iterated Denoising Energy Matching

for sampling from Boltzmann densities

Tara Akhound-Sadegh et al

paper presentation - arXiv:2402.06121

Chivintar Amenty, June 2025

Introduction

$$\mu_{\text{target}}(x) = \frac{\exp\left(-\mathcal{E}(x)\right)}{\mathcal{Z}}, \ \mathcal{Z} = \int_{\mathbb{R}^d} \exp\left(-\mathcal{E}(x)\right) dx.$$

- Boltzmann Probability Density
- Partition function Z: intractable
- Goal: sample from distribution μ only having access to Boltzmann energy

Motivation

$$\mu_{\text{target}}(x) = \frac{\exp\left(-\mathcal{E}(x)\right)}{\mathcal{Z}}, \ \mathcal{Z} = \int_{\mathbb{R}^d} \exp\left(-\mathcal{E}(x)\right) dx.$$

- Statistical Physics
- Molecular Dynamics
- Protein Modeling
- Material Science
- Bayesian Inference in Astrophysics, Quantum Chromo-Dynamics and *many* more . . .

Related Work

$$\mu_{\text{target}}(x) = \frac{\exp\left(-\mathcal{E}(x)\right)}{\mathcal{Z}}, \ \mathcal{Z} = \int_{\mathbb{R}^d} \exp\left(-\mathcal{E}(x)\right) dx.$$

- MC techniques: AIS, SMC
  - Computationally expensive
  - Slow convergence in HD spaces
- Simulation techniques
  - Scalability issues
- Diffusion Models
  - Need training data

