Sampling and friends
with dynamic measure transport

Nikolay Malkin

THE UNIVERSITY of EDINBURGH
informatics

Mila
27 November 2025


WAIT, IT WAS
ENTROPY-REGULARISED OFF-POLICY RL
ALL ALONG?
ALWAYS
HAS BEEN
imgflip.com

Summary

- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

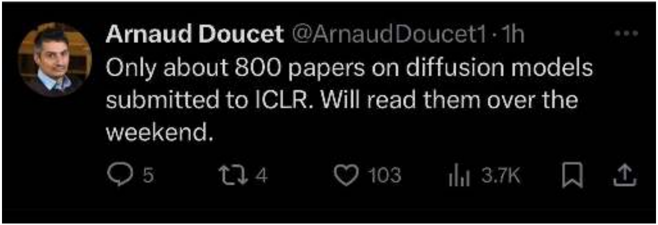
Summary

- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

Thank you to all [inspirers] and [collaborators].
In particular: J. Berner, L. Richter, M. Sendera
K. Tamogashev, S. Venkatraman.

- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

Diffusion models are everywhere. . .

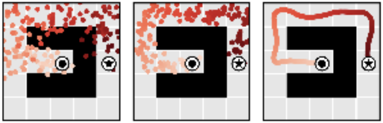


Diffusion models are everywhere. . .

'Edinburgh from Calton Hill, pointillist style'

Diffusion models are everywhere...

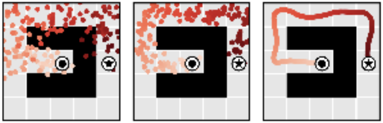
'Edinburgh from Calton Hill, pointillist style'



[Janner et al., ICML'22]

Diffusion models are everywhere. . .

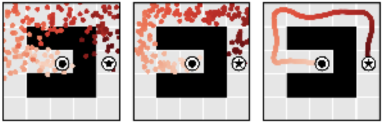
‘Edinburgh from Calton Hill, pointillist style’



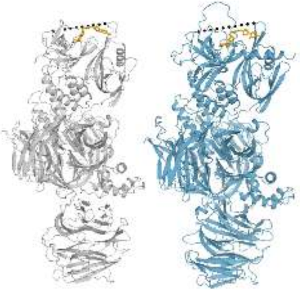
[Janner et al., ICML’22]

Diffusion models are everywhere. . .

'Edinburgh from Calton Hill, pointillist style'

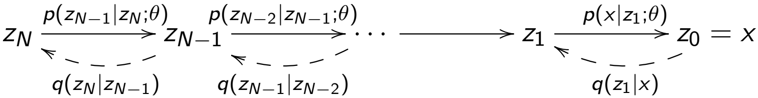


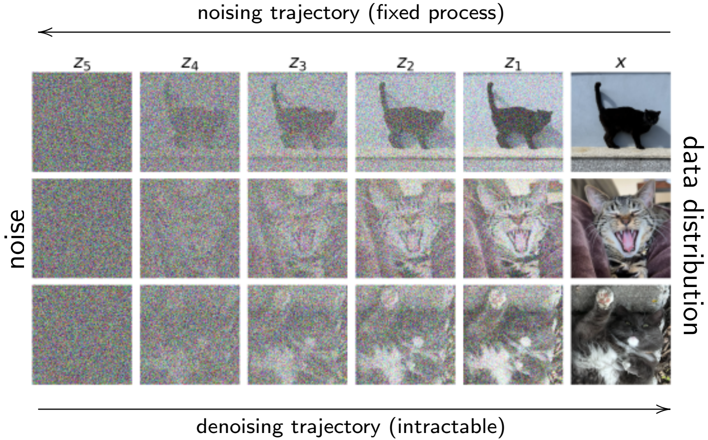
[Janner et al., ICML'22]



AlphaFold 3

Diffusion models and time discretisation





Hierarchical generative model training

The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t C_t x_t + D_t \sqrt{\Delta t} \varepsilon_t,, \ \varepsilon_t \sim \mathcal{N}(0, I)$$

Hierarchical generative model training

The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t C_t x_t + D_t \sqrt{\Delta t} \varepsilon_t,, \varepsilon_t \sim \mathcal{N}(0, I)$$

Learn to sample the denoising / reconstruction process?

- Approximate $x_{t+\Delta t} \mid x_t$ as Gaussian (valid as $\Delta t \to 0$)
- Learn its (conditional) mean and variance by MLE

Hierarchical generative model training

The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t C_t x_t + D_t \sqrt{\Delta t} \varepsilon_t,, \varepsilon_t \sim \mathcal{N}(0, I)$$

Learn to sample the denoising / reconstruction process?

- Approximate $x_{t+\Delta t} \mid x_t$ as Gaussian (valid as $\Delta t \to 0$)
- Learn its (conditional) mean and variance by MLE
  - Sample noising trajectories from data

Hierarchical generative model training

The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t C_t x_t + D_t \sqrt{\Delta t} \varepsilon_t,, \ \varepsilon_t \sim \mathcal{N}(0, I)$$

Learn to sample the denoising / reconstruction process?

- Approximate $x_{t+\Delta t} \mid x_t$ as Gaussian (valid as $\Delta t \to 0$)
- Learn its (conditional) mean and variance by MLE
  - Sample noising trajectories from data
  - Maximise their log-likelihood under denoising model

Hierarchical generative model training

The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t C_t x_t + D_t \sqrt{\Delta t} \varepsilon_t,, \varepsilon_t \sim \mathcal{N}(0, I)$$

Learn to sample the denoising / reconstruction process?

- Approximate $x_{t+\Delta t} \mid x_t$ as Gaussian (valid as $\Delta t \to 0$)
- Learn its (conditional) mean and variance by MLE
  - Sample noising trajectories from data
  - Maximise their log-likelihood under denoising model
  - $$$

Hierarchical generative model training

The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t C_t x_t + D_t \sqrt{\Delta t} \varepsilon_t,, \ \varepsilon_t \sim \mathcal{N}(0, I)$$

Learn to sample the denoising / reconstruction process?

- Approximate $x_{t+\Delta t} \mid x_t$ as Gaussian (valid as $\Delta t \to 0$)
- Learn its (conditional) mean and variance by MLE
  - Sample noising trajectories from data
  - Maximise their log-likelihood under denoising model
  - $$$
  - At optimality, as $\Delta t \to 0$, the reconstruction process is the reverse of the destruction process

Hierarchical generative model training

The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t\, C_t x_t + D_t \sqrt{\Delta t}\,\varepsilon_t,, \ \varepsilon_t \sim \mathcal{N}(0, I)$$

Learn to sample the denoising / reconstruction process?

- Approximate $x_{t+\Delta t} \mid x_t$ as Gaussian (valid as $\Delta t \to 0$)
- Learn its (conditional) mean and variance by MLE
  - Sample noising trajectories from data
  - Maximise their log-likelihood under denoising model
  - $$$
  - At optimality, as $\Delta t \to 0$, the reconstruction process is the reverse of the destruction process

$$x_{t+\Delta t} = x_t + \Delta_t \mu_\theta(x_t, t) + \sqrt{\Delta_t}\,\sigma_\theta(x_t, t)\varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, I)$$

Hierarchical generative model training

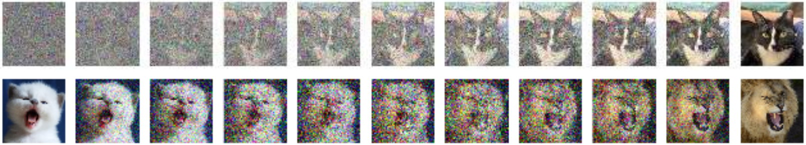
The noising / destruction process q is a discretised SDE:

$$x_{t-\Delta t} = x_t - \Delta t C_t x_t + D_t \sqrt{\Delta t} \varepsilon_t,, \ \varepsilon_t \sim \mathcal{N}(0, I)$$

Learn to sample the denoising / reconstruction process?

- Approximate $x_{t+\Delta t} \mid x_t$ as Gaussian (valid as $\Delta t \to 0$)
- Learn its (conditional) mean and variance by MLE
  - Sample noising trajectories from data
  - Maximise their log-likelihood under denoising model
  - $$$
  - At optimality, as $\Delta t \to 0$, the reconstruction process is the reverse of the destruction process

$$x_{t+\Delta t} = x_t + \Delta_t \mu_\theta(x_t, t) + \sqrt{\Delta_t} \sigma_\theta(x_t, t) \varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0, I)$$

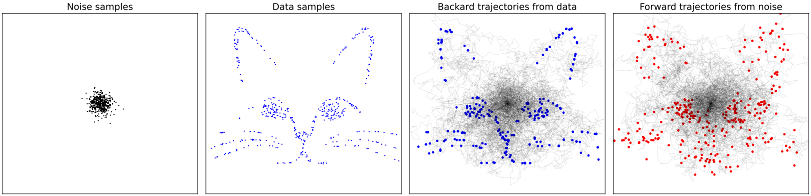


Variational interpretation of diffusion model training

- Diffusion model training matches two distributions over trajectories (sequences of latents):
  - Backward (noising) from data
  - Forward (denoising) from noise

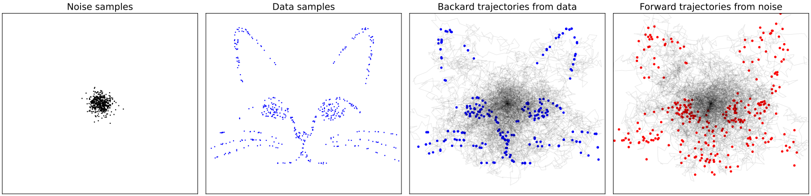
Variational interpretation of diffusion model training

- Diffusion model training matches two distributions over trajectories (sequences of latents):
  - Backward (noising) from data
  - Forward (denoising) from noise



Variational interpretation of diffusion model training

- Diffusion model training matches two distributions over trajectories (sequences of latents):
  - Backward (noising) from data
  - Forward (denoising) from noise



In continuous time, denoising $\leftrightarrow$ score matching $\leftrightarrow$ minimising KL divergence between two path space measures

- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

Diffusion models without data?

- Diffusion models are trained from data...

$$\mathrm{KL}(\text{target distribution} \cdot \text{noising process} \,\|\, \text{denoising process}_\theta)$$

Diffusion models without data?

- Diffusion models are trained from data...

$$\mathrm{KL}(\text{target distribution} \cdot \text{noising process} \,\|\, \text{denoising process}_\theta)$$

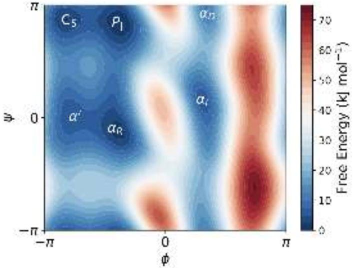
- Bayesian inference / sampling setting: we have only a target density / energy $R(\mathbf{x}) = \exp(-\mathcal{E}(\mathbf{x}))$
  - Thought of as unnormalised 'reward' (*e.g.*, a Bayesian posterior $p(\mathbf{x} \mid \mathbf{y}) \propto p(\mathbf{x})p(\mathbf{y} \mid \mathbf{x})$)
  - Related problem: product of diffusion prior $p(\mathbf{x})$ and constraint

Diffusion models without data?

