



Oops I took a gradient!

Scalable sampling for discrete distributions



Will Grathwohl

Energy-Based Models

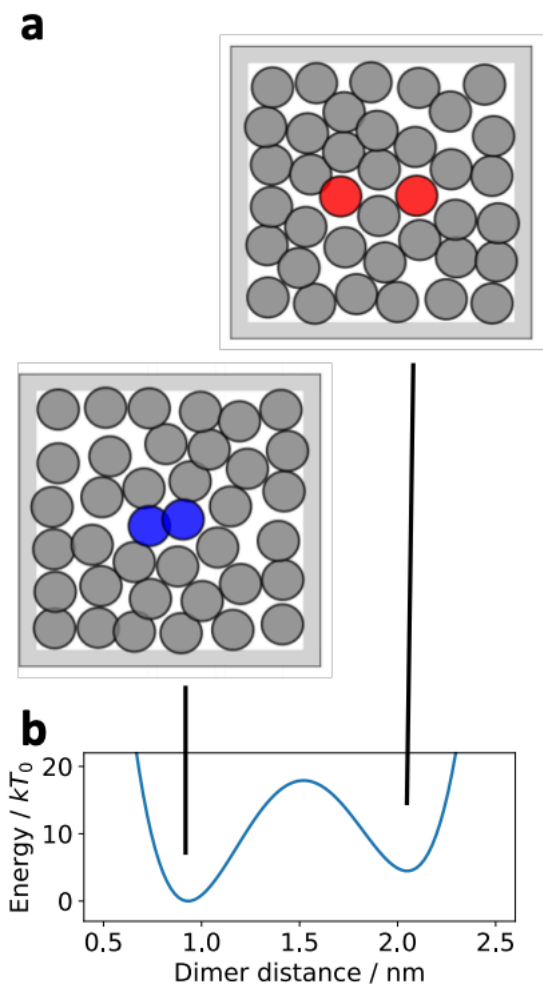
- An energy-based model (EBM) is a probability model in the following form:

$$p_{\theta}(x) = \frac{e^{-E_{\theta}(x)}}{Z(\theta)} \quad Z(\theta) = \int_{\mathcal{X}} e^{-E_{\theta}(x)} dx$$

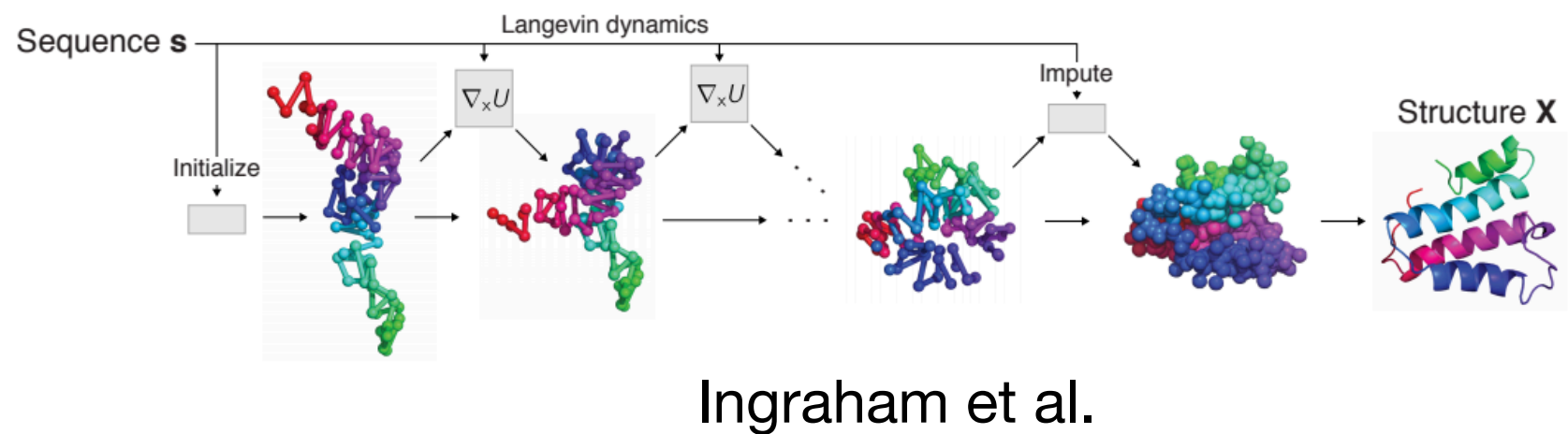
- Where $E_{\theta}(x) : \mathcal{X} \rightarrow \mathbb{R}$ fully specifies the model so $Z(\theta)$ does not need to be modelled

Energy-Based Models

- Long been popular in biology, physics, chemistry, and natural sciences



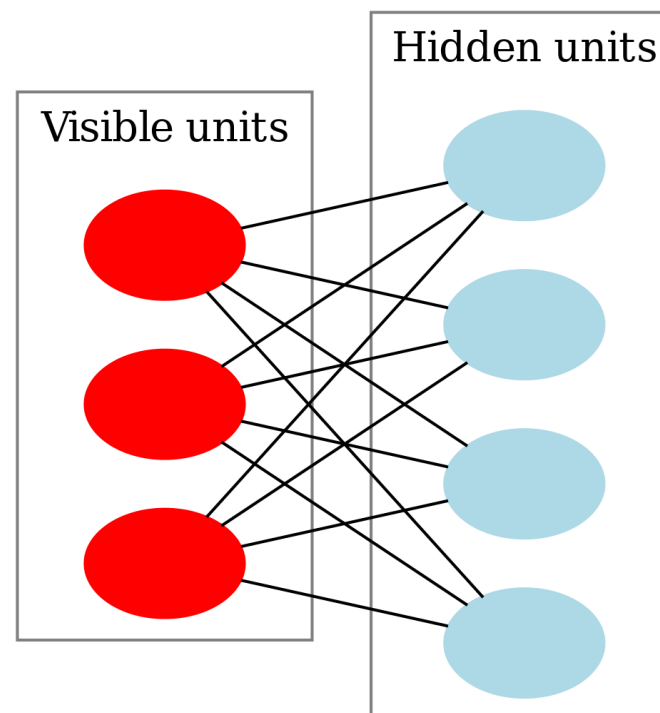
Noe et al.



Energy-Based Models

- Foundational to the history of deep learning and generative models
- For example, RBMs:

$$E(v, h) = -a^T v - b^T h - v^T W h, \quad p(v, h) = \frac{e^{-E(v, h)}}{Z}$$

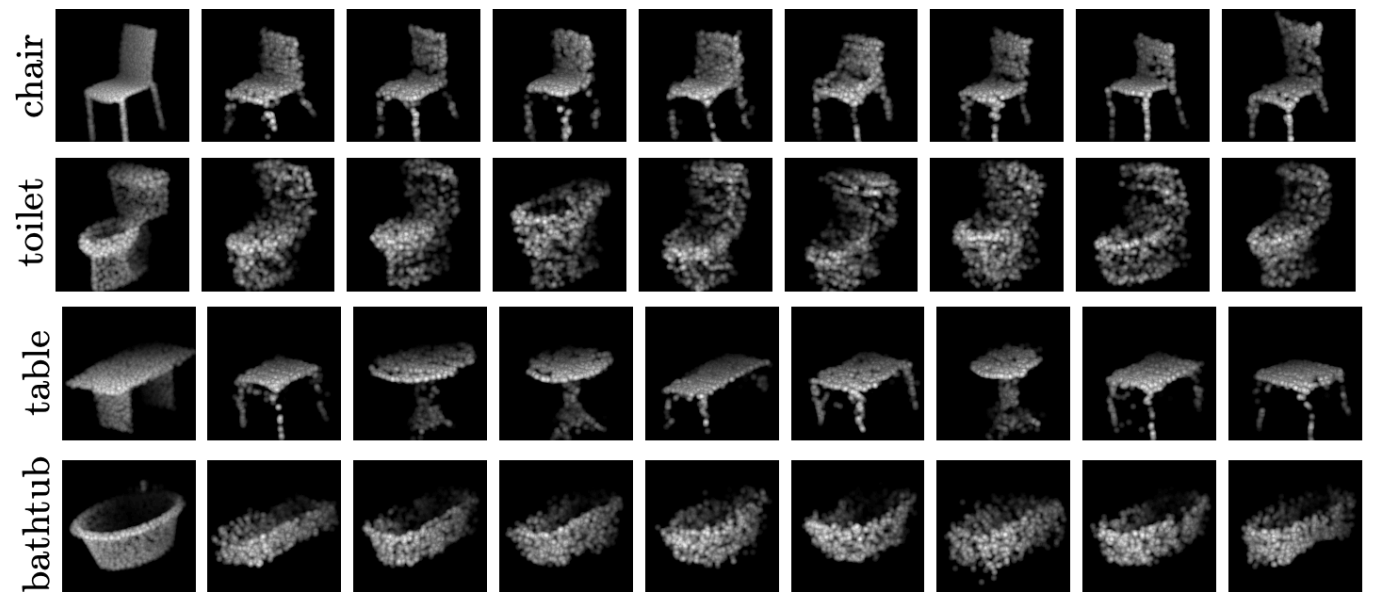


Energy-Based Models

- Fell out of favour in the 2000's as other approaches scaled better — VAEs, Flows, GANs, etc...
- But, have recently been re-popularized and are now one of the strongest approaches for generative modelling



Song et al. (2021)



Xie et al. (2020)

The comeback kid?

- **What explains the rapid success of recent EBM methods?**
 - **Deep neural nets**
 - **Gradient-based MCMC**
 - **Score matching**

Training EBMs

- To maximize likelihood we must compute

$$\log p_{\theta}(x) = -E_{\theta}(x) - \log Z(\theta)$$

$$\log p_{\theta}(x) = -E_{\theta}(x) - \log \int e^{-E_{\theta}(x)} dx$$

- Which is intractable
- The gradient however is simpler

$$\nabla_{\theta} \log p_{\theta}(x) = -\nabla_{\theta} E_{\theta}(x) - \mathbf{E}_{p_{\theta}(x)}[\nabla_{\theta} E_{\theta}(x)]$$

- If we can sample from model then we can derive an unbiased gradient estimator
- We can use this to train

Training EBMs (in the past)

- We design $E_{\theta}(x)$ to make sampling easy
- For example, an RBM

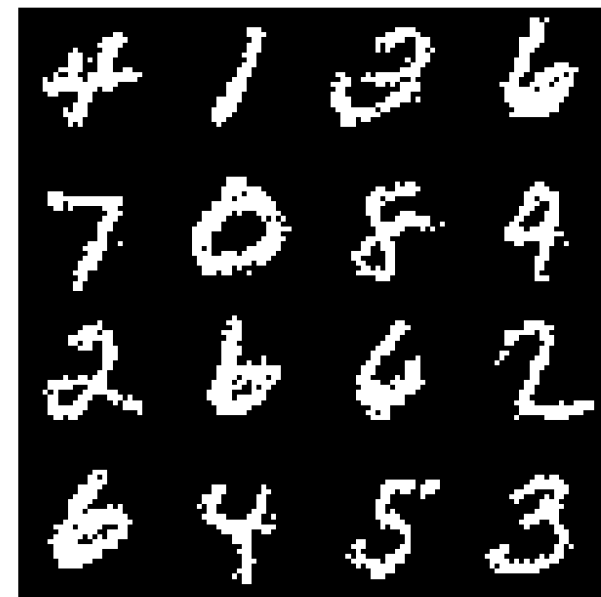
$$E_{\theta}(x, h) = h^T W x + c^T x + b^T h$$

- Which enables fast sampling through Gibbs

$$p(x | h) = \text{Bernoulli}(x; W^T h + c)$$

$$p(h | x) = \text{Bernoulli}(h; W x + b)$$

- Consequences are:
 - discrete data+latents
 - restricted architectures



Salakhutdinov and Murray (2008)

Training EBMs (today)

- We design $E_\theta(x)$ to be flexible as possible
- We let be a deep neural network $E_\theta(x) = -f_\theta(x)$
- There is no longer structure which can be exploited for sampling
- *Unless* we make data continuous...
- We can use gradient-based samplers

$$x_{t+t} = x_t + \frac{\epsilon}{2} \nabla_x f_\theta(x) + \epsilon \eta, \quad \eta \sim N(0, I)$$

- Consequences are:
 - Data must be continuous
 - Energy must be differentiable



Discrete Data

- **Now EBMs compete with (or outperform) other generative models**
- **Are powerful method for**
 - **Semi-supervised learning**
 - **Adversarial robustness**
 - **Image generation**
 - **Out-of-distribution detection**
 - **And more!**
- **We would love to apply these new advances to EBMs for discrete data such as tabular data, text, proteins, DNA**
- **When gradient-based MCMC cannot be applied, we're out of luck**
- **We could use de-quantization techniques, but these don't work (trust me...)**
- **How do we move forward???**

In this work...

- **We focus on MCMC sampling from discrete distributions where minimal structure is known a priori**
- **Motivated by EBMs we study a wide-class of discrete distributions**
- **We find a simple structure common to many discrete distributions**
- **We present a very simple sampler which exploits this structure in a novel way**
- **We find this sampler outperforms prior samplers for discrete data**
- **Our sampler enables the training of deep EBMs on discrete data**

Discrete Sampling

- We focus on sampling from $p(x) = \frac{e^{f(x)}}{Z}$ where
- $x \in \{0,1\}^D$ or $x \in \{0,\dots,K\}^D$

Discrete Sampling

- Simplest and most general approach is Gibbs sampling
- Pick dim i then re-sample $x_i \sim p(x_i | x_{-i})$
- For discrete x has form

$$p(x_i = k | x_{-i}) = \frac{\exp f(x_1, \dots, x_{i-1}, k, x_{i+1}, \dots, x_D)}{\sum_{j=1}^K \exp f(x_1, \dots, x_{i-1}, j, x_{i+1}, \dots, x_D)}$$

Discrete Sampling

- Simplest and most general approach is Gibbs sampling
- Pick dim i then re-sample $x_i \sim p(x_i | x_{-i})$
- For discrete x has form

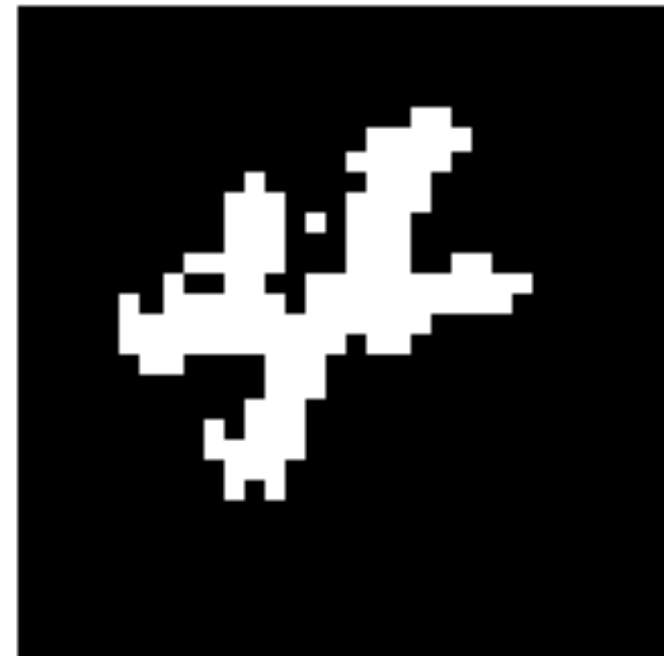


Typically fix an ordering
and iterate through

$$p(x_i = k | x_{-i}) = \frac{\exp f(x_1, \dots, x_{i-1}, k, x_{i+1}, \dots, x_D)}{\sum_{j=1}^K \exp f(x_1, \dots, x_{i-1}, j, x_{i+1}, \dots, x_D)}$$

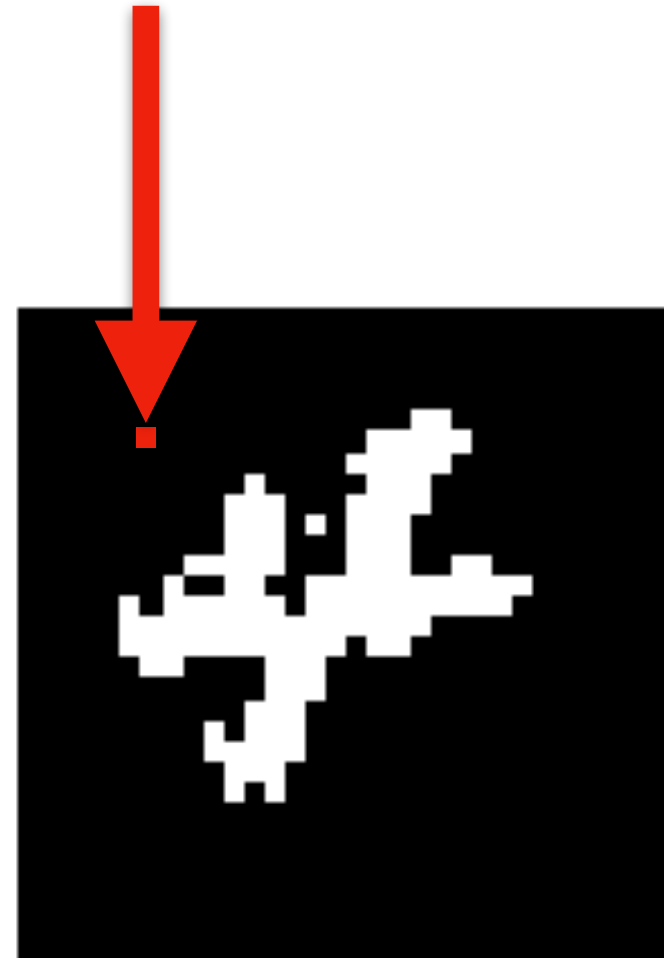
An example...

- **Consider sampling from a model of MNIST**
- **Most pixels are black**



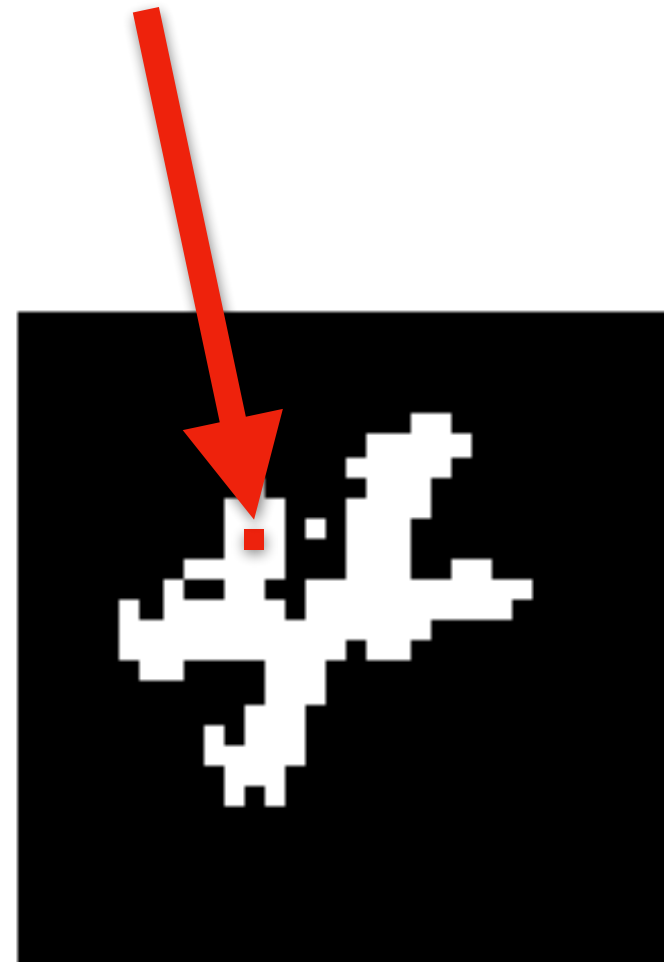
An example...

- **Consider sampling from a model of MNIST**
- **Most pixels are black**
- **If we propose dim in background**
- **Will not change \rightarrow computation wasted**



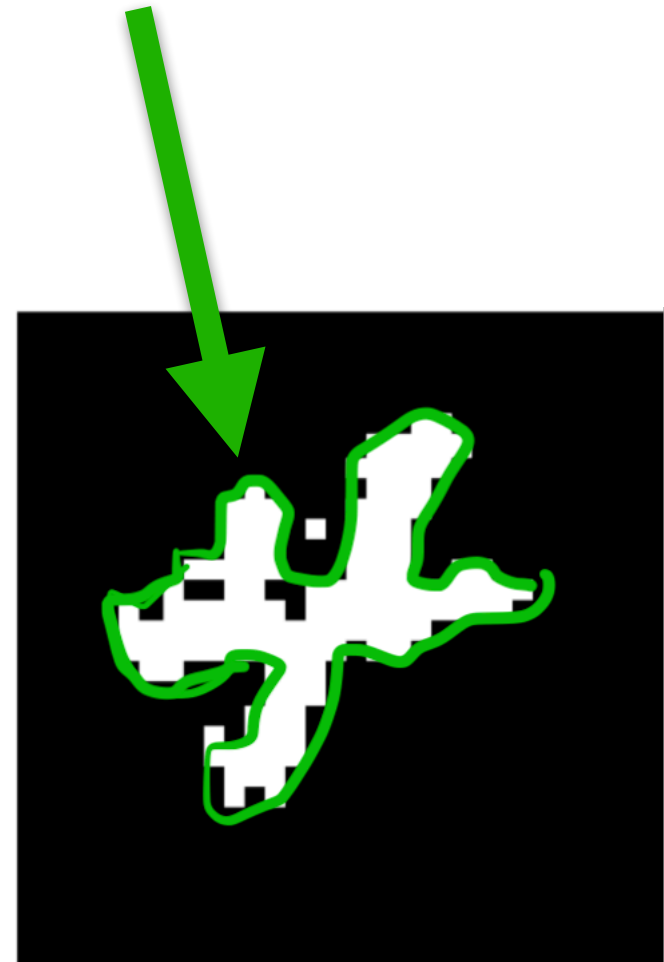
An example...

- **Consider sampling from a model of MNIST**
- **Most pixels are black**
- **If we propose dim in background**
- **Will not change → computation wasted**
- **If we propose dim in middle of digit**
- **Will not change → computation wasted**



An example...

- **Consider sampling from a model of MNIST**
- **Most pixels are black**
- **If we propose dim in background**
- **Will not change → computation wasted**
- **If we propose dim in middle of digit**
- **Will not change → computation wasted**
- **Dims on edge will change**



Adaptive Gibbs

- With this intuition we can come up with a related sampler
- Metropolis-Hastings with proposal $q(x' | x)$
- Proposal places distribution $q(i)$ over dimensions
- To generate x' , sample $i \sim q(i)$ and set $x' = \text{flip_dim}(x, i)$ accept with prob

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(x | x')}{q(x' | x)}, 1 \right\} \text{ (Metropolis-Hastings)}$$

Adaptive Gibbs

- With this intuition we can come up with a related sampler
- Metropolis-Hastings with proposal $q(x' | x)$
- Proposal places distribution $q(i)$ over dimensions
- To generate x' , sample $i \sim q(i)$ and set $x' = \text{flip_dim}(x, i)$ accept with prob

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(i)}{q(i)}, 1 \right\} \text{ (Metropolis-Hastings)}$$

Adaptive Gibbs

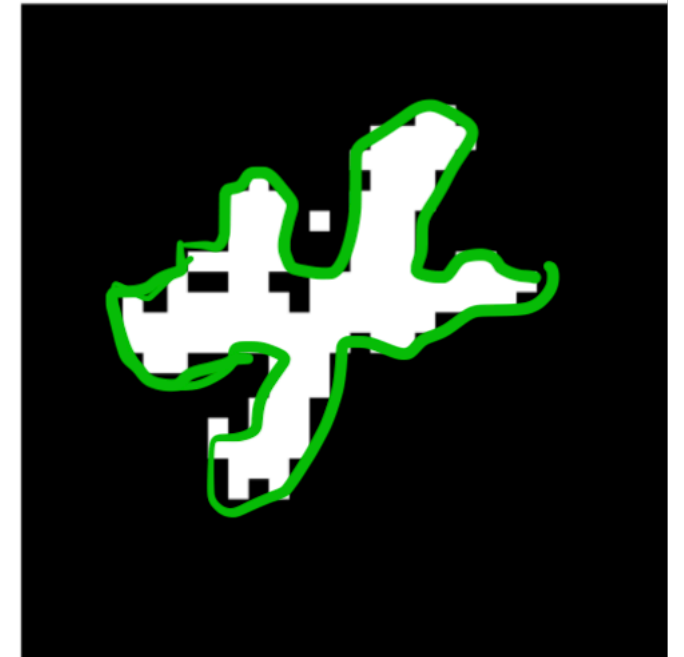
- With this intuition we can come up with a related sampler
- Metropolis-Hastings with proposal $q(x' | x)$
- Proposal places distribution $q(i)$ over dimensions
- To generate x' , sample $i \sim q(i)$ and set $x' = \text{flip_dim}(x, i)$ accept with prob

$$\min \{ \exp(f(x') - f(x)), 1 \} \text{ (Metropolis-Hastings)}$$

- Bias $q(i)$ towards dims which are likely to flip \rightarrow Efficient sampling!

