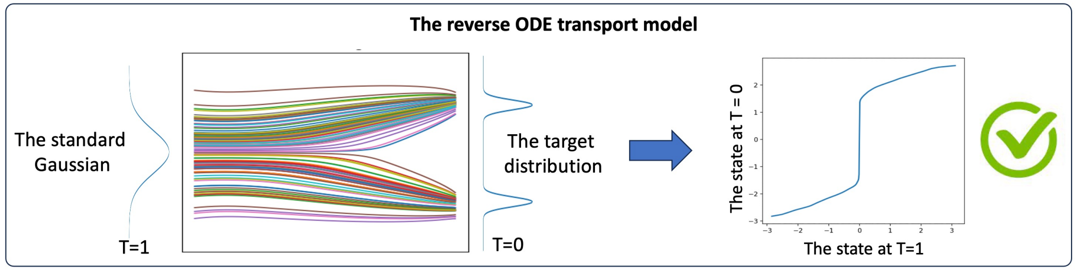
Reverse SDE:

$$dZ_t = \left[b(t)Z_t - \sigma^2(t)S(Z_t, t)\right] dt + \sigma(t) dB_t \quad \text{with } Z_0 = X|Y \text{ and } Z_1 = Z,$$



Reverse ODE:

$$dZ_t = \left[b(t)Z_t - \frac{1}{2}\sigma^2(t)S(Z_t, t)\right] dt \quad \text{with } \; Z_0 = X|Y \text{ and } Z_1 = Z,$$



- The ODE shows smooth and unique mapping between Z and X|Y;
- We solve ODE to generate training samples to learn the mapping between Z and X|Y.

Our Diffusion-Based Uncertainty Quantification (DBUQ) method

Our DBUQ method for parameter uncertainty quantification

Input: Prior sample set $\mathcal{D}_{\text{prior}} = \{(x_j, y_j)\}_{j=1}^{J}$;

Output: Trained generative model $F(Y, Z; \theta)$;

Procedure:

1. **for** $m = 1, \ldots, M$
    - Estimate score function using Monte Carlo estimation through Eq. (13)–(15);
    - Solve the ODE in Eq. (16) with the estimated score function;
    - Obtain one sample (x_m, y_m, z_m) in the dataset $\mathcal{D}_{\text{label}}$;

   end
2. Train a NN to approximate the generative model $X = F(Y, Z; \theta)$ using the training data $\mathcal{D}_{\text{label}} = \{(x_m, y_m, z_m)\}_{m=1}^{M}$.

Generate parameter posterior samples: for a given observation y, evaluate the trained F at standard Gaussian samples Z to generate parameter posterior samples to approximate the target distribution $p(X|Y = y)$

DBUQ method is computationally and memory efficient

Traditional score-based diffusion models

Computationally expensive, memory-intensive

- ❖ Use a NN to learn the score function;
  - The training data are generated by solving the forward SDE process;
  - It requires storing many stochastic paths of the forward SDE, which is computationally expensive and memory intensive;
- ❖ Solve a reverse SDE repeatedly using the learned score function to generate targe samples;
  - For each sample generation, it requires to solve the SDE for many time steps.

Our DBUQ method

Computationally and memory efficient
Amortized Bayesian inference

- ❖ Formulate a supervised learning problem to estimate the sample generator *F;*
  - After the generator is trained, it can quickly generate numerous parameter posterior samples for any given observations;
- ❖ Solve a reverse ODE to generate the training data to train a NN to estimate the *F;*
  - Computationally and memory efficient as solving the ODE only needs to store the initial and terminal states of the transport path;

An illustrative example of DBUQ

Problem

We use a simple 1D problem to illustrate our DBUQ method. The forward model $g(X)$ in the likelihood function of Eq. (2) is defined by

