

Bayesian Computation with Generative Diffusion Models via Multilevel Monte Carlo

Abdul-Lateef Haji-Ali^{1,2}, Marcelo Pereyra^{1,2}, Luke Shaw³,
Konstantinos Zygalakis^{1,3}

¹Maxwell Institute ²Heriot-Watt University

³Universitat Jaume I

⁴University of Edinburgh

The 30th Biennial Numerical Analysis Conference
24 June 2025

Outline

- 1 Motivation
- 2 Diffusion Models for Bayesian Inversion
- 3 Multilevel Monte Carlo (MLMC)
- 4 Numerical Results and Discussion

Motivation: The Promise of Diffusion Models

- **Diffusion Models:** Class of stochastic samplers that rely on neural networks to generate samples from a posterior distribution.
- **High-quality sampling:** DMs provide state-of-the-art sample quality for complex distributions.
- **Flexible conditioning:** Handle diverse inverse problems (denoising, inpainting, super-resolution) through unified framework.
- Well-defined **probability model** enables proper UQ.

Motivation: The Promise of Diffusion Models

- **Diffusion Models:** Class of stochastic samplers that rely on neural networks to generate samples from a posterior distribution.
- **High-quality sampling:** DMs provide state-of-the-art sample quality for complex distributions.
- **Flexible conditioning:** Handle diverse inverse problems (denoising, inpainting, super-resolution) through unified framework.
- Well-defined **probability model** enables proper UQ.

The Computational Challenge

- **Large-scale UQ:** Reliable uncertainty estimates may require 10^4 - 10^6 samples, especially when variability is significant.
- **Slow sampling:** Typical DM requires 10^2 - 10^3 neural function evaluations (NFEs) per sample.
- **Prohibitive cost:** Full UQ analysis can require 10^7 - 10^9 NFEs.

Motivation: Why Diffusion Models for UQ?

The need for speed

- **Practical deployment** in time-sensitive applications (medical imaging, real-time systems).
- **Large-scale uncertainty studies** (global sensitivity analysis, robust optimization).
- **Hyperparameter tuning** (prior selection, likelihood calibration).

Solutions:

- Reduce cost per NFE (pruning, quantization).
- Reduce number of NFE's per sample (distillation).
- Reduce number of NFE's per statistic computation.

Bayesian framework

- Aim of a Machine Learning algorithm: Estimate $x \in \mathbb{R}^n$, $n \gg 1$, given observed data y .
- Bayesian framework: $Y \sim \mathcal{P}(\mathcal{A}(x))$, conditional on $X = x$.
- Sampling from the conditional density of X , $p(\cdot|y) \propto \mathcal{L}(y|\cdot)\pi(\cdot)$, for likelihood \mathcal{L} and prior π .
- Our goal is to quantify the uncertainty that such a probability distribution entails.

Denoising Diffusion Probabilistic Models

- **Finite sequence of noising kernels**, or a continuous-time SDE.
- Build sequence of states $\{X_i\}_{i=0}^T$ with $X_0 = X$.
- Sample from the joint density of (X_0, \dots, X_T) given $Y = y$; joint density admits the desired conditional density as marginal.
- Write

$$p_{0:T}(x_0, \dots, x_T | y) = \left(\prod_{t=1}^T \hat{\mathcal{K}}_{t-1}(x_{t-1} | x_t, y) \right) p_T(x_T | y),$$

for **reverse transition kernels**, $(\hat{\mathcal{K}}_t)_{t=0}^{T-1}$, and a *pre-determined* posterior density $p_T(\cdot | y)$ of the final state X_T .

- For example, $p_T(\cdot | y) \in \{\phi_n(\cdot), \delta_y(\cdot), \phi_n(\cdot; (\mathbb{I}_n - \mathbb{M})y, \mathbb{M})\}$.