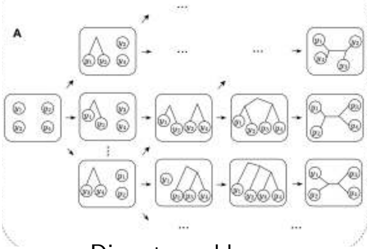
- Diffusion models are trained from data...

$$\mathrm{KL}(\text{target distribution} \cdot \text{noising process} \,\|\, \text{denoising process}_\theta)$$

- Bayesian inference / sampling setting: we have only a target density / energy $R(\mathbf{x}) = \exp(-\mathcal{E}(\mathbf{x}))$
  - Thought of as unnormalised 'reward' (*e.g.*, a Bayesian posterior $p(\mathbf{x} \mid \mathbf{y}) \propto p(\mathbf{x})p(\mathbf{y} \mid \mathbf{x})$)
  - Related problem: product of diffusion prior $p(\mathbf{x})$ and constraint



[Phillips et al., χ:2408.15905]



Discrete problems:
[Zhou et al., ICLR'24, χ:2310.08774]

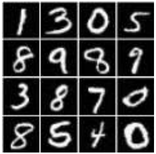
Diffusion models without data?

- Diffusion models are trained from data...

$$\mathrm{KL}(\text{target distribution} \cdot \text{noising process} \,\|\, \text{denoising process}_\theta)$$

- Bayesian inference / sampling setting: we have only a target density / energy $R(\mathbf{x}) = \exp(-\mathcal{E}(\mathbf{x}))$
  - Thought of as unnormalised 'reward' (*e.g.*, a Bayesian posterior $p(\mathbf{x} \mid \mathbf{y}) \propto p(\mathbf{x})p(\mathbf{y} \mid \mathbf{x})$)
  - Related problem: product of diffusion prior $p(\mathbf{x})$ and constraint

diffusion model



$+$ classifier $p(7 \mid x)$ $\rightarrow$

conditional samples

Diffusion models without data?

Approaches to training a diffusion model without data:

- Optimise the reverse KL ($\leftrightarrow$ stochastic control methods)

  KL(denoising process$_\theta$ $\|$ target distribution $\cdot$ noising process)

  - KL: Memory issues from deep reparametrisation trick
  - Mode-seeking behaviour
- PDE approaches

  [Nüsken & Richter, PDEA, χ:2005.05409], [Máté & Fleuret, TMLR, χ:2301.07388], [Sun et al., χ:2407.07873] and others

Diffusion models without data?

Approaches to training a diffusion model without data:

- Optimise the reverse KL ($\leftrightarrow$ stochastic control methods)

  KL(denoising process$_\theta$ $\|$ target distribution $\cdot$ noising process)

  - KL: Memory issues from deep reparametrisation trick
  - Mode-seeking behaviour
- PDE approaches

  [Nüsken & Richter, PDEA, χ:2005.05409], [Máté & Fleuret, TMLR, χ:2301.07388], [Sun et al., χ:2407.07873] and others
- Monte Carlo methods to estimate $\nabla \log(R * \mathcal{N}(0, V(t)))$
  - Diffusion samplers are annealed importance samplers [Doucet et al., NeurIPS'22, χ:2208.07698]
  - SMC to sample posterior under diffusion priors [Cardoso et al., ICLR'24, χ:2308.07983] and others
  - High variance (but sometimes amortisable) [Akhound-Sadegh et al., ICML'24, χ:2402.06121] and others

Diffusion models without data?

Approaches to training a diffusion model without data:

- Optimise the reverse KL ($\leftrightarrow$ stochastic control methods)

  KL(denoising process$_\theta$ ‖ target distribution · noising process)

- PDE approaches
- Monte Carlo methods to estimate $\nabla \log(R * \mathcal{N}(0, V(t)))$
  Examples of estimates amenable to importance sampling:
  - DEM [Akhound-Sadegh et al., ICML'24, χ:2402.06121]:

$$\nabla \log(R * \mathcal{N}(0, V_t))(x_t) = \frac{\mathbb{E}_{x_0 \sim \mathcal{N}(x_t, V_t)}[\nabla R(x_0)]}{\mathbb{E}_{x_0 \sim \mathcal{N}(x_t, V_t)}[R(x_0)]}$$

    (estimated using diagonal joint proposal)

  - RDMC [Huang et al., ICLR'24, χ:2307.02037]:

$$\nabla \log(R * \mathcal{N}(0, V_t))(x_t) = \frac{\mathbb{E}_{x_0 \sim \mathcal{N}(x_t, V_t)}[R(x_0) \nabla \log \mathcal{N}(x_0; x_t, V_t)]}{\mathbb{E}_{x_0 \sim \mathcal{N}(x_t, V_t)}[R(x_0)]}$$

  - Others proposed for diffusion posterior sampling

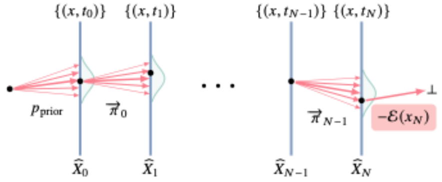
Diffusion models without data?

Approaches to training a diffusion model without data:

- Optimise the reverse KL ($\leftrightarrow$ stochastic control methods)

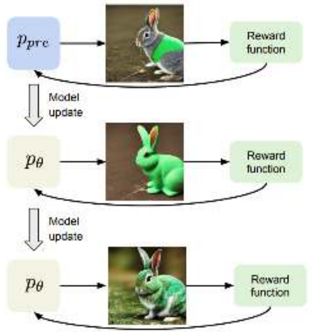
  KL(denoising process$_\theta$ $\|$ target distribution $\cdot$ noising process)

- PDE approaches
- Monte Carlo methods to estimate $\nabla \log(R * \mathcal{N}(0, V(t)))$
- Off-policy RL: diffusion samplers are diversity-seeking agents



[Berner at al., χ:2501.06148] $\uparrow$

[Fan et et al., 'DPOK...'] $\rightarrow$



Diffusion models without data?

Approaches to training a diffusion model without data:

- Optimise the reverse KL ($\leftrightarrow$ stochastic control methods)

  $\mathrm{KL}(\text{denoising process}_\theta \,\|\, \text{target distribution} \cdot \text{noising process})$

- PDE approaches
- Monte Carlo methods to estimate $\nabla \log(R * \mathcal{N}(0, V(t)))$
- Off-policy RL: diffusion samplers are diversity-seeking agents

Example of a **consistency objective**: For a denoising trajectory $\tau = \mathbf{x}_0 \to \mathbf{x}_{\Delta t} \to \cdots \to \mathbf{x}_1$, minimise a divergence such as

$$\mathcal{L}_{\mathrm{TB}}(\tau) = \left(\log \frac{Z_\theta \cdot \text{denoising process}_\theta(\tau)}{R(\mathbf{x}_1) \cdot \text{noising process}(\tau \mid \mathbf{x}_1)} \right)^2$$

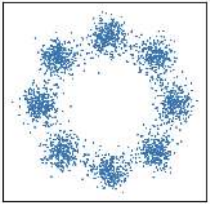
- Multi-objective problem; need to select τ
- ‘Off-policy’ = preconditioning
- But, on-policy, we recover the reverse KL gradient (this later)

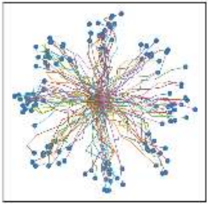
Time reversal for SDEs

We have two SDEs $\rightsquigarrow$ path space measures:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\,dt + \sigma(t)\,dW_t, \qquad X_0 \sim p_{\text{prior}},$$

$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\,dt + \sigma(t)\,d\overleftarrow{W}_t, \qquad X_1 \sim p_{\text{target}}$$





Time reversal for SDEs

We have two SDEs $\rightsquigarrow$ path space measures:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\,dt + \sigma(t)\,dW_t, \qquad X_0 \sim p_{\text{prior}},$$

$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\,dt + \sigma(t)\,\overleftarrow{dW}_t, \qquad X_1 \sim p_{\text{target}}$$

Radon-Nikodym derivative via Girsanov theorem:

$$\log \frac{d\overrightarrow{\mathbb{P}}}{d\overleftarrow{\mathbb{P}}} = \log \frac{p_{\text{prior}}(X_0)}{p_{\text{target}}(X_1)} + \int_0^1 \frac{\|\overleftarrow{\mu}(X_t, t)\|^2 - \|\overrightarrow{\mu}(X_t, t)\|^2}{2\sigma(t)^2}\,dt + \int_0^1 \frac{\overrightarrow{\mu}(X_t, t)}{\sigma(t)^2} \cdot dX_t - \int_0^1 \frac{\overleftarrow{\mu}(X_t, t)}{\sigma(t)^2} \cdot d\overleftarrow{X}_t$$

Time reversal for SDEs

We have two SDEs $\rightsquigarrow$ path space measures:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\,dt + \sigma(t)\,dW_t, \qquad X_0 \sim p_{\text{prior}},$$
$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\,dt + \sigma(t)\,\overleftarrow{dW}_t, \qquad X_1 \sim p_{\text{target}}$$

Radon-Nikodym derivative via Girsanov theorem:

$$\log \frac{d\overrightarrow{\mathbb{P}}}{d\overleftarrow{\mathbb{P}}} = \log \frac{p_{\text{prior}}(X_0)}{p_{\text{target}}(X_1)} + \int_0^1 \frac{\|\overleftarrow{\mu}(X_t, t)\|^2 - \|\overrightarrow{\mu}(X_t, t)\|^2}{2\sigma(t)^2}\,dt$$
$$+ \int_0^1 \frac{\overrightarrow{\mu}(X_t, t)}{\sigma(t)^2} \cdot dX_t - \int_0^1 \frac{\overleftarrow{\mu}(X_t, t)}{\sigma(t)^2} \cdot d\overleftarrow{X}_t$$

and the KL, giving a stochastic control cost with control $\overrightarrow{\mu}$:

$$\mathrm{KL}(\overrightarrow{\mathbb{P}} \,\|\, \overleftarrow{\mathbb{P}}) = \log Z + \mathbb{E}_{X \sim \overrightarrow{\mathbb{P}}}\Bigg[\log p_{\text{prior}}(X_0) + \mathcal{E}(X_T)$$
$$+ \int_0^1 \left(\frac{\|\overrightarrow{\mu}(X_t, t) - \overleftarrow{\mu}(X_t, t)\|^2}{2\sigma(t)^2} - \nabla \cdot \overleftarrow{\mu}(X_t, t) \right) dt\Bigg]$$

PDE perspective

The two SDEs define the same process with marginal densities p_t if and only if the following three are satsified:

- Boundary conditions: $p_0 = p_{\text{prior}}$ or $p_1 = p_{\text{target}}$
- Nelson's (1965) / Anderson's (1982) identity:

$$\overleftarrow{\mu}(x,t) = \overrightarrow{\mu}(x,t) - \sigma(t)^2 \nabla \log p_t(x)$$

- Fokker-Planck equation for either process:

$$\partial_t p_t = -\nabla \cdot (p_t \overrightarrow{\mu}) + \frac{\sigma^2}{2} \Delta p_t$$