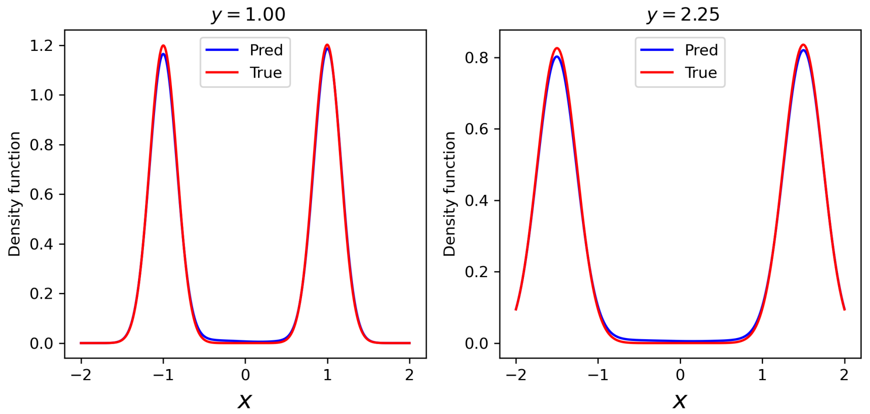
$$g(X) = X^2,$$

where the prior distribution of X is defined by a uniform distribution $\mathcal{U}([-2,2])$ in the domain $[-2,2]$. The observation variable Y is defined by

$$Y = g(X) + \varepsilon,$$

where ε follows the Gaussian distribution $\mathcal{N}(0,\sigma^2)$ with $\sigma = 0.1$.

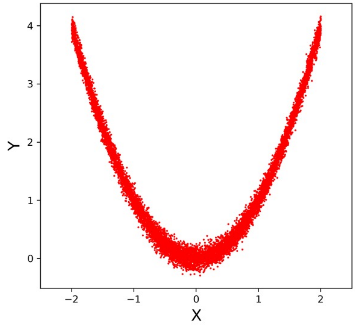
Result



Procedure

Raw data

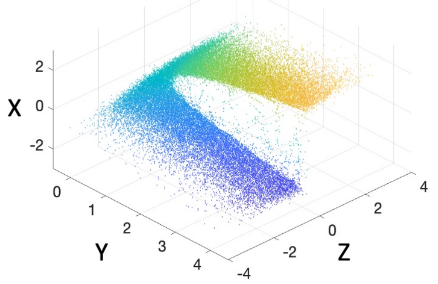
$\mathcal{D}_{\text{prior}} = \{(x_j, y_j)\}_{j=1}^{J}$ in Eq. (4)



Diffusion model

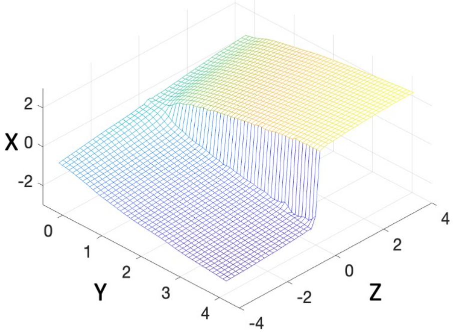
Labeled data

$\mathcal{D}_{\text{label}} := \{(x_m, y_m, z_m)\}_{m=1}^{M}$ in Eq. (17)



NN training

The generator $X = F(Y, Z; \theta)$



- Multimodal distributions can be common for earth system model parameter estimation;
- It is challenging for UQ methods to capture all the possible modes;
- DBUQ accurately approximates the target bi-modal distributions;
- DBUQ is computationally efficient, taking < 2min for this problem.

Apply DBUQ to improve ELM calibration

- Problem: Use DBUQ to estimate 8 ELM parameters;
- Observation: Annual averaged latent heat flux (LH) for 5 years at the Missouri Ozark AmeriFlux site in 2006-2010;
- Prior sample: 1000 paired samples $\mathcal{D}_{\text{prior}} = \{(x_j, y_j)\}_{j=1}^{J}$
- Two case studies:
  - Synthetic case for method verification
  - Real observations application
- Compare DBUQ with MCMC for performance evaluation

Parameter name	Parameter range
rootb_par	[0.5, 4]
slatop	[0.01, 0.05]
flnr	[0.1, 0.4]
frootcn	[25, 60]
froot_leaf	[0.3, 1.5]
br_mr	[1.5e-6, 4e-6]
crit_dayl	[35000, 45000]
crit_onset_gdd	[600, 1000]