Back to our example...

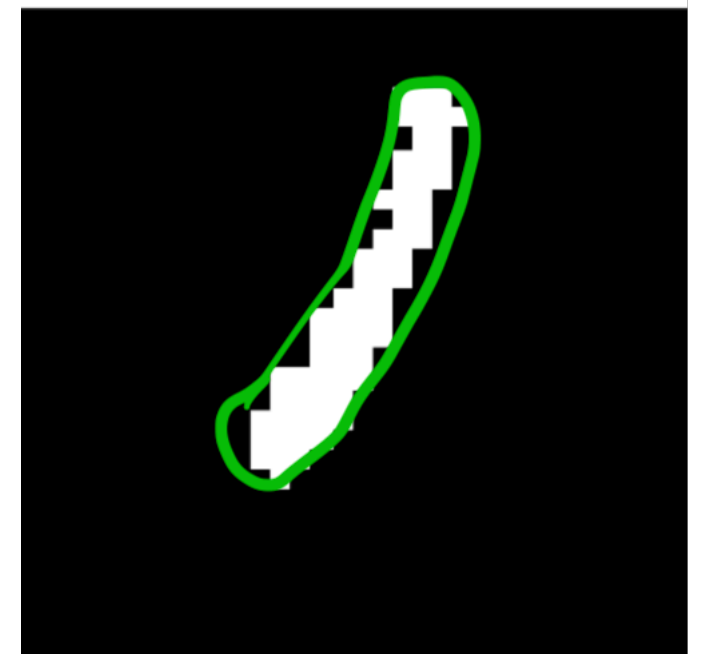
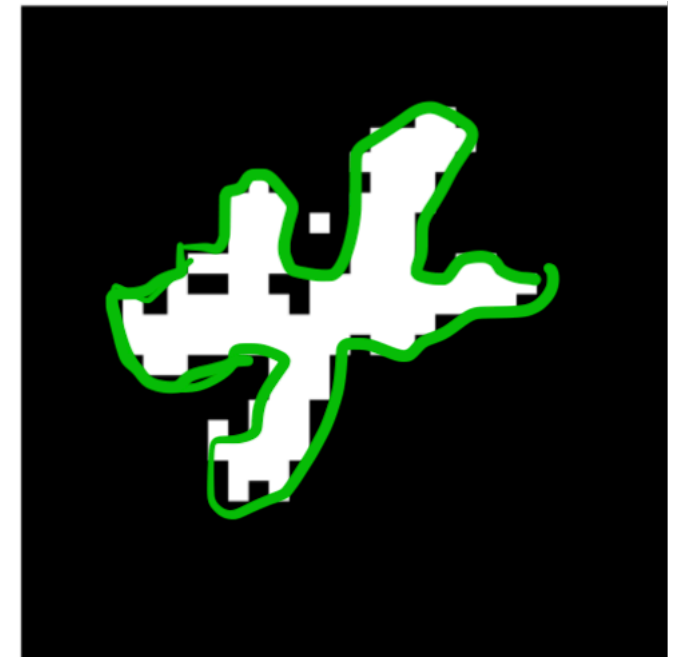
- Dims most likely to flip depend on input



Back to our example...

- Dims most likely to flip depend on input
- Thus, an input-dependent proposal $q(i | x)$ should be most efficient
- Changes accept probability to

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(x | x')}{q(x' | x)}, 1 \right\}$$



Proposals for Discrete Sampling

- How to design $q(x' | x)$? Acceptance rate:

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(x | x')}{q(x' | x)}, 1 \right\}$$

- Want $f(x') - f(x)$ high to proposals have high likelihood
- Want $q(x | x')$ high (reversible) to have high acceptance rate
- Need $q(x' | x)$ to balance these for good sampling

Proposals for Discrete Sampling

- How to design $q(x' | x)$? Acceptance rate:

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(x | x')}{q(x' | x)}, 1 \right\}$$

- Want $f(x') - f(x)$ high to proposals have high likelihood
- Want $q(x | x')$ high (reversible) to have high acceptance rate
- Need $q(x' | x)$ to balance these for good sampling

- Idea: let $q_\tau(x' | x) = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{Z(x)} = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{\sum_{x'' \in H(x)} \exp\left(\frac{f(x'') - f(x)}{\tau}\right)}$

Proposals for Discrete Sampling

- How to design $q(x' | x)$? Acceptance rate:

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(x | x')}{q(x' | x)}, 1 \right\}$$

- Want $f(x') - f(x)$ high to proposals have high likelihood
- Want $q(x | x')$ high (reversible) to have high acceptance rate
- Need $q(x' | x)$ to balance these for good sampling

- Idea: let $q_\tau(x' | x) = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{Z(x)} = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{\sum_{x'} \exp\left(\frac{f(x') - f(x)}{\tau}\right)}$

Tempered softmax over

$$\frac{f(x') - f(x)}{\tau}$$

For possible x'

Proposals for Discrete Sampling

- How to design $q(x' | x)$? Acceptance rate:

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(x | x')}{q(x' | x)}, 1 \right\}$$

- Want $f(x') - f(x)$ high to proposals have high likelihood

Make τ small

- Want $q(x | x')$ high (reversible) to have high acceptance rate

- Need $q(x' | x)$ to balance these for good sampling

Idea: let $q_\tau(x' | x) = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{Z(x)} = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{\sum_{x'} \exp\left(\frac{f(x') - f(x)}{\tau}\right)}$

Tempered softmax over

$$\frac{f(x') - f(x)}{\tau}$$

For possible x'

Proposals for Discrete Sampling

- How to design $q(x' | x)$? Acceptance rate:

$$\min \left\{ \exp(f(x') - f(x)) \frac{q(x | x')}{q(x' | x)}, 1 \right\}$$

- Want $f(x') - f(x)$ high to proposals have high likelihood
- Want $q(x | x')$ high (reversible) to have high acceptance rate
- Need $q(x' | x)$ to balance these for good sampling

Make τ big

Idea: let $q_\tau(x' | x) = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{Z(x)} = \frac{\exp\left(\frac{f(x') - f(x)}{\tau}\right)}{\sum_{x'} \exp\left(\frac{f(x') - f(x)}{\tau}\right)}$

Tempered softmax over

$$\frac{f(x') - f(x)}{\tau}$$

For possible x'

Choosing τ

- Rewrite acceptance probability wrt $q_\tau(x' | x)$

$$\begin{aligned} & \min \left\{ \exp(f(x') - f(x)) \frac{q_\tau(x | x')}{q_\tau(x' | x)}, 1 \right\} \\ &= \min \left\{ \exp(f(x') - f(x)) \frac{\exp(\frac{1}{\tau}(f(x) - f(x'))) \frac{Z(x')}{Z(x)}}{\exp(\frac{1}{\tau}(f(x') - f(x)))}, 1 \right\} \\ &= \min \left\{ \exp \left(\left(1 - \frac{2}{\tau} \right) (f(x') - f(x)) \right) \frac{Z(x')}{Z(x)}, 1 \right\} \end{aligned}$$

Choosing τ

- Rewrite acceptance probability wrt $q_\tau(x' | x)$

$$\begin{aligned} & \min \left\{ \exp(f(x') - f(x)) \frac{q_\tau(x | x')}{q_\tau(x' | x)}, 1 \right\} \\ &= \min \left\{ \exp(f(x') - f(x)) \frac{\exp(\frac{1}{\tau}(f(x) - f(x'))) \frac{Z(x')}{Z(x)}}{\exp(\frac{1}{\tau}(f(x') - f(x)))}, 1 \right\} \\ &= \min \left\{ \exp \left(\underbrace{\left(1 - \frac{2}{\tau} \right) (f(x') - f(x))}_{\text{Set } \tau = 2 \text{ to cancel}} \right) \frac{Z(x')}{Z(x)}, 1 \right\} \end{aligned}$$

Set $\tau = 2$ to cancel

Choosing τ

- Rewrite acceptance probability wrt $q_2(x' | x)$

$$\min \left\{ \exp(f(x') - f(x)) \frac{q_2(x | x')}{q_2(x' | x)}, 1 \right\}$$
$$= \min \left\{ \frac{Z(x')}{Z(x)}, 1 \right\}$$

Choosing τ

- Rewrite acceptance probability wrt $q_2(x' | x)$

$$\min \left\{ \exp(f(x') - f(x)) \frac{q_2(x | x')}{q_2(x' | x)}, 1 \right\}$$

$$= \min \left\{ \frac{Z(x')}{Z(x)}, 1 \right\}$$



Should be
near 1

Difference Functions

- **We want a proposal**

$$q(x' | x) = \frac{\exp\left(\frac{f(x') - f(x)}{2}\right)}{Z(x)}$$

- **To sample we must compute $f(x') - f(x)$ for all $x' \in H(x)$**
- **If this is Hamming ball of size 1 this means $O(D)$ function evals**
- **Bad news if D big...**

A surprisingly common structure

Bernoulli: $\log p(x) = \theta x - \log Z$

Categorical: $\log p(x) = \theta^T x - \log Z$

Ising: $\log p(x) = x^T W x + b^T x - \log Z$

Potts: $\log p(x) = \sum_{i=1}^D h_i^T x_i + \sum_{ij} x_i^T J_{ij} x_j - \log Z$

RBM: $\log p(x) = \sum_i \text{softplus}(Wx + b)_i + c^T x$

HMM: $\log p(x | y) = \sum_{t=1}^T x_t A x_{t-1} + \frac{(w^T x_t - y_t)^2}{\sigma^2}$

Deep EBM: $\log p(x) = f_{\theta}(x) - \log Z$

A surprisingly common structure

Bernoulli: $\log p(x) = \theta x - \log Z$

Categorical: $\log p(x) = \theta^T x - \log Z$

Ising: $\log p(x) = x^T W x + b^T x - \log Z$

Potts: $\log p(x) = \sum_{i=1}^D h_i^T x_i + \sum_{ij} x_i^T J_{ij} x_j - \log Z$

RBM: $\log p(x) = \sum_i \text{softplus}(Wx + b)_i + c^T x$

HMM: $\log p(x|y) = \sum_{t=1}^T x_t^T A x_{t-1} + \frac{(w^T x_t - y_t)^2}{\sigma^2}$

Deep EBM: $\log p(x) = f_\theta(x) - \log Z$

These are all continuous,
differentiable functions of
real-valued inputs!

Discrete structure is
created by restricting
input to $\{0,1\} \subset R$

- What's going on here?

Exploiting a surprisingly common structure

- If the unnormalized log-probability function $f(x)$ is a continuous, differentiable real-valued function restricted to a discrete set
- We can use Taylor-series to estimate

$$f(x') \approx (x' - x)^T \nabla_x f(x)$$

- For binary data, we estimate $f(x') - f(x)$ for all $x' \in H(x)$ with:

$$\tilde{d}(x) = - (2x - 1) \odot \nabla_x f(x)$$

- Where $\tilde{d}(x)_i = f(\text{flip_dim}(x, i)) - f(x)$

- For categorical data:

$$\tilde{d}(x)_{ij} = \nabla_x f(x)_{ij} - x_i^T \nabla_x f(x)_i$$

Gibbs With Gradients

- We propose a new sampler for discrete distributions
- We do Metropolis-Hastings with a proposal $q(x' | x)$
- The proposal approximates:

$$q(x' | x) = \frac{\exp\left(\frac{f(x') - f(x)}{2}\right)}{Z(x)}$$

- Using a Taylor-series

$$q(x' | x) = \frac{\exp\left(\frac{(x' - x)^T \nabla_x f(x)}{2}\right)}{\tilde{Z}(x)}$$

- Where $x' \in H(x)$ and we use $\tilde{d}(x)$ to efficiently compute all differences using $O(1)$ function evaluations

Gibbs With Gradients (pseudo-code)

Algorithm 1 Gibbs With Gradients

Input: unnormalized log-prob $f(\cdot)$, current sample x

Compute $\tilde{d}(x)$ {Eq. 3 if binary, Eq. 4 if categorical.}

Compute $q(i|x) = \text{Categorical} \left(\text{Softmax} \left(\frac{\tilde{d}(x)}{2} \right) \right)$

Sample $i \sim q(i|x)$

$x' = \text{flipdim}(x, i)$

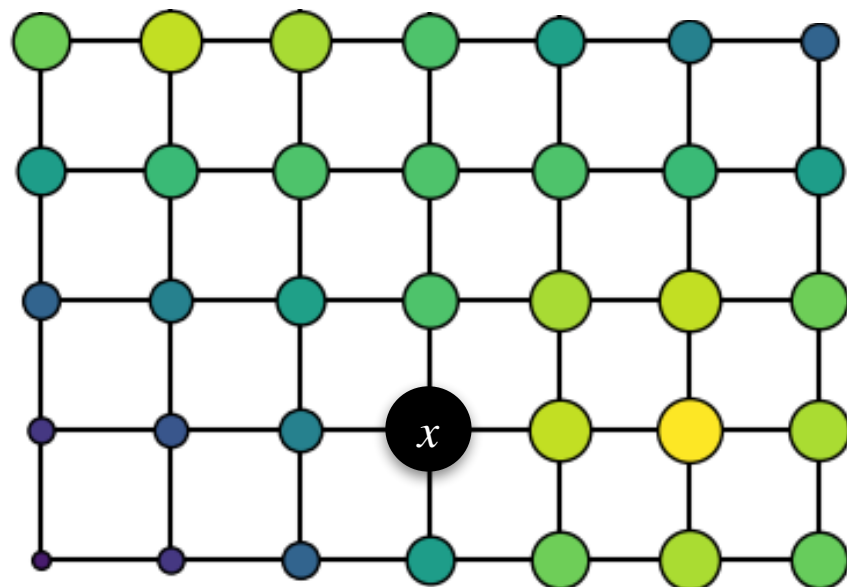
Compute $q(i|x') = \text{Categorical} \left(\text{Softmax} \left(\frac{\tilde{d}(x')}{2} \right) \right)$

Accept with probability:

$$\min \left(\exp(f(x') - f(x)) \frac{q(i|x')}{q(i|x)}, 1 \right)$$

Gibbs With Gradients (visually)

