CSCI 699 - ProbGen

Probabilistic and Generative Models

Willie Neiswanger

Lecture 4 - Approximate Inference

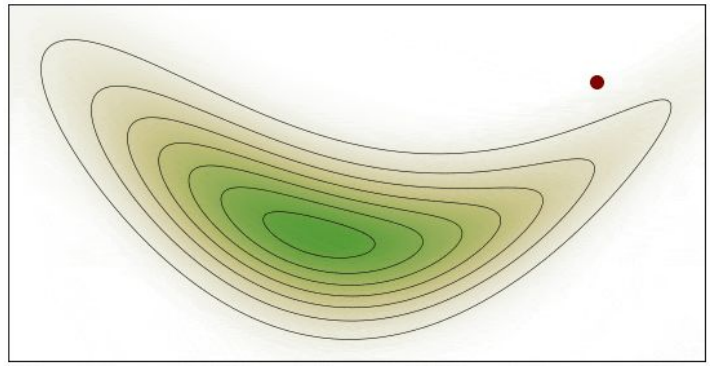
Today

Lecture: Approximate inference algorithms in probabilistic models, including the classics — Markov chain Monte Carlo (MCMC), and variational inference (VI).

Today

Lecture: Approximate inference algorithms in probabilistic models, including the classics — Markov chain Monte Carlo (MCMC), and variational inference (VI).

- Markov chain Monte Carlo (MCMC) methods, including:
  - MC integration, rejection/importance sampling.
  - Metropolis–Hastings algorithm, Gibbs sampling.
  - Gradient-based methods: Langevin Monte Carlo, Hamiltonian Monte Carlo.

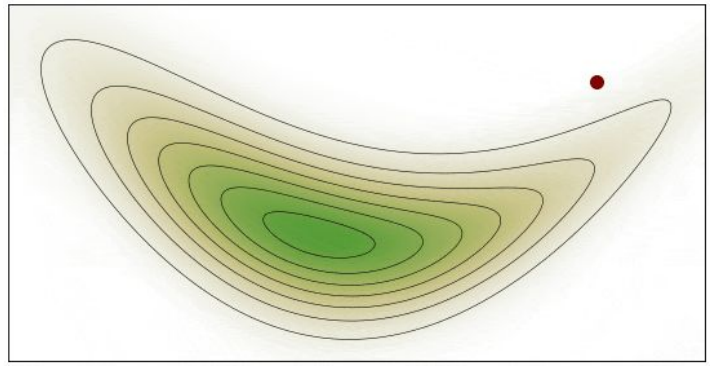


Source: "Creating animations with MCMC", Krepl 2018, Wikimedia commons,

Today

Lecture: Approximate inference algorithms in probabilistic models, including the classics — Markov chain Monte Carlo (MCMC), and variational inference (VI).

- Markov chain Monte Carlo (MCMC) methods, including:
  - MC integration, rejection/importance sampling.
  - Metropolis–Hastings algorithm, Gibbs sampling.
  - Gradient-based methods: Langevin Monte Carlo, Hamiltonian Monte Carlo.
- Variational inference (VI) methods, including
  - Evidence lower bound (ELBO), Stochastic VI, black-box VI.

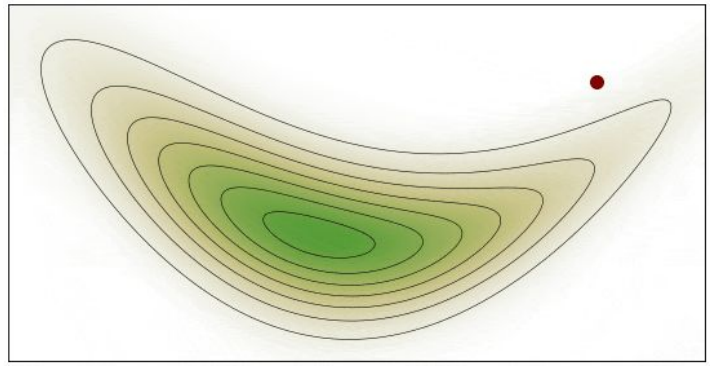


Source: "Creating animations with MCMC", Krepl 2018, Wikimedia commons,

Today

Lecture: Approximate inference algorithms in probabilistic models, including the classics — Markov chain Monte Carlo (MCMC), and variational inference (VI).