DBUQ

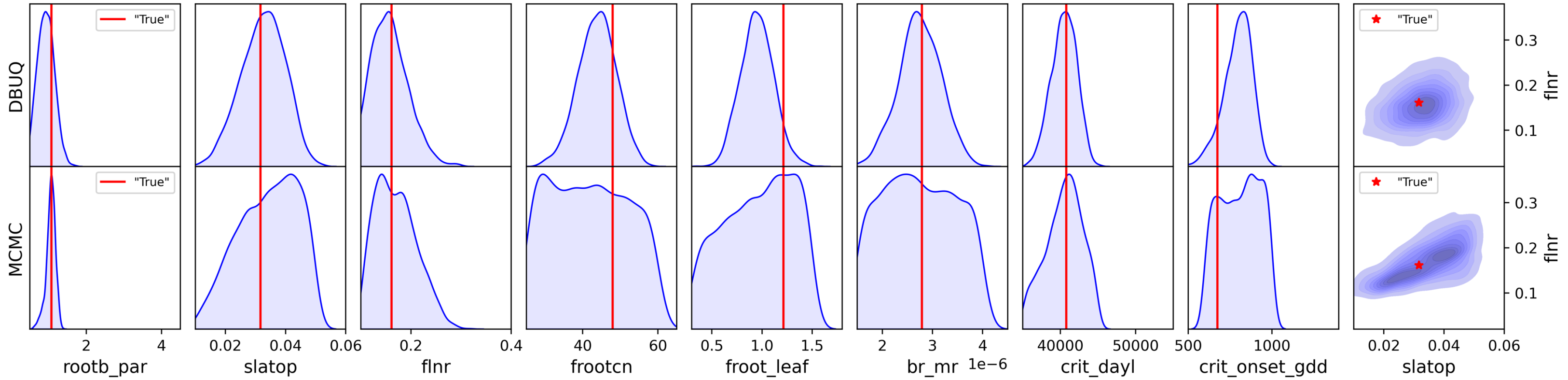
- Input: 1000 prior samples $\mathcal{D}_{\text{prior}} = \{(x_j, y_j)\}_{j=1}^{J}$
- Output: **a trained generator** which can be quickly evaluated to generate target samples for any given observations;
- Computing time: < 10 min for solving both cases
- **Particularly suitable for site-specific earth system model calibration at a global scale** due to its computational efficiency and amortized inference.

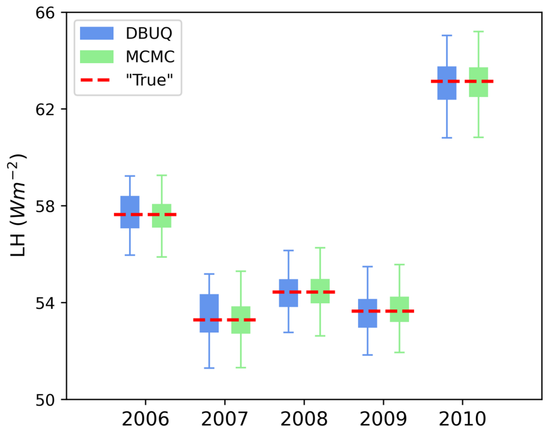
MCMC

- Input: 1000 prior samples $\mathcal{D}_{\text{prior}} = \{(x_j, y_j)\}_{j=1}^{J}$
- Procedure: build a surrogate model on the prior samples, and then perform MCMC simulations on the surrogate;
- Output: **a set of posterior samples;** For a different observation, we need to re-run MCMC;
- Computing time: ~ 5 hours for one case to generate the same number of posterior samples as DBUQ.