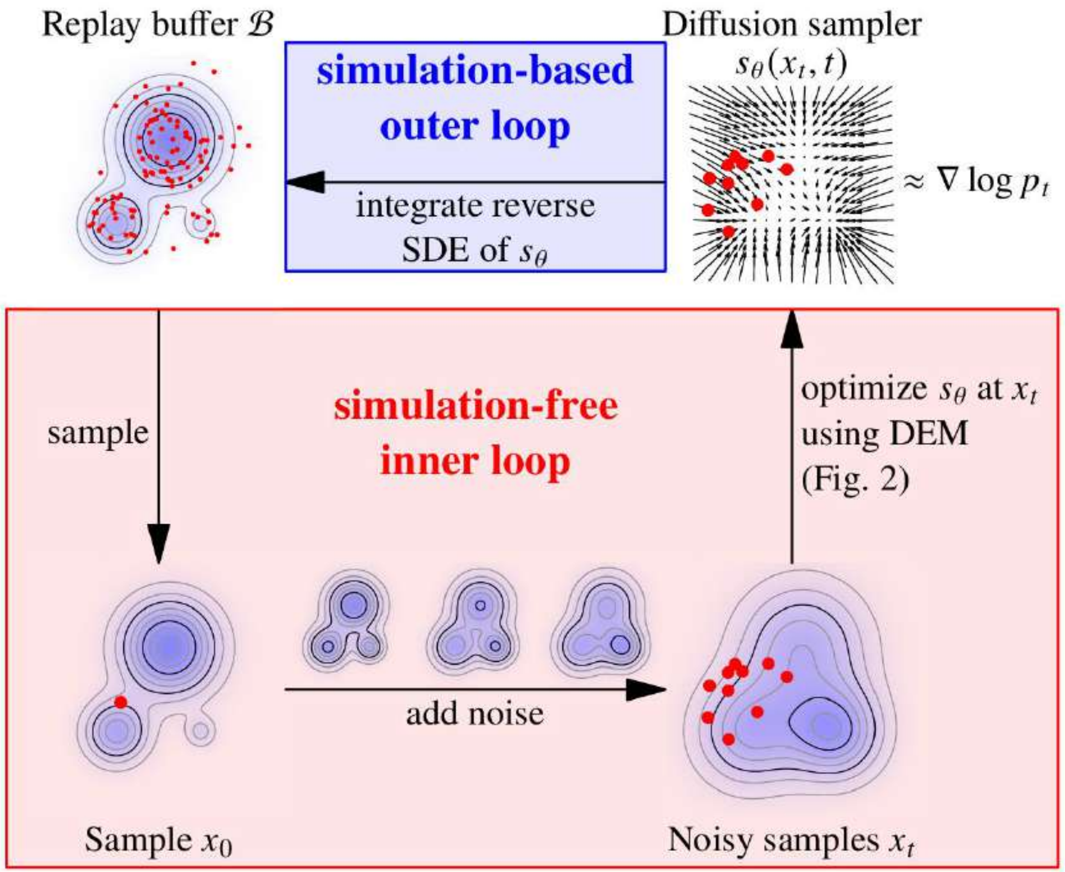
iDEM

iDEM introduction

- Neural sampler
- Diffusion-style
- Simulation-free (in inner loop)
- Computationally tractable
- Stochastic regression objective

- Diffusion sampled data

- Good coverage of all modes
- Imbues symmetries (SE(3) group)