- Markov chain Monte Carlo (MCMC) methods, including:
  - MC integration, rejection/importance sampling.
  - Metropolis–Hastings algorithm, Gibbs sampling.
  - Gradient-based methods: Langevin Monte Carlo, Hamiltonian Monte Carlo.
- Variational inference (VI) methods, including
  - Evidence lower bound (ELBO), Stochastic VI, black-box VI.



Source: "Creating animations with MCMC", Krepl 2018, Wikimedia commons,

After:

- Project Pitches #1: Groups 1-5

Review: Exact Inference in PGMs

Review — Last Class on Exact Inference

Lecture: Classic algorithms in probabilistic graphical models (PGMs) for exact and approximate inference & learning.

- Exact inference algorithms in PGMs.
- Variable Elimination algorithm.
- Belief Propagation algorithm (sum/max-product message passing).
- Famous algorithms: Forward-Backward & Viterbi.
- Expectation-Maximization (EM) algorithm.

Source: USC Viterbi Magazine, "The Viterbi Algorithm at 50"

Review — Last Class on Exact Inference

Variable elimination algorithm → for PGMs with discrete random variables.

Review — Last Class on Exact Inference

Variable elimination algorithm:

For each variable x_i,

(1) Form a product of all factors ϕ_i containing x_i.

(2) Marginalize out x_i to obtain a new factor τ.

(3) Replace the factors ϕ_i with τ.

A function of all other variables in factor.

⇒ Computationally efficient way of computing a marginal inference query.

Review — Last Class on Exact Inference

Variable elimination algorithm:

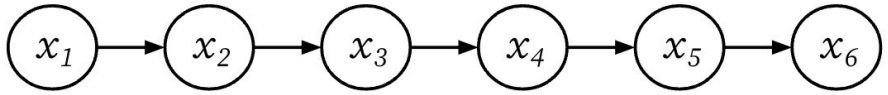
Showed examples on:

Markov chains

Variable Elimination — An Illustrative Example

Consider first the problem of marginal inference.

Suppose we are given this Markov chain PGM.



Joint PDF can be written:

Markovian property

$$p(x_1, \ldots, x_n) = p(x_1) \prod_{i=2}^{n} p(x_i \mid x_{i-1})$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Review — Last Class on Exact Inference

Variable elimination algorithm:

Showed examples on:

Student-grade-letter PGM

⇒ Eliminated: *d*, *i*, *s*, *g*
(to get marginal $p(l)$) .

Variable Elimination — General Form

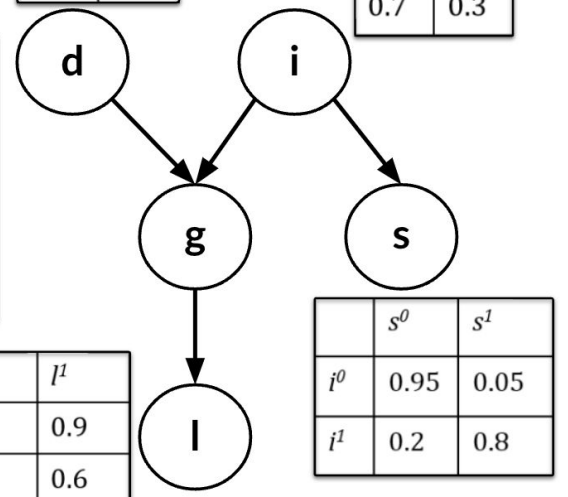
Recall the ***Student PGM*** from last class:

	g^1	g^2	g^3
i^0,d^0	0.3	0.4	0.3
i^0,d^1	0.05	0.25	0.7
i^1,d^0	0.9	0.08	0.02
i^1,d^1	0.5	0.3	0.2

d - Class difficulty.
i - Student's intelligence.
g - Student's grade in the class.
s - Student's SAT test score.
l - Professor's letter of recommendation.
(good vs bad letter)

d^0	d^1
0.6	0.4

i^0	i^1
0.7	0.3



	l^0	l^1
g^1	0.1	0.9
g^2	0.4	0.6
g^3	0.99	0.01

	s^0	s^1
i^0	0.95	0.05
i^1	0.2	0.8

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Review — Last Class on Exact Inference