Target Distribution

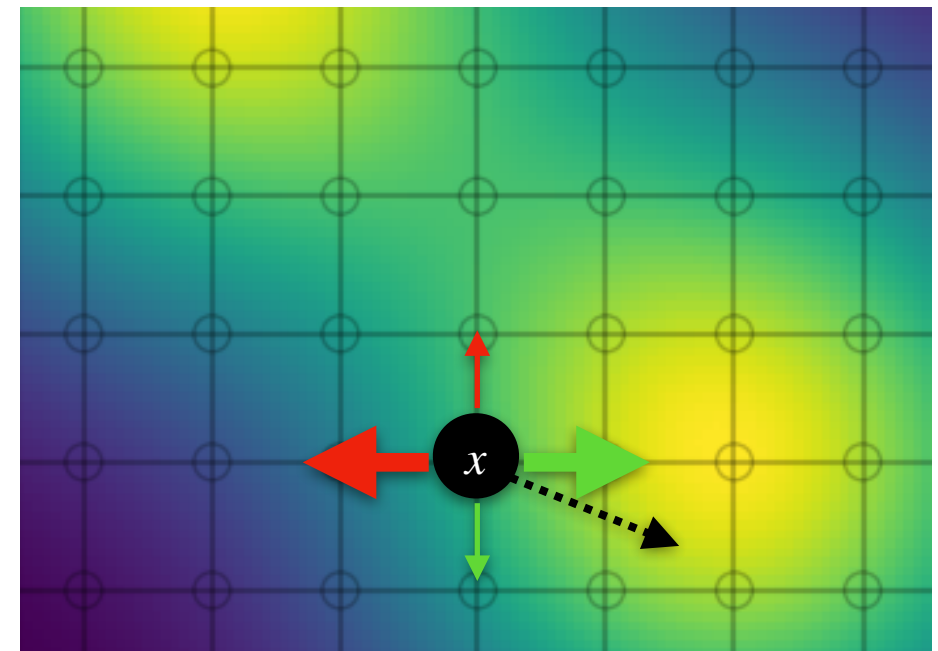


Compute gradients of
continuous function



Estimate
likelihood ratios

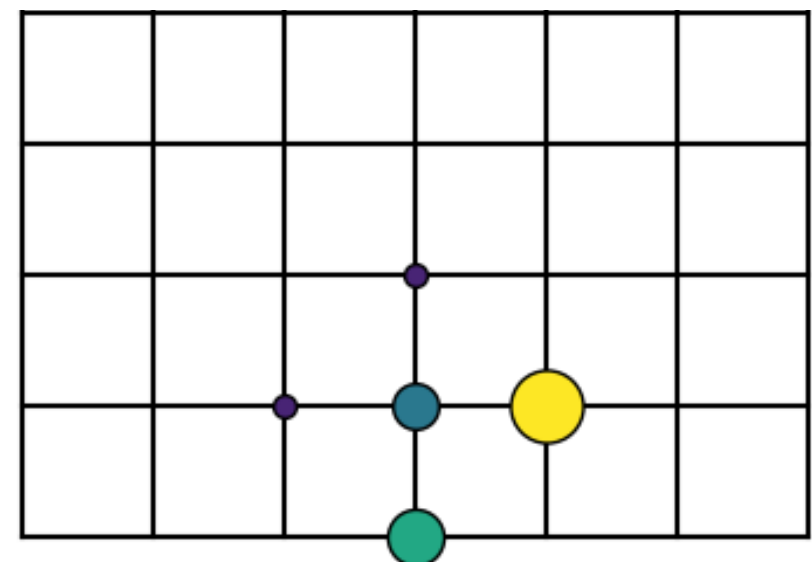
Underlying Continuous Function



Take softmax to obtain
proposal in original
discrete space



Proposal Distribution

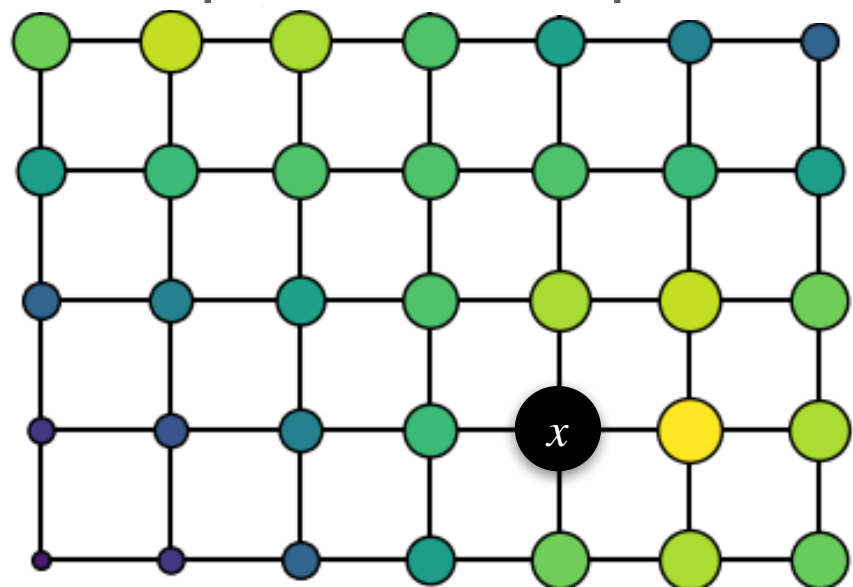


Sample from proposal

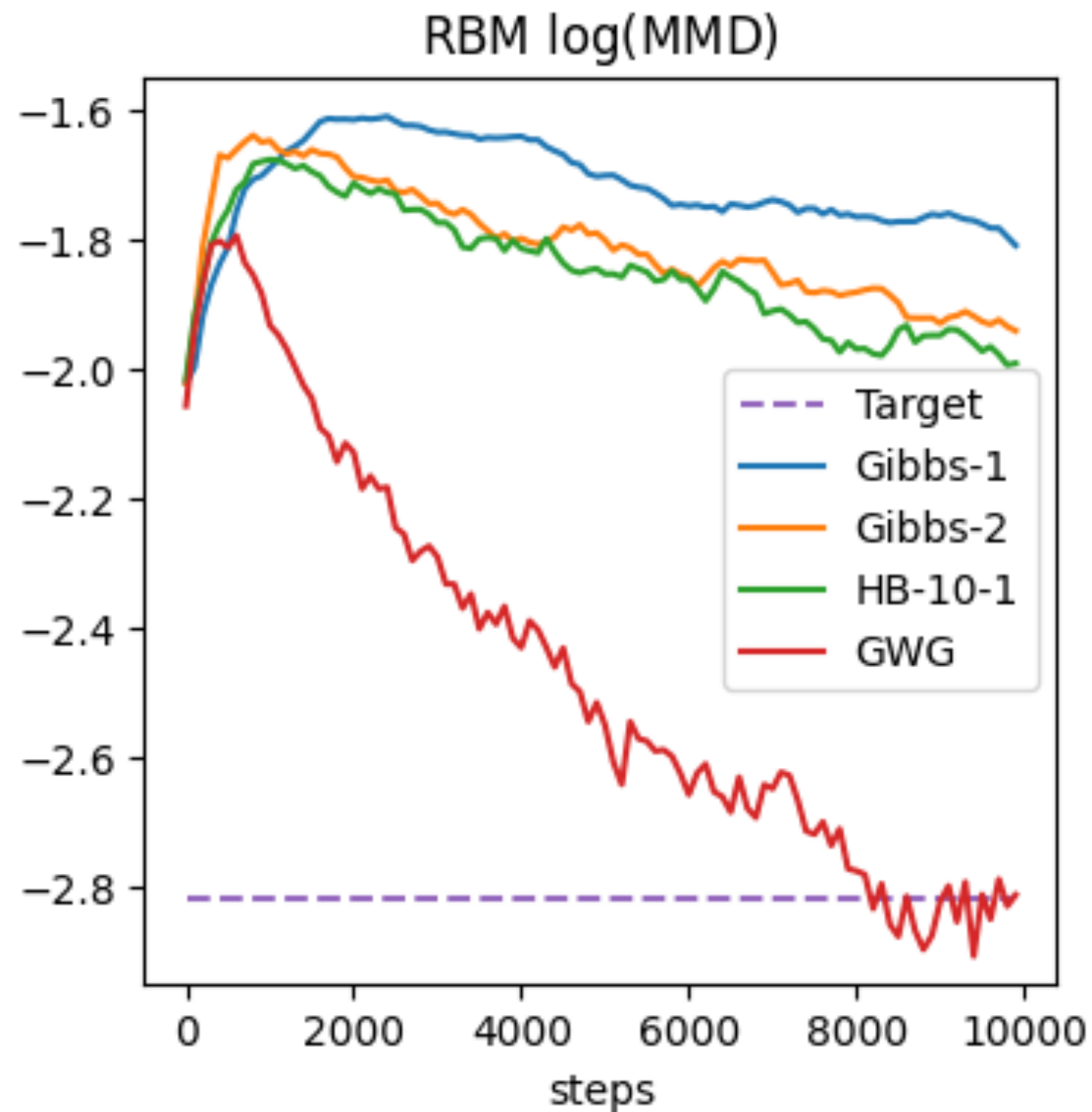


Metropolis-Hastings
Step

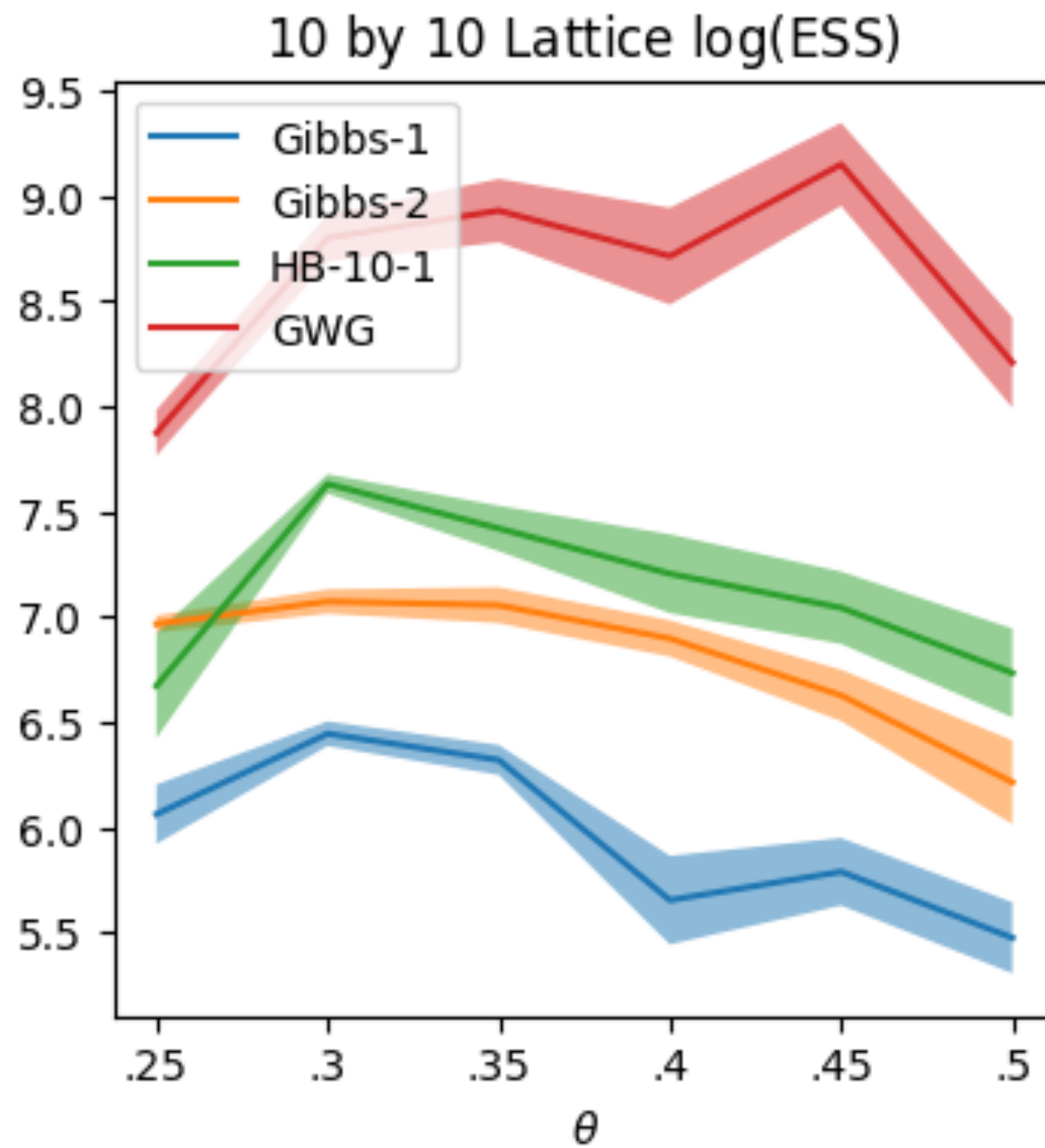
Updated Sample



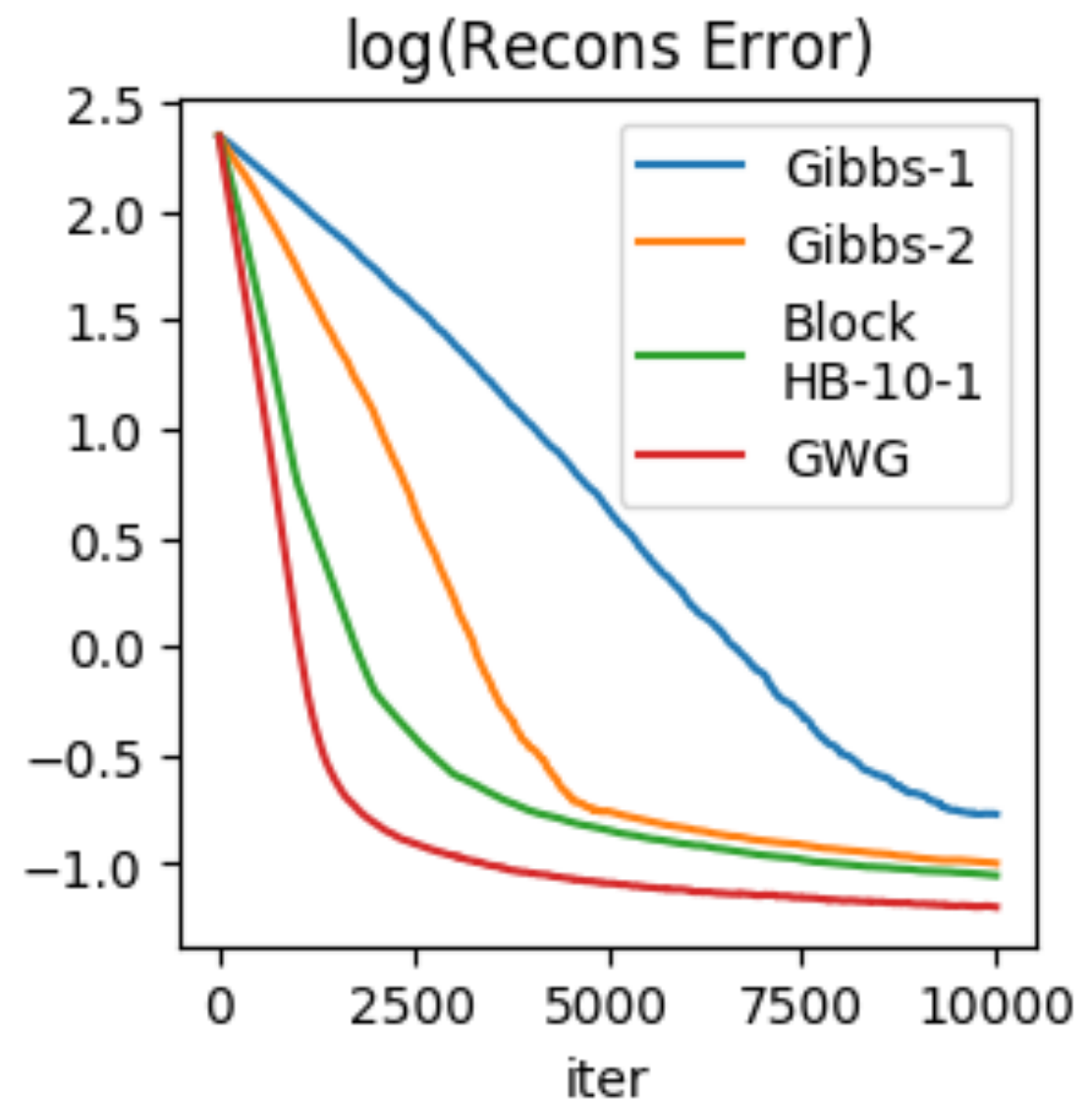
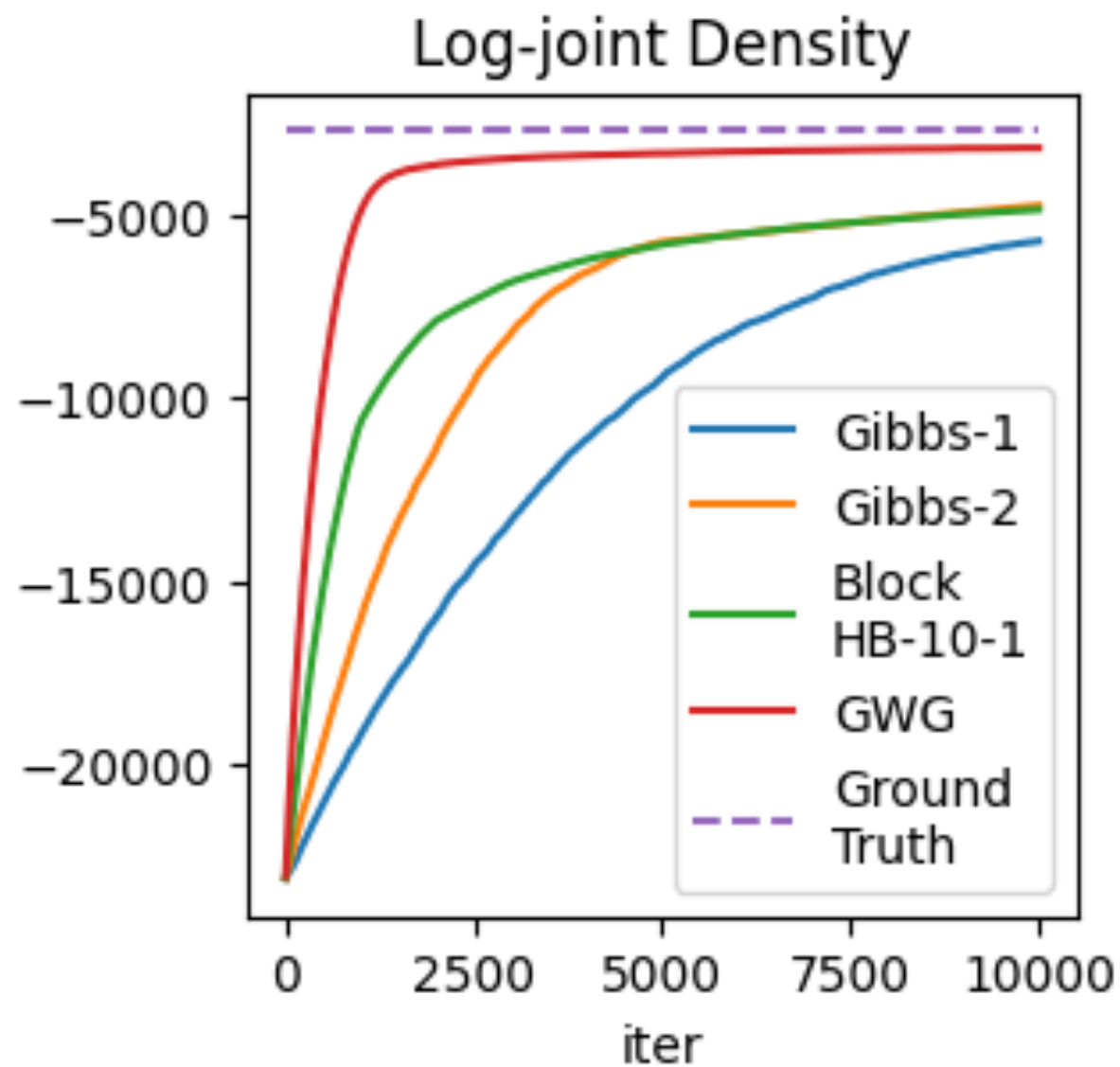
Sampling Experiments...



Sampling Experiments...



Sampling Experiments...



Relationship to Relaxations

- Ours is not the first method to use the functional form and gradients of a discrete distribution for sampling
- Use discrete target $p_d(x)$, $x \in X$ to create a related distribution $p_c(y)$ over continuous $y \in Y$, and a mapping $\Gamma(y) : Y \rightarrow X$ such that if

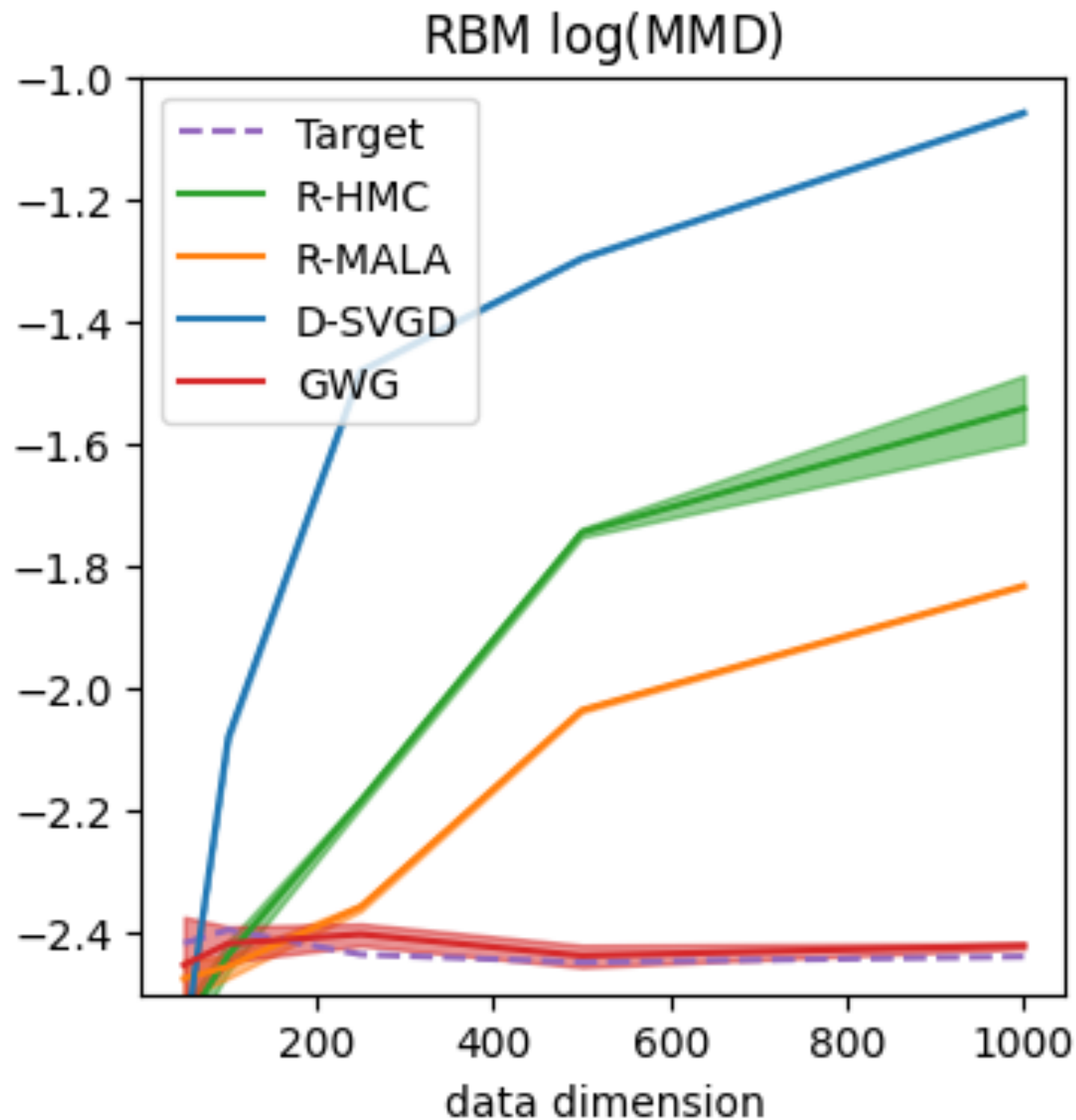
$$y \sim p_c(y) \quad x = \Gamma(y)$$

- Then we have $x \sim p_d(x)$
- Then gradient-based MCMC is used to sample from $p_c(y)$
- For binary x , we can let $\Gamma(y) = \text{sign}(y)$ and then

$$p_c(y) = N(0, I) p_d(\Gamma(y))$$

Unfortunately...

- These approaches do not scale to high dimensions



Block Gibbs



Gibbs-With-Gradients



D-SVGD



R-MALA



R-HMC



Unfortunately...

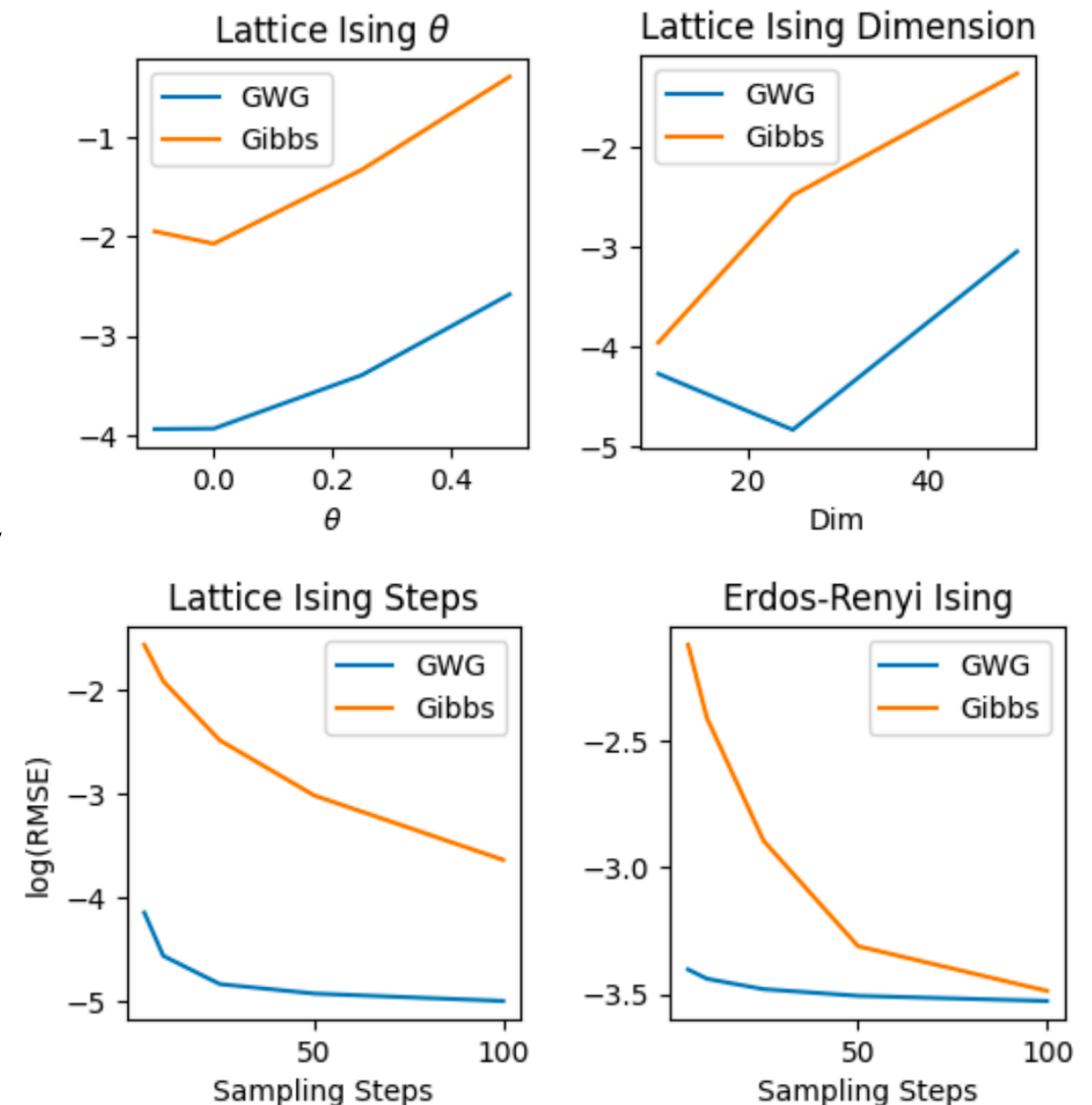
- These approaches do not scale to high dimensions
- They behave well with < 200 dimensions
- Perform poorly above
- Relaxation can *technically* use gradient-based MCMC but
 - Can be arbitrarily multi-modal
 - Approximately discontinuous
 - Metrics of mixing and convergence in y -space are meaningless in x -space
 - Infinite possibilities for relaxation, choices matter
 - Lots of hyper-parameters: base distribution, temperature, length-scale, ...

Training EBMs

- **Recall** $\nabla_{\theta} \log p(x) = -E_{\theta}(x) + \mathbf{E}_{p_{\theta}(x)}[E_{\theta}(x)]$
- **So MCMC sampling can enable parameter inference for EBMs**
- **We train two traditional EBMs**
 - **Ising model:** $E_{\theta}(x) = x^T W x + b^T x$, where $\theta = \{W, b\}$
 - **Potts model:** $E_{\theta}(x) = \sum_{i=1}^D h_i^T x_i + \sum_{ij} x_j^T J_{ij} x_j$, where $\theta = \{J, h\}$

Training EBM

- Ising model: $E_{\theta}(x) = x^T W x + b^T x$, where $\theta = \{W, b\}$
- We generate Ising models where W is the adjacency of:
 - 2D Lattice
 - Random Erdos-Renyi Graph
- Compared to Gibbs sampling
 - GWG finds better solutions
 - Does so much more efficiently



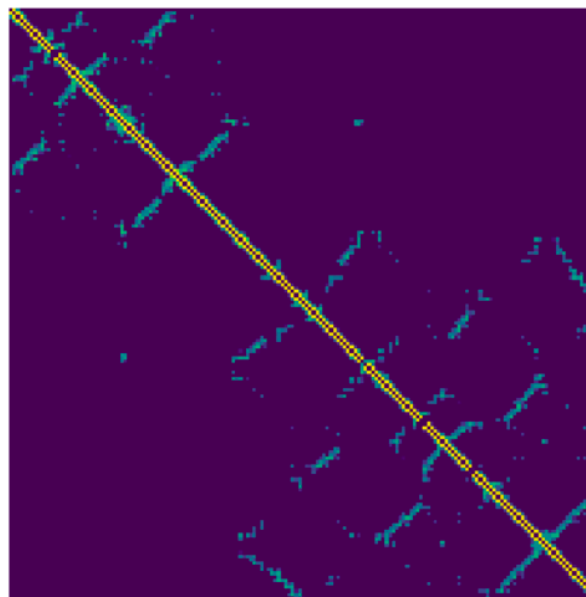
Training EBM

- **Potts model:** $E_{\theta}(x) = \sum_{i=1}^D h_i^T x_i + \sum_{ij} x_j^T J_{ij} x_j$, where $\theta = \{J, h\}$
- **Train on evolutionary protein data** $x \in \{0, \dots, 20\}^D$
- **Potts model likelihood is sum of pairwise interactions** $x_j^T J_{ij} x_j$
- **Strength of inferred interactions can predict 3D structure of protein**
- **Normally trained with pseudo-likelihood**

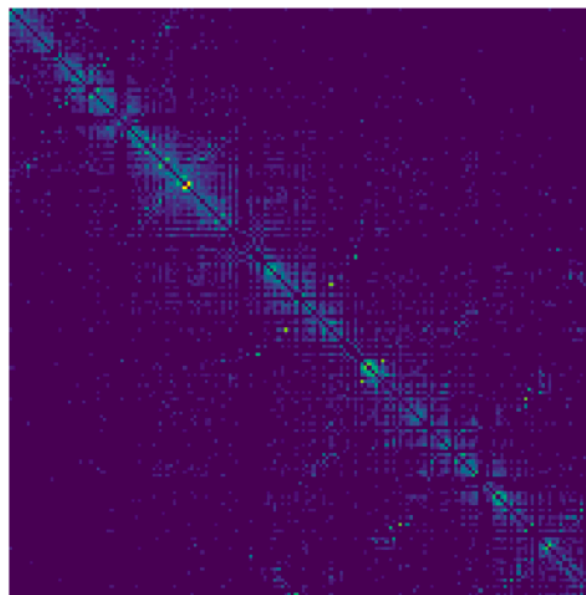
Training EBM

- Potts model: $E_{\theta}(x) = \sum_{i=1}^D h_i^T x_i + \sum_{ij} x_j^T J_{ij} x_j$, where $\theta = \{J, h\}$