iDEM introduction

Bi-Level algorithm

Inner loop

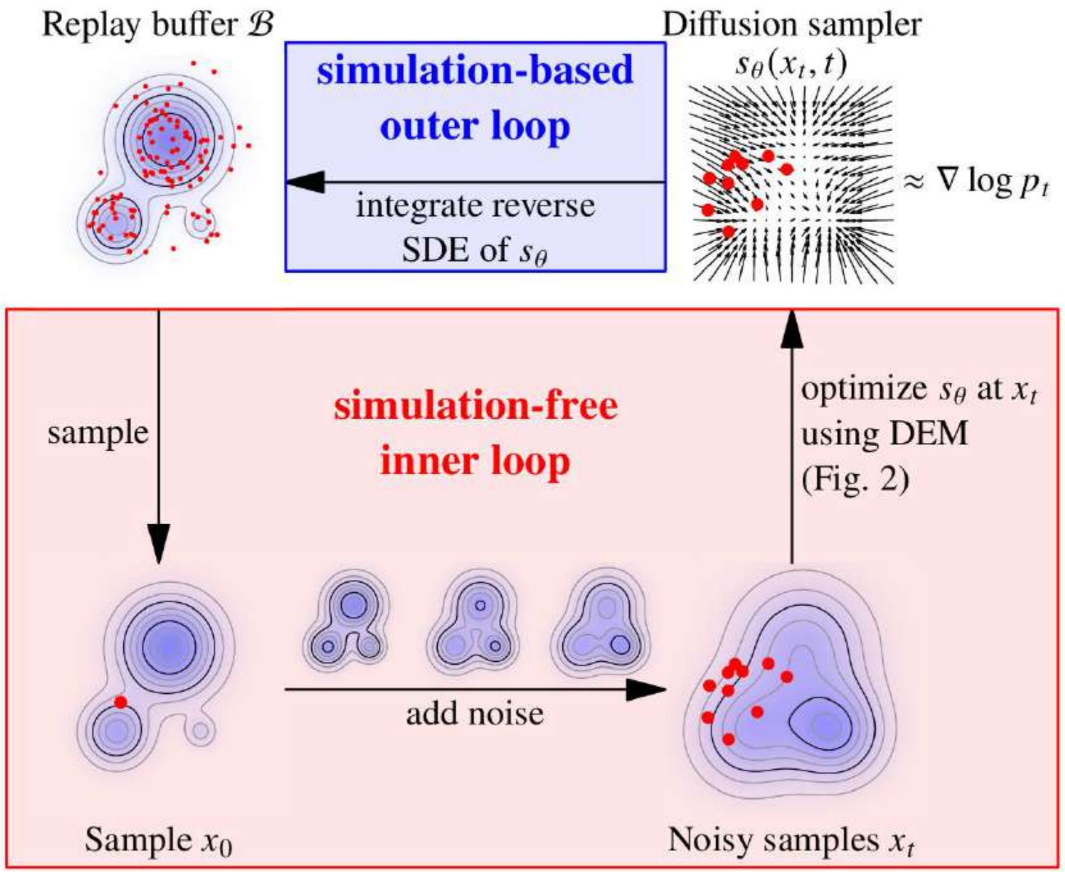
- Optimizes score function

  1. How, when we do not have samples from the distribution?

Outer loop

- Reverse SDE of score function
- Actual samples are generated

  2. Where to get good samples?

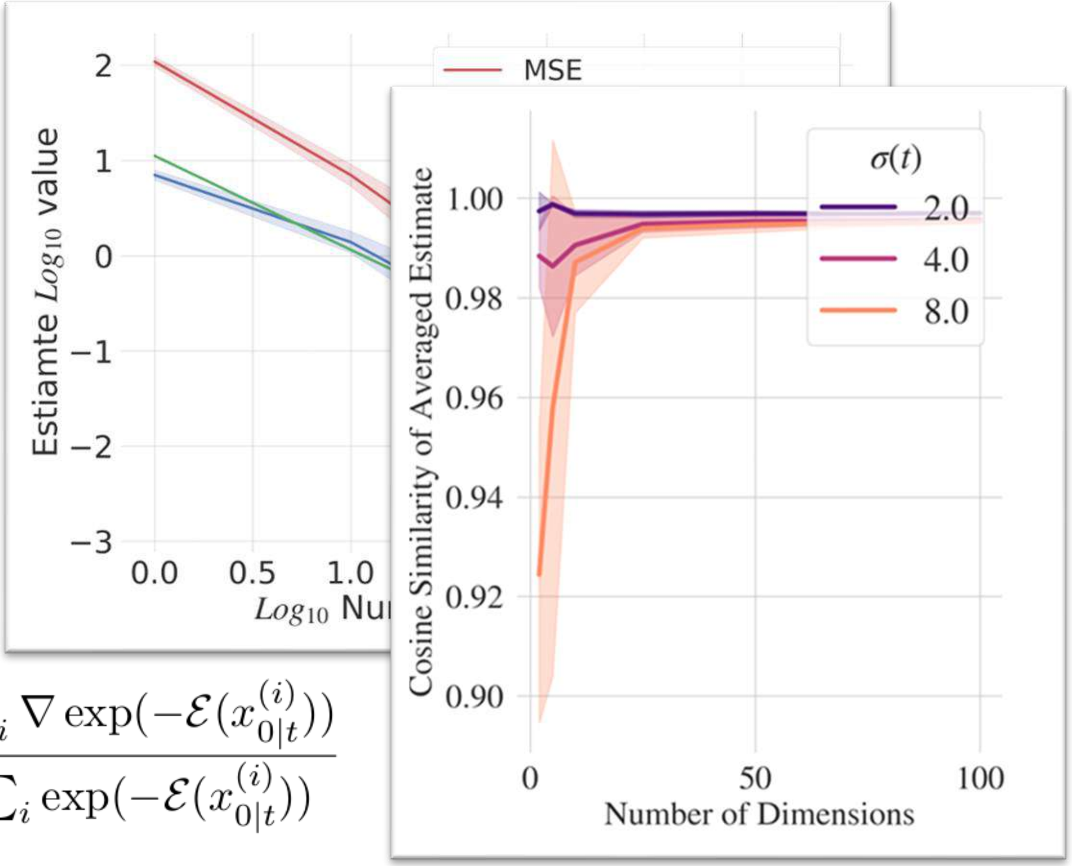


C1 – Inner Loop

$$\nabla \log p_t(x_t) = \frac{((\nabla p_0) * \mathcal{N}(0, \sigma_t^2))(x_t)}{p_t(x_t)}$$

$$= \frac{\mathbb{E}_{x_{0|t} \sim \mathcal{N}(x_t, \sigma_t^2)}[\nabla p_0(x_{0|t})]}{\mathbb{E}_{x_{0|t} \sim \mathcal{N}(x_t, \sigma_t^2)}[p_0(x_{0|t})]}$$

$$= \frac{\mathbb{E}_{x_{0|t} \sim \mathcal{N}(x_t, \sigma_t^2)}[\nabla \exp(-\mathcal{E}(x_{0|t}))]}{\mathbb{E}_{x_{0|t} \sim \mathcal{N}(x_t, \sigma_t^2)}[\exp(-\mathcal{E}(x_{0|t}))]}, \quad \approx \frac{\frac{1}{K}\sum_i \nabla \exp(-\mathcal{E}(x_{0|t}^{(i)}))}{\frac{1}{K}\sum_i \exp(-\mathcal{E}(x_{0|t}^{(i)}))}$$