DDPMs: Forward transition kernels, K

Gaussian, Markovian forward process:

$$K_t(x|x_{t-1}) = \phi_n \left(x; \sqrt{\frac{\gamma_t}{\gamma_{t-1}}} x_{t-1}, \left(1 - \frac{\gamma_t}{\gamma_{t-1}} \right) \mathbb{I}_n \right),$$

for $x \in \mathbb{R}^n$ and where $1 = \gamma_0 > \gamma_1 > \dots > \gamma_T > 0$.

Hence

$$K_{t;0}(x|x_0) = \phi_n(x; \sqrt{\gamma_t} x_0, (1 - \gamma_t) \mathbb{I}_n), \quad 1 \leq t \leq T,$$

and we define the score function, for any $x \in \mathbb{R}^n$,

$$\nabla \log K_{t;0}(x_t|x_0) = (\sqrt{\gamma_t} x_0 - x_t) / (1 - \gamma_t),$$

and we solve for x_0

$$x_0(x_t; y, t) = \frac{1}{\sqrt{\gamma_t}} (x_t + (1 - \gamma_t) \nabla \log K_{t;0}(x_t|x_0))$$

DDPMs: Reverse Kernel, λ

By conditioning on the initial state and assuming intermediate states are independent of the observation Y ,

$$\hat{\lambda}_{t-1}(x|x_t, y) = \int_{\mathbb{R}^n} \lambda_{t-1;0,t}(x|x_0, x_t) \hat{\lambda}_{0;t}(x_0|x_t, y) dx_0.$$

We use

$$\hat{\lambda}_{0;t}(\cdot|x_t, y) = \delta_{\hat{x}_0}(\cdot)$$

where

$$\hat{x}_0(x_t; y, t) = \frac{1}{\sqrt{\gamma_t}}(x_t + (1 - \gamma_t)\nabla \log K_{t;0}(x_t|x_0))$$

Finally,

$$\lambda_{t-1;0,t}(x|x_0, x_t) = \phi_n(\cdot, \mu_{t-1;0,t}, \sigma_{t-1;0,t}^2 \mathbb{I}_n)$$

for some known mean and variance.

DDPMs: Reverse Kernel, λ

By conditioning on the initial state and assuming intermediate states are independent of the observation Y ,

$$\hat{\lambda}_{t-1}(x|x_t, y) = \int_{\mathbb{R}^n} \lambda_{t-1;0,t}(x|x_0, x_t) \hat{\lambda}_{0;t}(x_0|x_t, y) dx_0.$$

We use

$$\hat{\lambda}_{0;t}(\cdot|x_t, y) = \delta_{\hat{x}_0}(\cdot)$$

where

$$\hat{x}_0(x_t; y, t, \theta) = \frac{1}{\sqrt{\gamma_t}}(x_t + (1 - \gamma_t) s_{\theta}(x_t, y, t))$$

Finally,

$$\lambda_{t-1;0,t}(x|x_0, x_t) = \phi_n(\cdot, \mu_{t-1;0,t}, \sigma_{t-1;0,t}^2 \mathbb{I}_n)$$

for some known mean and variance.

DDPMs: Four Sources of Errors

$$x_{t-1} = \sqrt{\frac{\gamma_{t-1}}{\gamma_t}} x_t - \sqrt{\frac{\gamma_t}{\gamma_{t-1}}} \left(\Gamma_{t-1} - \frac{\gamma_{t-1}}{\gamma_t} \Gamma_t \right) s_\theta(x_t, y, t) + \sqrt{\frac{\Gamma_{t-1}}{\Gamma_t} \left(1 - \frac{\gamma_t}{\gamma_{t-1}} \right)} \xi_t, \quad \text{where } \Gamma_t = 1 - \gamma_t$$

- **Model error:** From imperfect score network training based on a finite training-set.
- **Finite-time error:** We have to choose $\gamma_T > 0$ (to ensure γ_{t-1}/γ_t is small), hence X_T will not be exactly Gaussian (not fully diffused).
- **Truncation error:** The posterior $p(\cdot | y)$ has smaller support than $p_t(\cdot | y)$, leading to blow up in s_θ as $t \rightarrow 0$; need to stop early or have a Gaussian approximation independent of the score.
- **Discretization error** (our focus today): We can skip M steps, sampling directly x_{t-M} given x_t .