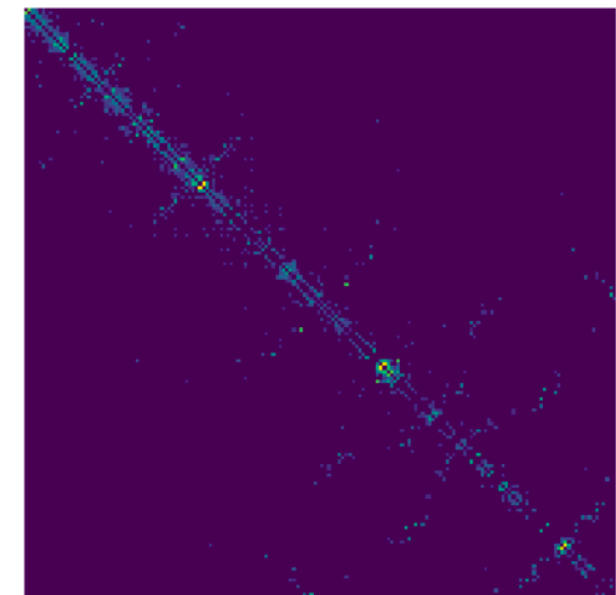
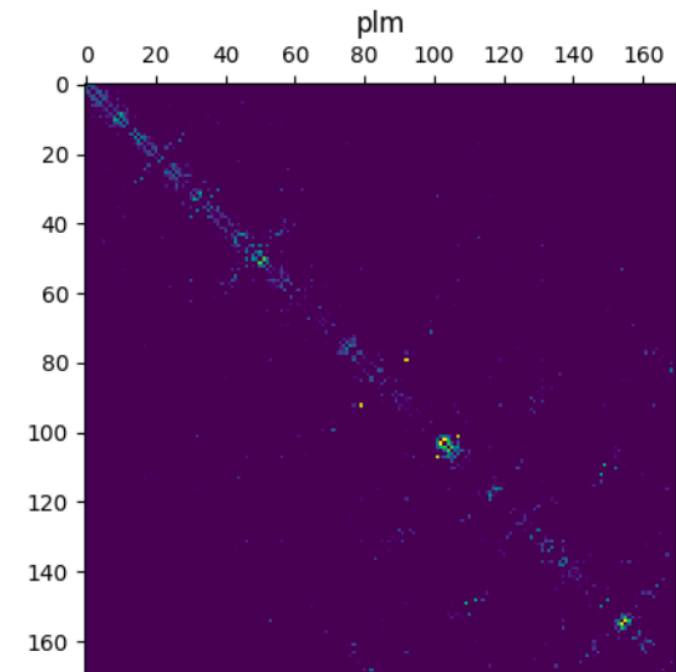
Ground Truth



gibbs

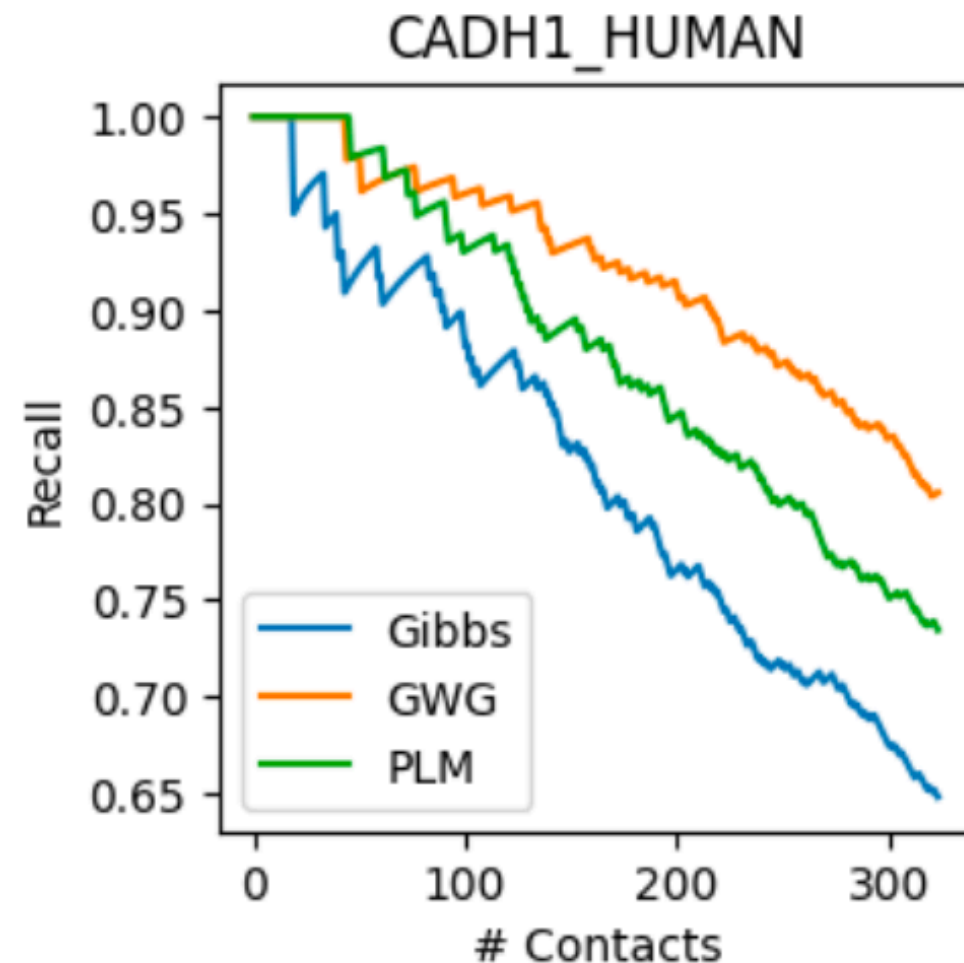
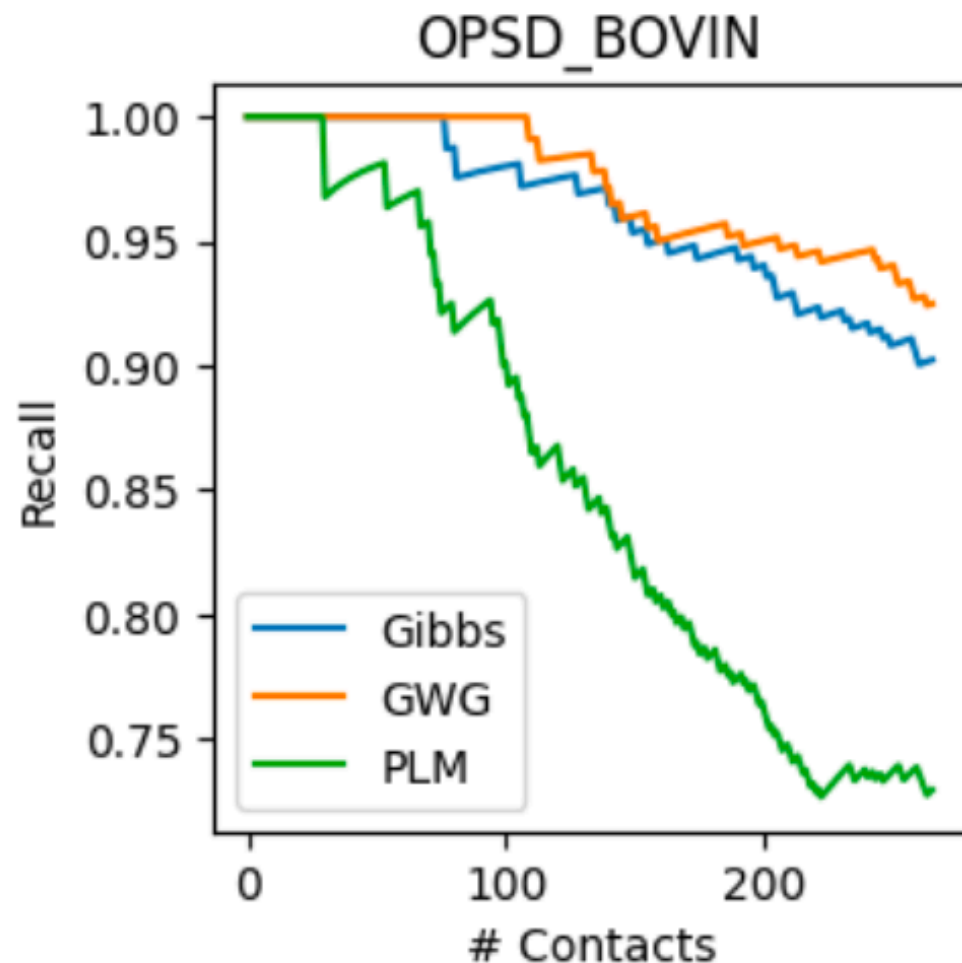


gwg



Training EBM

- Potts model: $E_{\theta}(x) = \sum_{i=1}^D h_i^T x_i + \sum_{ij} x_j^T J_{ij} x_j$, where $\theta = \{J, h\}$
- Contact prediction



Deep EBMs for Discrete Data

- Recent success in EBMs comes from powerful, expressive energy functions $p_{\theta}(x) = \frac{e^{f_{\theta}(x)}}{Z}$
- For high dimensional data, Gibbs is too slow to converge
- For k -outcome categorical data we must evaluate $f_{\theta}(x)$ k -times vs 2 for GWG
 - For large k GWG is much faster per iterations AND converges in fewer iterations
- We train Deep ResNet EBMs on binary and categorical image data
- Binary pixel values are 0, 1
- For categorical we take continuous pixel values and treat as 1-of-256 way categorical
 - This means 256 function evals for 1 step of Gibbs

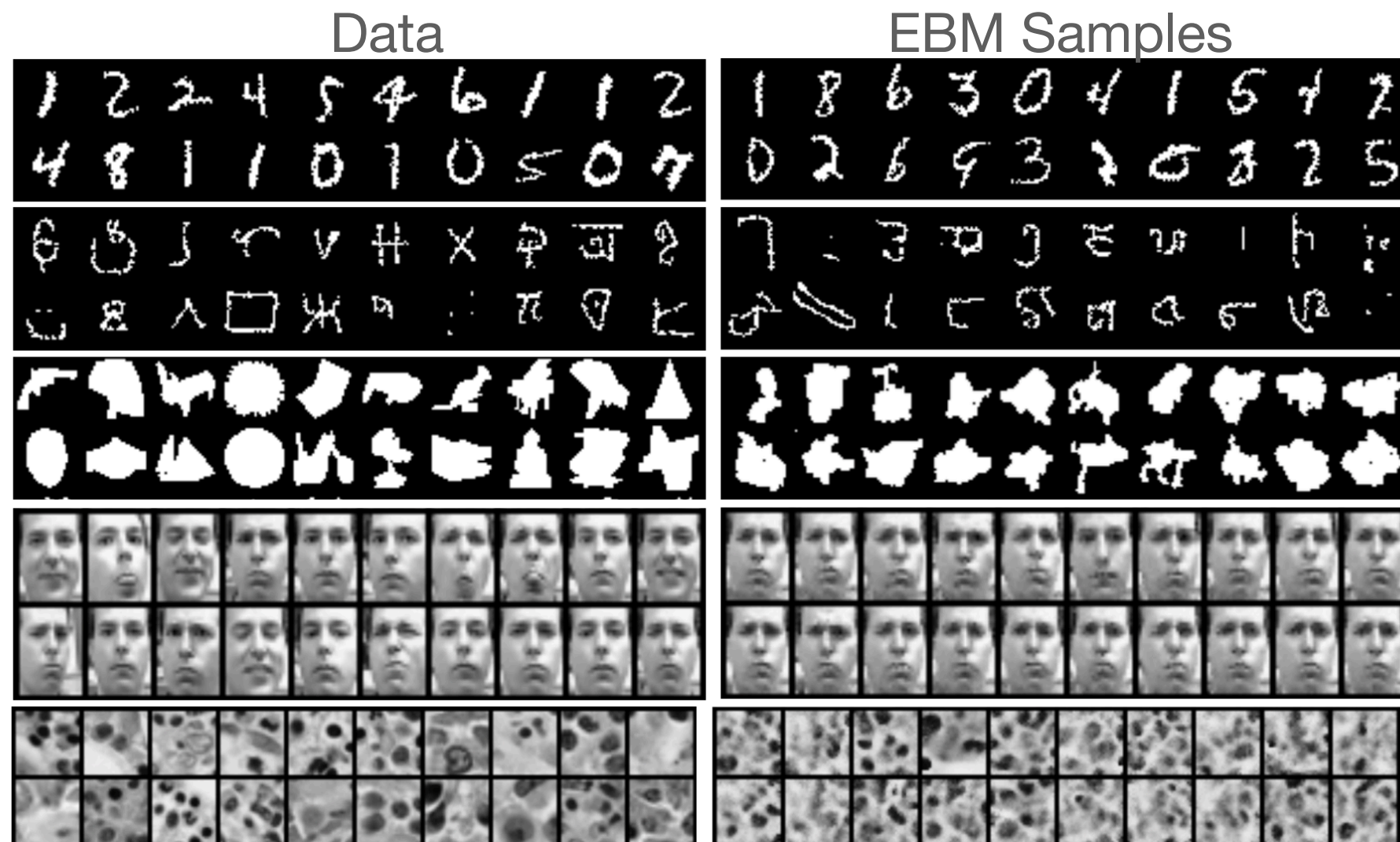
Deep EBMs for Discrete Data

- Train with PCD
- Outperforms VAEs, RBM, and Deep belief net in log-likelihood
- GWG greatly outperforms Gibbs on binary data and Gibbs is completely unable to train because of high cost per-iteration

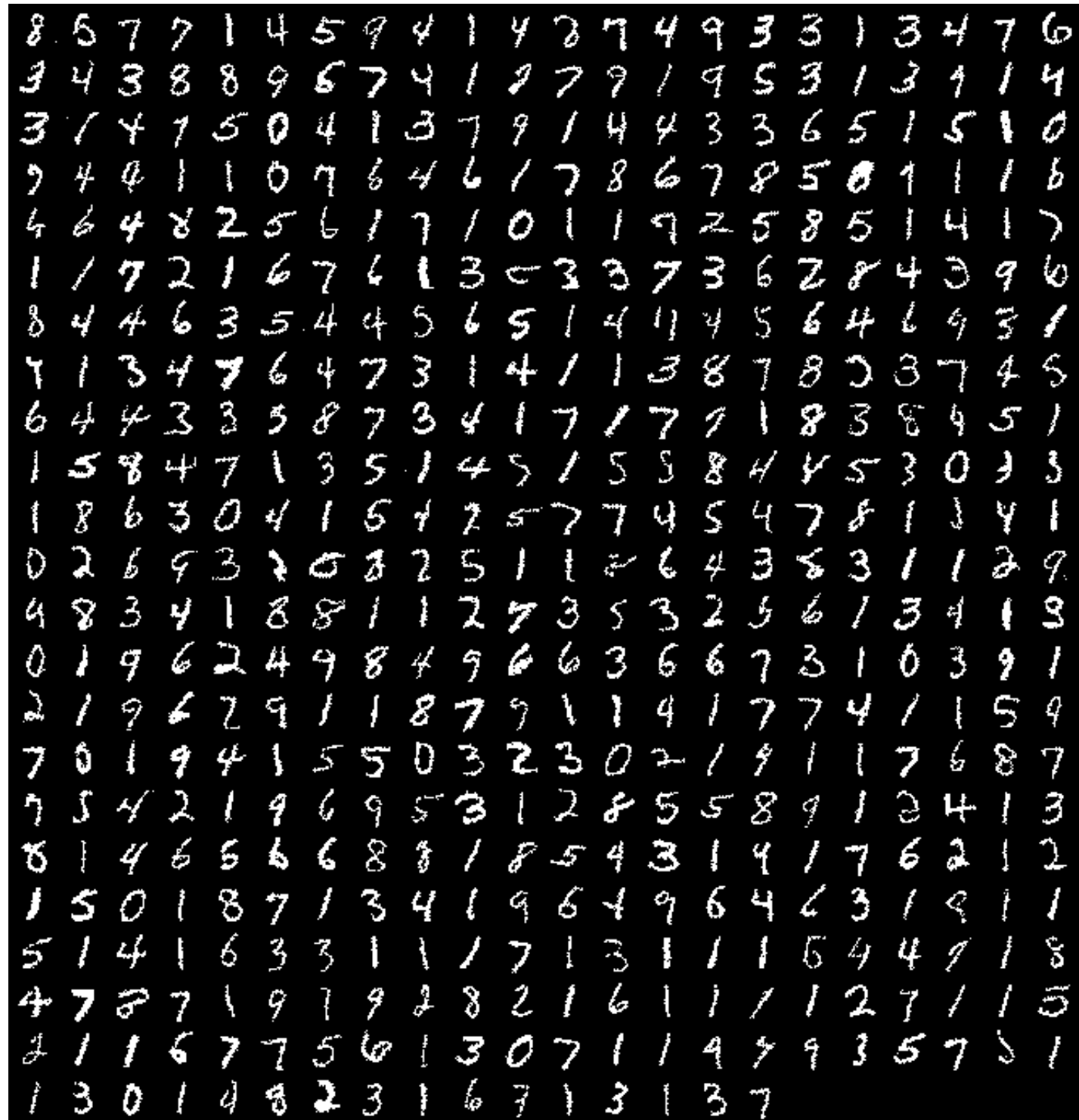
Data Type	Dataset	VAE (MLP)	VAE (Conv)	EBM (GWG)	EBM (Gibbs)	RBM	DBN
Binary (log-likelihood \uparrow)	Static MNIST	-86.05	-82.41	-80.01	-117.17	-86.39	-85.67
	Dynamic MNIST	-82.42	-80.40	-80.51	-121.19	—	—
	Omniglot	-103.52	-97.65	-94.72	-142.06	-100.47	-100.78
	Caltech Silhouettes	-112.08	-106.35	-96.20	-163.50	—	—
Categorical (bits/dim \downarrow)	Frey Faces	4.61	4.49	4.65	—	—	—
	Histopathology	5.82	5.59	5.08	—	—	—

Deep EBMs for Discrete Data

- Train with PCD
- Outperforms VAEs, RBM, and Deep belief net in log-likelihood
- GWG greatly outperforms Gibbs on binary data and Gibbs is completely unable to train because of high cost per-iteration



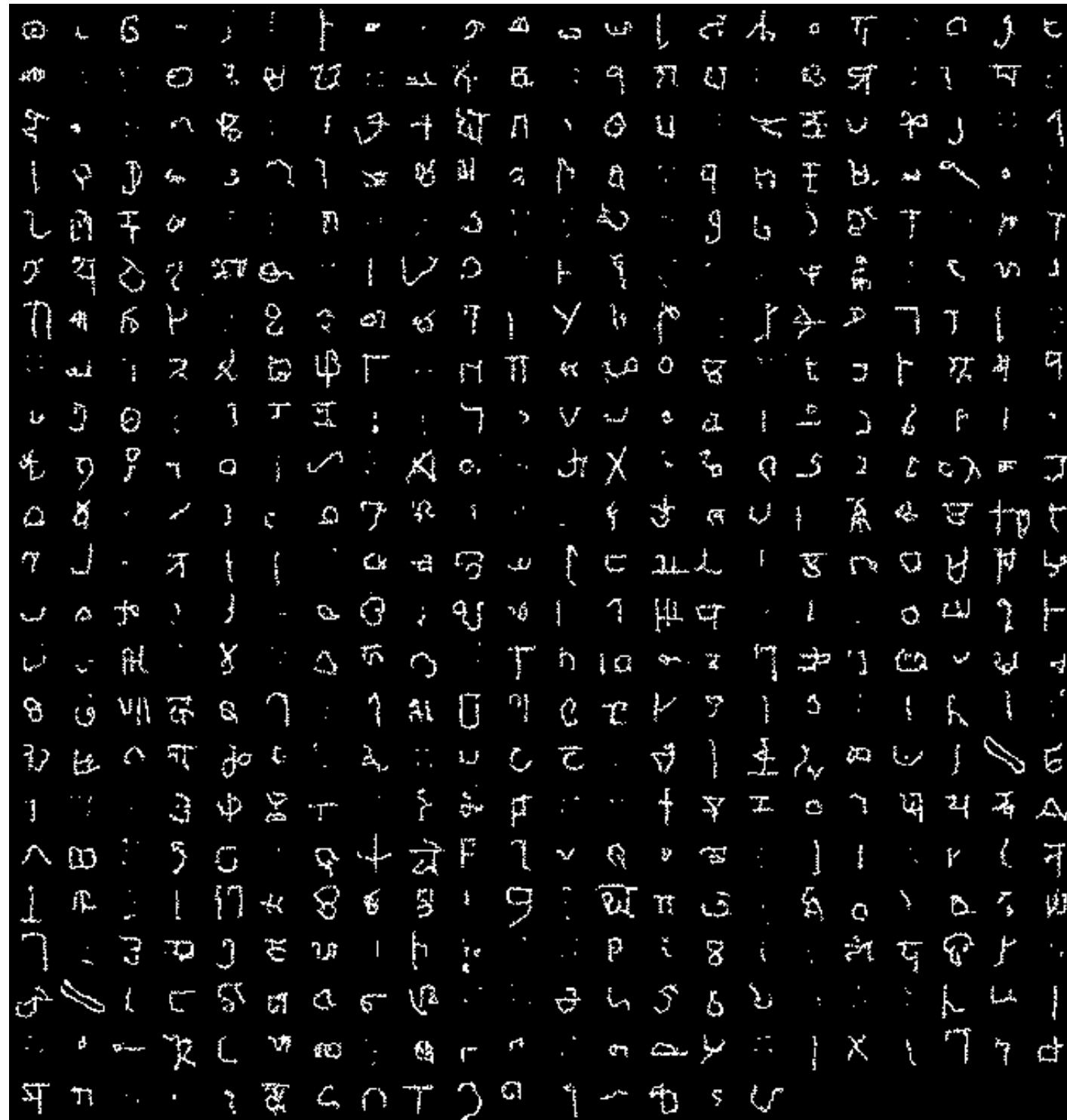
Deep EBMs for Discrete Data



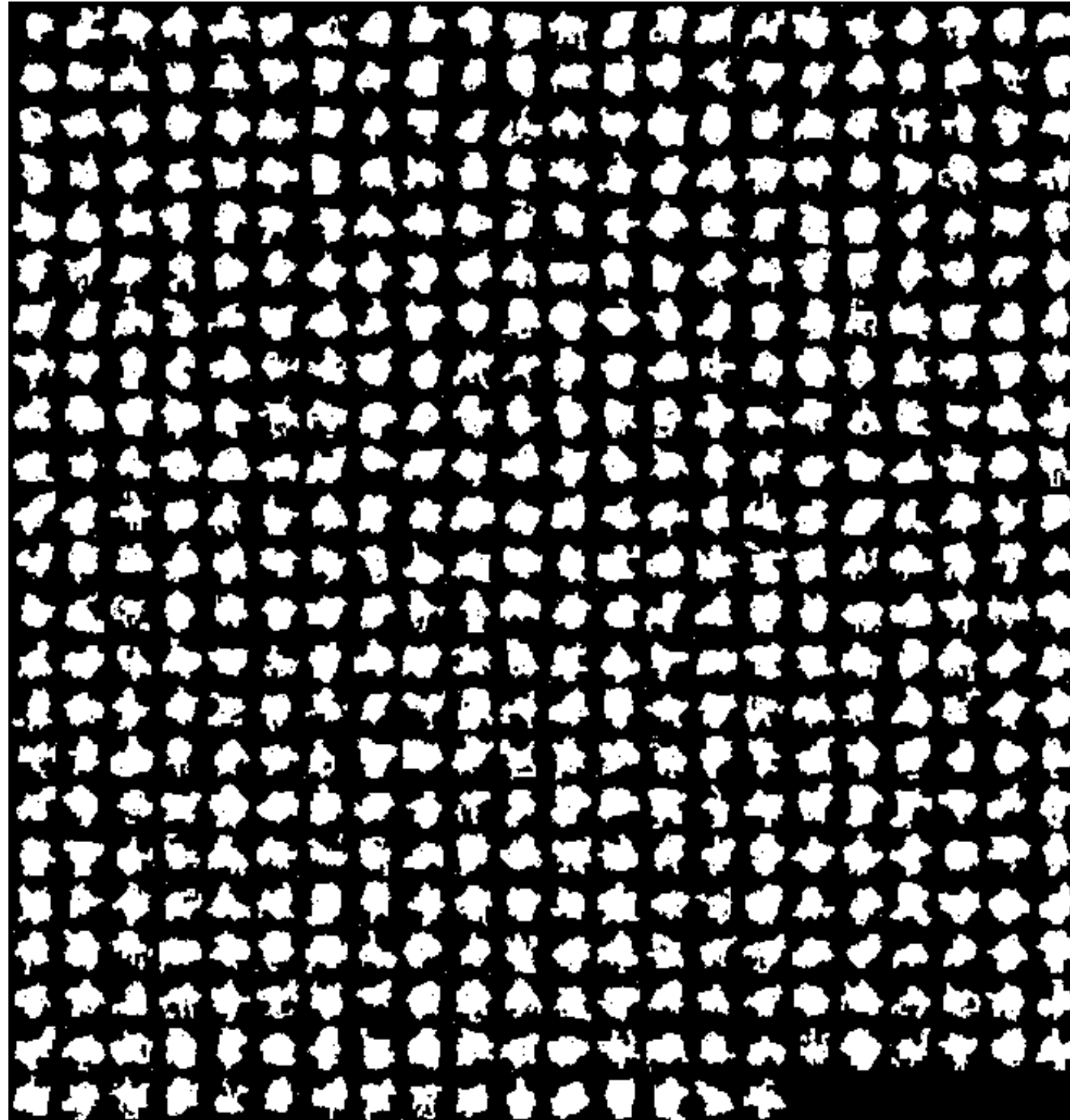
A 20x20 grid of handwritten digits, resembling a noisy or corrupted version of the MNIST dataset, used for Deep EBMs. The digits are displayed in a black, slightly blurred font on a white background. The grid contains the following sequence of digits (row by row):