Variable elimination algorithm:

Showed examples on:

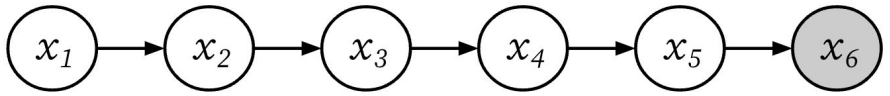
Extensions to conditional distributions (e.g., posterior distributions)

Variable Elimination — Extensions to Conditional Probabilities

We can extend this algorithm to conditional distributions (*e.g.*, posteriors)!

Suppose we have a joint probability $p(x_1, \ldots, x_n)$.

And our **goal** is to compute a given conditional probability, *e.g.*, $p(x_1 \mid x_n = 5)$.



Conditional PDF:

$$p(x_1 \mid x_n = 5) = \frac{p(x_1, x_n = 5)}{p(x_n = 5)}$$

Two marginal probabilities!

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Review — Last Class on Exact Inference

The **belief propagation** algorithm.

⇒ If you want to be able to answer *any* inference query.

(And not have to repeat belief propagation for each query).

Caveat: this is for *tree graph PGMs* … for general PGMs, need to use the Junction Tree algorithm, which we didn't cover in class.

Review — Last Class on Exact Inference

Definition. The **belief propagation** algorithm on tree graphs has two variants: *sum-product message passing*, and *max-product message passing*.

Sum-product message passing algorithm:

Used for marginal/conditional inference, *i.e.*, computing $p(x_i)$ or $p(x_i \mid x_k)$.

Max-product message passing algorithm:

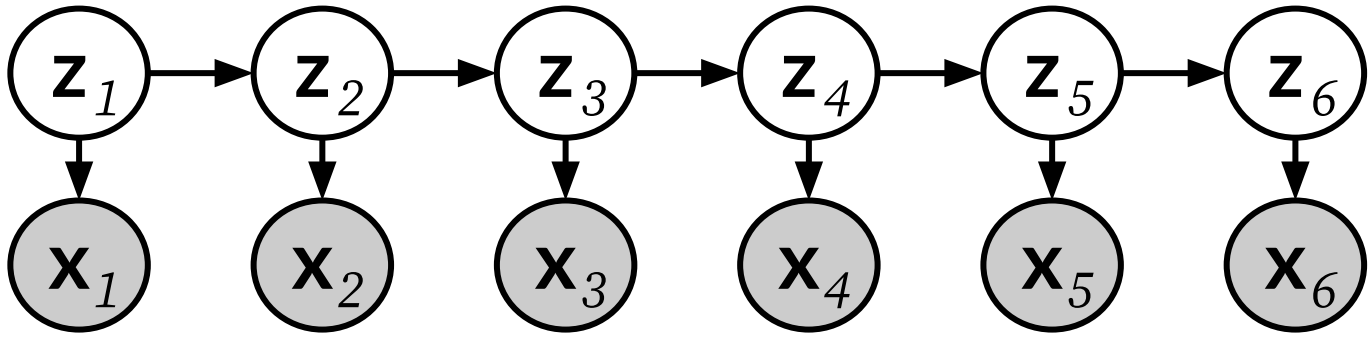
Used for MAP (*maximum a posteriori*) inference, *i.e.*, computing:

$$\max_{x_1,\ldots,x_n} p(x_1, \ldots, x_n) \quad \text{or} \quad \max_{x_i} p(x_i \mid x_k) \quad \text{or} \quad \arg\max_{x_i} p(x_i \mid x_k)$$

Review — Last Class on Exact Inference

Famous Example of Sum-Product Algorithm:

Forward-Backward Algorithm in HMMs!