DDPMs: Four Sources of Errors

$$x_{t-M} = \sqrt{\frac{\gamma_{t-M}}{\gamma_t}} x_t - \sqrt{\frac{\gamma_t}{\gamma_{t-M}}} \left(\Gamma_{t-M} - \frac{\gamma_{t-M}}{\gamma_t} \Gamma_t \right) s_\theta(x_t, y, t) \\ + \sqrt{\frac{\Gamma_{t-M}}{\Gamma_t} \left(1 - \frac{\gamma_t}{\gamma_{t-M}} \right)} \xi_t, \quad \text{where } \Gamma_t = 1 - \gamma_t$$

- **Model error:** From imperfect score network training based on a finite training-set.
- **Finite-time error:** We have to choose $\gamma_T > 0$ (to ensure γ_{t-1}/γ_t is small), hence X_T will not be exactly Gaussian (not fully diffused).
- **Truncation error:** The posterior $p(\cdot | y)$ has smaller support than $p_t(\cdot | y)$, leading to blow up in s_θ as $t \rightarrow 0$; need to stop early or have a Gaussian approximation independent of the score.
- **Discretization error** (our focus today): We can skip M steps, sampling directly x_{t-M} given x_t .

Monte Carlo Estimation

- Goal: Estimate $E[f(X)]$, where $X \sim p(\cdot)$.
- Monte Carlo estimator:

$$E[f(X)] \approx E[f(\hat{X})] \approx \frac{1}{N} \sum_{i=1}^N \hat{x}^{(i)},$$

- Variance $\frac{1}{N} \text{Var}[f(\hat{X})]$, Bias $|E[f(X)] - E[f(\hat{X})]|$.
- For RMSE ε : Number of samples grow as $O(\varepsilon^{-2})$ to reduce variance.
Cost per sample grows as ε decreases to decrease bias.

Multilevel Monte Carlo: Basic Identity

- Use a hierarchy of approximations: $\hat{X}_0, \hat{X}_1, \dots, \hat{X}_L$ with increasing cost and accuracy (each adding M intermediate steps compared to the previous approximation).
- Telescoping sum:

$$E[f(\hat{X}_L)] = E[f(\hat{X}_0)] + \sum_{\ell=1}^L E[f(\hat{X}_\ell) - f(\hat{X}_{\ell-1})]$$

- Define level estimators:

$$Y_\ell = \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} \left(f(\hat{x}_\ell^{(i)}) - f(\hat{x}_{\ell-1}^{(i)}) \right) \quad \text{with coupled samples}$$

- Total estimator:

$$Y = \sum_{\ell=0}^L Y_\ell \quad \text{with} \quad Y_0 = \frac{1}{N_0} \sum_{i=1}^{N_0} f(x_0^{(i)})$$

MLMC Efficiency and Complexity

- Bias: $|E[f(\hat{X}_\ell) - f(\hat{X}_{\ell-1})]| \sim M^{-\alpha\ell}$
 Variance: $\text{Var}[f(\hat{X}_\ell) - f(\hat{X}_{\ell-1})] =: V_\ell \sim M^{-\beta\ell}$
 Sampling cost: $C_\ell \sim M^\ell$
- Optimal total cost: Choosing N_ℓ to minimize cost, yields

$$C_{\text{MLMC}} \lesssim \varepsilon^{-2} \left(\sum_{\ell=0}^L \sqrt{V_\ell C_\ell} \right)^2$$

$$\lesssim \begin{cases} \varepsilon^{-2} V_0 C_0 & V_\ell C_\ell \rightarrow 0 \\ \varepsilon^{-2} L^2 & V_\ell C_\ell \approx \text{const} \\ \varepsilon^{-2} V_L C_L & V_\ell C_\ell \rightarrow \infty \end{cases}$$