PDE perspective

The two SDEs define the same process with marginal densities p_t if and only if the following three are satsified:

- Boundary conditions: $p_0 = p_{\text{prior}}$ or $p_1 = p_{\text{target}}$
- Nelson's (1965) / Anderson's (1982) identity:

$$\overleftarrow{\mu}(x,t) = \overrightarrow{\mu}(x,t) - \sigma(t)^2 \nabla \log p_t(x)$$

- Fokker-Planck equation for either process:

$$\partial_t p_t = -\nabla \cdot (p_t \overrightarrow{\mu}) + \frac{\sigma^2}{2} \Delta p_t$$

This leads to objectives that enforce the above conditions through appropriate parametrisations or losses (see [Máté & Fleuret, TMLR, χ:2301.07388], [Sun et al., χ:2407.07873], others)

Key references on the various approaches

- KL minimisation: [Zhang & Chen, ICLR'22, χ:2111.15141], [Vargas et al., ICLR'23, χ:2302.13834]
- Off-policy losses: [Nüsken & Richter, PDEA, χ:2005.05409], [Richter & Berner, ICLR'24, χ:2307.01198]
- Connections with SMC, control, etc.: [Vargas et al., ICLR'24, χ:2307.01050], [Chen et al., ICLR'25, χ:2412.07081], [Albergo & Vanden-Eijnden, ICML'25, χ:2410.02711], [Choi et al., χ:2510.11711]

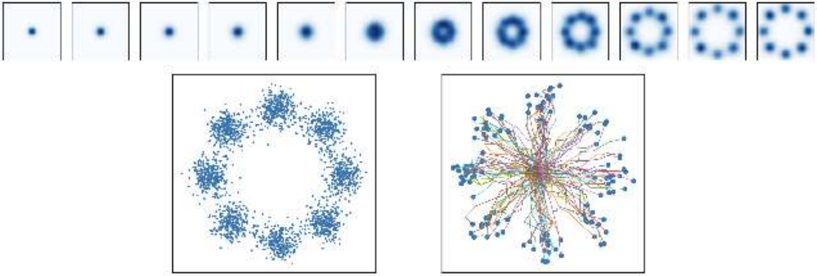
Key references on the various approaches

- KL minimisation: [Zhang & Chen, ICLR'22, χ:2111.15141], [Vargas et al., ICLR'23, χ:2302.13834]
- Off-policy losses: [Nüsken & Richter, PDEA, χ:2005.05409], [Richter & Berner, ICLR'24, χ:2307.01198]
  My work on this (shameless plug):
  - RL techniques: [Sendera et al., NeurIPS'24, χ:2402.05098], [Kim et al., ICLR'25, χ:2410.01432], [Gritsaev et al., χ:2506.01541], . . .
  - Unifying theory and continuous-time limit: [Lahlou et al., ICML'23, χ:2301.12594], [Berner et al., χ:2501.06148]
  - Inverse problems and scaling: [Venkatraman et al., NeurIPS'24, χ:2405.20971], [Venkatraman et al., ICML'25, χ:2502.06999]
- Connections with SMC, control, etc.: [Vargas et al., ICLR'24, χ:2307.01050], [Chen et al., ICLR'25, χ:2412.07081], [Albergo & Vanden-Eijnden, ICML'25, χ:2410.02711], [Choi et al., χ:2510.11711]

- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

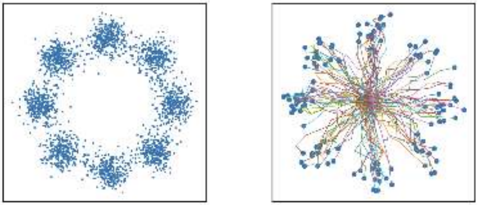
Again: Sampling with learned diffusions

Recall the problem: sampling a distribution p_{target} on $\mathbb{R}^d$ given its unnormalised density $\rho = \exp(-\mathcal{E}(\cdot))$



Again: Sampling with learned diffusions

Recall the problem: sampling a distribution p_{target} on $\mathbb{R}^d$ given its unnormalised density $\rho = \exp(-\mathcal{E}(\cdot))$



- Assume a pair of SDEs $\rightsquigarrow$ path space measures:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\,dt + \sigma(t)\,dW_t, \qquad X_0 \sim p_{\text{prior}},$$

$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\,dt + \sigma(t)\,d\overleftarrow{W}_t, \qquad X_1 \sim p_{\text{target}}$$

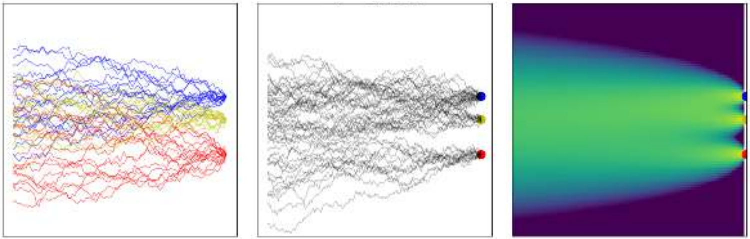
Again: Sampling with learned diffusions

Recall the problem: sampling a distribution p_{target} on $\mathbb{R}^d$ given its unnormalised density $\rho = \exp(-\mathcal{E}(\cdot))$

- Assume a pair of SDEs $\rightsquigarrow$ path space measures:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\,dt + \sigma(t)\,dW_t, \qquad X_0 \sim p_{\text{prior}},$$
$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\,dt + \sigma(t)\,\overleftarrow{dW}_t, \qquad X_1 \sim p_{\text{target}}$$

- Match the two processes (PINN/PDEs, KL, off-policy divergences)
- If they are equal, then $(\mathrm{ev}_1)_\# \overrightarrow{\mathbb{P}} = (\mathrm{ev}_1)_\# \overleftarrow{\mathbb{P}}$, so $X_1 \sim p_{\text{target}}$.



Again: Sampling with learned diffusions

Recall the problem: sampling a distribution p_{target} on $\mathbb{R}^d$ given its unnormalised density $\rho = \exp(-\mathcal{E}(\cdot))$

- Assume a pair of SDEs $\rightsquigarrow$ path space measures:

$$\begin{aligned} \overrightarrow{\mathbb{P}}: &\quad \mathrm{d}X_t = \overrightarrow{\mu}(X_t, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W_t, &\qquad X_0 \sim p_{\text{prior}}, \\ \overleftarrow{\mathbb{P}}: &\quad \mathrm{d}Y_t = \overleftarrow{\mu}(Y_t, t)\,\mathrm{d}t + \sigma(t)\,\overleftarrow{\mathrm{d}W}_t, &\qquad X_1 \sim p_{\text{target}} \end{aligned}$$

- Match the two processes (PINN/PDEs, KL, off-policy divergences)
- If they are equal, then $(\mathrm{ev}_1)_\# \overrightarrow{\mathbb{P}} = (\mathrm{ev}_1)_\# \overleftarrow{\mathbb{P}}$, so $X_1 \sim p_{\text{target}}$.

The discrete-time version of this: hierchical variational inference

Hierarchical variational inference

- Assume a Markov chain with states valued in $\mathbb{R}^d$:

$$X_0 \xrightarrow{\overrightarrow{p}} X_1 \xrightarrow{\overrightarrow{p}} X_2 \xrightarrow{\overrightarrow{p}} \dots \xrightarrow{\overrightarrow{p}} X_T, \quad X_0 \sim p_{\text{prior}}$$

  where the $\overrightarrow{p}$ are (densities of) Lebesgue-a.c. transition kernels
- For $\overrightarrow{p}$ to satisfy $X_T \sim p_{\text{target}}$, need

$$p_{\text{target}}(x_T) = \int p_{\text{prior}}(x_0) \prod_{t=0}^{T-1} \overrightarrow{p}(x_{t+1} \mid x_t) \, \mathrm{d}x_0 \, \mathrm{d}x_1 \dots \mathrm{d}x_{T-1}$$

Hierarchical variational inference

- Assume a Markov chain with states valued in $\mathbb{R}^d$:

$$X_0 \xrightarrow{\vec{p}} X_1 \xrightarrow{\vec{p}} X_2 \xrightarrow{\vec{p}} \dots \xrightarrow{\vec{p}} X_T, \quad X_0 \sim p_{\text{prior}}$$

  where the $\vec{p}$ are (densities of) Lebesgue-a.c. transition kernels
- For $\vec{p}$ to satisfy $X_T \sim p_{\text{target}}$, need

$$p_{\text{target}}(x_T) = \int p_{\text{prior}}(x_0) \prod_{t=0}^{T-1} \vec{p}(x_{t+1} \mid x_t)\, \mathrm{d}x_0\, \mathrm{d}x_1 \dots \mathrm{d}x_{T-1}$$

  - Introduce a variational distribution with reverse factorisation

$$Y_0 \xleftarrow{\overleftarrow{p}} Y_1 \xleftarrow{\overleftarrow{p}} Y_2 \xleftarrow{\overleftarrow{p}} \dots \xleftarrow{\overleftarrow{p}} Y_T, \quad Y_T \sim p_{\text{target}}$$

    where $\overleftarrow{p}$ are fixed kernels

Hierarchical variational inference

- Assume a Markov chain with states valued in $\mathbb{R}^d$:

$$X_0 \xrightarrow{\overrightarrow{p}} X_1 \xrightarrow{\overrightarrow{p}} X_2 \xrightarrow{\overrightarrow{p}} \dots \xrightarrow{\overrightarrow{p}} X_T, \quad X_0 \sim p_{\text{prior}}$$

where the $\overrightarrow{p}$ are (densities of) Lebesgue-a.c. transition kernels
- For $\overrightarrow{p}$ to satisfy $X_T \sim p_{\text{target}}$, need

$$p_{\text{target}}(x_T) = \int p_{\text{prior}}(x_0) \prod_{t=0}^{T-1} \overrightarrow{p}(x_{t+1} \mid x_t)\, \mathrm{d}x_0\, \mathrm{d}x_1 \dots \mathrm{d}x_{T-1}$$

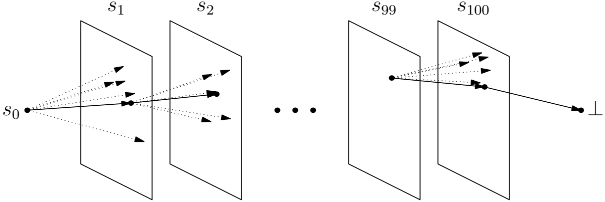
  - Introduce a variational distribution with reverse factorisation

$$Y_0 \xleftarrow{\overleftarrow{p}} Y_1 \xleftarrow{\overleftarrow{p}} Y_2 \xleftarrow{\overleftarrow{p}} \dots \xleftarrow{\overleftarrow{p}} Y_T, \quad Y_T \sim p_{\text{target}}$$

  where $\overleftarrow{p}$ are fixed kernels
  - HVI: Match $p_{\text{prior}} \otimes \overrightarrow{p} \otimes \cdots \otimes \overrightarrow{p}$ and $p_{\text{target}} \otimes \overleftarrow{p} \otimes \cdots \otimes \overleftarrow{p}$ by minimising the KL divergence
  - Data processing inequality: $0 \leq \mathrm{KL}(X_T \,\|\, Y_T) \leq \mathrm{KL}\left(p_{\text{prior}} \otimes \overrightarrow{p} \otimes \cdots \otimes \overrightarrow{p} \,\|\, p_{\text{target}} \otimes \overleftarrow{p} \otimes \cdots \otimes \overleftarrow{p}\right)$