$= \mathcal{S}_K(x_t, t)$

$$x_{0|t}^{(1)}, \ldots, x_{0|t}^{(K)} \sim \mathcal{N}(x_t, \sigma_t^2)$$

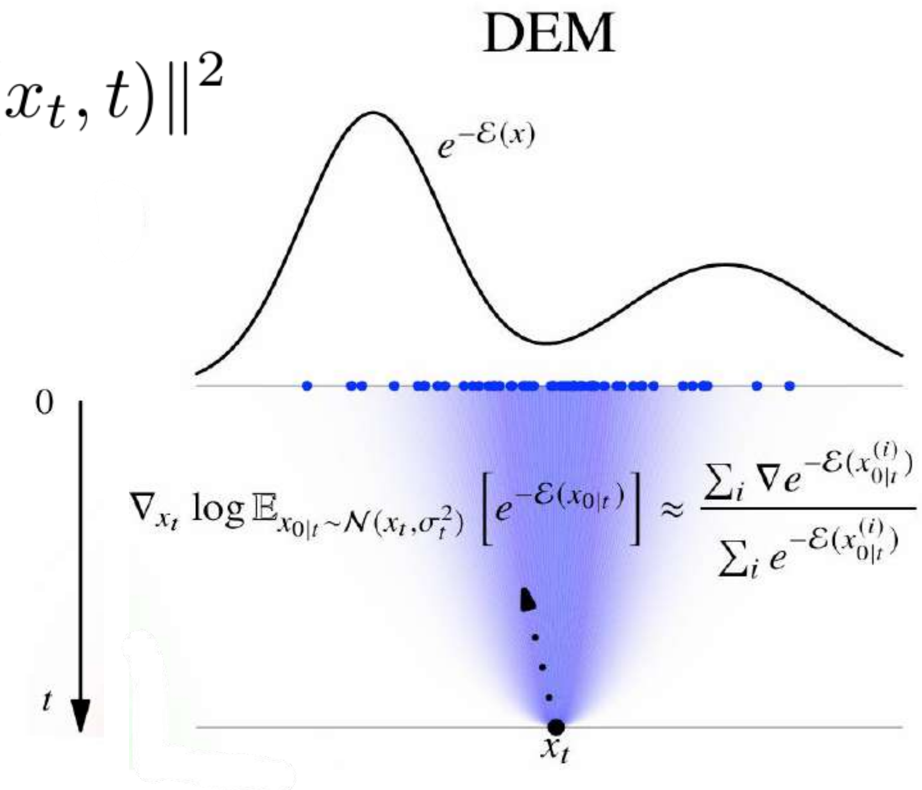
C1 – Inner Loop

$$\mathcal{S}_K(x_t, t) \Leftarrow s_\theta(x_t, t)$$

$$\mathcal{L}_{\text{DEM}}(x_t, t) := \|\mathcal{S}_K(x_t, t) - s_\theta(x_t, t)\|^2$$

inner loop summary:

- Estimate the smoothed score function using noisy samples
- Train a neural network with DEM loss
- Use network in outer loop



C2 – Outer Loop

- With $s_\theta(x_t, t)$ frozen:
- Reverse time SDE
- Generate samples
- Store in replay buffer

Algorithm 1 ITERATED DENOISING ENERGY MATCHING

Input: Network s_θ, Batch size b, Noise schedule σ_t^2, Prior p_1, Num. integration steps L, Replay buffer $\mathcal{B}$, Max Buffer Size $|\mathcal{B}|$, Num. MC samples K.

while Outer-Loop **do**

 $\{x_1\}_{i=1}^b \sim p_1(x_1)$

 $\{x_0\}_{i=1}^b \leftarrow \texttt{sde\_int}(\{x_1\}_{i=1}^b, s_\theta, L)$ {Sample}

 $\mathcal{B} = (\mathcal{B} \cup \{x_0\}_{i=1}^b)$ {Update Buffer $\mathcal{B}$}

 while Inner-Loop **do**

  $x_0 \leftarrow \mathcal{B}.\texttt{sample()}$ {Uniform sampling from $\mathcal{B}$}