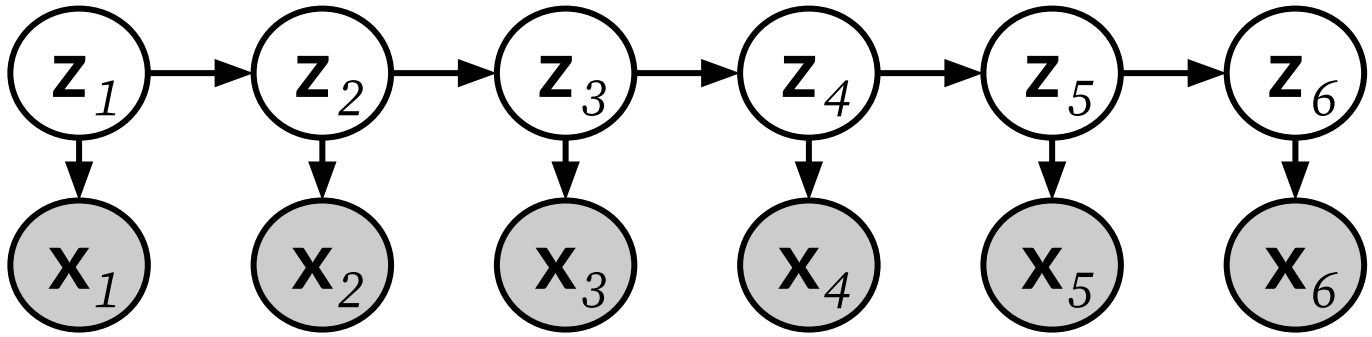
Suppose we want to query $p(z_i \mid x_1, \ldots, x_n)$ for all $z_i \in \{z_1, \ldots, z_n\}$.

Often called “smoothing”.

Review — Last Class on Exact Inference

Famous Example of Max-Product Algorithm:

Viterbi Algorithm in HMMs!



Suppose we want to query: $\max_{z_1,\ldots,z_n} p(z_1,\ldots,z_n \mid x_1,\ldots,x_n)$

MAP estimate of latent z's

Often called "Viterbi Path".

(Joint maximum over all)

Review — Last Class on Exact Inference

Then: Learning (parameter estimation) in PGMs.

Review — Last Class on Exact Inference

Then: Learning (parameter estimation) in PGMs.

Expectation maximization (EM) algorithm is a famous algorithm, which can be remembered as

Maximum likelihood estimation in models with latent variables.

Note: this is a key parameter estimation algorithm for PGMs — more general than inference/MAP in discrete-variable PGMs with tree structure.

(*E.g.*, which we assumed for belief propagation algorithms)