8	6	7	7	1	4	5	9	4	1	4	2	7	4	9	3	3	1	3	4	7	6
3	4	3	8	8	9	6	7	4	1	2	7	9	1	9	5	3	1	3	1	1	4
3	1	4	7	5	0	4	1	3	7	9	1	4	4	3	3	6	5	1	5	1	0
9	4	4	1	1	0	7	6	4	6	1	7	8	6	7	8	5	0	1	1	1	6
4	6	4	8	2	5	6	1	7	1	0	1	1	9	2	5	8	5	1	4	1	7
1	1	7	2	1	6	7	6	1	3	5	3	3	7	3	6	2	8	4	3	9	6
8	4	4	6	3	5	4	4	5	6	5	1	4	1	4	5	6	4	6	4	3	1
7	1	3	4	7	6	4	7	3	1	4	1	1	3	8	7	8	3	3	7	4	5
6	4	4	3	3	5	8	7	3	4	1	7	1	7	7	1	8	3	8	4	5	1
1	5	8	4	7	1	3	5	1	4	5	1	5	5	8	4	4	5	3	0	3	3
1	8	6	3	0	4	1	5	4	2	5	7	7	4	5	4	7	8	1	3	4	1
0	2	6	9	3	1	0	8	2	5	1	1	2	6	4	3	8	3	1	1	2	9
4	8	3	4	1	8	8	1	1	2	7	3	5	3	2	5	6	1	3	4	1	8
0	1	9	6	2	4	9	8	4	9	6	6	3	6	6	7	3	1	0	3	9	1
2	1	9	6	2	9	1	1	8	7	9	1	1	4	1	7	7	4	1	1	5	9
7	0	1	9	4	1	5	5	0	3	2	3	0	2	1	9	1	1	7	6	8	7
7	5	4	2	1	9	6	9	5	3	1	2	8	5	5	8	9	1	2	4	1	3
8	1	4	6	6	6	6	8	8	1	8	5	4	3	1	9	1	7	6	2	1	2
1	5	0	1	8	7	1	3	4	1	9	6	4	9	6	4	6	3	1	9	1	1
5	1	4	1	6	3	3	1	1	1	7	1	3	1	1	1	5	4	4	7	1	8
4	7	2	7	1	9	1	9	2	8	2	1	6	1	1	1	1	2	7	1	1	5
2	1	1	6	7	7	5	6	1	3	0	7	1	1	4	7	9	3	5	7	3	1
1	3	0	1	4	8	2	3	1	6	7	1	3	1	3	7						

Deep EBMs for Discrete Data



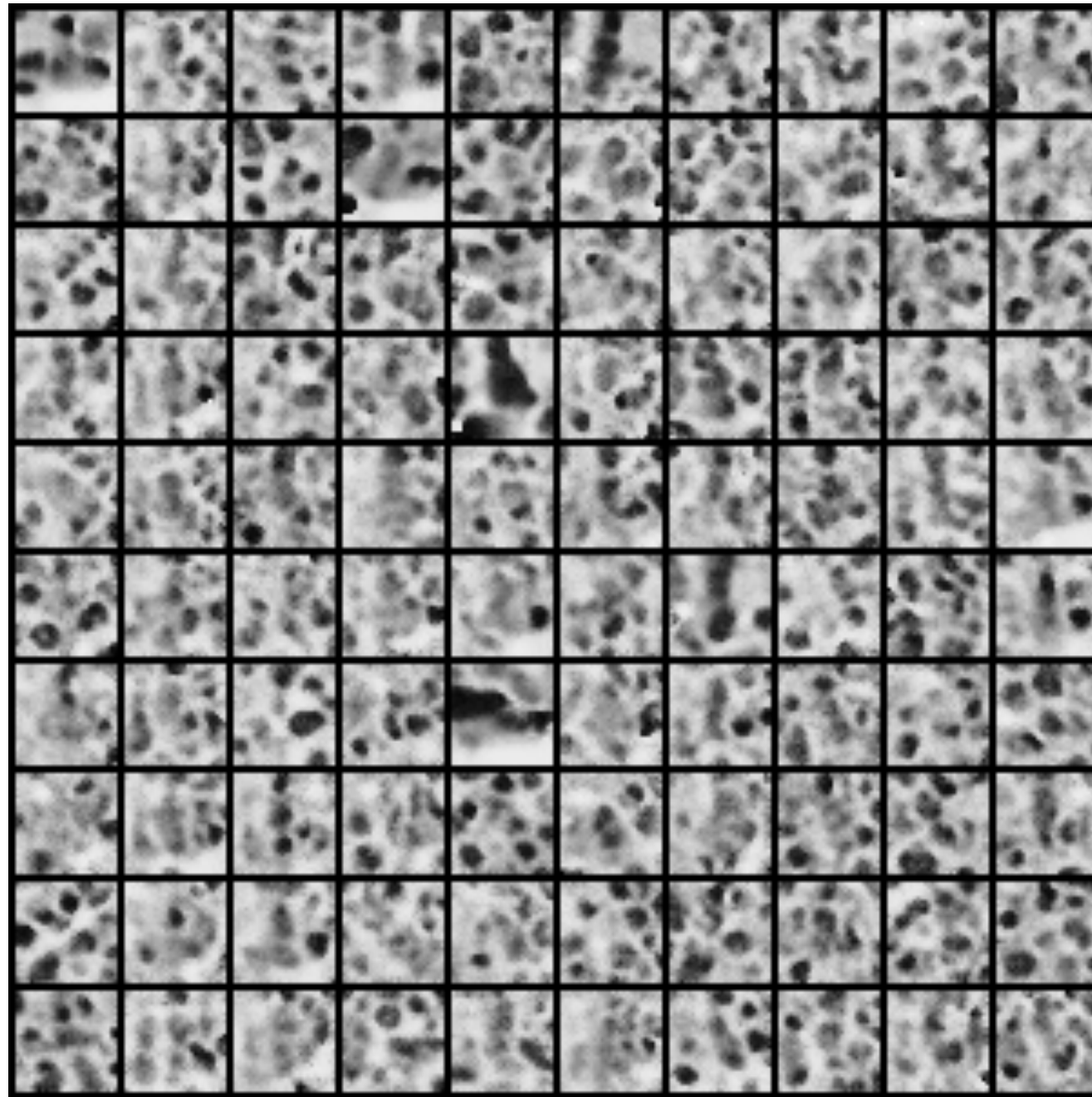
Deep EBMs for Discrete Data



Deep EBMs for Discrete Data



Deep EBMs for Discrete Data

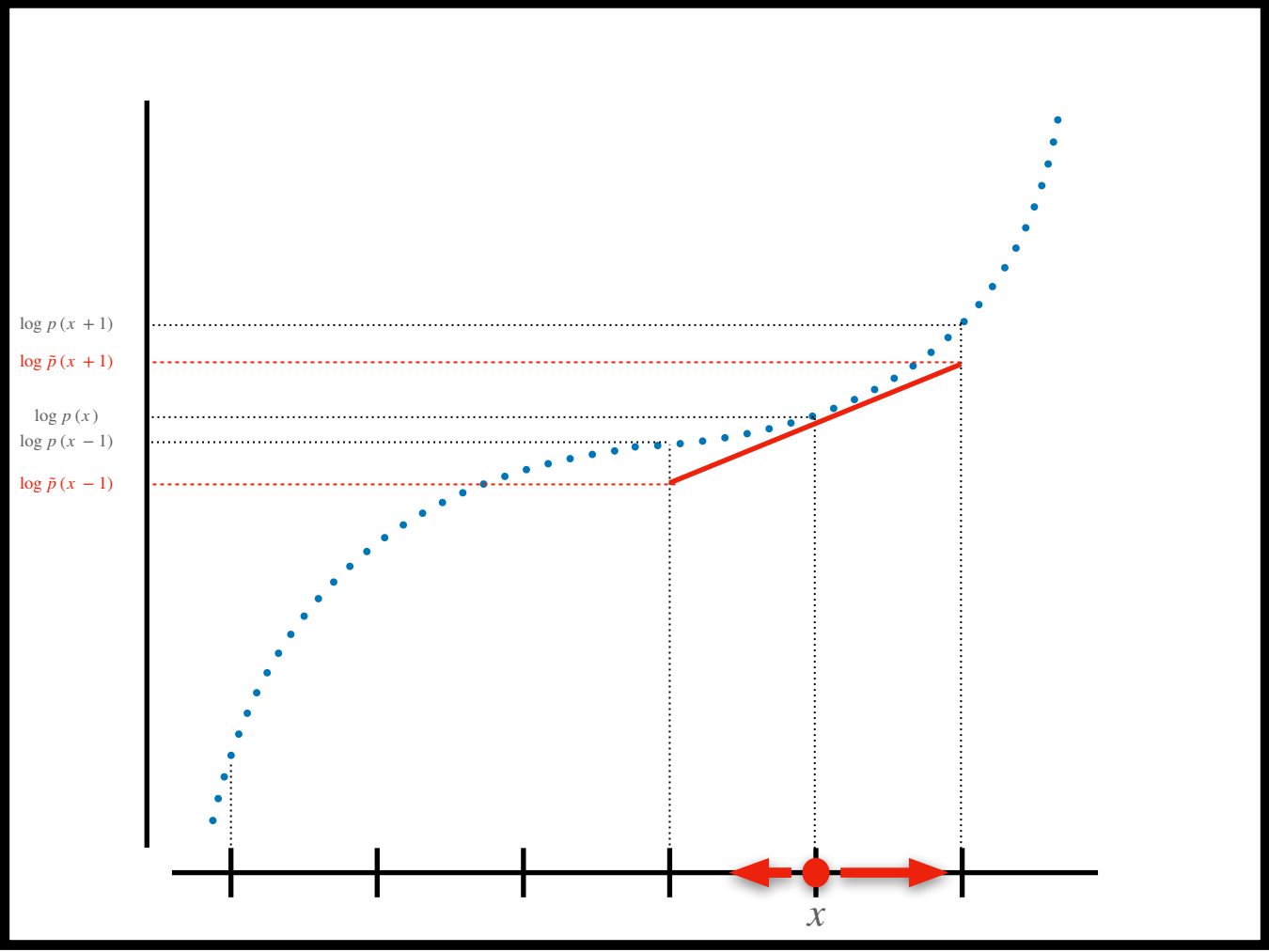


Deep EBMs for text?

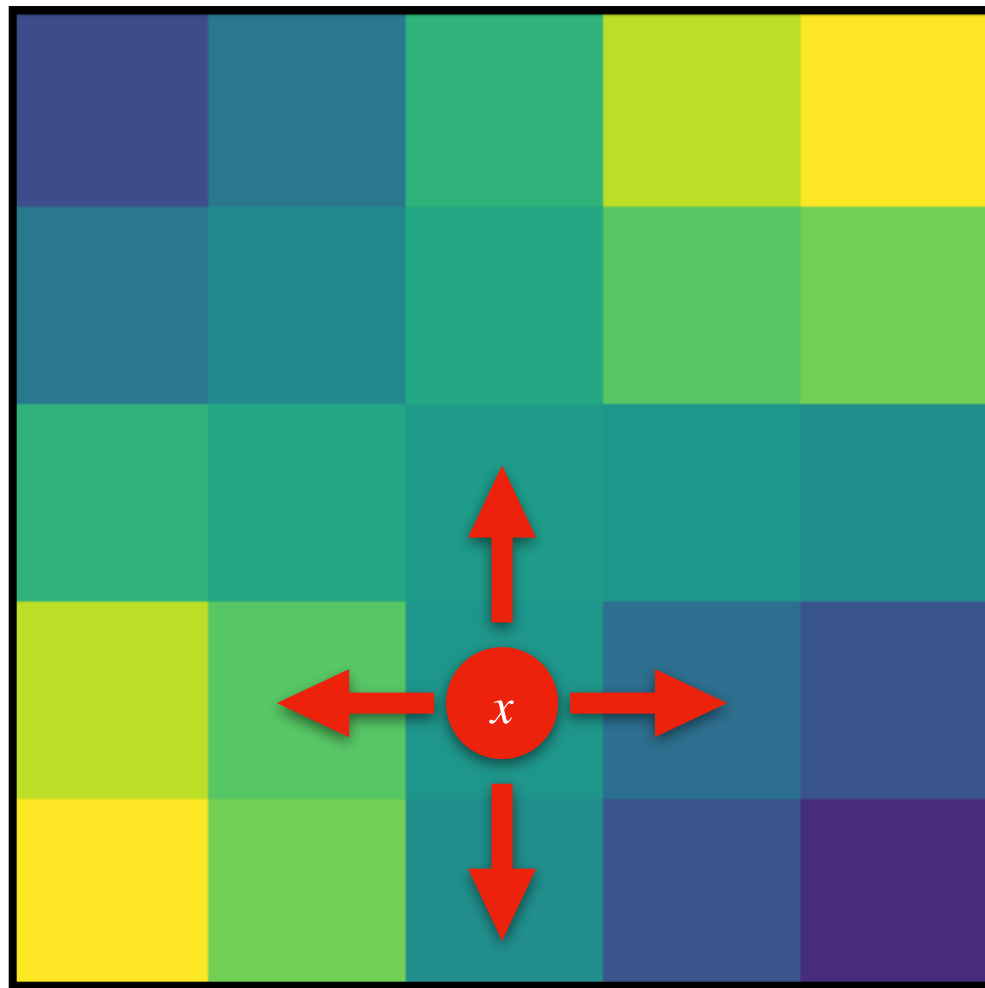
- A more extreme categorical domain is text
- Most language models today have $> 10,000$ words in their vocabulary
- HUGE categorical sampling space
- Standard Gibbs would need over 10,000 function evals per step
- Can GWG enable text EBMs?
- We train a model on short sentences with 10k vocab
 - 20 words x 10k vocab = 200,000 possible moves for GWG
- Autoregressive model gets -74 test set log-likelihood
- EBM gets -77.14
- Uniform is -184.18 and categorical distribution gets -100.05
- GWG is competitive with AR despite HUGE sampling space

Next Steps

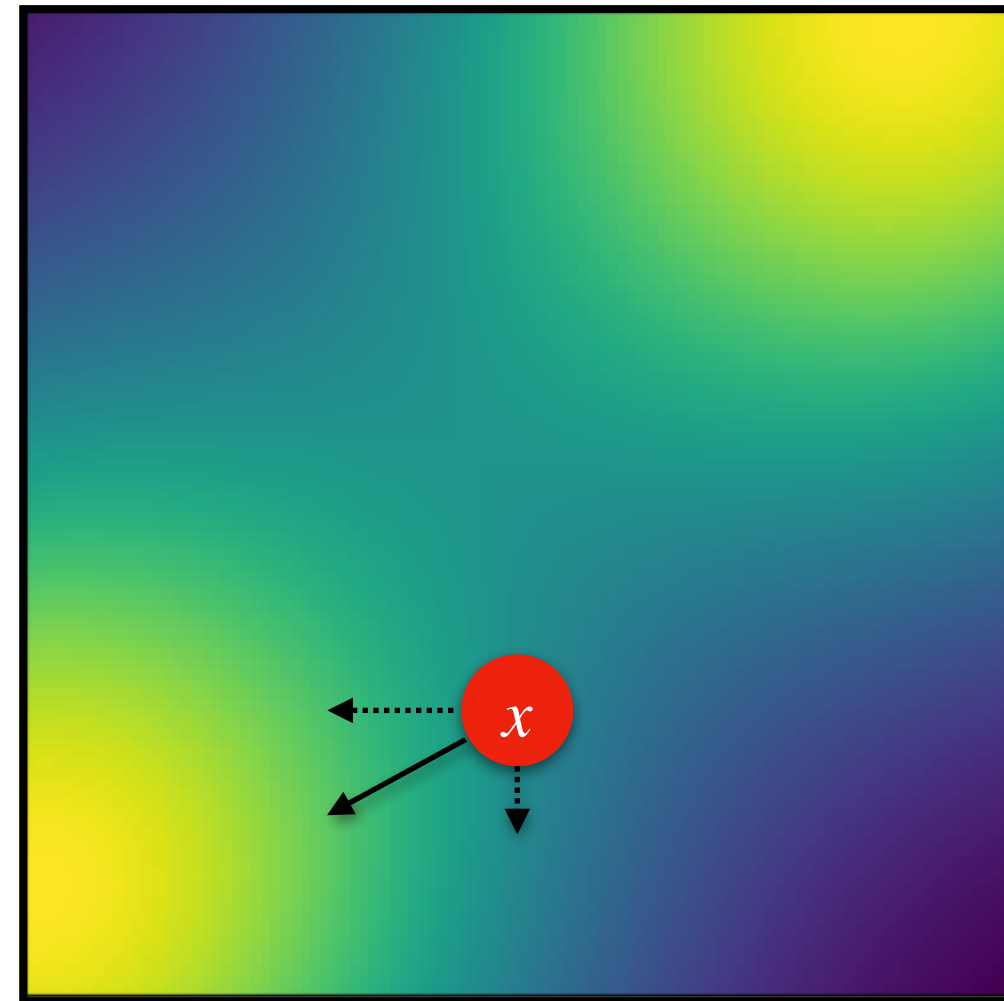
- Improvements and new approximations for large categoricals
- New approximations when gradients can't be computed (graphs, programs)
- New applications of gradient-based approximations
 - Discrete Score Matching
 - Discrete Stein Discrepancies
- Extend to larger window sizes and change multiple dimensions at a time
- Incorporating momentum
- Integrate into probabilistic programming frameworks



Target Distribution



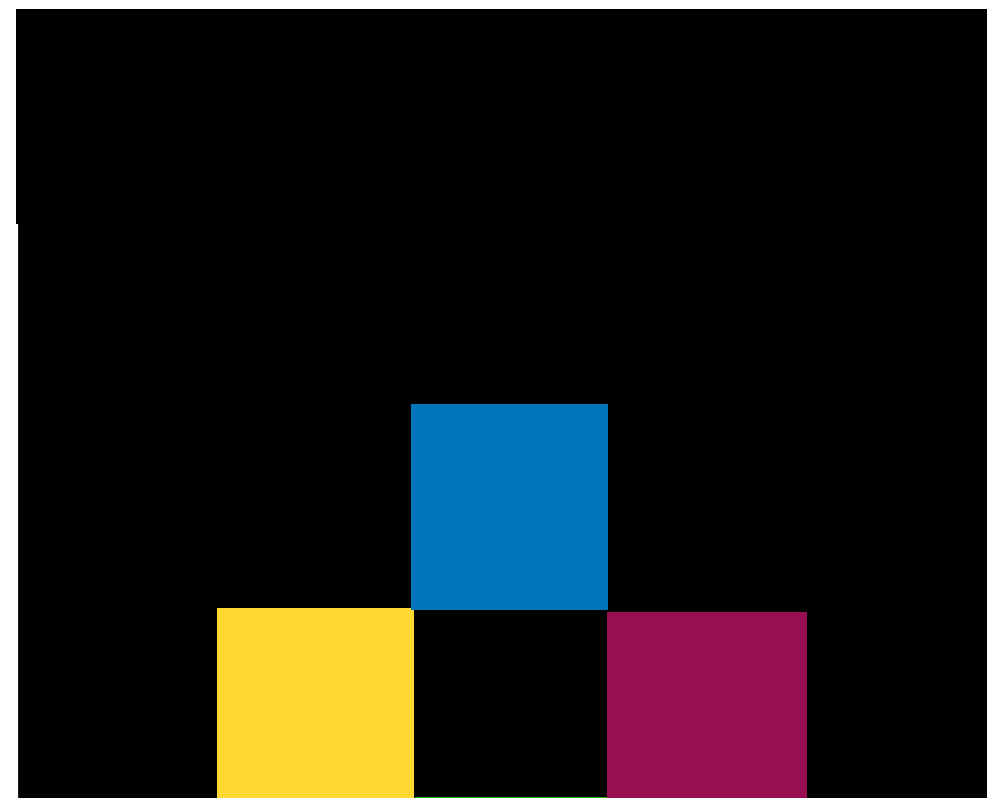
Underlying Continuous Function



Compute gradients of
Continuous function



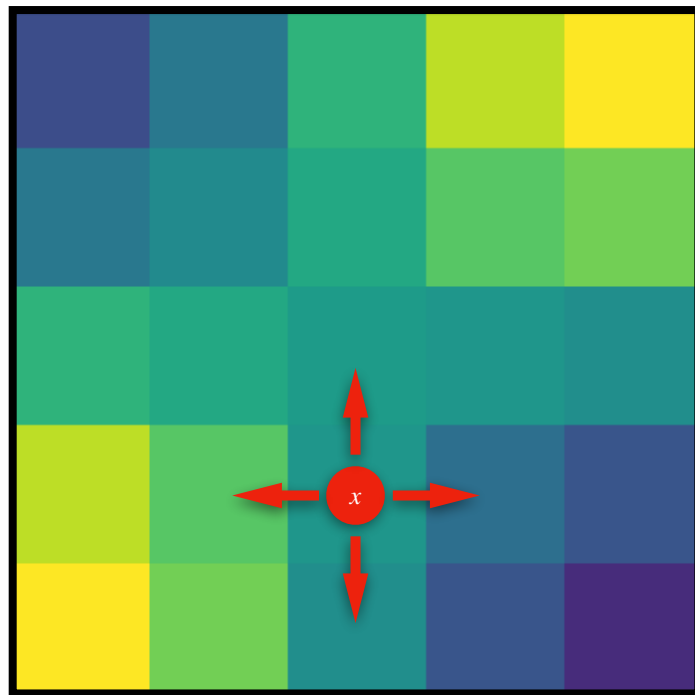
Proposal



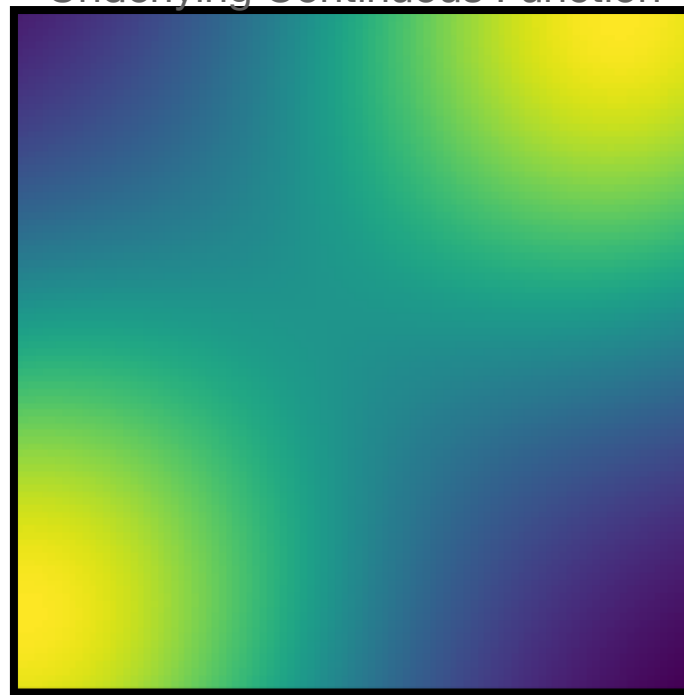
Use estimated likelihood
ratios to parameterize proposal
In original discrete space