  $t \sim \mathcal{U}(0,1)$, $x_t \sim \mathcal{N}(x_0, \sigma_t^2)$

  $\mathcal{L}_{\text{DEM}}(x_t, t) = \|S_K(x_t, t) - s_\theta(x_t, t)\|^2$

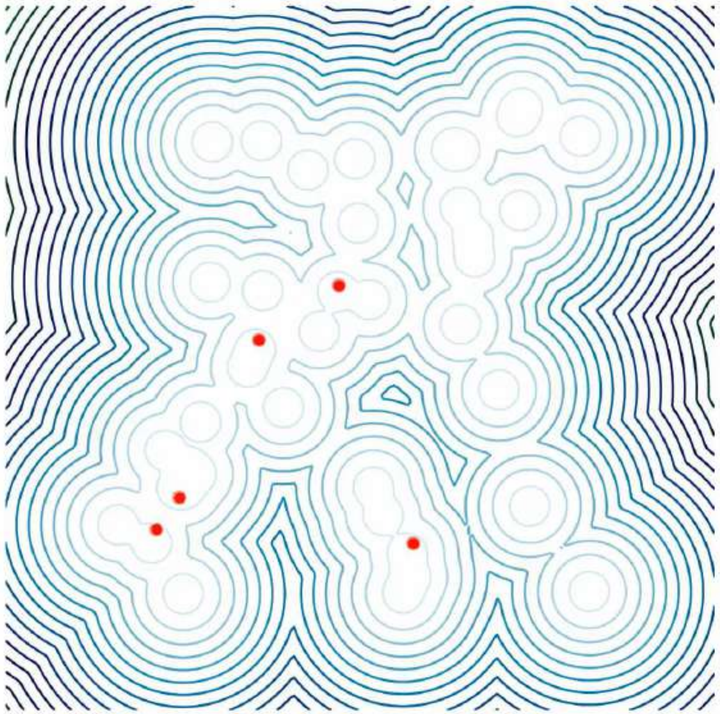
  $\theta \leftarrow \texttt{Update}(\theta, \nabla_\theta \mathcal{L}_{\text{DEM}})$

 end while

end while

output s_θ

iDEM algorithm



Algorithm 1 ITERATED DENOISING ENERGY MATCHING

Input: Network s_θ, Batch size b, Noise schedule σ_t^2, Prior p_1, Num. integration steps L, Replay buffer $\mathcal{B}$, Max Buffer Size $|\mathcal{B}|$, Num. MC samples K.

while Outer-Loop **do**

  $\{x_1\}_{i=1}^b \sim p_1(x_1)$

  $\{x_0\}_{i=1}^b \leftarrow \texttt{sde\_int}(\{x_1\}_{i=1}^b, s_\theta, L)$ {Sample}

  $\mathcal{B} = (\mathcal{B} \cup \{x_0\}_{i=1}^b)$ {Update Buffer $\mathcal{B}$}

  while Inner-Loop **do**

    $x_0 \leftarrow \mathcal{B}.\texttt{sample()}$ {Uniform sampling from $\mathcal{B}$}

    $t \sim \mathcal{U}(0,1), x_t \sim \mathcal{N}(x_0, \sigma_t^2)$

    $\mathcal{L}_{\text{DEM}}(x_t, t) = \|S_K(x_t, t) - s_\theta(x_t, t)\|^2$

    $\theta \leftarrow \texttt{Update}(\theta, \nabla_\theta \mathcal{L}_{\text{DEM}})$