Reinforcement learning setup

- Consider a **deterministic graded Markov decision process** $\approx$ directed graph with set of states $\mathcal{S} = \mathcal{S}_0 \sqcup \mathcal{S}_1 \sqcup \cdots \sqcup \mathcal{S}_T$, reward $r(s_t, s_{t+1})$ associated with transition from s_t to s_{t+1}
- A **policy** π is a collection of functions $\pi_{\text{prior}} \in \mathcal{P}(\mathcal{S}_0)$, $\pi_t : \mathcal{S}_t \to \mathcal{P}(\mathcal{S}_{t+1}))$ satisfying reachability constraints



Reinforcement learning setup

- Consider a **deterministic graded Markov decision process** $\approx$ directed graph with set of states $\mathcal{S} = \mathcal{S}_0 \sqcup \mathcal{S}_1 \sqcup \cdots \sqcup \mathcal{S}_T$, reward $r(s_t, s_{t+1})$ associated with transition from s_t to s_{t+1}
- A **policy** π is a collection of functions $\pi_{\text{prior}} \in \mathcal{P}(\mathcal{S}_0)$, $\pi_t : \mathcal{S}_t \to \mathcal{P}(\mathcal{S}_{t+1}))$ satisfying reachability constraints
- Goal: find a policy that maximises the expected reward

$$\mathcal{R}(\pi) = \mathbb{E}_{X_0, X_1, \dots, X_T \sim \pi_{\text{prior}} \otimes \pi_0 \otimes \cdots \otimes \pi_{T-1}} \left[\sum_{t=0}^{T-1} r(s_t, s_{t+1}) \right]$$

(Solution not always unique; deterministic maximiser exists.)

Reinforcement learning setup

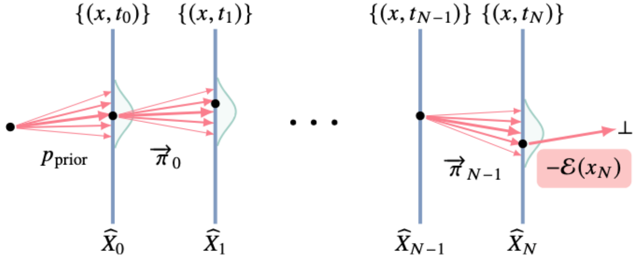
- Consider a **deterministic graded Markov decision process** $\approx$ directed graph with set of states $\mathcal{S} = \mathcal{S}_0 \sqcup \mathcal{S}_1 \sqcup \cdots \sqcup \mathcal{S}_T$, reward $r(s_t, s_{t+1})$ associated with transition from s_t to s_{t+1}
- A **policy** π is a collection of functions $\pi_{\text{prior}} \in \mathcal{P}(\mathcal{S}_0)$, $\pi_t : \mathcal{S}_t \to \mathcal{P}(\mathcal{S}_{t+1}))$ satisfying reachability constraints
- Goal: find a policy that maximises the expected reward

$$\mathcal{R}(\pi) = \mathbb{E}_{x_0, x_1, \ldots, x_T \sim \pi_{\text{prior}} \otimes \pi_0 \otimes \cdots \otimes \pi_{T-1}} \left[\sum_{t=0}^{T-1} r(s_t, s_{t+1}) \right]$$

(Solution not always unique; deterministic maximiser exists.)
- Entropy-regularised objective: $R(\pi) + \alpha \mathcal{H}[\pi]$
- Solution to maximum-entropy RL problem:

$$\pi^*(x_0, x_1, \ldots x_T) \propto \exp\left(\frac{1}{\alpha} \sum_{t=0}^{T-1} r(x_t, x_{t+1}) \right)$$

MDPs and policies associated with diffusion



The policies are given by neural networks predicting the parameters of transition kernels (e.g., Gaussian mean and variance) from (x_t, t)

- Note that the reverse of a process with Gaussian transitions is not generally Gaussian (but it is in the continuous-time limit)

HVI as entropy-regularised RL

Setting up HVI as a maximum-entropy RL problem:

- Recall:

$$X_0 \xrightarrow{\overrightarrow{p}} X_1 \xrightarrow{\overrightarrow{p}} X_2 \xrightarrow{\overrightarrow{p}} \dots \xrightarrow{\overrightarrow{p}} X_T, \quad X_0 \sim p_{\text{prior}}$$

$$Y_0 \xleftarrow{\overleftarrow{p}} Y_1 \xleftarrow{\overleftarrow{p}} Y_2 \xleftarrow{\overleftarrow{p}} \dots \xleftarrow{\overleftarrow{p}} Y_T, \quad Y_T \sim p_{\text{target}}$$

- X_i, Y_i take values in space $\mathcal{S}_i$

HVI as entropy-regularised RL

Setting up HVI as a maximum-entropy RL problem:

- Recall:

$$X_0 \xrightarrow{\vec{p}} X_1 \xrightarrow{\vec{p}} X_2 \xrightarrow{\vec{p}} \dots \xrightarrow{\vec{p}} X_T, \quad X_0 \sim p_{\text{prior}}$$

$$Y_0 \xleftarrow{\overleftarrow{p}} Y_1 \xleftarrow{\overleftarrow{p}} Y_2 \xleftarrow{\overleftarrow{p}} \dots \xleftarrow{\overleftarrow{p}} Y_T, \quad Y_T \sim p_{\text{target}}$$

- X_i, Y_i take values in space $\mathcal{S}_i$
- Set $p_{\text{init}} = p_{\text{prior}}$, $\alpha = 1$, reward

$$r(\overbrace{x_t}^{\in\mathcal{S}_t}, \overbrace{x_{t+1}}^{\in\mathcal{S}_{t+1}}) = \begin{cases} \log \overleftarrow{p}(x_t \mid x_{t+1}), & t < T-1, \\ \log \overleftarrow{p}(x_t \mid x_{t+1}) - \mathcal{E}(x_T), & t = T-1 \end{cases}$$

HVI as entropy-regularised RL

Setting up HVI as a maximum-entropy RL problem:

- Recall:

$$X_0 \xrightarrow{\overrightarrow{p}} X_1 \xrightarrow{\overrightarrow{p}} X_2 \xrightarrow{\overrightarrow{p}} \dots \xrightarrow{\overrightarrow{p}} X_T, \quad X_0 \sim p_{\text{prior}}$$

$$Y_0 \xleftarrow{\overleftarrow{p}} Y_1 \xleftarrow{\overleftarrow{p}} Y_2 \xleftarrow{\overleftarrow{p}} \dots \xleftarrow{\overleftarrow{p}} Y_T, \quad Y_T \sim p_{\text{target}}$$

- X_i, Y_i take values in space $\mathcal{S}_i$
- Set $p_{\text{init}} = p_{\text{prior}}$, $\alpha = 1$, reward

$$r(\overbrace{x_t}^{\in\mathcal{S}_t}, \overbrace{x_{t+1}}^{\in\mathcal{S}_{t+1}}) = \begin{cases} \log \overleftarrow{p}(x_t \mid x_{t+1}), & t < T-1, \\ \log \overleftarrow{p}(x_t \mid x_{t+1}) - \mathcal{E}(x_T), & t = T-1 \end{cases}$$

- Optimal policy $\pi^* \rightsquigarrow$ kernel $\overrightarrow{p}$ such that

$$p_{\text{prior}}(x_0) \prod_{t=0}^{T-1} \overrightarrow{p}(x_{t+1} \mid x_t) \propto \exp(-\mathcal{E}(x_T)) \prod_{t=0}^{T-1} \overleftarrow{p}(x_t \mid x_{t+1})$$

Note: no assumption that spaces $\mathcal{S}_t$ are all identical (more later)

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Learn **value functions** $V_t : \mathcal{S}_t \to \mathbb{R}$ to enforce **soft Bellman equation**:

$$V_t(x_t) = \overbrace{\log \int \exp}^{\text{max in unreg. RL!}} \left(r(x_t, x_{t+1}) + V_{t+1}(x_{t+1}) \right) \mathrm{d}x_{t+1}$$

with boundary condition $V_T(x_T) \equiv 0$

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Learn **value functions** $V_t : \mathcal{S}_t \to \mathbb{R}$ to enforce **soft Bellman equation**:

$$V_t(x_t) = \overbrace{\log \int \exp}^{\text{max in unreg. RL!}} \left(r(x_t, x_{t+1}) + V_{t+1}(x_{t+1})\right) \mathrm{d}x_{t+1}$$

    with boundary condition $V_T(x_T) \equiv 0$
  - Policy π given in terms of value function by

$$\pi_0(x_0) \propto \exp(V_0(x_0)), \quad \pi(x_{t+1} \mid x_t) \propto \exp(V^*_{t+1}(x_{t+1}) + r(x_t, x_{t+1}))$$

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Learn **value functions** $V_t : \mathcal{S}_t \to \mathbb{R}$ to enforce **soft Bellman equation**:

$$V_t(x_t) = \overbrace{\log \int \exp}^{\text{max in unreg. RL!}} \left(r(x_t, x_{t+1}) + V_{t+1}(x_{t+1})\right) \mathrm{d}x_{t+1}$$

    with boundary condition $V_T(x_T) \equiv 0$
  - Policy π given in terms of value function by

$$\pi_0(x_0) \propto \exp(V_0(x_0)), \quad \pi(x_{t+1} \mid x_t) \propto \exp(V^*_{t+1}(x_{t+1}) + r(x_t, x_{t+1}))$$

  - Algebraic manipulation recovers the nested VI [Zimmermann et al., NeurIPS'21, χ:2106.11302] / detailed balance constraint for the transition kernels:

$$\tilde{V}_t(x_t) + \log \overrightarrow{p}(x_{t+1} \mid x_t) = \tilde{V}_{t+1}(x_{t+1}) + \log \overleftarrow{p}(x_t \mid x_{t+1})$$

    where $\tilde{V}_t(x_t) = V_t(x_t)$ for $t < T$ and $\tilde{V}_T(x_T) = -\mathcal{E}(x_T)$

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Nested VI / detailed balance constraint:

$$\tilde{V}_t(x_t) + \log \overrightarrow{p}(x_{t+1} \mid x_t) = \tilde{V}_{t+1}(x_{t+1}) + \log \overleftarrow{p}(x_t \mid x_{t+1})$$

    where $\tilde{V}_T(x_T) = -\mathcal{E}(x_T)$
  - The $\tilde{V}_t$ are log-marginal densities up to a constant

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Nested VI / detailed balance constraint:

$$\tilde{V}_t(x_t) + \log \overrightarrow{p}(x_{t+1} \mid x_t) = \tilde{V}_{t+1}(x_{t+1}) + \log \overleftarrow{p}(x_t \mid x_{t+1})$$

    where $\tilde{V}_T(x_T) = -\mathcal{E}(x_T)$
  - The $\tilde{V}_t$ are log-marginal densities up to a constant
- Global objective (path consistency):
  - Iterating the soft Bellman equation gives a **path consistency** condition [Nachum et al., NIPS'17, χ:1702.08892]
  - In our setting, this recovers the following HVI constraint:

$$V_0(x_0) + \log \prod_{t=0}^{T-1} \overrightarrow{p}(x_{t+1} \mid x_t) = -\mathcal{E}(x_T) + \log \prod_{t=0}^{T-1} \overleftarrow{p}(x_t \mid x_{t+1})$$

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Nested VI / detailed balance constraint:

$$\tilde{V}_t(x_t) + \log \overrightarrow{p}(x_{t+1} \mid x_t) = \tilde{V}_{t+1}(x_{t+1}) + \log \overleftarrow{p}(x_t \mid x_{t+1})$$

    where $\tilde{V}_T(x_T) = -\mathcal{E}(x_T)$
  - The $\tilde{V}_t$ are log-marginal densities up to a constant
- Global objective (path consistency):
  - Iterating the soft Bellman equation gives a **path consistency** condition [Nachum et al., NIPS'17, χ:1702.08892]
  - In our setting, this recovers the following HVI constraint:

$$V_0(x_0) + \log \prod_{t=0}^{T-1} \overrightarrow{p}(x_{t+1} \mid x_t) = -\mathcal{E}(x_T) + \log \prod_{t=0}^{T-1} \overleftarrow{p}(x_t \mid x_{t+1})$$

  - Does not involve intermediate value functions!