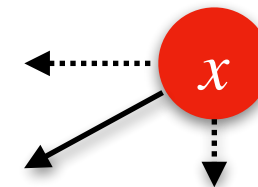
Target Distribution



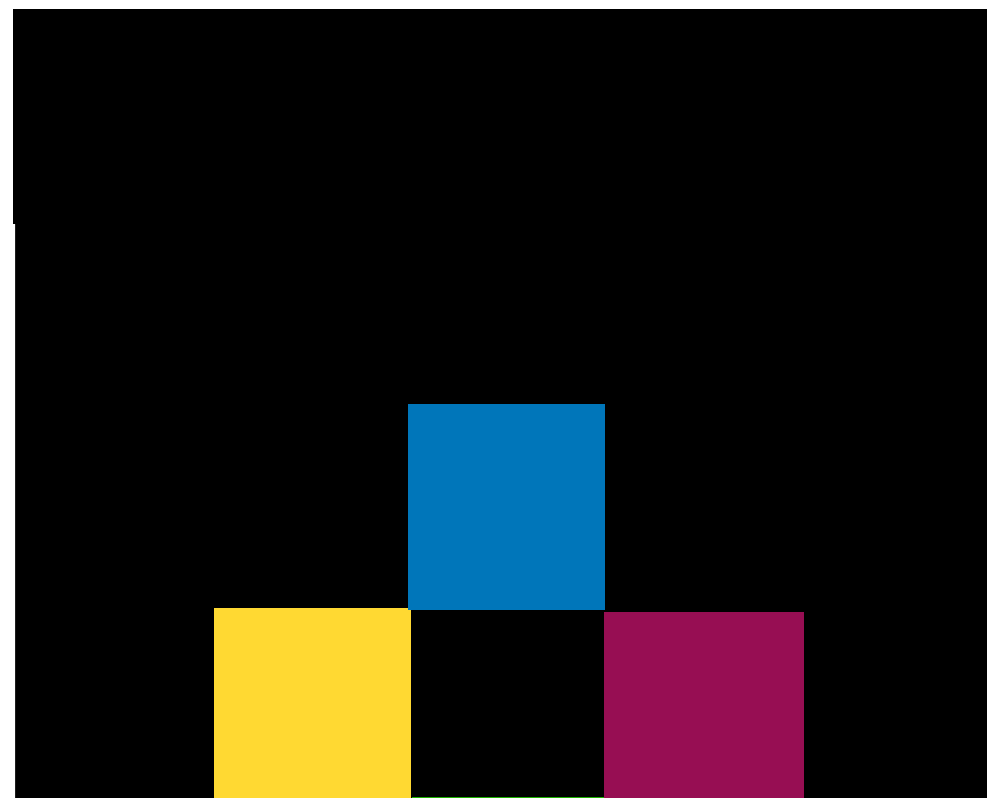
Underlying Continuous Function



Compute gradients of
Continuous function



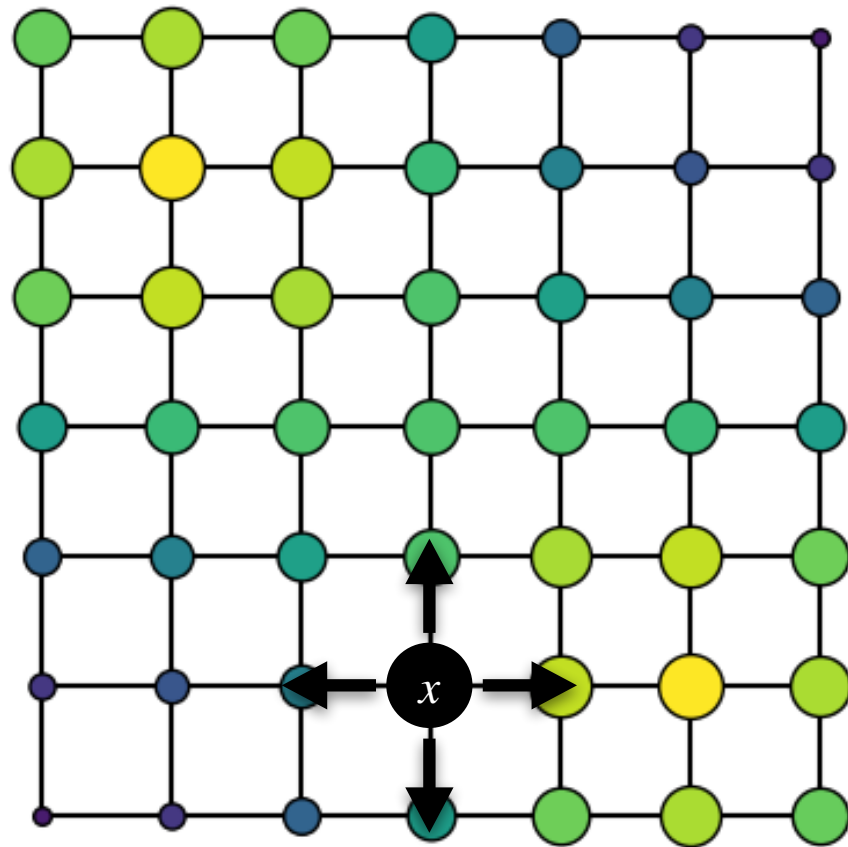
Proposal



Use estimated likelihood
ratios to parameterize proposal
In original discrete space



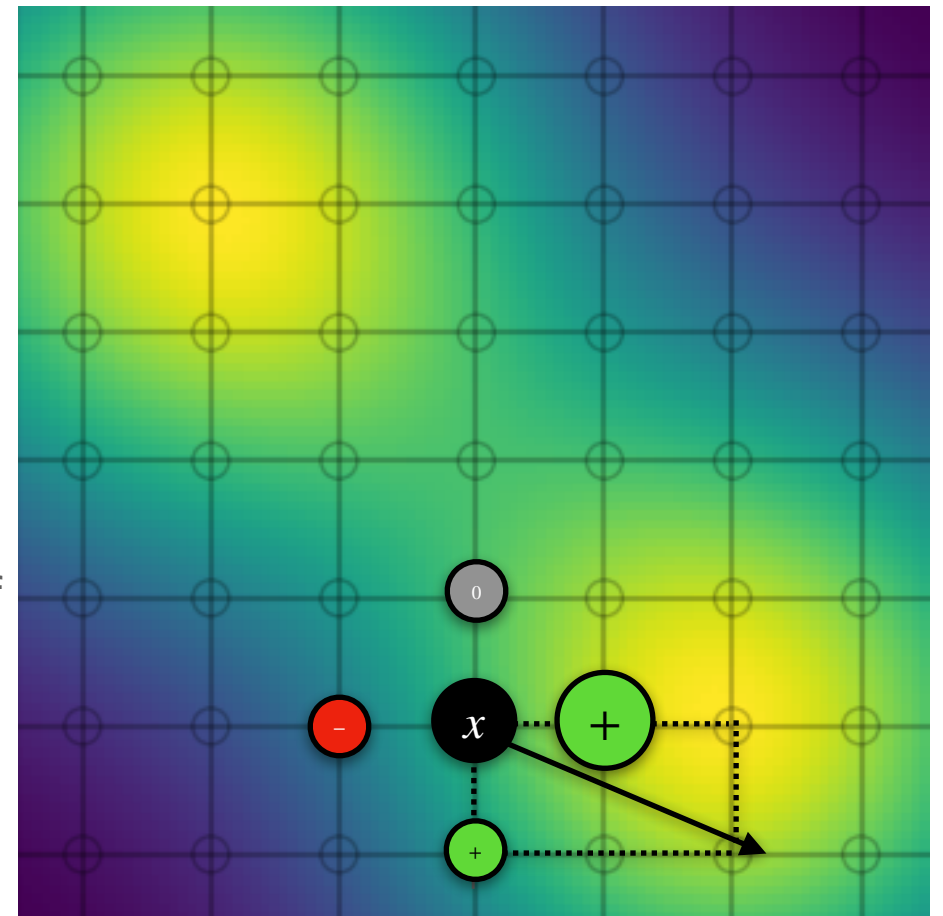
Target Distribution



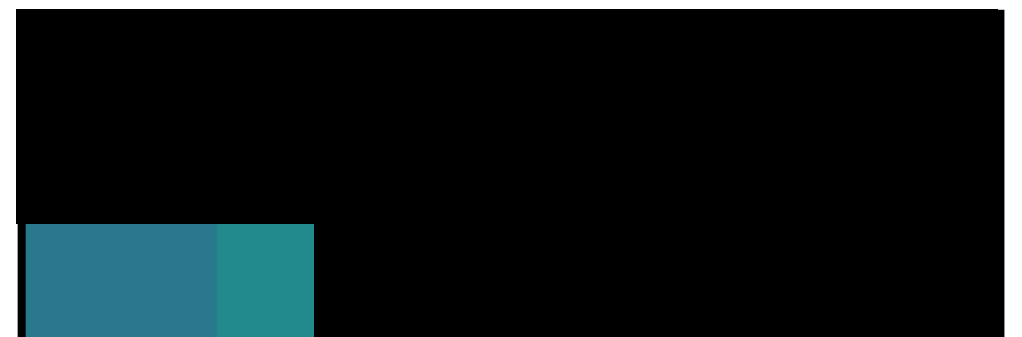
Compute gradients of
Continuous function



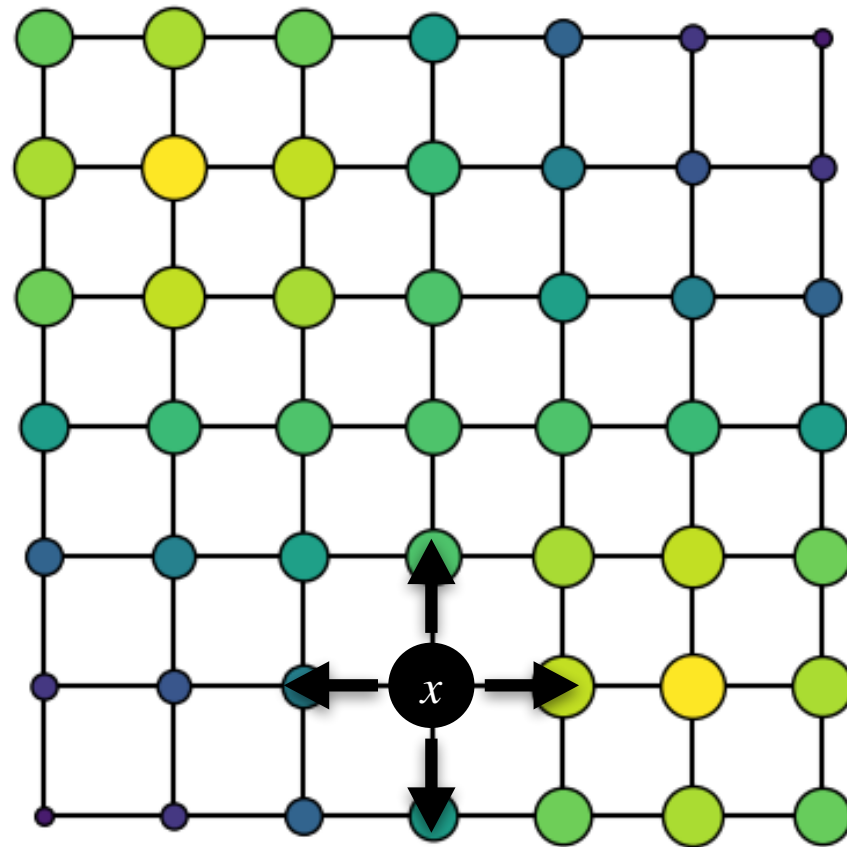
Underlying Continuous Function



Proposal



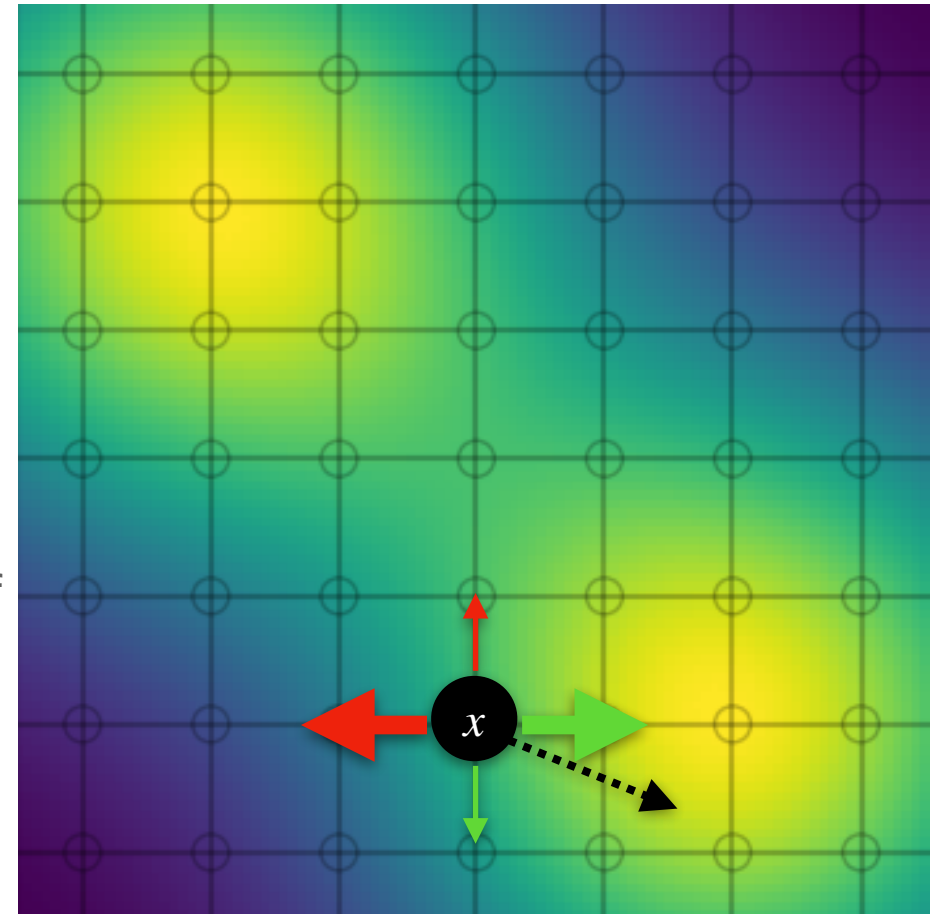
Target Distribution



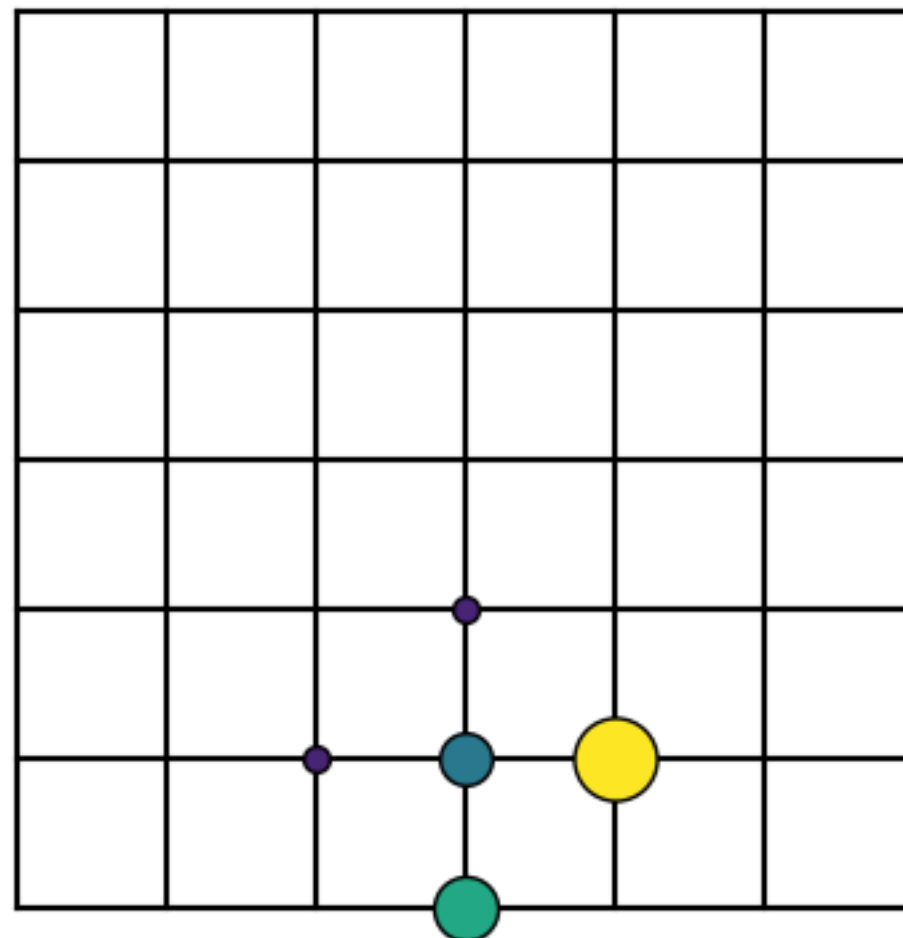
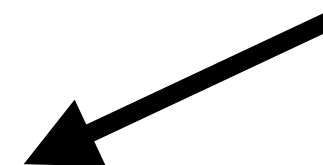
Compute gradients of continuous function