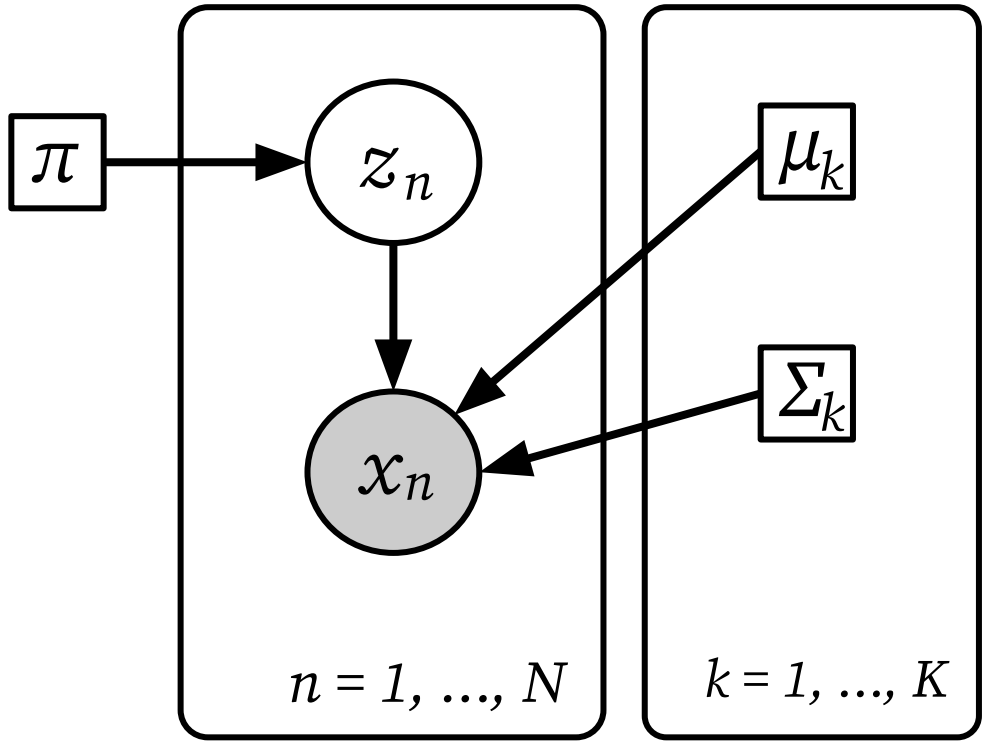
Review — Last Class on Exact Inference

Suppose we want to estimate the parameters: π, μ_k, Σ_k

Gaussian mixture model (GMM)

Chicken and egg problem!

- If we knew the parameters, then we could do inference to infer the latents.
- If we knew the latents, then we could run MLE (optimize) to estimate the parameters.



Review — Last Class on Exact Inference

Let's investigate the two steps of EM algorithm:

Step (1): "The E Step" — infer the latent variables: $z \sim p_\theta(z \mid x)$

So that you could compute this expected value:

$$\mathbb{E}_{z \sim p_\theta(z|x)} \left[\log p_\theta(x, z) \right]$$

⇒ The expected value of the (log) likelihood function, w.r.t. the conditional distribution of latents given observed (under current estimates of the parameters).

Review — Last Class on Exact Inference

Let's investigate the two steps of EM algorithm:

Step (2): "The M Step" — Run MLE to estimate the parameters:

$$\theta \leftarrow \arg\max_\theta \mathbb{E}_z \left[\log p_\theta(x, z)\right]$$

Objective: expectation from previous slide (optimizing over θ in the joint PDF)

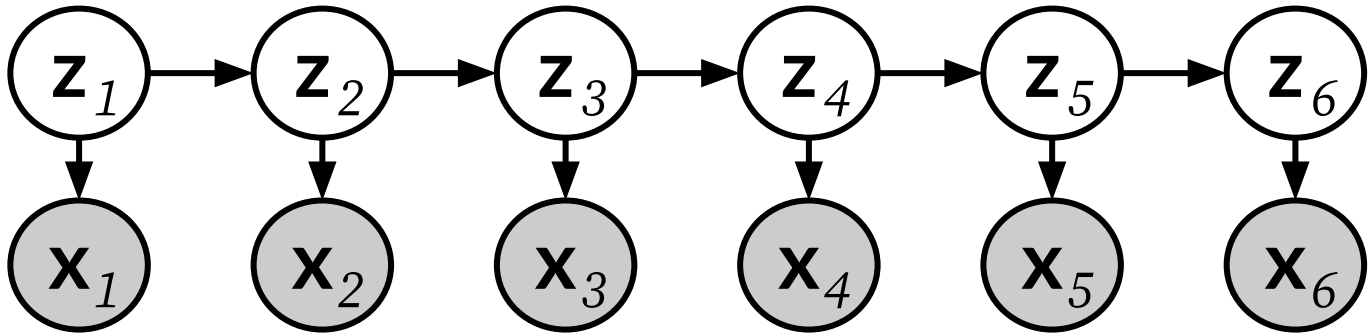
Given our expression for the expected log-likelihood (from the E-step), optimize as usual to get an estimate for the MLE.

Review — Last Class on Exact Inference

Famous Example of EM Algorithm:

Baum–Welch Algorithm in HMMs!