  end while

end while

output s_θ

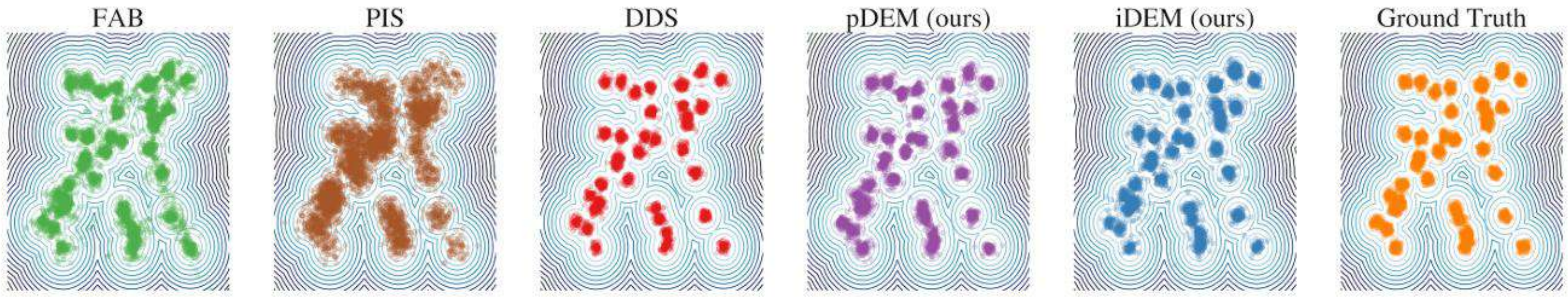
Evaluation on 4 tasks, 5 benchmarks

- 40-mode GMM
- Lennard-Jones 13
- Lennard-Jones 55
- 4-particle double-well potential