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Nested VI / detailed balance constraint:
    $$\tilde{V}_t(x_t) + \log \overrightarrow{p}(x_{t+1} \mid x_t) = \tilde{V}_{t+1}(x_{t+1}) + \log \overleftarrow{p}(x_t \mid x_{t+1})$$
    where $\tilde{V}_T(x_T) = -\mathcal{E}(x_T)$
  - The $\tilde{V}_t$ are log-marginal densities up to a constant
- Global objective (path consistency):
  - HVI / 'trajectory balance' [M. et al., NeurIPS'22, 2201.13259]:
    $$V_0(x_0) + \log \prod_{t=0}^{T-1} \overrightarrow{p}(x_{t+1} \mid x_t) = -\mathcal{E}(x_T) + \log \prod_{t=0}^{T-1} \overleftarrow{p}(x_t \mid x_{t+1})$$
  - If $\overleftarrow{p}$ is fixed and transports the target to p_{prior}, then $V_0(x_0) = \log p_{\text{prior}}(x_0) + \text{const.}$

Local and global objectives for entropic RL

How to learn the optimal policy π^*? [M. et al., ICLR'23, χ:2210.00580], [Deleu et al., UAI'24, χ:2402.10309]

- Local objective (soft Q-learning):
  - Nested VI / detailed balance constraint:

$$\tilde{V}_t(x_t) + \log \overrightarrow{p}(x_{t+1} \mid x_t) = \tilde{V}_{t+1}(x_{t+1}) + \log \overleftarrow{p}(x_t \mid x_{t+1})$$

    where $\tilde{V}_T(x_T) = -\mathcal{E}(x_T)$
  - The $\tilde{V}_t$ are log-marginal densities up to a constant
- Global objective (path consistency):
  - HVI / 'trajectory balance' [M. et al., NeurIPS'22, 2201.13259]:

$$V_0(x_0) + \log \prod_{t=0}^{T-1} \overrightarrow{p}(x_{t+1} \mid x_t) = -\mathcal{E}(x_T) + \log \prod_{t=0}^{T-1} \overleftarrow{p}(x_t \mid x_{t+1})$$

  - If $\overleftarrow{p}$ is fixed and transports the target to p_{prior}, then $V_0(x_0) = \log p_{\text{prior}}(x_0) + \text{const}$.
- Both constraints can be turned into optimisation objectives
  - Minimising some divergence between the two sides over trajectories/transitions sampled from some behaviour policy

Off-policy hierarchical VI

- Recall the trajectory balance constraint:

$$V_0(x_0)+\log\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)=-\mathcal{E}(x_T)+\log\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})$$

Off-policy hierarchical VI

- Recall the trajectory balance constraint:

$$V_0(x_0)+\log\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)=-\mathcal{E}(x_T)+\log\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})$$

- Optimising the squared difference over a training distribution $p_{\rm beh}$ recovers VarGrad [Richter et al., NeurIPS'20, χ:2010.10436]:

$$\mathcal{L}_{\rm LV}^{p_{\rm beh}}(\overrightarrow{p},\overleftarrow{p})=\mathrm{Var}_{p_{\rm beh}}\left(\log\frac{p_{\rm prior}(x_0)\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)}{\exp(-\mathcal{E}(x_T))\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})}\right)$$

Off-policy hierarchical VI

- Recall the trajectory balance constraint:

$$V_0(x_0)+\log\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t) = -\mathcal{E}(x_T)+\log\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})$$

- Optimising the squared difference over a training distribution p_{beh} recovers VarGrad [Richter et al., NeurIPS'20, χ:2010.10436]:

$$\mathcal{L}_{\text{LV}}^{p_{\text{beh}}}(\overrightarrow{p},\overleftarrow{p}) = \text{Var}_{p_{\text{beh}}}\left(\log\frac{p_{\text{prior}}(x_0)\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)}{\exp(-\mathcal{E}(x_T))\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})}\right)$$

How to choose the behaviour policy p_{beh}?

Off-policy hierarchical VI

- Recall the trajectory balance constraint:

$$V_0(x_0)+\log\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)=-\mathcal{E}(x_T)+\log\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})$$

- Optimising the squared difference over a training distribution p_{beh} recovers VarGrad [Richter et al., NeurIPS'20, χ:2010.10436]:

$$\mathcal{L}_{\text{LV}}^{p_{\text{beh}}}(\overrightarrow{p},\overleftarrow{p})=\text{Var}_{p_{\text{beh}}}\left(\log\frac{p_{\text{prior}}(x_0)\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)}{\exp(-\mathcal{E}(x_T))\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})}\right)$$

How to choose the behaviour policy p_{beh}?

- **Fact** [rediscovered many times]: If $\pi_{\text{beh}}=\overrightarrow{p}$ ('on-policy'), then the gradient of $\mathcal{L}$ w.r.t. $\overrightarrow{p}$ is the reverse KL gradient:

$$\nabla_{\overrightarrow{p}}\mathcal{L}_{\text{LV}}^{p_{\text{beh}}}(\overrightarrow{p},\overleftarrow{p})\Big|_{p_{\text{beh}}=\overrightarrow{p}}=2\nabla_{\overrightarrow{p}}\text{KL}(p_{\text{prior}}\otimes\overrightarrow{p}\,\|\,p_{\text{target}}\otimes\overleftarrow{p})$$

Off-policy hierarchical VI

- Recall the trajectory balance constraint:

$$V_0(x_0)+\log\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)=-\mathcal{E}(x_T)+\log\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})$$

- Optimising the squared difference over a training distribution p_{beh} recovers VarGrad [Richter et al., NeurIPS'20, χ:2010.10436]:

$$\mathcal{L}_{\text{LV}}^{p_{\text{beh}}}(\overrightarrow{p},\overleftarrow{p})=\text{Var}_{p_{\text{beh}}}\left(\log\frac{p_{\text{prior}}(x_0)\prod_{t=0}^{T-1}\overrightarrow{p}(x_{t+1}\mid x_t)}{\exp(-\mathcal{E}(x_T))\prod_{t=0}^{T-1}\overleftarrow{p}(x_t\mid x_{t+1})}\right)$$

How to choose the behaviour policy p_{beh}?

- **Fact** [rediscovered many times]: If $\pi_{\text{beh}}=\overrightarrow{p}$ ('on-policy'), then the gradient of $\mathcal{L}$ w.r.t. $\overrightarrow{p}$ is the reverse KL gradient:

$$\nabla_{\overrightarrow{p}}\mathcal{L}_{\text{LV}}^{p_{\text{beh}}}(\overrightarrow{p},\overleftarrow{p})\Big|_{p_{\text{beh}}=\overrightarrow{p}}=2\nabla_{\overrightarrow{p}}\text{KL}(p_{\text{prior}}\otimes\overrightarrow{p}\,\|\,p_{\text{target}}\otimes\overleftarrow{p})$$

- But we can do better than reverse KL...

Exploratory policies for training diffusion samplers

Rather than sampling trajectories from the current distribution $\overrightarrow{p}$, we can sample from a more exploratory policy π_{beh}:

- Add extra variance to Gaussian kernels

Exploratory policies for training diffusion samplers

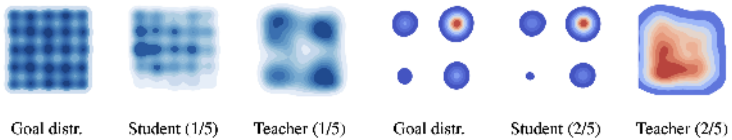
Rather than sampling trajectories from the current distribution $\overrightarrow{p}$, we can sample from a more exploratory policy π_{beh}:

- Add extra variance to Gaussian kernels
- Maintain a **replay buffer** of terminal states x_T [Sendera et al., NeurIPS'24, χ:2402.05098]
  - Update buffer using your favourite MCMC (e.g., Langevin)
  - Train $\overrightarrow{p}$ on trajectories formed by sampling from the replay buffer and then following $\overleftarrow{p}$

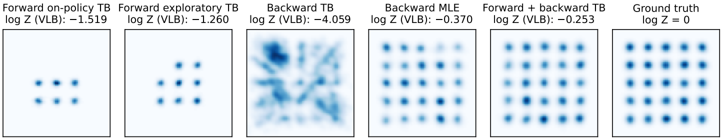
Exploratory policies for training diffusion samplers

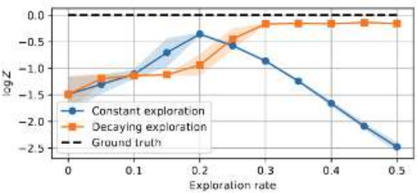
Rather than sampling trajectories from the current distribution $\overrightarrow{p}$, we can sample from a more exploratory policy π_{beh}:

- Add extra variance to Gaussian kernels
- Maintain a **replay buffer** of terminal states x_T [Sendera et al., NeurIPS'24, χ:2402.05098]
  - Update buffer using your favourite MCMC (e.g., Langevin)
  - Train $\overrightarrow{p}$ on trajectories formed by sampling from the replay buffer and then following $\overleftarrow{p}$
- Or even **learn** the exploratory policy to favour high-loss trajectories [Kim et al., ICLR'25, χ:2410.01432]



Exploration methods work





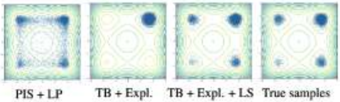


Figure 1: Two-dimensional projections of **Manywell** samples from models trained by different algorithms. Our proposed replay buffer with local search is capable of preventing mode collapse.