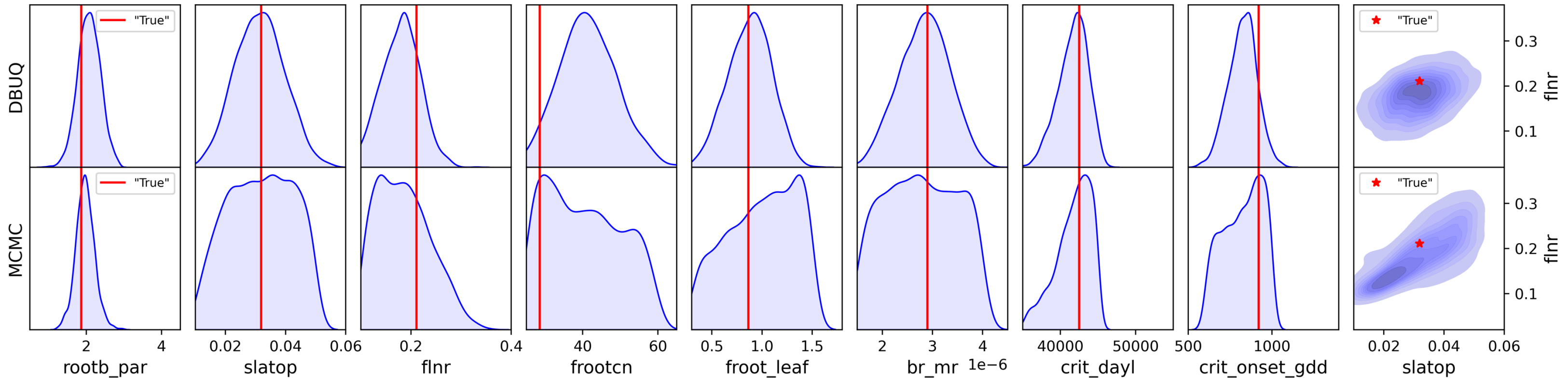
DBUQ accurately and efficiently estimates parameter PDFs