Lloyd Welch is also USC affiliated :-).



Goal: estimate parameters of HMM (*i.e.*, parameters of: initial state distribution, transition distribution, and emission distributions).

Approximate Inference: Overview

Exact vs. Approximate Inference

In practice

- The probabilistic models that we use are often quite complex.
- Simple algorithms like variable elimination may be too slow.
- Many interesting classes of models do not admit tractable inference.
- ⇒ Much research effort is spent on *approximate inference* algorithms.

Main Approximate Inference Algorithms

At a high level: two popular "types" of approximate inference algorithms:

Markov chain Monte Carlo (MCMC)

E.g., Gibbs Sampling

Source: Jessica Stringham, Gibbs Sampling in Python

Variational Inference (VI)

E.g., black-box variational inference

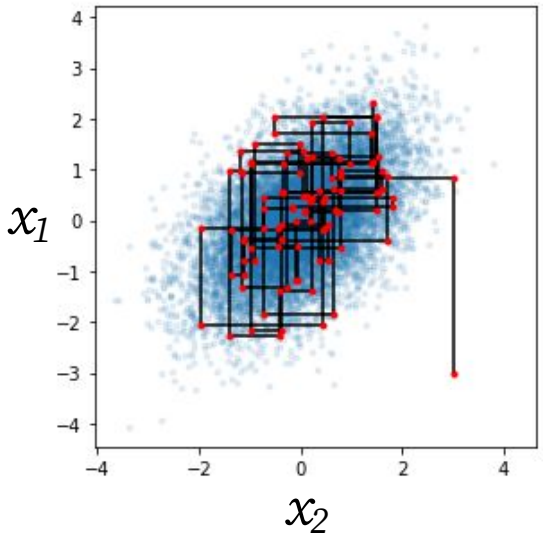
Source: Dave Blei, "Variational Inference"

Main Approximate Inference Algorithms

At a high level: two popular "types" of approximate inference algorithms:

Markov chain Monte Carlo (MCMC)

E.g., Gibbs Sampling



Source: Jessica Stringham, Gibbs Sampling in Python

Variational Inference (VI)

E.g., black-box variational inference

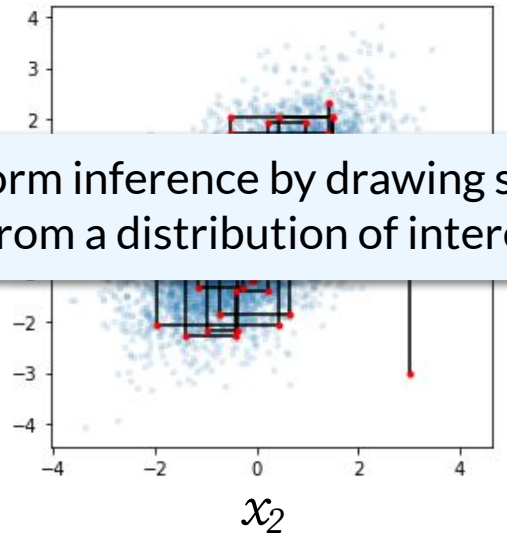
Source: Dave Blei, "Variational Inference"

Main Approximate Inference Algorithms

At a high level: two popular "types" of approximate inference algorithms:

Markov chain Monte Carlo (MCMC)

E.g., Gibbs Sampling



Perform inference by drawing samples from a distribution of interest.

Source: Jessica Stringham, Gibbs Sampling in Python

Variational Inference (VI)