- PIS
- DDS
- FAB
- pDEM
- iDEM

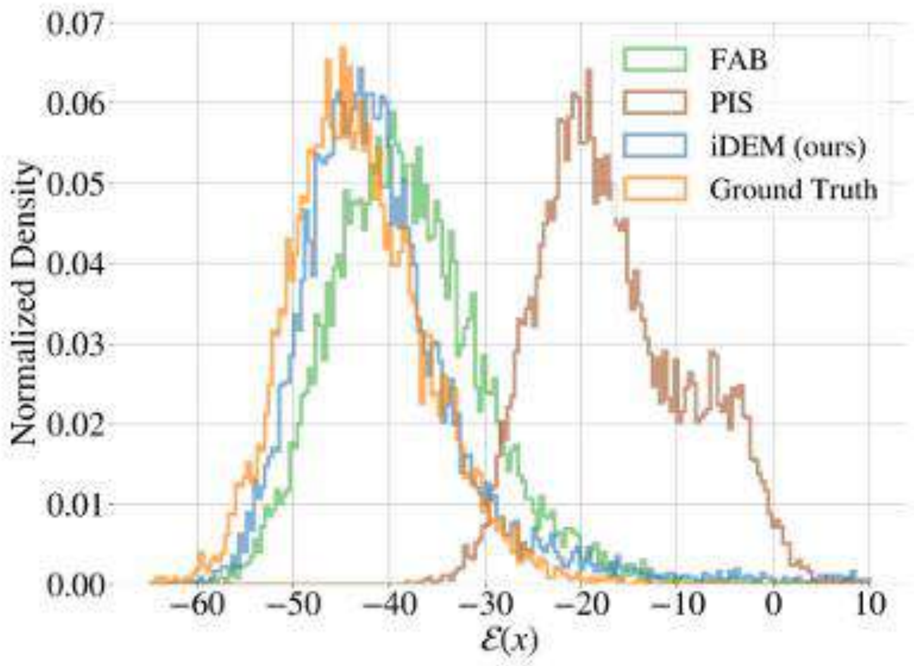
Evaluation on 4 tasks, 5 benchmarks

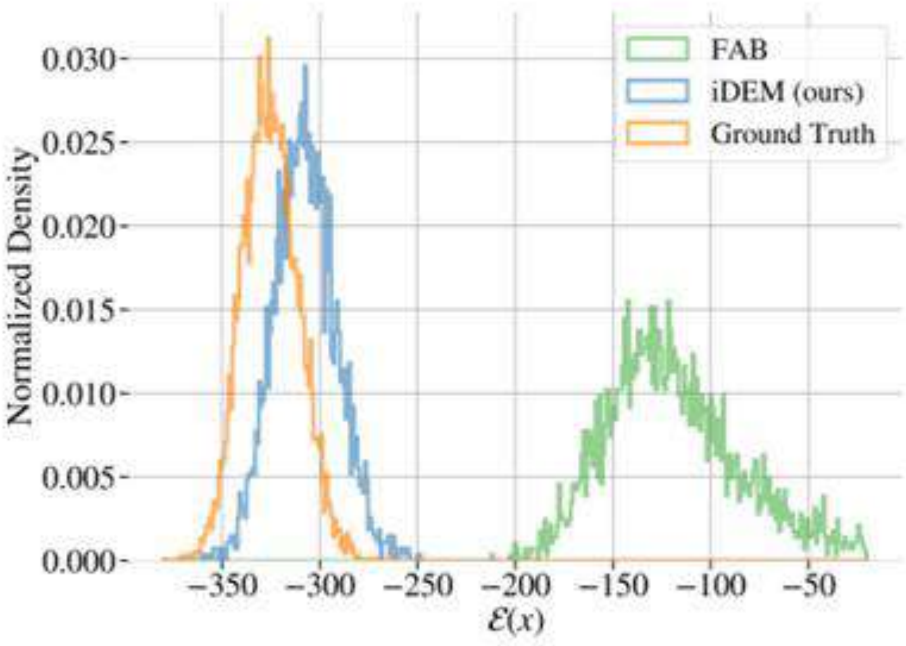
40-mode GMM



Evaluation on 4 tasks, 5 benchmarks

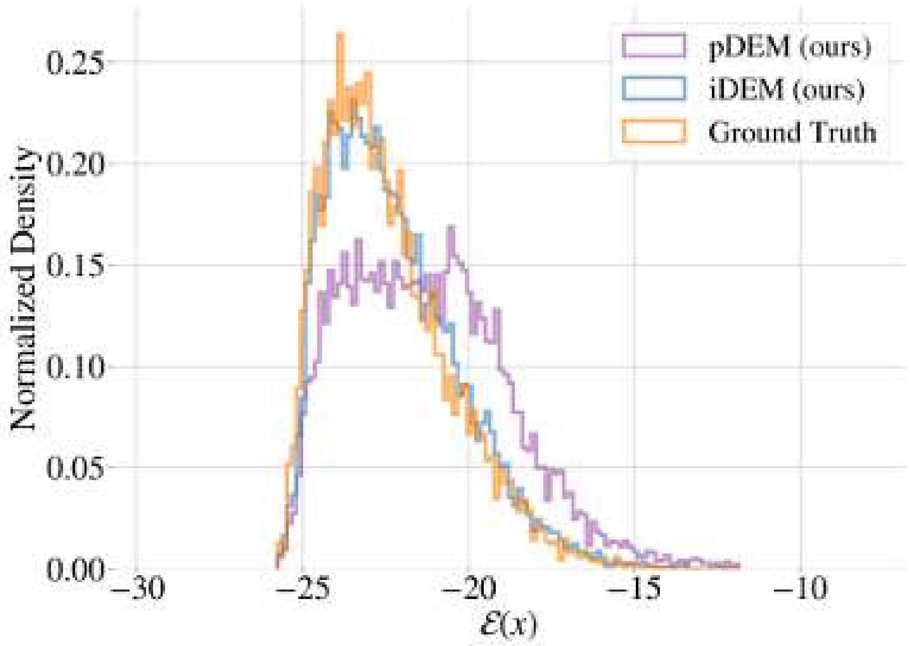
Lennard-Jones 13 , Lennard-Jones 55

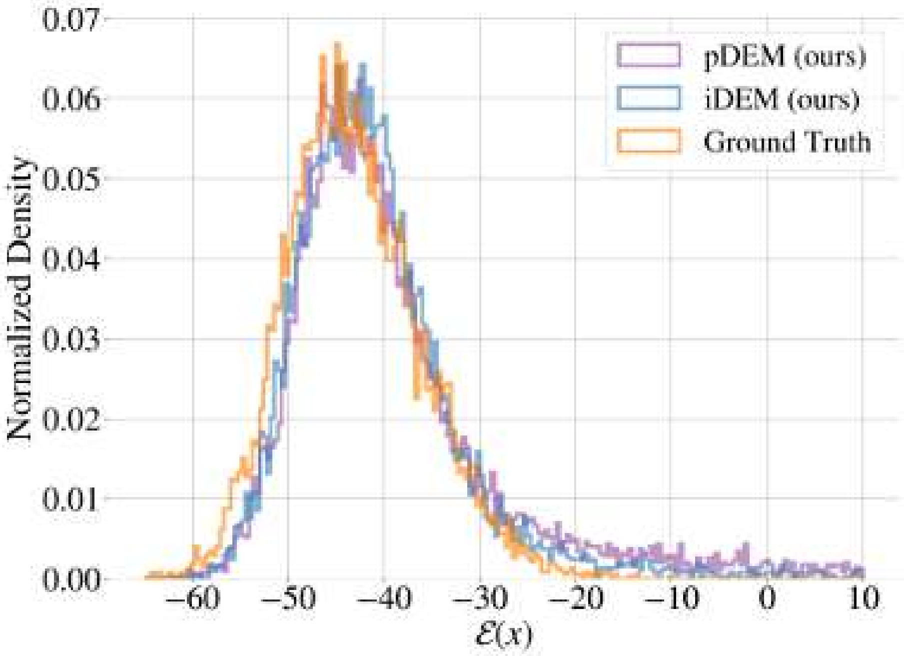




Evaluation on 4 tasks, 5 benchmarks

4-particle double-well potential , LJ-13





Performance Results

- Negative Log Likelihood
- Effective Sample Size
- Wasserstein distance

- Efficient
- High quality samples

Algorithm ↓ Dataset →	GMM	DW-4	LJ-13	LJ-55
FAB (Midgley et al., 2023b)	1.71	6.87	21.78	40.35
PIS (Zhang & Chen, 2022)	4.11	11.29	17.36	$*$
DDS (Vargas et al., 2023)	1.81	5.65	$*$	$*$
pDEM (ours)	0.36	1.40	1.79	$*$
iDEM (ours)	0.87	4.30	6.55	7.75

Energy →	GMM ($d = 2$)			DW-4 ($d = 8$)			LJ-13 ($d = 39$)			LJ-55 ($d = 165$)		
Algorithm ↓	NLL	ESS	$\mathcal{W}_2$	NLL	ESS	$\mathcal{W}_2$	NLL	ESS	$\mathcal{W}_2$	NLL	ESS	$\mathcal{W}_2$
FAB (Midgley et al., 2023b)	7.14±0.01	0.653 ±0.017	12.0±5.73	**7.16**±0.01	**0.947** ±0.007	2.15±0.02	**17.52**±0.17	0.101 ±0.059	4.35±0.01	200.32±62.3	0.063 ±0.001	18.03±1.21
PIS (Zhang & Chen, 2022)	7.72±0.03	0.295 ±0.018	**7.64**±0.92	7.19±0.01	0.901 ±0.003	**2.13**±0.02	47.05±12.46	0.004 ±0.002	4.67±0.11	$*$	$*$	$*$
DDS (Vargas et al., 2023)	7.43±0.46	0.687 ±0.208	9.31±0.82	11.27±1.24	0.408 ±0.001	2.15±0.04	$*$	$*$	$*$	$*$	$*$	$*$