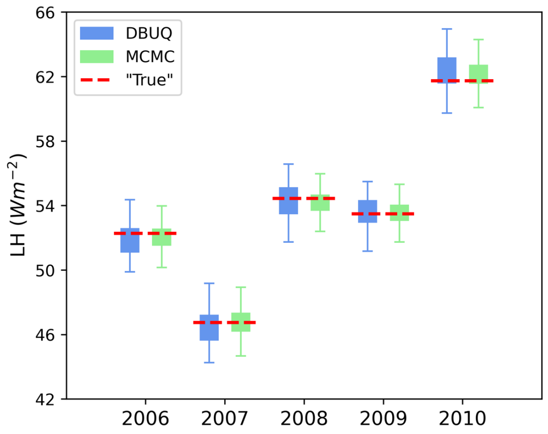
Synthetic case I





Synthetic case II

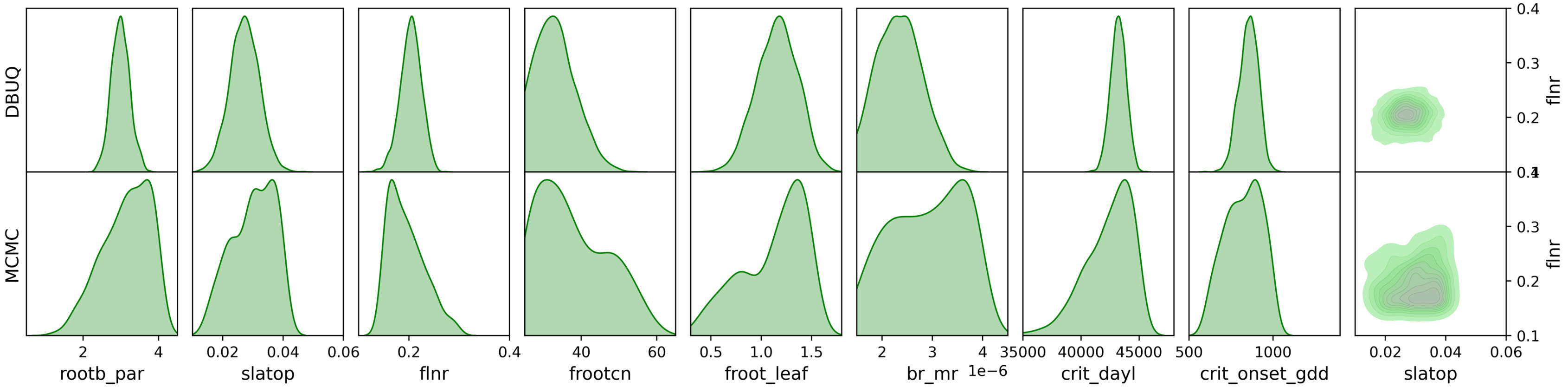


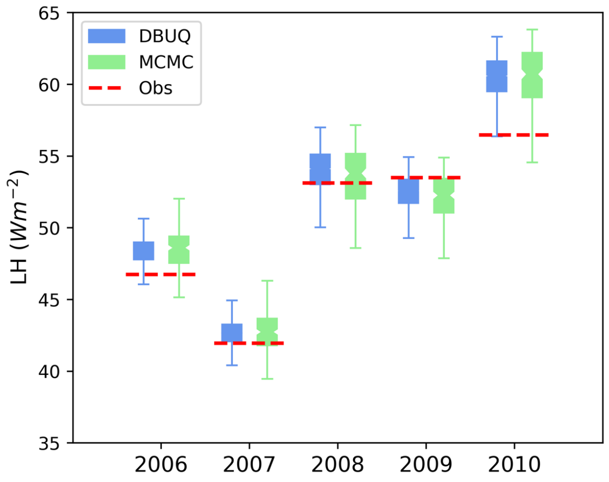


- ❖ DBUQ shows high accuracy in approximating the parameter posterior distributions.
  - Similar to the MCMC results, both accurately estimate the “true” parameter values with high probability.
- ❖ DBUQ demonstrates an accurate model calibration, as the prediction samples simulated from the parameter posterior samples are closely around the “true” observation.

DBUQ accurately and efficiently estimates parameter PDFs

Real observation case





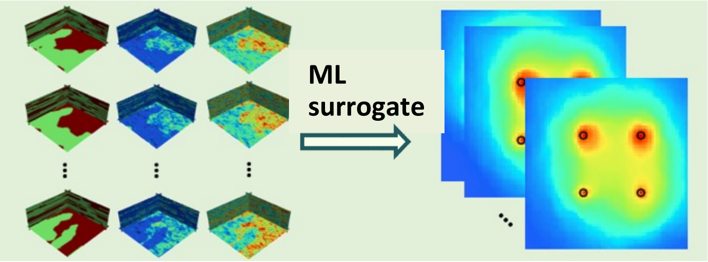
- ❖ DBUQ again shows high accuracy in approximating the parameter posterior distributions.
  - Similar to the MCMC results in estimating both the marginal and joint PDFs.
- ❖ DBUQ demonstrates an accurate model calibration, as the prediction samples simulated from the parameter posterior samples are closely around the observation.
- ❖ Note, DBUQ achieves comparable accuracy with MCMC with significantly less computational time.
  - DBUQ: 10 mins for all the three case studies;
  - MCMC: 5 hours for one case study;

Lu et al., ICLR 2024.

Summary: advanced ML methods for efficient UQ

Surrogate Modeling

Build a fast surrogate of the ELM, and then evaluate the surrogate in the standard UQ process

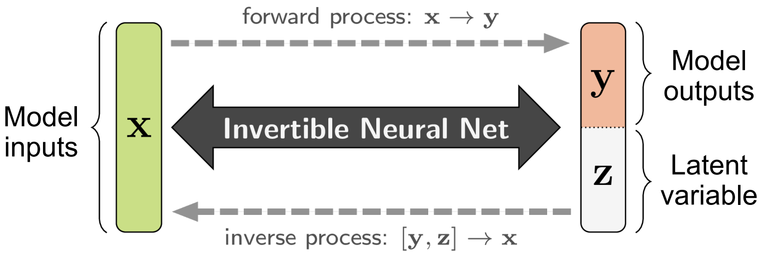


Input ensemble **Output ensemble**

- Surrogate modeling demands an accurate surrogate across the entire parameter space.
- It requires a new MCMC simulation whenever likelihood functions vary.

Invertible Neural Network

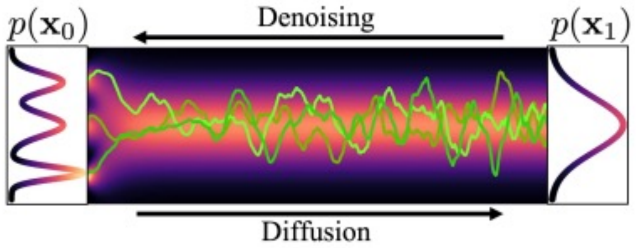
INN solves UQ problem directly based on ELM model simulation samples



- It is limited to INN structure.
- The dimension of [x] and [y, z] should be the same.
- The training of INN is unstable, heavily dependent on the hyperparameters.

Diffusion-based UQ

Our DBUQ method generates parameter posterior samples by evaluating the NN.



- It can accurately quantify parameter uncertainty.
- It is computationally and memory efficient.
- It performs amortized Bayesian inference.
- It enables real-time and large-scale model calibration and UQ.