E.g., black-box variational inference

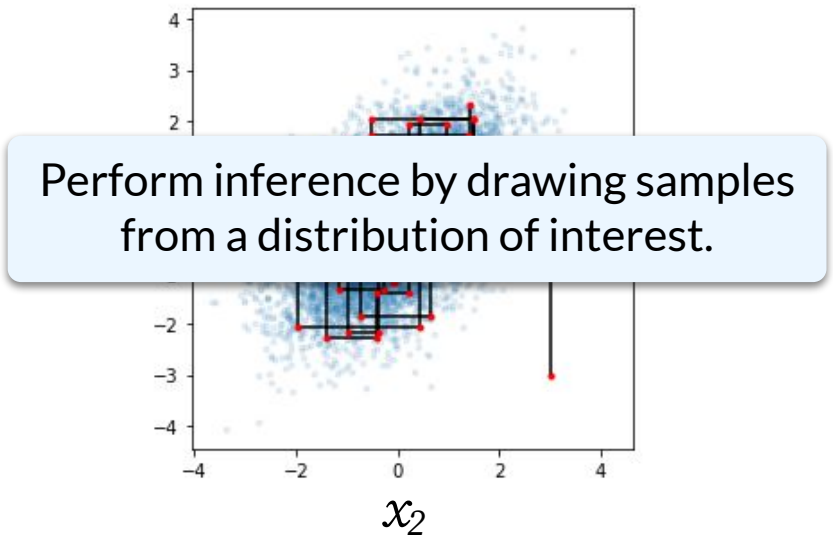
Source: Dave Blei, "Variational Inference"

Main Approximate Inference Algorithms

At a high level: two popular "types" of approximate inference algorithms:

Markov chain Monte Carlo (MCMC)

E.g., Gibbs Sampling

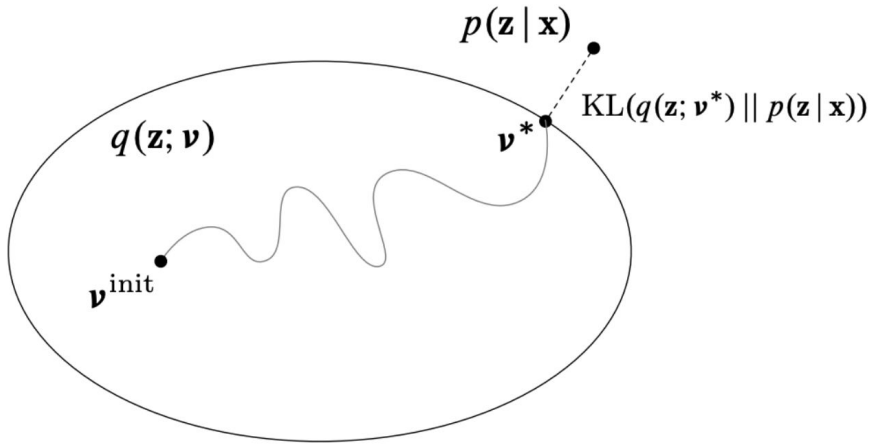


Perform inference by drawing samples from a distribution of interest.

Source: Jessica Stringham, Gibbs Sampling in Python

Variational Inference (VI)

E.g., black-box variational inference



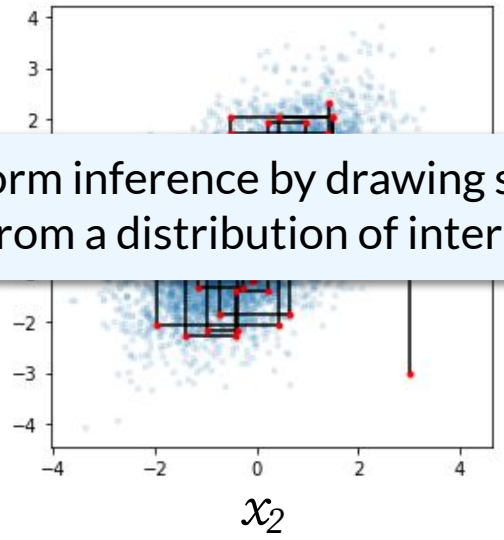
Source: Dave Blei, "Variational Inference"

Main Approximate Inference Algorithms

At a high level: two popular "types" of approximate inference algorithms:

Markov chain Monte Carlo (MCMC)