pDEM (ours)	7.10±0.02	0.634 ±0.084	12.20±0.14	7.44±0.05	0.547 ±0.010	**2.11**±0.03	18.80±0.48	0.044 ±0.013	**4.21**±0.06	$*$	$*$	$*$
iDEM (ours)	**6.96**±0.07	**0.734** ±0.092	**7.42**±3.44	**7.17**±0.00	0.825 ±0.002	**2.13**±0.04	**17.68**±0.14	**0.231** ±0.005	**4.26**±0.03	**125.86**±18.03	**0.106** ±0.022	**16.128**±0.071

Discussion & Future Work

- Consistent DEM objective
  - can it be biased still?

- reverse SDE: simulation step still
  - can it be replaced by a learned policy? RL-like or GFlowNets inspired methods?

- Molecule's energy is *really* complex (low temperatures, huge variations)
  - maybe flatten a little the energy landscape with more energy?
    - collect samples and then somehow simulate the trained model at lower temperature progressively until we reach the target temperature?

Conclusion & Acknowledgements

paper

code

iDEM strong step forward:

- scalable
- symmetry-aware
- simulation-efficient sampling
- general
- extensible
  - Feynman-Kac Correctors, Schrodinger Bridges, Transition Matching . . .

- special thanks to the authors for their work and the code availability →

Tara Akhound-Sadegh*, Jarrid Rector-Brooks*, Avishek Joey Bose*, Sarthak Mittal, Pablo Lemos, Cheng-Hao Liu, Marcin Sendera, Siamak Ravanbakhsh, Gauthier Gidel, Yoshua Bengio, Nikolay Malkin, Alexander Tong