Connections with SMC

- The error in the detailed balance constraint

$$\tilde{V}_i(x_{t_i}) + \log \overrightarrow{p}(x_{t_{i+1}} \mid x_{t_i}) - \tilde{V}_{t+1}(x_{t_{i+1}}) - \log \overleftarrow{p}(x_{t_i} \mid x_{t_{i+1}})$$

is precisely the **log-importance weight** accumulated by AIS with intermediate targets $\propto \exp(\tilde{V}_i(x_{t_i}))$

Connections with SMC

- The error in the detailed balance constraint

$$\tilde{V}_i(x_{t_i}) + \log \overrightarrow{p}(x_{t_{i+1}} \mid x_{t_i}) - \tilde{V}_{t+1}(x_{t_{i+1}}) - \log \overleftarrow{p}(x_{t_i} \mid x_{t_{i+1}})$$

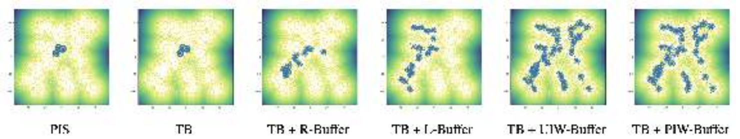
  is precisely the **log-importance weight** accumulated by AIS with intermediate targets $\propto \exp(\tilde{V}_i(x_{t_i}))$
- NVI/DB (resp. HVI/VarGrad) training minimise **variance of log-IWs** over steps (resp. over trajectories)
- Deep entropic RL is an twisted SMC algorithm (cf. [Chen et al., χ:2412.07081])

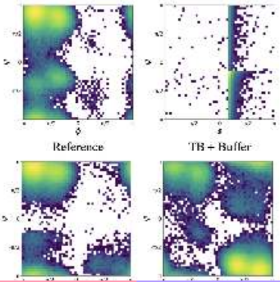
Connections with SMC

- Deep entropic RL is an twisted SMC algorithm (cf. [Chen et al., χ:2412.07081])

[Choi et al., χ:2510.11711]:

particle filters (SMC) + group importance sampling in replay buffers





Why drop the continuous-time assumption?

- In the continuous-time setting, we are matching two processes:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\, dt + \sigma(t)\, dW_t, \qquad X_0 \sim p_{\text{prior}},$$
$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\, dt + \sigma(t)\, \overleftarrow{dW}_t, \qquad X_1 \sim p_{\text{target}}$$

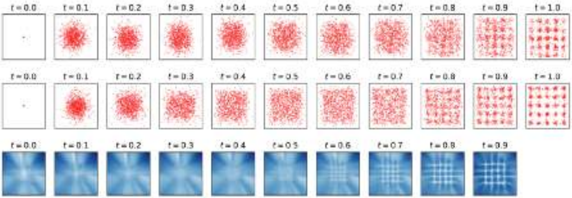
  - This constrains us to use the same diffusion coefficients $\sigma(t)$ in both processes (otherwise, RND not defined, cf. Girsanov)

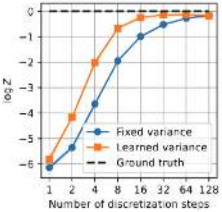
Why drop the continuous-time assumption?

- In the continuous-time setting, we are matching two processes:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\,dt + \sigma(t)\,dW_t, \qquad X_0 \sim p_{\text{prior}},$$
$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\,dt + \sigma(t)\,d\overleftarrow{W}_t, \qquad X_1 \sim p_{\text{target}}$$

  - This constrains us to use the same diffusion coefficients $\sigma(t)$ in both processes (otherwise, RND not defined, cf. Girsanov)
- In discrete time, we can relax this assumption and learn kernel variances freely, correcting for time discretisation error



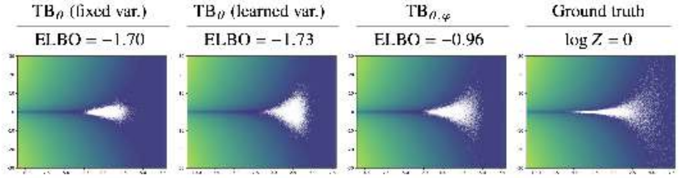


Why drop the continuous-time assumption?

- In the continuous-time setting, we are matching two processes:

$$\overrightarrow{\mathbb{P}}: \quad dX_t = \overrightarrow{\mu}(X_t, t)\,dt + \sigma(t)\,dW_t, \qquad X_0 \sim p_{\text{prior}},$$
$$\overleftarrow{\mathbb{P}}: \quad dY_t = \overleftarrow{\mu}(Y_t, t)\,dt + \sigma(t)\,d\overleftarrow{W}_t, \qquad X_1 \sim p_{\text{target}}$$

  - This constrains us to use the same diffusion coefficients $\sigma(t)$ in both processes (otherwise, RND not defined, cf. Girsanov)
- In discrete time, we can relax this assumption and learn kernel variances freely, correcting for time discretisation error
- We can also learn the reverse kernel $\overleftarrow{p}$, introducing a new degree of freedom (related to bridge sampling)



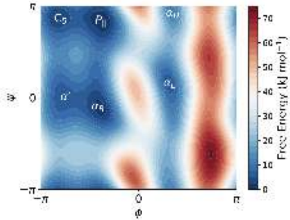
[Gritsaev et al., χ:2506.01541]

Why drop the continuous-time assumption?

- In the continuous-time setting, we are matching two processes:

$$\overrightarrow{\mathbb{P}}: \quad \mathrm{d}X_t = \overrightarrow{\mu}(X_t, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W_t, \qquad X_0 \sim p_{\text{prior}},$$
$$\overleftarrow{\mathbb{P}}: \quad \mathrm{d}Y_t = \overleftarrow{\mu}(Y_t, t)\,\mathrm{d}t + \sigma(t)\,\overleftarrow{\mathrm{d}W}_t, \qquad X_1 \sim p_{\text{target}}$$

  - This constrains us to use the same diffusion coefficients $\sigma(t)$ in both processes (otherwise, RND not defined, cf. Girsanov)
- In discrete time, we can relax this assumption and learn kernel variances freely, correcting for time discretisation error
- We can also learn the reverse kernel $\overleftarrow{p}$, introducing a new degree of freedom (related to bridge sampling)
- Easy generalisation to non-Gaussian kernels, kernels on manifolds, etc. [Phillips et al., χ:2408.15905], mixture-of-von-Mises kernel on torus



- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

VI with Euler-Maruyama kernels and consistency

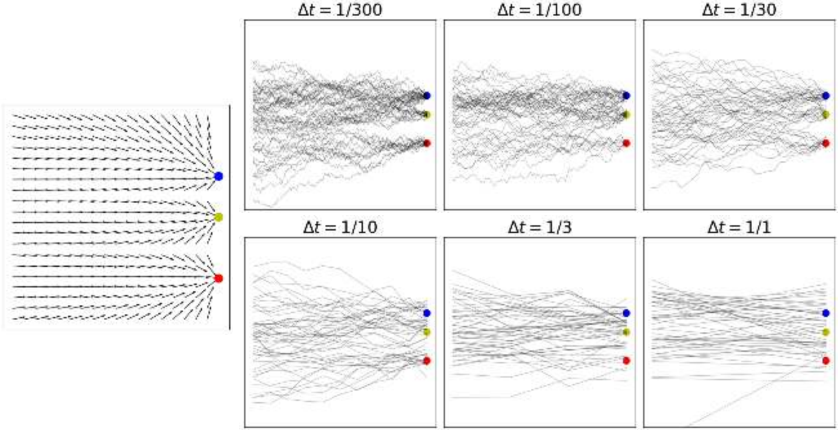
If we **do** assume underlying SDEs, how are HVI/RL approaches related to the continuous-time setting? [Berner et al., χ:2501.06148]

$$\overrightarrow{\mathbb{P}}: \quad \mathrm{d}X_t = \overrightarrow{\mu}(X_t, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W_t, \quad X_0 \sim p_{\text{prior}}$$

VI with Euler-Maruyama kernels and consistency

If we **do** assume underlying SDEs, how are HVI/RL approaches related to the continuous-time setting? [Berner et al., χ:2501.06148]

$$\overrightarrow{\mathbb{P}}: \quad \mathrm{d}X_t = \overrightarrow{\mu}(X_t, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W_t, \quad X_0 \sim p_{\text{prior}}$$



VI with Euler-Maruyama kernels and consistency

If we **do** assume underlying SDEs, how are HVI/RL approaches related to the continuous-time setting? [Berner et al., χ:2501.06148]

$$\overrightarrow{\mathbb{P}}: \quad \mathrm{d}X_t = \overrightarrow{\mu}(X_t, t)\,\mathrm{d}t + \sigma(t)\,\mathrm{d}W_t, \quad X_0 \sim p_{\text{prior}}$$

SDE $\rightsquigarrow$ Euler-Maruyama discretisation as a policy:

- Given a time discretisation $0 < t_0 < t_1 < \cdots < t_T = 1$ with $\Delta t_i := t_{i+1} - t_i$, we get a Gaussian Markov kernel by

  $$X_{t_{i+1}} = X_{t_i} + \overrightarrow{\mu}(X_{t_i}, t_i)\Delta t_i + \sigma(t_i)\Delta W_i, \quad \Delta W_i \sim \mathcal{N}(0, \Delta t_i)$$

  defining a discrete-time process $\overrightarrow{p}$
- Similar possible for reverse-time SDE

VI with Euler-Maruyama kernels and consistency

SDE $\rightsquigarrow$ Euler-Maruyama discretisation as a policy:

- Given a time discretisation $0 < t_0 < t_1 < \cdots < t_T = 1$ with $\Delta t_i := t_{i+1} - t_i$, we get a Gaussian Markov kernel by

$$X_{t_{i+1}} = X_{t_i} + \overrightarrow{\mu}(X_{t_i}, t_i)\Delta t_i + \sigma(t_i)\Delta W_i, \quad \Delta W_i \sim \mathcal{N}(0, \Delta t_i)$$

  defining a discrete-time process $\overrightarrow{p}$
- Similar possible for reverse-time SDE

What happens as $\max_i \Delta t_i \to 0$?