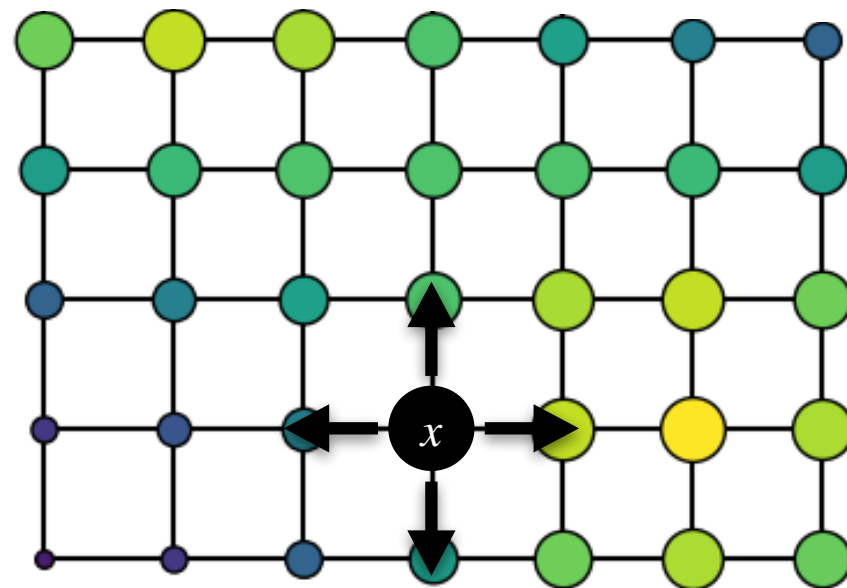
Underlying Continuous Function



Take softmax to obtain proposal in original discrete space



Target Distribution

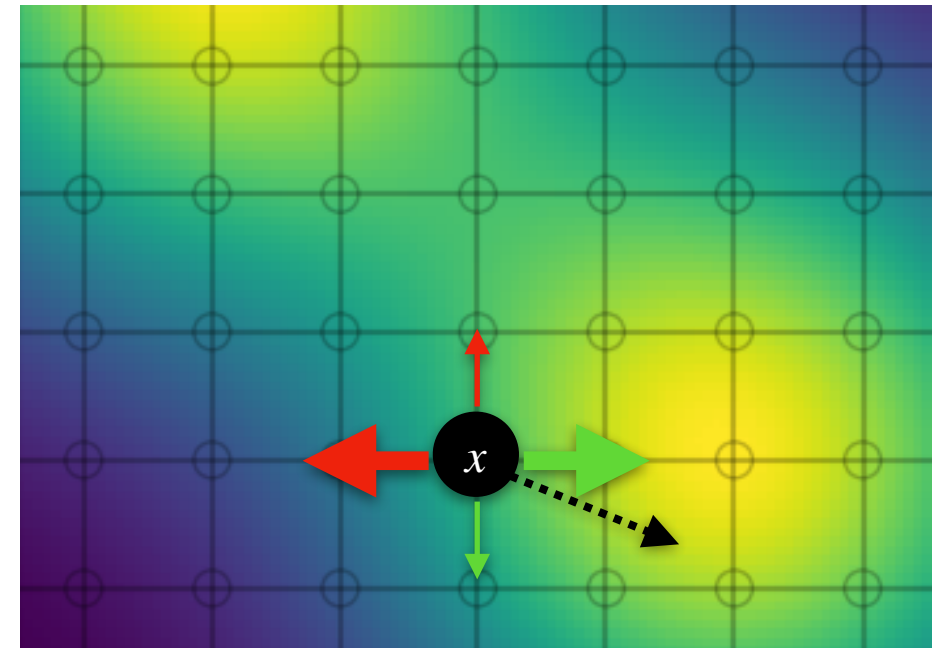


Compute gradients of
continuous function

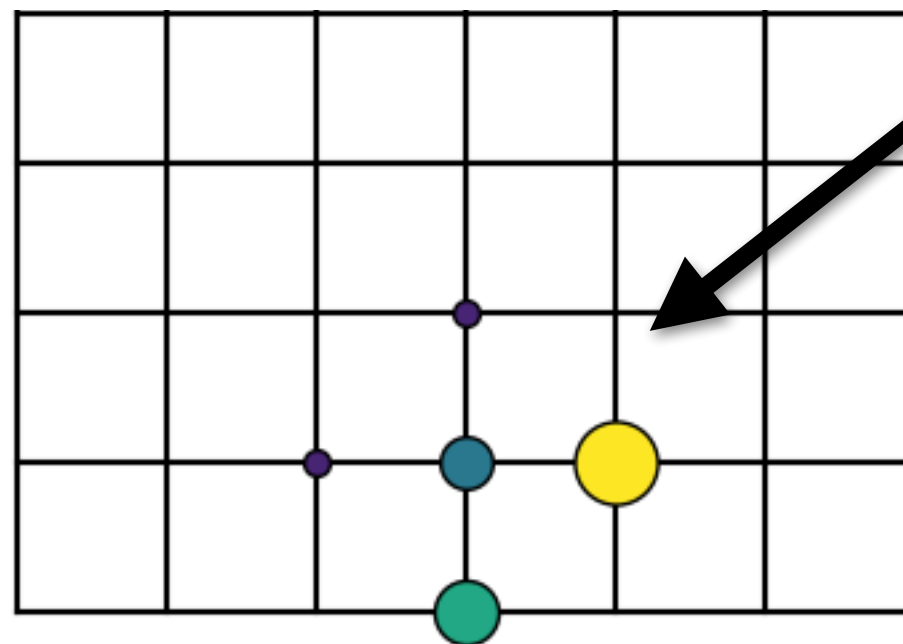


Estimate
likelihood ratios

Underlying Continuous Function

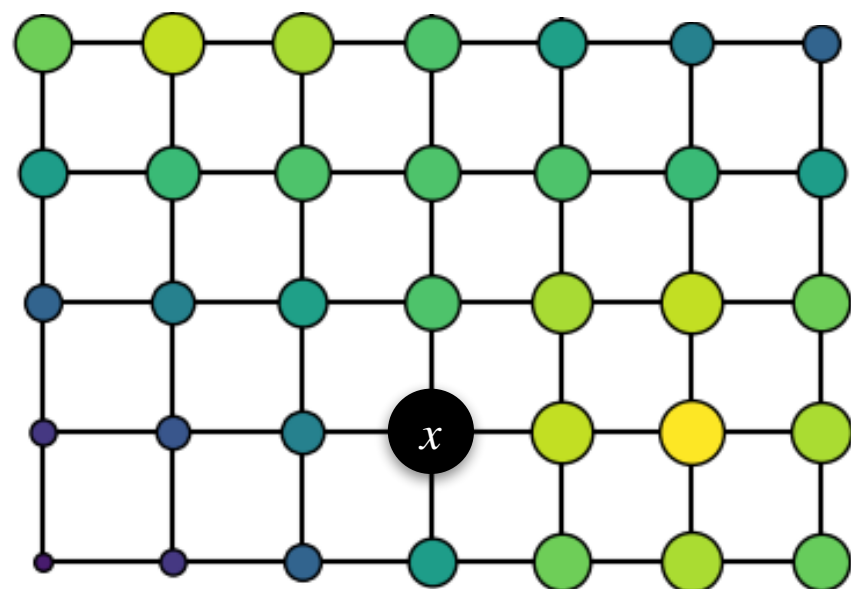


Proposal Distribution



Take softmax to obtain
proposal in original
discrete space

Target Distribution

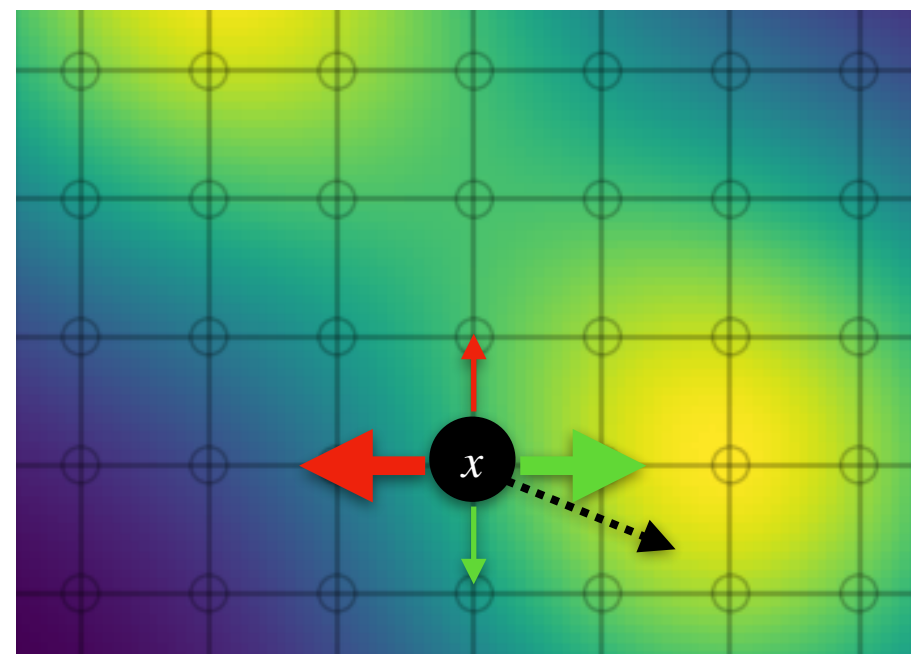


Compute gradients of continuous function



Estimate likelihood ratios

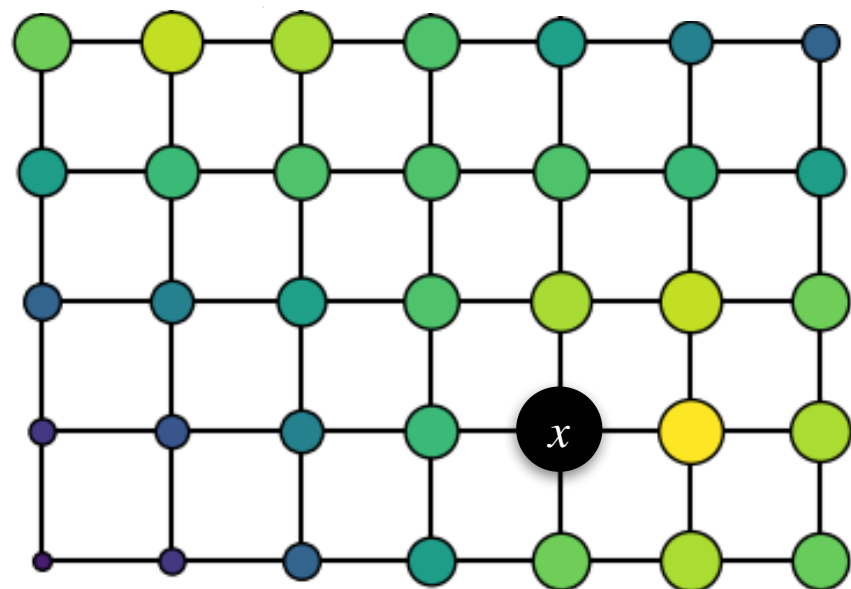
Underlying Continuous Function



Take softmax to obtain proposal in original discrete space



Updated Sample

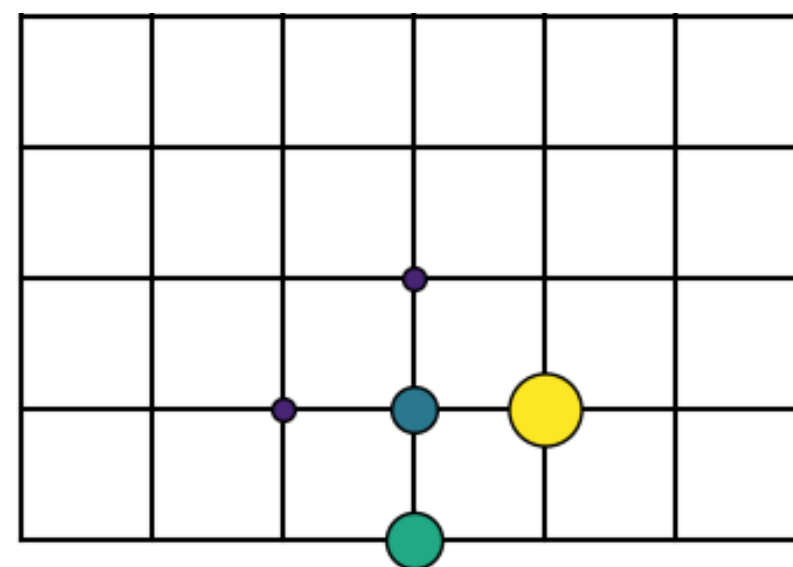


Sample from proposal

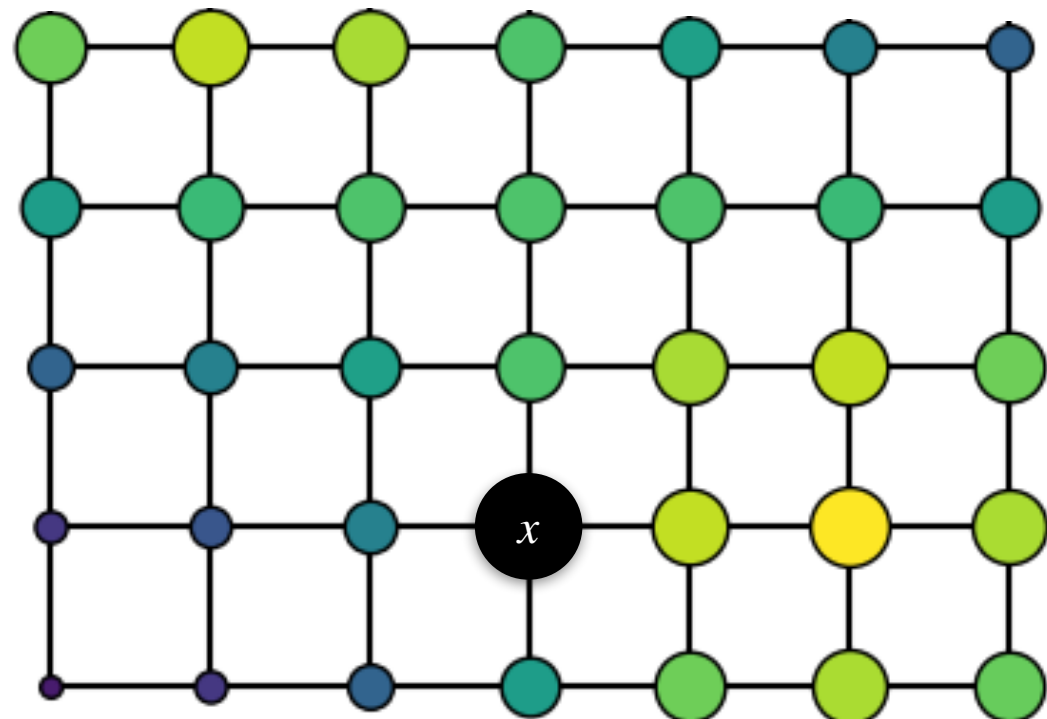


Metropolis-Hastings Step

Proposal Distribution



Target Distribution

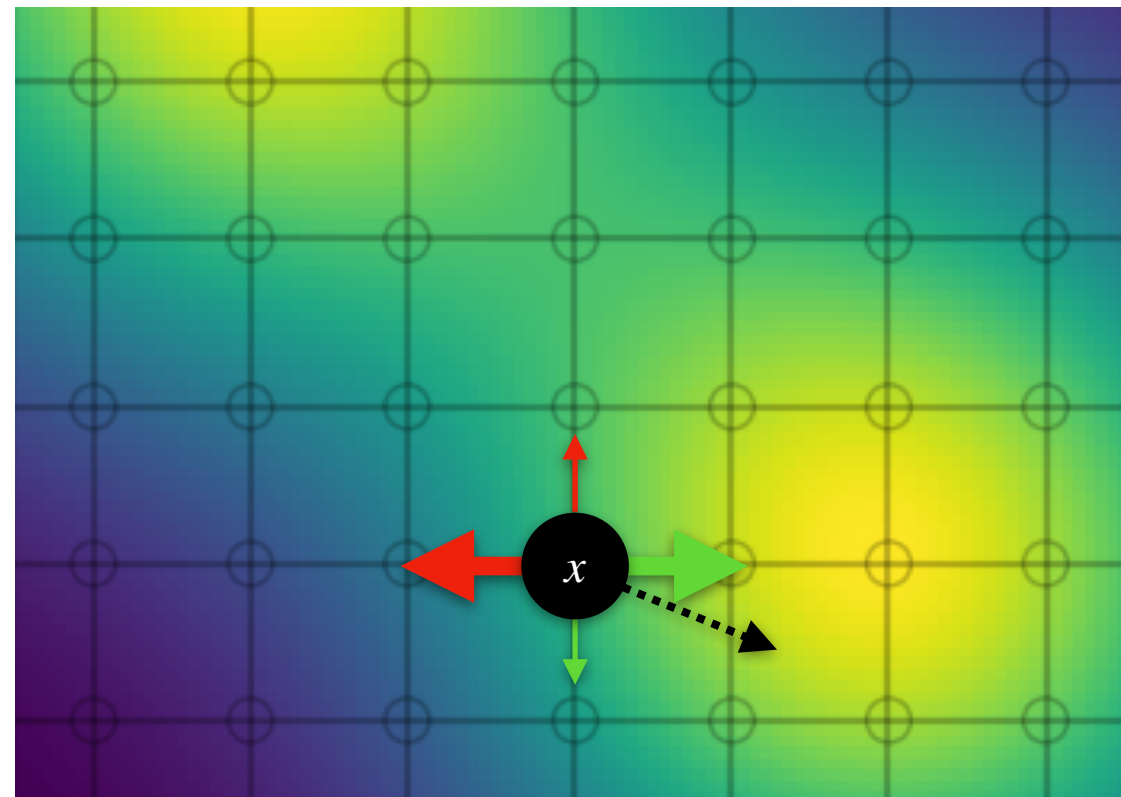


Compute gradients of continuous function



Estimate likelihood ratios

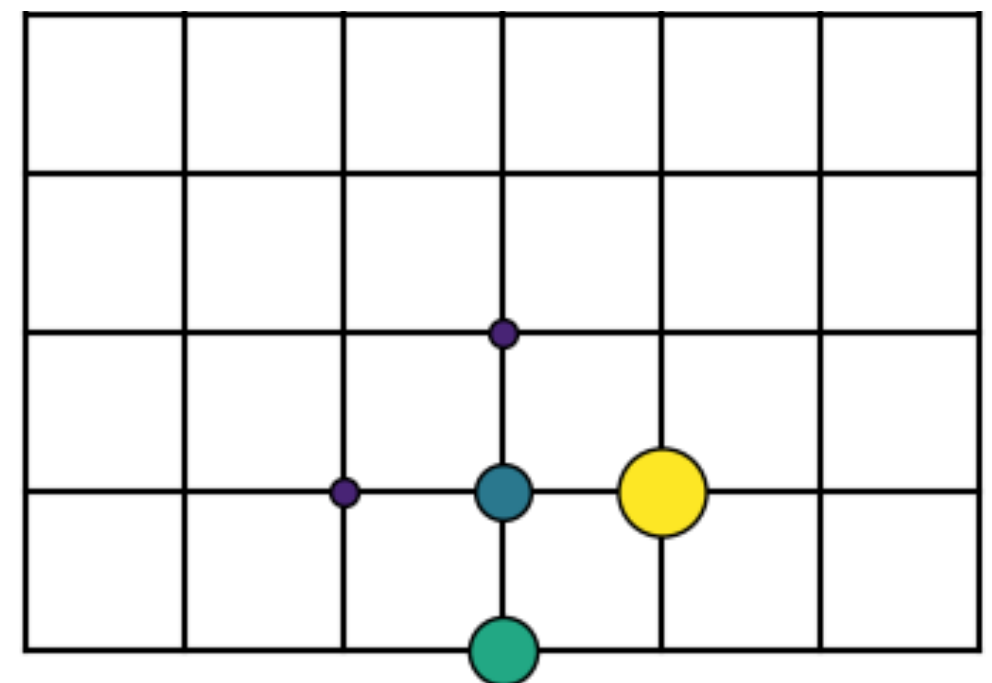
Underlying Continuous Function



Take softmax to obtain proposal in original discrete space



Proposal Distribution

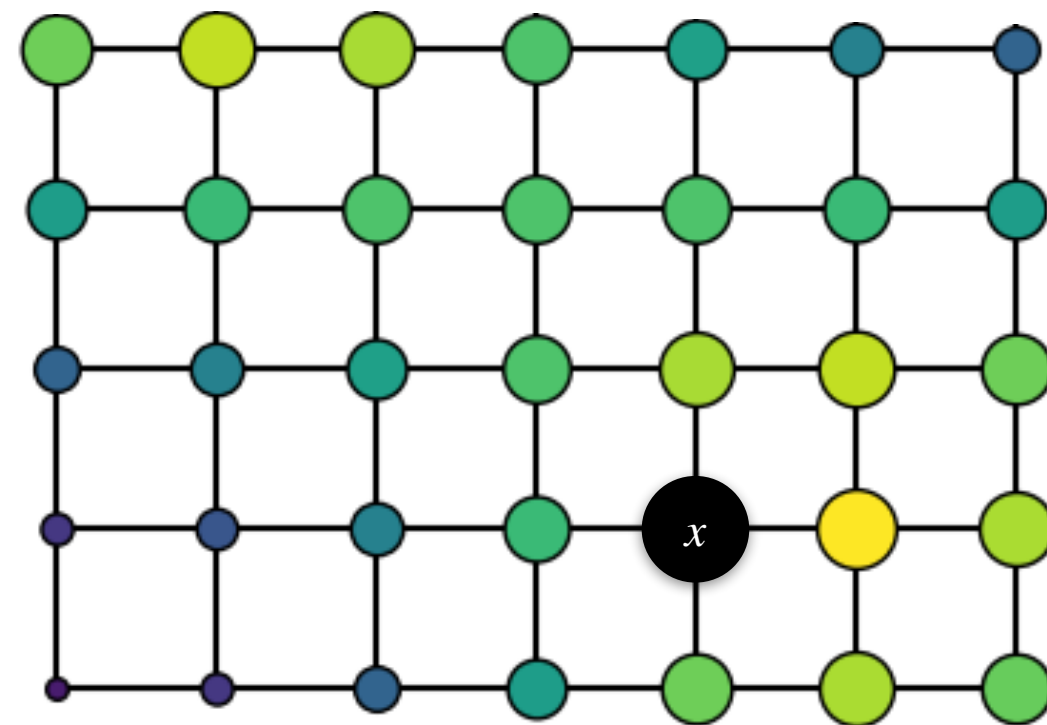


Sample from proposal

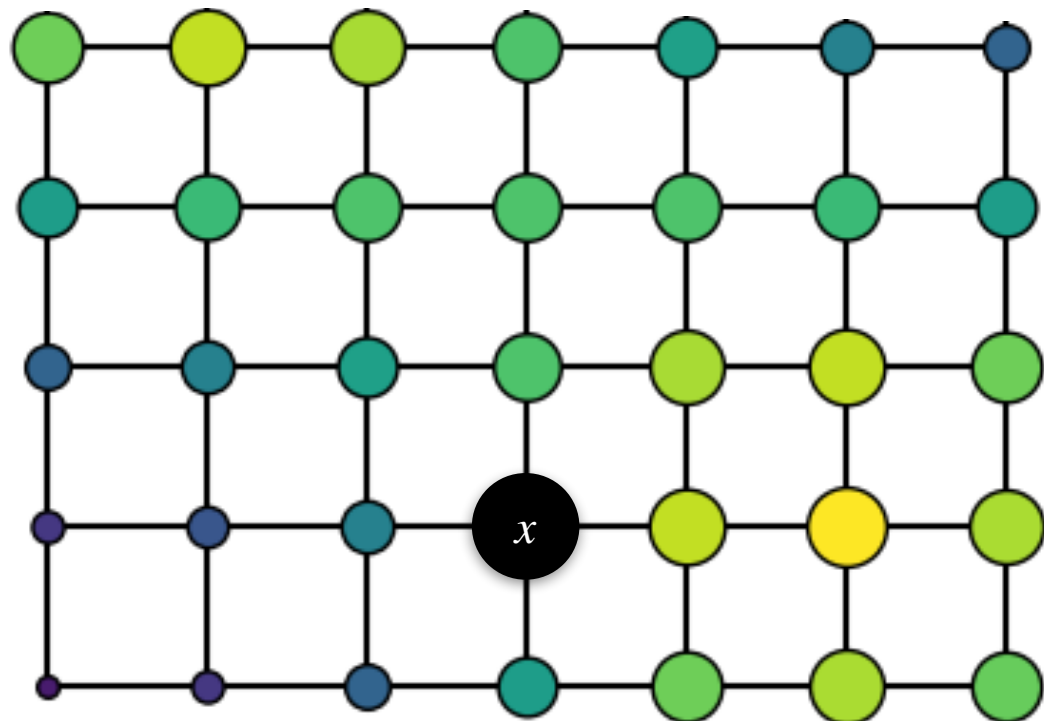


Metropolis-Hastings Step

Updated Sample



Target Distribution

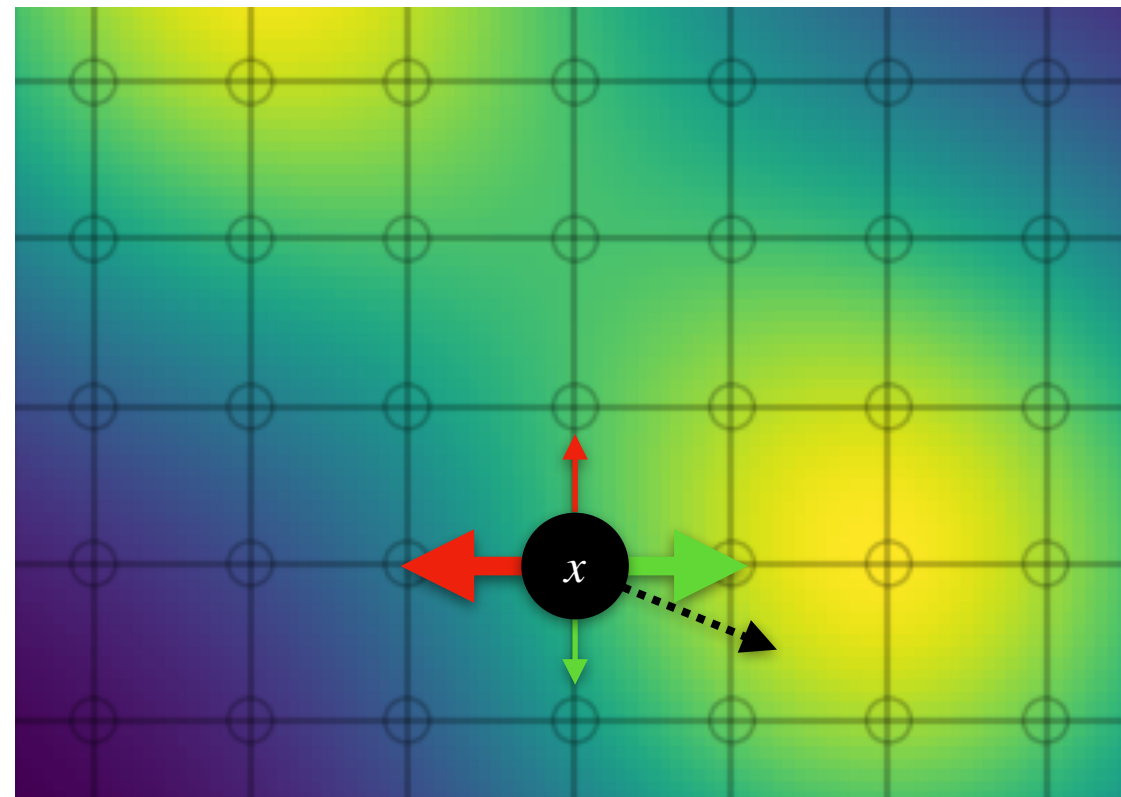


Compute gradients
of continuous
function



Estimate
likelihood ratios

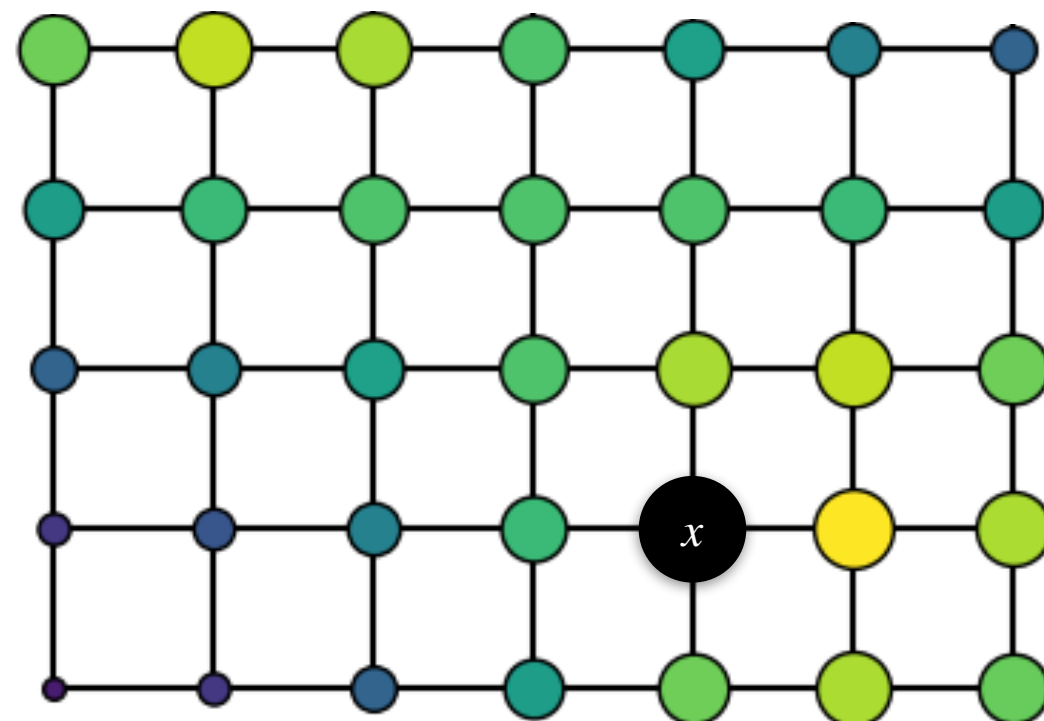
Underlying Continuous Function



Take softmax to obtain
proposal in original
discrete space



Updated Sample

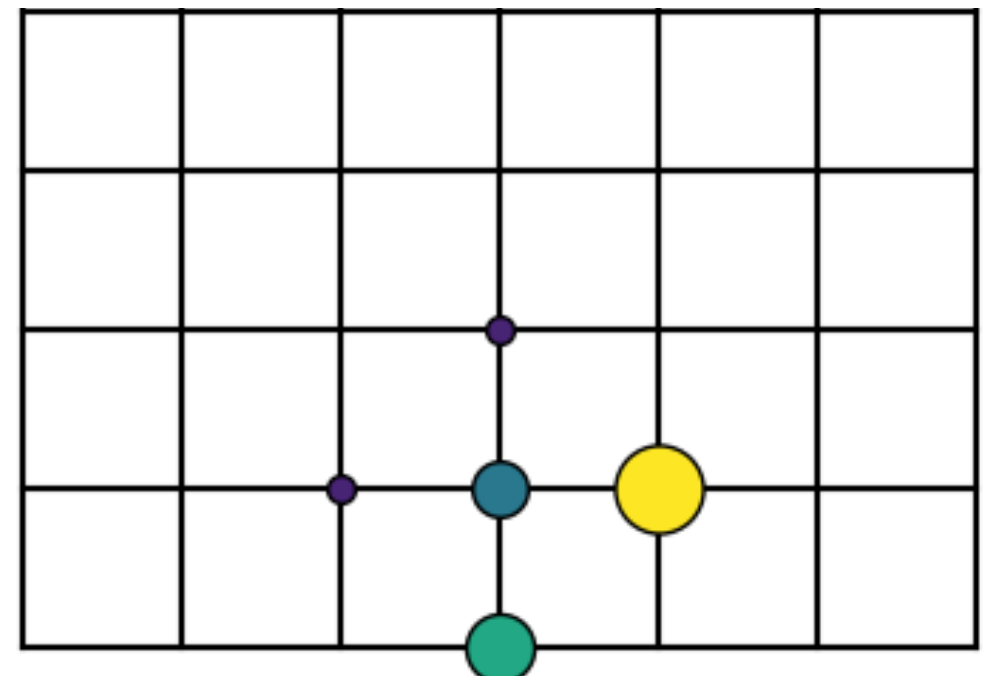


Sample
from proposal

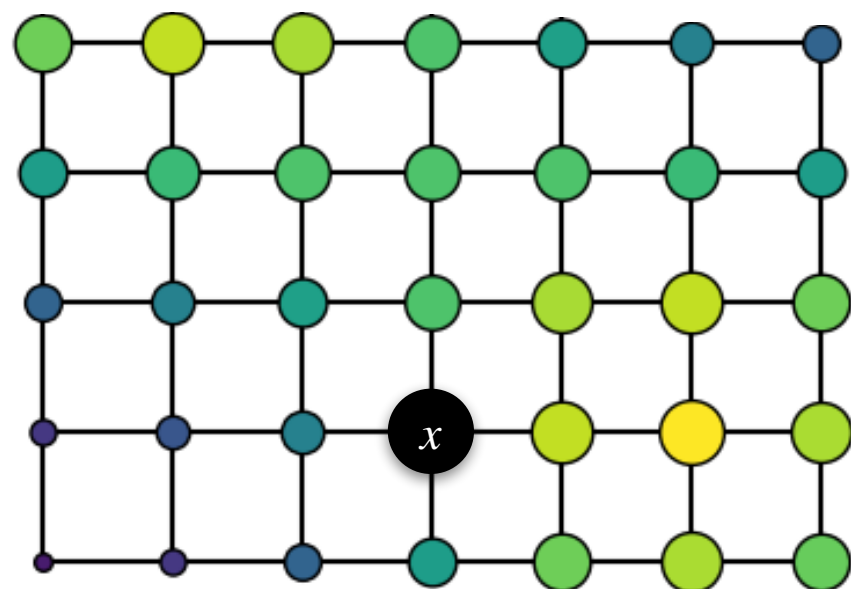


Metropolis-Hastings
Step

Proposal Distribution



Target Distribution

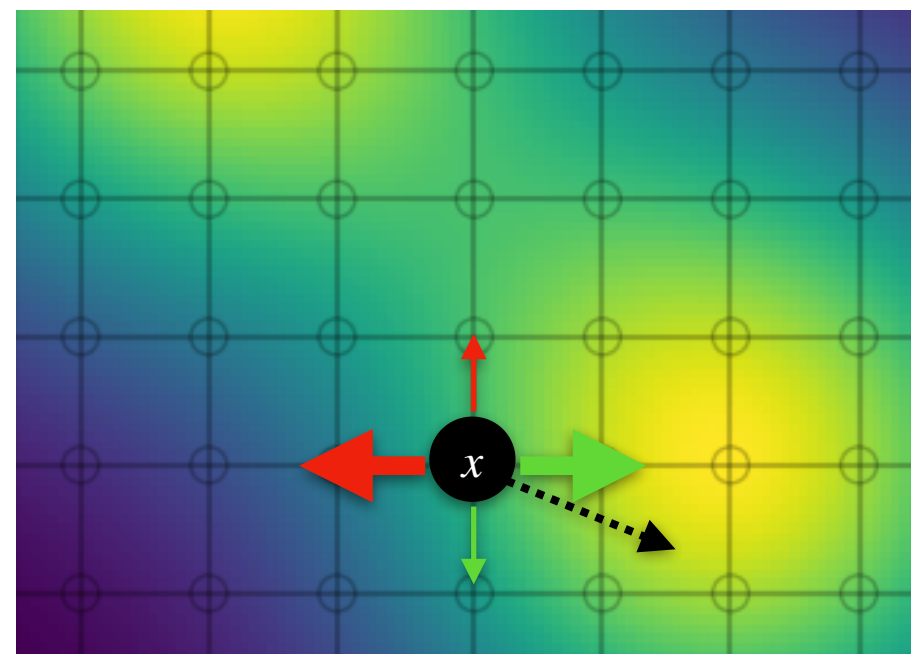


Compute gradients of
continuous function



Estimate
likelihood ratios

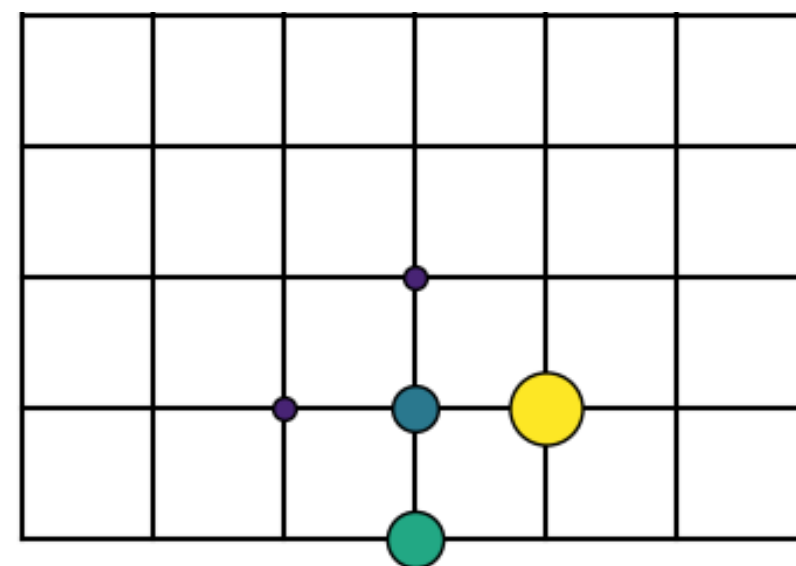
Underlying Continuous Function



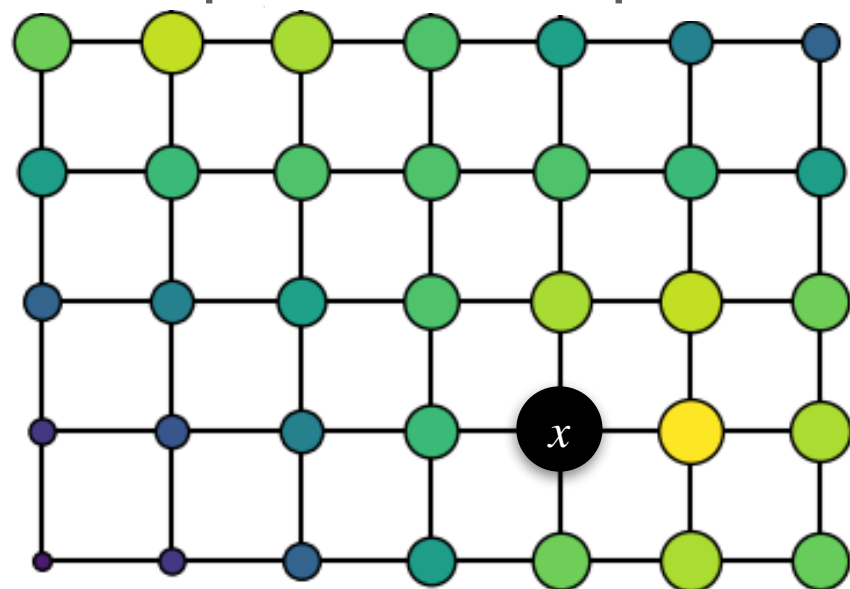
Take softmax to obtain
proposal in original
discrete space



Proposal Distribution



Updated Sample



Sample from proposal



Metropolis-Hastings
Step

Block Gibbs



Gibbs-With-Gradients



D-SVGD



R-MALA



R-HMC



1 2 2 4 5 4 6 1 1 2
4 8 1 1 0 1 0 5 0 7

1 8 6 3 0 4 1 5 7 2
0 2 6 9 3 2 5 8 2 5

6 5 3 4 5 4 3 2 1
0 8 7 6 5 4 3 2 1