- Compare with standard MC at finest level:

$$C_{\text{MC}} \lesssim \varepsilon^{-2} V_0 C_L \Rightarrow \text{MLMC much cheaper if } V_\ell C_\ell \downarrow$$

Need for Exponential Integrators

- Diffusion models discretise stiff SDEs in reverse:

$$dX_t = (A_t X_t - B_t^2 \nabla \log K_{t;0}(x_t|x_0)) dt + B_t dW_t$$

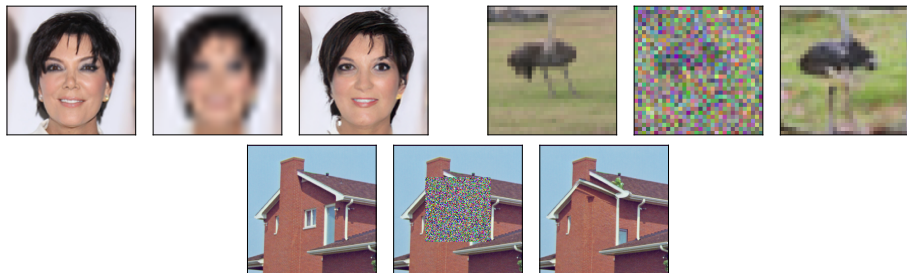
- As $t \rightarrow T$, we have $\gamma_t \rightarrow 0$, hence A_t becomes unbounded, leading to instability.
- Exponential integrators mitigate instability due to the linear terms:

$$\begin{aligned} x_{t-M}^r = & e^{-\int_t^{t-M} A_\tau d\tau} x_t \\ & - s_\theta(x_t, y, t) \int_t^{t-M} e^{\int_s^{t-M} A_\tau d\tau} B_s^2 ds \\ & + \int_t^{t-M} e^{\int_s^{t-M} A_\tau d\tau} B_s dW_s \end{aligned}$$

- Crucially, correlating the fine and coarse paths needs to take into account previous variance coefficient.

Three Imaging Inverse Problems

left to right: truth x ; observation y ; posterior sample from a DM.



- ① Super-resolution: $y = Ax$, A downsampling, ill-posed
- ② Denoising: $y = x + \eta$, high noise
- ③ Inpainting: $y = Mx$, partial masking

Use $f(x) = x^2$ to estimate marginal second moment (modelling pixel-wise uncertainty).

Convergence Rates

- Empirically fit α and β :

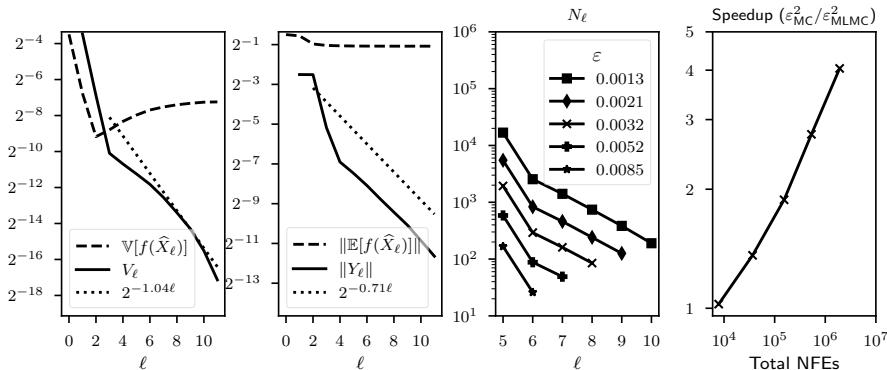
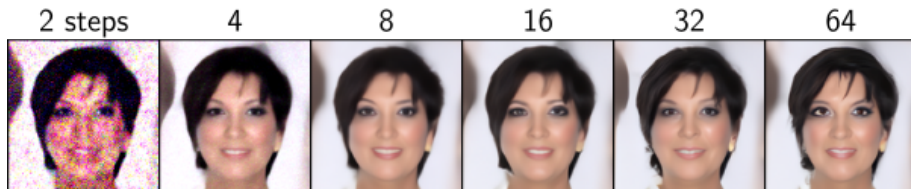
$$\left| \mathbb{E}[f(X) - f(\hat{X}_\ell)] \right| \sim M^{-\alpha\ell}$$