VI with Euler-Maruyama kernels and consistency

What happens as $\max_i \Delta t_i \to 0$? Under mild assumptions:

- **Theorem 1:** Global objectives (VarGrad) are consistent: $\lim_{\max_i \Delta t_i \to 0} \mathcal{L}_{\mathrm{LV}}^{p_{beh}}(\overrightarrow{p}, \overleftarrow{p}) = \mathcal{L}_{\mathrm{LV}}^{\mathbb{P}_{beh}}(\overrightarrow{\mathbb{P}}, \overleftarrow{\mathbb{P}})$ almost surely

VI with Euler-Maruyama kernels and consistency

What happens as $\max_i \Delta t_i \to 0$? Under mild assumptions:

- **Theorem 1:** Global objectives (VarGrad) are consistent: $\lim_{\max_i \Delta t_i \to 0} \mathcal{L}_{\mathrm{LV}}^{p_{beh}}(\overrightarrow{p}, \overleftarrow{p}) = \mathcal{L}_{\mathrm{LV}}^{\mathbb{P}_{beh}}(\overrightarrow{\mathbb{P}}, \overleftarrow{\mathbb{P}})$ almost surely
- **Theorem 2:** Local constraints (soft Q-learning) approach PDEs. Considering the detailed balance discrepancy

$$\tilde{V}_i(x_{t_i}) + \log \overrightarrow{p}(x_{t_{i+1}} \mid x_{t_i}) - \tilde{V}_{t+1}(x_{t_{i+1}}) - \log \overleftarrow{p}(x_{t_i} \mid x_{t_{i+1}}),$$

  - Vanishing of the $O(\sqrt{\Delta t_i}) \to$ Nelson's identity:

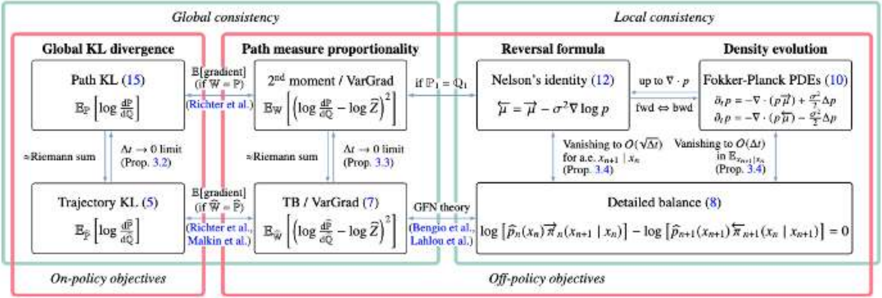
$$\overrightarrow{\mu}(x_{t_i}, t_i) = \overleftarrow{\mu}(x_{t_i}, t_i) + \sigma(t_i)^2 \nabla V_i(x_{t_i})$$

  - Vanishing of the expected $O(\Delta t_i) \to$ Fokker-Planck:

$$\partial_t p_t = -\nabla \cdot (\overrightarrow{\mu}(x_t, t) p_t) + \frac{\sigma(t)^2}{2} \nabla \cdot \nabla p_t$$

    where $p_{t_i}(x) = \exp(V_i(x))$.
  - The two jointly imply the forward and reverse SDEs define the same process and have marginal densities p_t

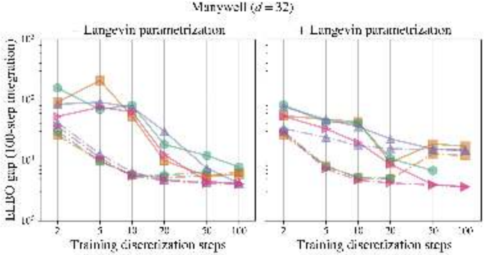
VI with Euler-Maruyama kernels and consistency

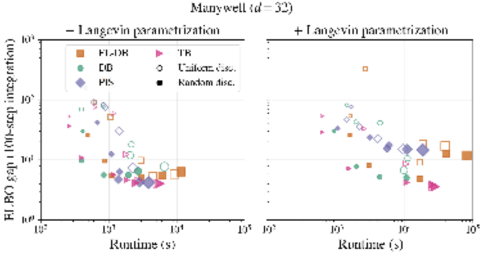


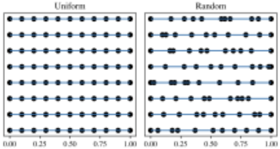
[Berner et al., χ:2501.06148]

Implications for training with variable time steps

We can train models using HVI/RL losses with very few time steps, then sample by simulating SDEs with much finer discretisation:







Interestingly, the coarse discretisation needs to be nonuniform.

- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

Amortising intractable posteriors under diffusion priors

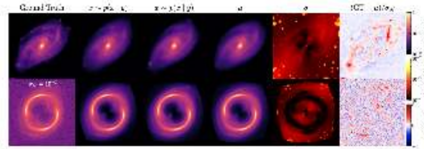
What about sampling x_T from $p(x_T \mid y) \propto p(x_T)p(y \mid x_T)$, where $p(x_T)$ is a pretrained **diffusion prior** and $p(y \mid x_T)$ is a likelihood?

- Intractable in general; MC and SMC-based methods exist
- Extracting information from pretrained foundation models for images, text, proteins, etc. is important in generative AI

Amortising intractable posteriors under diffusion priors

What about sampling x_T from $p(x_T \mid y) \propto p(x_T)p(y \mid x_T)$, where $p(x_T)$ is a pretrained **diffusion prior** and $p(y \mid x_T)$ is a likelihood?

- Intractable in general; MC and SMC-based methods exist
- Extracting information from pretrained foundation models for images, text, proteins, etc. is important in generative AI



a lion reading the newspaper*

a steam engine train, high resolution*

Amortising intractable posteriors under diffusion priors

What about sampling x_T from $p(x_T \mid y) \propto p(x_T)p(y \mid x_T)$, where $p(x_T)$ is a pretrained **diffusion prior** and $p(y \mid x_T)$ is a likelihood?

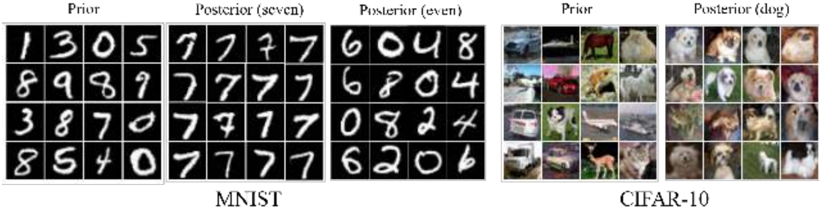
- Intractable in general; MC and SMC-based methods exist
- Extracting information from pretrained foundation models for images, text, proteins, etc. is important in generative AI

By renormalising the base measure from Lebesgue to one defined by the prior diffusion model, convert this into an entropic RL problem as above

- 'Relative' VarGrad and other objectives [Venkatraman et al., NeurIPS'24, χ:2405.20971]
- Apply the same methods to **fine-tune** the prior diffusion model into a posterior model

Amortising intractable posteriors under diffusion priors

Class-conditional image models from unconditional priors

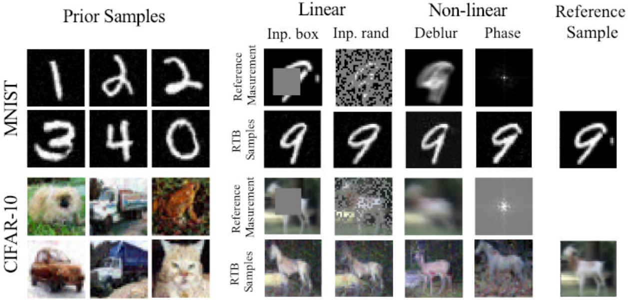


[Venkatraman et al., NeurIPS'24, χ:2405.20971]

- Unconditional diffusion model $+$ classifier $\rightsquigarrow$ class-conditional model
- Classifier guidance approximations and RL baselines are biased

Amortising intractable posteriors under diffusion priors

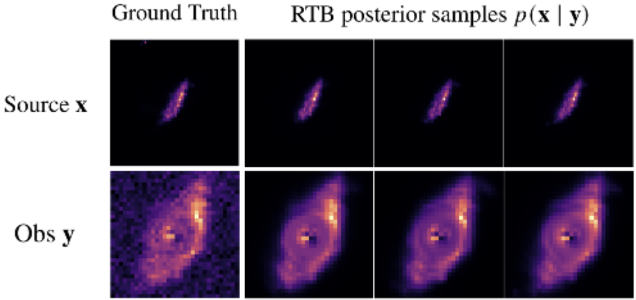
(Non-!)linear inverse problems (with applications in inverse imaging)



[Venkatraman et al., NeurIPS'24, χ:2405.20971]

Amortising intractable posteriors under diffusion priors

(Non-!)linear inverse problems (with applications in inverse imaging)



[Venkatraman et al., NeurIPS'24, χ:2405.20971]

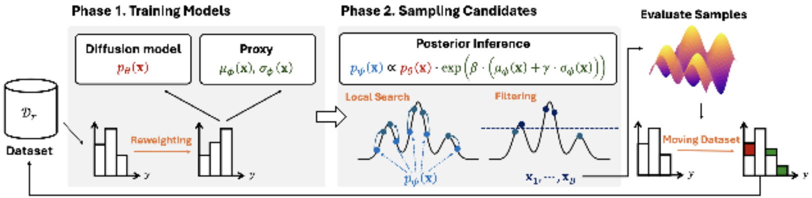
Amortising intractable posteriors under diffusion priors

Table 1: Sources of diffusion priors and constraints.

Domain	Prior $p(\mathbf{x})$	Constraint $r(\mathbf{x})$	Posterior
Conditional image generation (§4.1)	Image diffusion model $p(\mathbf{x})$	Classifier likelihood $p(c \mid \mathbf{x})$	Class-conditional distribution $p(\mathbf{x} \mid c)$
Text-to-image generation (§4.2)	Text-to-image foundation model	RLHF reward model	Aligned text-to-image model
Language infilling (§4.3)	Discrete diffusion model	Autoregressive completion likelihood	Infilling distribution
Offline RL policy extraction (§4.4)	Diffusion model as behavior policy	Boltzmann dist. of Q-function	Optimal KL-constrained policy

Other applications:

- Discrete-space diffusion (text)
- Offline RL policy extraction
- Black-box Bayesian optimisation [Yun et al., χ:2502.16824]



Inference in latent spaces of generative models

'Outsourced' diffusion sampling: sample posteriors in latent spaces of GANs, VAEs, etc., given a constraint on the output space

Table 2. The priors and constraints studied in §5. Outsourced diffusion sampling works in noise spaces of a wide range of generative models and is agnostic to their specific form.

Task	Constraint	Prior	Prior type	d_{noise}	d_{data}
CIFAR-10 classifer guidance	CIFAR-10 classifer	SN-GAN I-CFM	GAN CNF	128 $3 \times 32 \times 32$	$3 \times 32 \times 32$ $3 \times 32 \times 32$
FFHQ text conditioning	ImageReward	StyleGAN3 NVAE	GAN Hierarchical VAE	512 $4 \times 20 \times 8 \times 8$	$3 \times 256 \times 256$ $3 \times 256 \times 256$
Text-to-Image model RLHF	ImageReward	Stable Diffusion 3	Latent-CNF	$16 \times 64 \times 64$	$3 \times 512 \times 512$
Protein structure	Structure Diversity	FoldFlow 2	Riemannian CNF	7×64	7×64

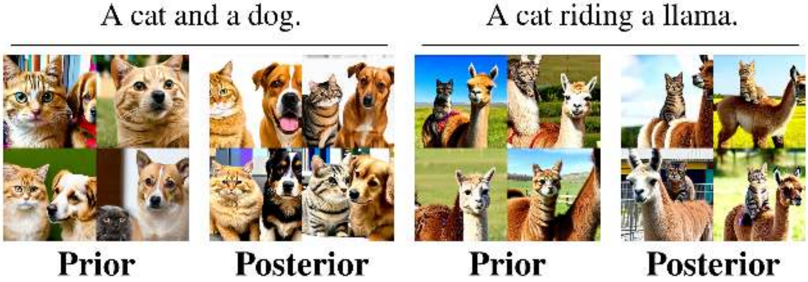
[Venkatraman et al., ICML'25, χ:2502.06999]

Inference in latent spaces of generative models

'Outsourced' diffusion sampling: sample posteriors in latent spaces of GANs, VAEs, etc., given a constraint on the output space

Table 2. The priors and constraints studied in §5. Outsourced diffusion sampling works in noise spaces of a wide range of generative models and is agnostic to their specific form.