7 6 5 4 3 2 1 0
9 8 7 6 5 4 3 2 1

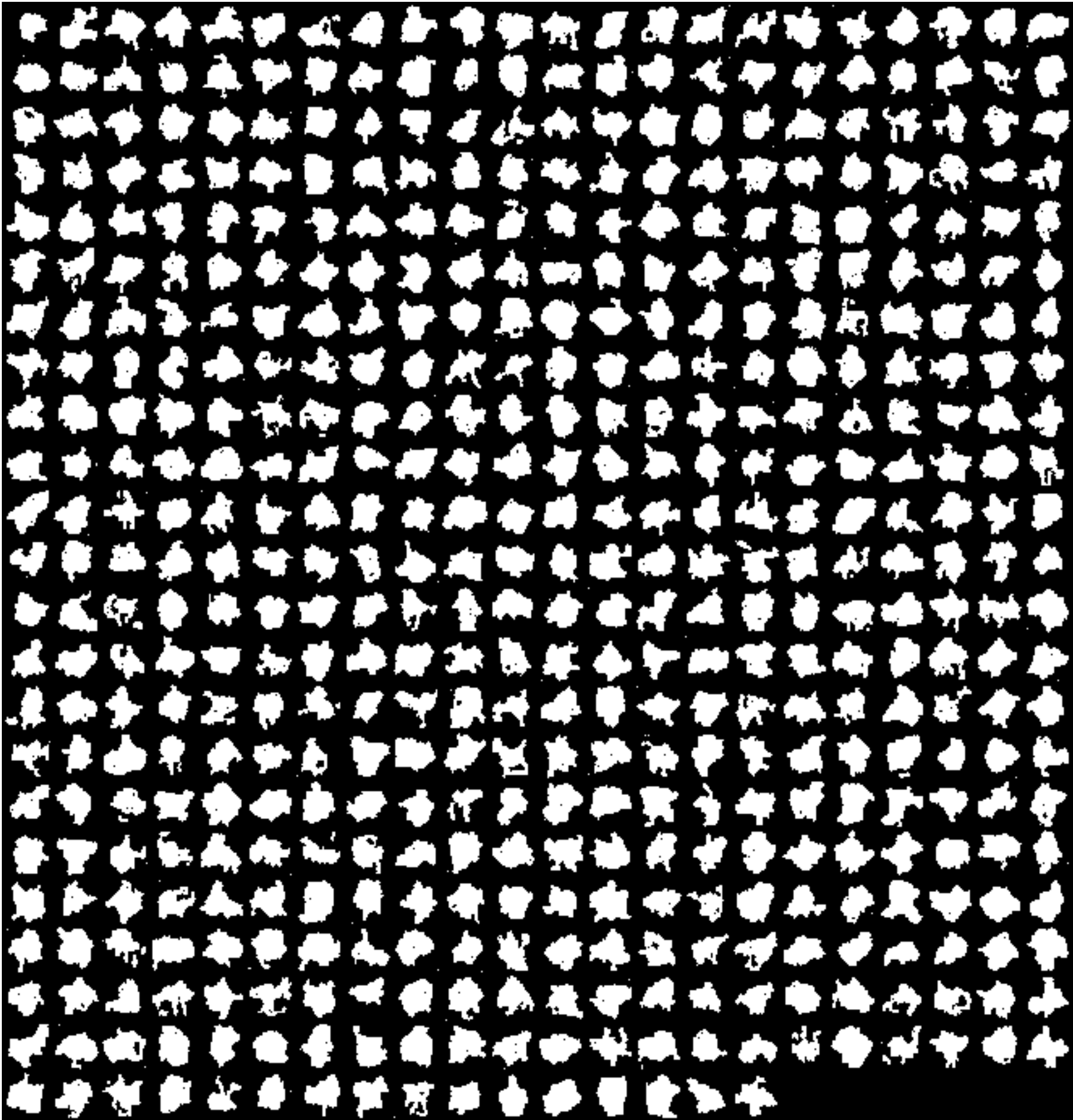


1 2 2 4 5 4 6 1 1 2
4 8 1 1 0 1 0 5 0 7

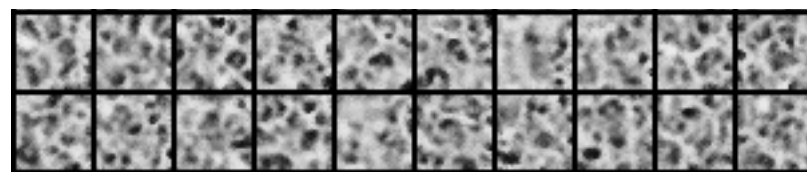
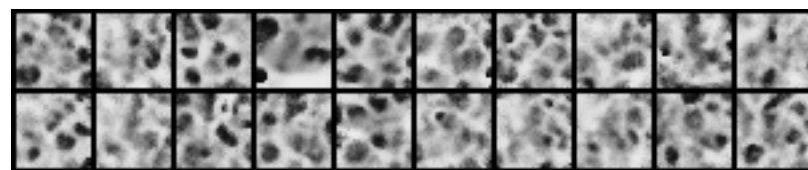
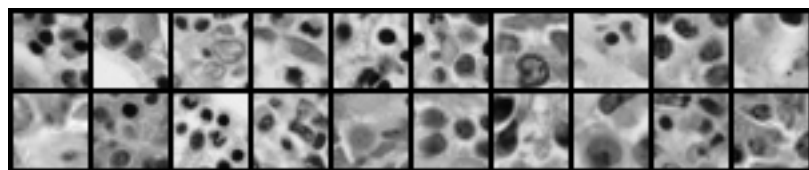
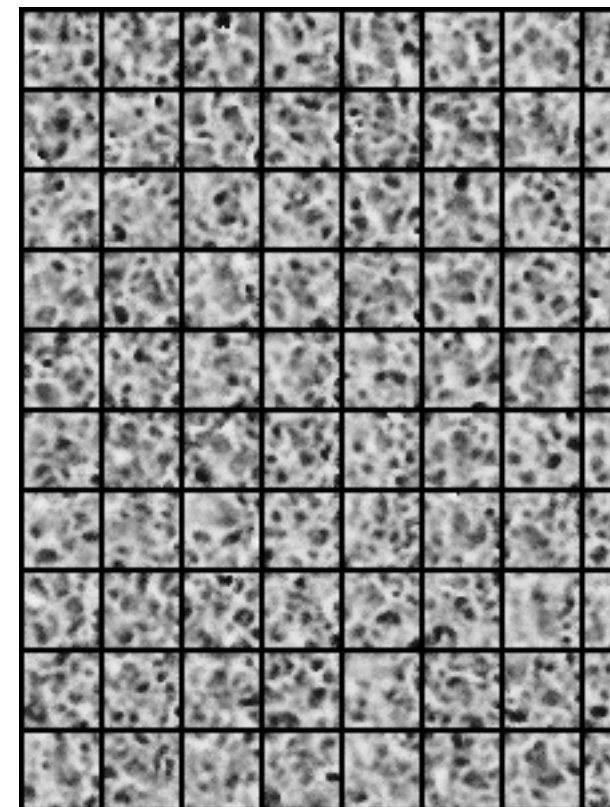
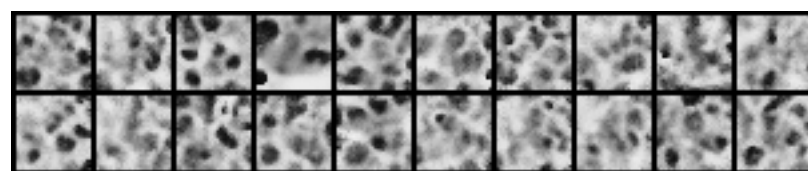
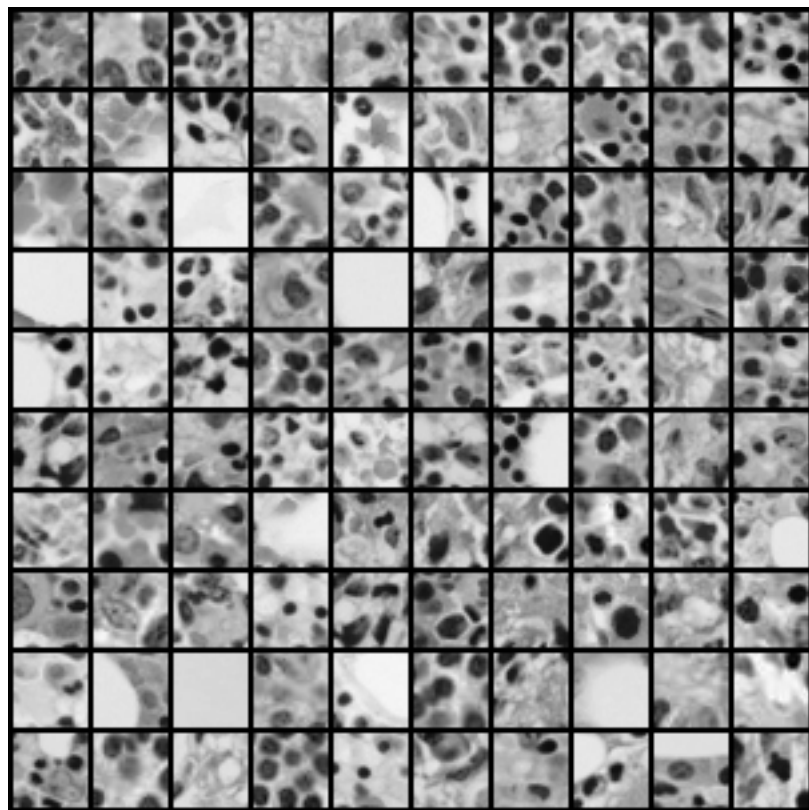
8 5 7 7 1 4 5 9 4 1 4 2 7 4 9 3 3 1 3 4 7 6
3 4 3 8 8 9 6 7 4 1 2 7 9 1 9 5 3 1 3 1 1 4
3 1 4 1 5 0 4 1 3 7 9 1 4 4 3 3 6 5 1 5 1 0
9 4 4 1 1 0 7 6 4 6 1 7 8 6 7 8 5 0 1 1 1 6
4 6 4 8 2 5 6 1 7 1 0 1 1 9 2 5 8 5 1 4 1 7
1 1 7 2 1 6 7 4 1 3 5 3 3 7 3 6 2 8 4 3 9 6
8 4 4 6 3 5 4 4 5 6 5 1 4 1 4 5 6 4 6 4 3 1
7 1 3 4 7 6 4 7 3 1 4 1 1 3 8 7 8 3 3 7 4 5
6 4 4 3 3 5 8 7 3 4 1 7 1 7 7 1 8 3 8 4 5 1
1 5 8 4 7 1 3 5 1 4 5 1 5 5 8 4 4 5 3 0 3 3
1 8 6 3 0 4 1 5 4 2 5 7 7 4 5 4 7 8 1 3 4 1
0 2 6 9 3 2 0 8 2 5 1 1 2 6 4 3 8 3 1 1 2 9
4 8 3 4 1 8 8 1 1 2 7 3 5 3 2 5 6 1 3 4 1 3
0 1 9 6 2 4 9 8 4 9 6 6 3 6 6 7 3 1 6 3 9 1
2 1 9 4 2 9 1 1 8 7 9 1 1 4 1 7 7 4 1 1 5 4
7 0 1 9 4 1 5 5 0 3 2 3 0 2 1 9 1 1 7 6 8 7
7 5 4 2 1 9 6 9 5 3 1 2 8 5 5 8 9 1 2 4 1 3
8 1 4 6 5 6 6 8 8 1 8 5 4 3 1 4 1 7 6 2 1 2
1 5 0 1 8 7 1 3 4 1 9 6 4 9 6 4 4 3 1 9 1 1
5 1 4 1 6 3 3 1 1 1 7 1 3 1 1 1 5 4 4 7 1 8
4 7 2 7 1 9 1 9 2 8 2 1 6 1 1 1 2 7 1 1 5
2 1 1 6 7 7 5 6 1 3 0 7 1 1 4 7 9 3 5 7 5 1
1 3 0 1 4 8 2 3 1 6 7 1 3 1 3 7

୧ ୨ ୩ ୪ ୫ ୬ ୭ ୮ ୯ ୧୦ ୧୧ ୧୨ ୧୩ ୧୪ ୧୫ ୧୬ ୧୭ ୧୮ ୧୯ ୨୦ ୨୧ ୨୨ ୨୩ ୨୪ ୨୫ ୨୬ ୨୭ ୨୮ ୨୯ ୩୦ ୩୧ ୩୨ ୩୩ ୩୪ ୩୫ ୩୬ ୩୭ ୩୮ ୩୯ ୪୦ ୪୧ ୪୨ ୪୩ ୪୪ ୪୫ ୪୬ ୪୭ ୪୮ ୪୯ ୫୦ ୫୧ ୫୨ ୫୩ ୫୪ ୫୫ ୫୬ ୫୭ ୫୮ ୫୯ ୬୦ ୬୧ ୬୨ ୬୩ ୬୪ ୬୫ ୬୬ ୬୭ ୬୮ ୬୯ ୭୦ ୭୧ ୭୨ ୭୩ ୭୪ ୭୫ ୭୬ ୭୭ ୭୮ ୭୯ ୮୦ ୮୧ ୮୨ ୮୩ ୮୪ ୮୫ ୮୬ ୮୭ ୮୮ ୮୯ ୯୦ ୯୧ ୯୨ ୯୩ ୯୪ ୯୫ ୯୬ ୯୭ ୯୮ ୯୯ ୧୦୦

୧ ୨ ୩ ୪ ୫ ୬ ୭ ୮ ୯ ୧୦ ୧୧ ୧୨ ୧୩ ୧୪ ୧୫ ୧୬ ୧୭ ୧୮ ୧୯ ୨୦ ୨୧ ୨୨ ୨୩ ୨୪ ୨୫ ୨୬ ୨୭ ୨୮ ୨୯ ୩୦ ୩୧ ୩୨ ୩୩ ୩୪ ୩୫ ୩୬ ୩୭ ୩୮ ୩୯ ୪୦ ୪୧ ୪୨ ୪୩ ୪୪ ୪୫ ୪୬ ୪୭ ୪୮ ୪୯ ୫୦ ୫୧ ୫୨ ୫୩ ୫୪ ୫୫ ୫୬ ୫୭ ୫୮ ୫୯ ୬୦ ୬୧ ୬୨ ୬୩ ୬୪ ୬୫ ୬୬ ୬୭ ୬୮ ୬୯ ୭୦ ୭୧ ୭୨ ୭୩ ୭୪ ୭୫ ୭୬ ୭୭ ୭୮ ୭୯ ୮୦ ୮୧ ୮୨ ୮୩ ୮୪ ୮୫ ୮୬ ୮୭ ୮୮ ୮୯ ୯୦ ୯୧ ୯୨ ୯୩ ୯୪ ୯୫ ୯୬ ୯୭ ୯୮ ୯୯ ୧୦୦









D-SVGD



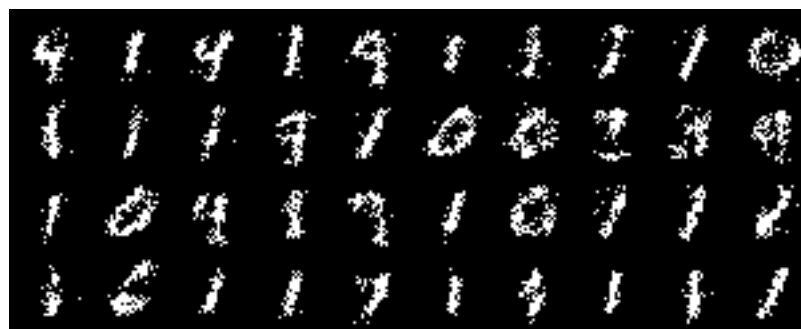
R-MALA



R-HMC



Block Gibbs



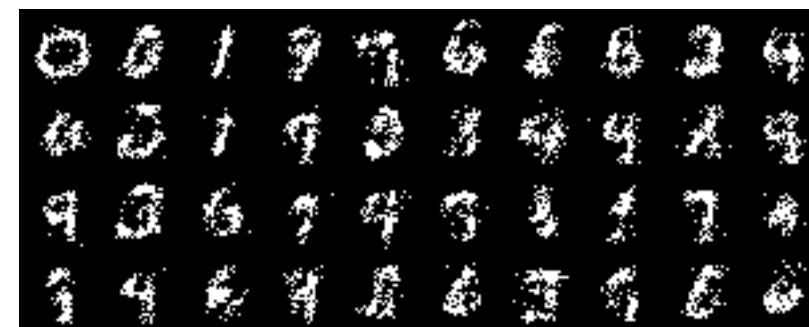
Gibbs-1



Gibbs-With-Gradients



Gibbs-2



Gibbs-With-Gradients-5



Hamming-Ball-10-1





D-SVGD



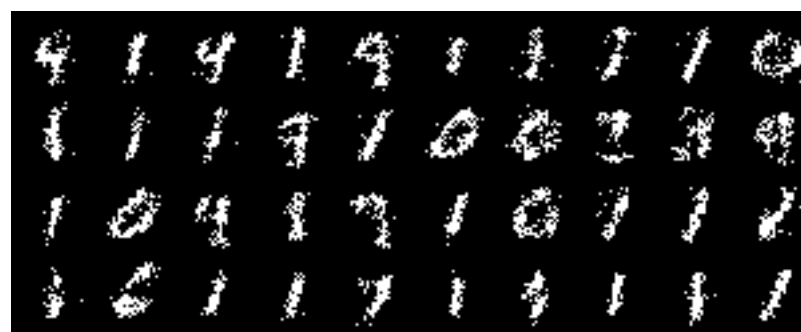
R-MALA



R-HMC



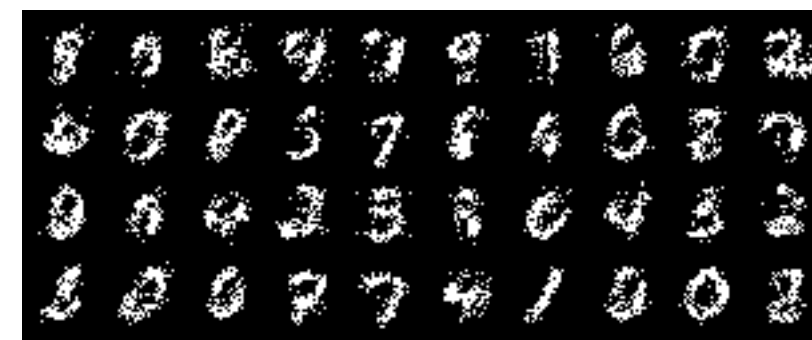
Block Gibbs



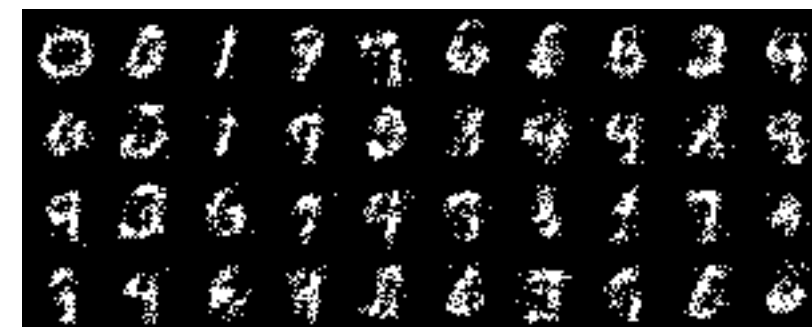
Gibbs-With-Gradients



Gibbs-1



Gibbs-2



Hamming-Ball-10-1