and

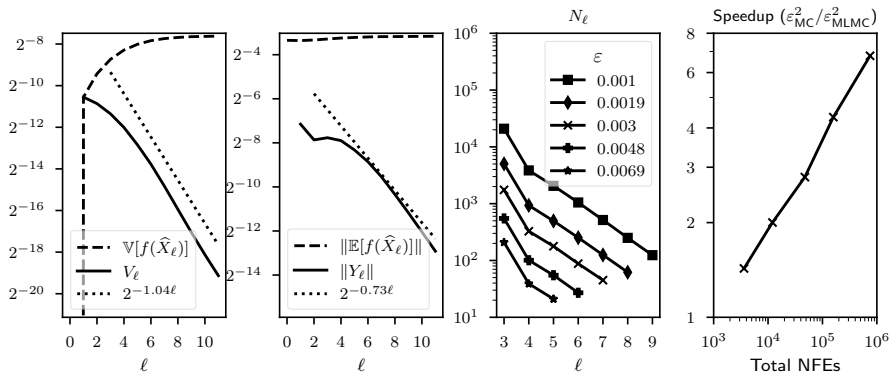
$$\text{Var}[f(\hat{X}_\ell) - f(\hat{X}_{\ell-1})] \sim M^{-\beta\ell}$$

- Expected: $\alpha = 1$ and $\beta = 2$; corresponding to rates from Milstein scheme.

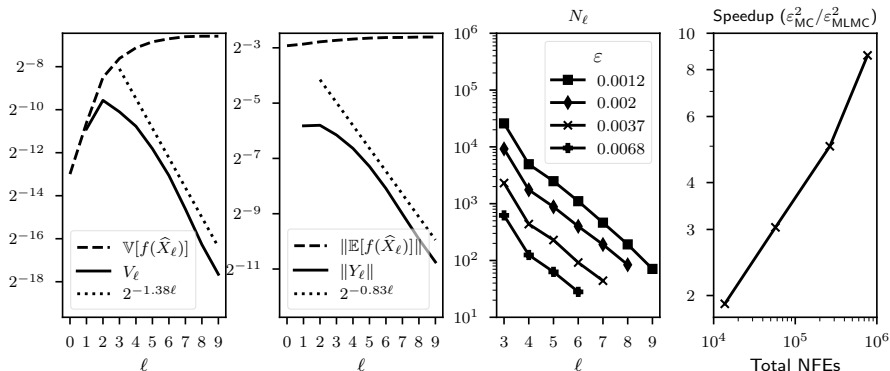
Results: Super-resolution



Results: Denoising



Results: Inpainting



Concluding Remarks

L. Shaw, A.-L. Haji-Ali, M. Pereyra, and K. Zygalakis. “Bayesian computation with generative diffusion models by Multilevel Monte Carlo”. In: *Philosophical Transactions A* (2025). DOI: [10.1098/rsta.2024.0333](https://doi.org/10.1098/rsta.2024.0333)

- MLMC is a powerful variance reduction tool – for diffusion-based Bayesian inference, MLMC reduces NFEs by 4–9 times for fixed accuracy.
- Requires careful time discretisation and coupling.
- Need better analysis of the approximation of the score function to understand degradation of convergence rates.
- Future work
 - Combine with distillation and quantisation.
 - Multilevel training of DMs.
 - Tackle cost associated to other approximations parameters (model, finite-time and truncation errors).