Task	Constraint	Prior	Prior type	d_{noise}	d_{data}
CIFAR-10 classifer guidance	CIFAR-10 classifer	SN-GAN	GAN	128	$3 \times 32 \times 32$
		I-CFM	CNF	$3 \times 32 \times 32$	$3 \times 32 \times 32$
FFHQ text conditioning	ImageReward	StyleGAN3	GAN	512	$3 \times 256 \times 256$
		NVAE	Hierarchical VAE	$4 \times 20 \times 8 \times 8$	$3 \times 256 \times 256$
Text-to-Image model RLHF	ImageReward	Stable Diffusion 3	Latent-CNF	$16 \times 64 \times 64$	$3 \times 512 \times 512$
Protein structure	Structure Diversity	FoldFlow 2	Riemannian CNF	7×64	7×64



[Venkatraman et al., ICML'25, χ:2502.06999]

Schrödinger bridge problem

- The SB problem (for processes on $[0,1]$ taking values in $\mathbb{R}^d$):

$$\mathbb{P}_t^* = \arg\min_{\mathbb{P}_t} \left\{ \mathrm{KL}\left(\mathbb{P}_t \,\|\, \mathbb{Q}_t\right) : (\pi_0)_\# \mathbb{P}_t = p_0, (\pi_1)_\# \mathbb{P}_t = p_1 \right\}$$

  where $\mathbb{Q}_t$ is a reference process and p_0, p_1 are given
- If $\mathbb{Q}_t$ is given by a SDE

$$\mathrm{d}X_t = F_{\mathrm{ref}}(X_t, t)\,\mathrm{d}t + \sigma_t\,\mathrm{d}W_t, \quad X_0 \sim q_0$$

  then so (under regularity conditions) is the solution $\mathbb{P}_t^*$

Schrödinger bridge problem

- The SB problem (for processes on $[0,1]$ taking values in $\mathbb{R}^d$):

$$\mathbb{P}_t^* = \arg\min_{\mathbb{P}_t} \left\{ \mathrm{KL}\left(\mathbb{P}_t \,\|\, \mathbb{Q}_t\right) : (\pi_0)_\# \mathbb{P}_t = p_0, (\pi_1)_\# \mathbb{P}_t = p_1 \right\}$$

  where $\mathbb{Q}_t$ is a reference process and p_0, p_1 are given
- If $\mathbb{Q}_t$ is given by a SDE

$$\mathrm{d}X_t = F_{\mathrm{ref}}(X_t, t)\,\mathrm{d}t + \sigma_t\,\mathrm{d}W_t, \quad X_0 \sim q_0$$

  then so (under regularity conditions) is the solution $\mathbb{P}_t^*$
- For $\mathbb{P}_t : \mathrm{d}X_t = F(X_t, t)\,\mathrm{d}t + \sigma_t\,\mathrm{d}W_t, X_0 \sim p_0$, KL is a control cost:

$$\mathrm{KL}(\mathbb{P}_t \,\|\, \mathbb{Q}_t) = \mathrm{KL}(p_0 \,\|\, q_0) + \mathbb{E}_{X_t \sim \mathbb{P}_t} \int_0^1 \frac{\|F_{\mathrm{ref}}(X_t, t) - F(X_t, t)\|^2}{2\sigma_t^2}\,\mathrm{d}t,$$

  showing that $\sigma_t \to 0$ gives dynamic optimal transport
  - Marginally entropic OT between p_0, p_1 with entropy coefficient $2\sigma_t^2$)

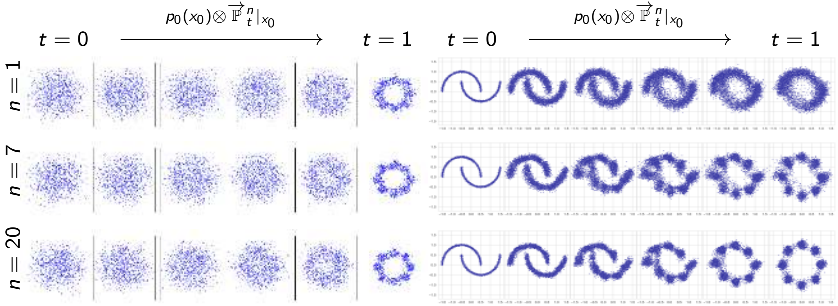
Iterative proportional fitting

IPF [Sinkhorn, 1964] is a recursion initialised at $\overrightarrow{\mathbb{P}}_t^{0} = \mathbb{Q}_t$:

$$\overleftarrow{\mathbb{P}}_t^{n+1} = \arg\min_{\mathbb{P}_t} \left\{ \mathrm{KL}\left(\mathbb{P}_t \,\|\, \overrightarrow{\mathbb{P}}_t^{n}\right) \text{ s.t. } (\pi_0)_{\#}\mathbb{P}_t = p_0 \right\},$$

$$\overrightarrow{\mathbb{P}}_t^{n+1} = \arg\min_{\mathbb{P}_t} \left\{ \mathrm{KL}\left(\mathbb{P}_t \,\|\, \overleftarrow{\mathbb{P}}_t^{n+1}\right) \text{ s.t. } (\pi_1)_{\#}\mathbb{P}_t = p_1 \right\}$$

where each step is a *half-bridge* problem (*e.g.*, $\overleftarrow{\mathbb{P}}_t^{n+1} = p_0 \otimes \overrightarrow{\mathbb{P}}_t^{n}|_{x_0}$)



The processes $\overrightarrow{\mathbb{P}}_t$ and $\overleftarrow{\mathbb{P}}_t$ converge in KL to the SB solution $\mathbb{P}_t^*$

Schrödinger bridge with diffusion sampling objectives

Existing IPF implementations assume samples from p_0, p_1 are given

- If p_1 is given by samples, training $\overrightarrow{\mathbb{P}}_t$ is maximum-likelihood training (as in diffusion)
- If p_0 is given by samples, training $\overleftarrow{\mathbb{P}}_t$ is also maximum-likelihood training (trivial in diffusion)
    - Diffusion training (with noising process converging to p_0) is a case of IPF that converges in one step

Schrödinger bridge with diffusion sampling objectives

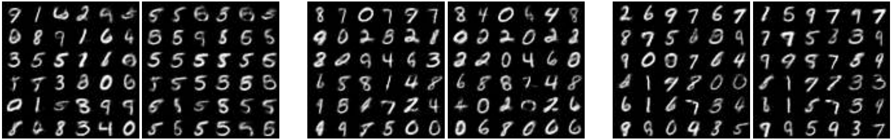
Existing IPF implementations assume samples from p_0, p_1 are given

- If p_1 is given by samples, training $\overrightarrow{\mathbb{P}}_t$ is maximum-likelihood training (as in diffusion)
- If p_0 is given by samples, training $\overrightarrow{\mathbb{P}}_t$ is also maximum-likelihood training (trivial in diffusion)
  - Diffusion training (with noising process converging to p_0) is a case of IPF that converges in one step
- If one of both of the distributions is given by an unnormalised density, we can use generalisations of the RL/VI objectives above (and appropriate off-policy training)

Outsourced Schrödinger bridge

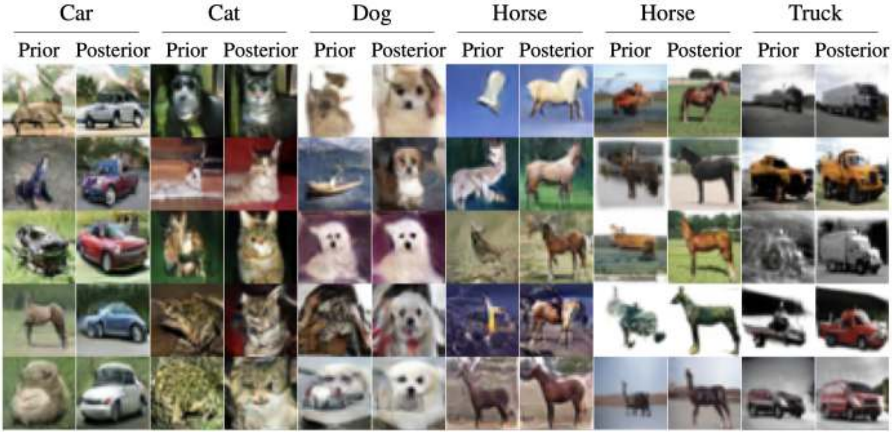
Translation $p_{\text{prior}} \leftrightarrow p_{\text{prior}} \cdot p(\text{class} \mid \cdot)$ in the latent space of a generative model

Prior **Fives** **Prior** **Even** **Prior** **Odd**



Outsourced Schrödinger bridge

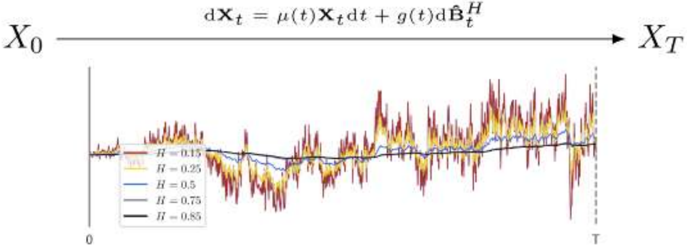
Translation $p_{\text{prior}} \leftrightarrow p_{\text{prior}} \cdot p(\text{class} \mid \cdot)$ in the latent space of a generative model



- Diffusion models review
- Survey of sampling with learned diffusions
  - Continuous-time case: Time reversal for SDEs
- Two views on stochastic measure transport in discrete time
  - Hierarchical variational inference
  - Deep entropy-regularised reinforcement learning
  - Limiting properties
- Some large-scale applications
  - Posteriors under diffusion and other generative model priors
- Schrödinger bridge generalisation
- Conclusion and outlook

Open directions in modelling

- SMC as an RL exploration strategy; diffusion samplers as adaptive importance samplers [with S. Choi, V. Elvira, . . .]
- Non-Markovian generation: Friction, momentum, persistent latent state [with R. Rajpal, B. Leimkuhler]
- Discrete-time optimal approximation with nondiagonal diffusion [with T. Gritsaev, D. Vetrov, . . .]
- Samplers and bridges in discrete space [with A. Carter, K. Tamogashev, . . .]



[Nobis et al., 'Generative fractional diffusion models', 2024]

Conclusion

- SDE generative processes as distribution approximators in inference/sampling tasks using RL and control methods
  - Discrete-time formulation allows for flexible models and training schemes
  - Connections with SMC, optimal transport, Schrödinger bridges
- Many open directions in modelling, algorithms, and applications
  - And, of course, theory: sample complexity bounds, discretisation error, ...

Conclusion

- SDE generative processes as distribution approximators in inference/sampling tasks using RL and control methods
  - Discrete-time formulation allows for flexible models and training schemes
  - Connections with SMC, optimal transport, Schrödinger bridges
- Many open directions in modelling, algorithms, and applications
  - And, of course, theory: sample complexity bounds, discretisation error, ...

Thank you for your attention.
More: `malkin1729.github.io`
[Always looking for new applications, collaborations, ...]

Temporary page!

LaTeX was unable to guess the total number of pages correctly. A
there was some unprocessed data that should have been added
the final page this extra page has been added to receive it.
If you rerun the document (without altering it) this surplus page
will go away, because LaTeX now knows how many pages to expe
for this document.