E.g., Gibbs Sampling

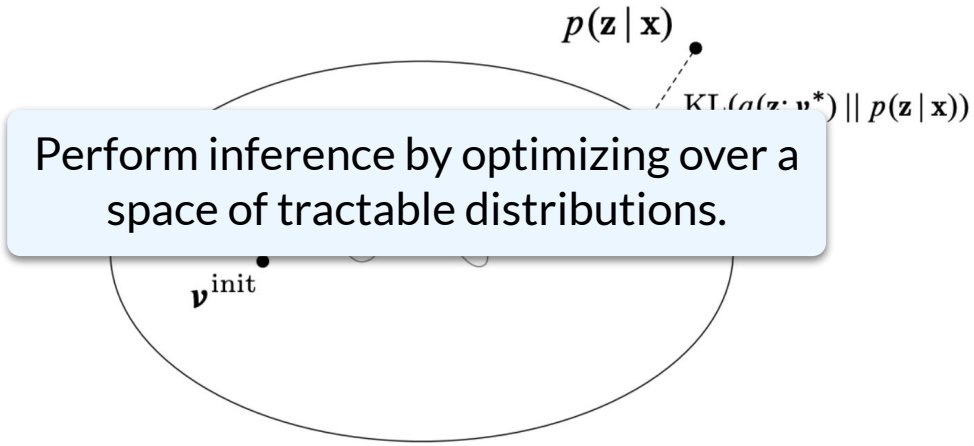


Perform inference by drawing samples from a distribution of interest.

Source: Jessica Stringham, Gibbs Sampling in Python

Variational Inference (VI)

E.g., black-box variational inference



Perform inference by optimizing over a space of tractable distributions.

Source: Dave Blei, "Variational Inference"

Main Approximate Inference Algorithms

How are these methods used in probabilistic/generative modeling?

Main Approximate Inference Algorithms

How are these methods used in probabilistic/generative modeling?

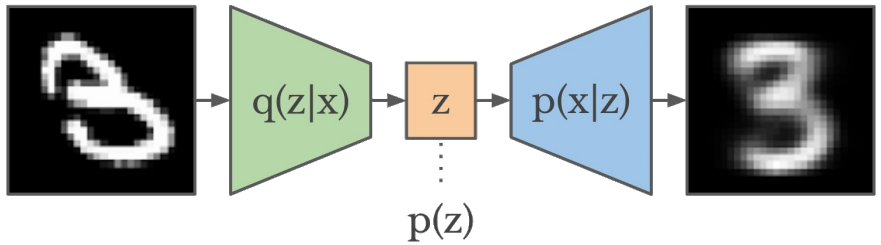
Posterior Inference (⇒ inferring a conditional dist.)

Main Approximate Inference Algorithms

How are these methods used in probabilistic/generative modeling?

Posterior Inference (⇒ inferring a conditional dist.)

- *E.g., Variational Inference Algorithms*
- ⇒ approximate posterior inference (conditional distribution of latent variables given observed variables) via an optimization procedure.



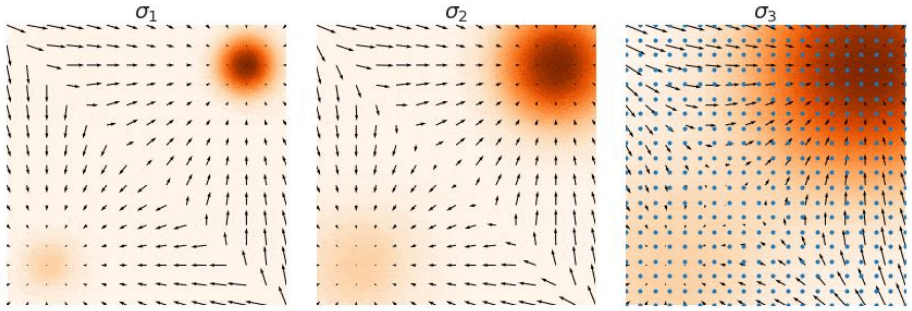
- VAE (Variational Autoencoder)
- DDPM (Denoising Diffusion Probabilistic Model)
- DDIM (Denoising Diffusion Implicit Model)
- Variational Inference in (Deep) Gaussian Processes

Main Approximate Inference Algorithms

How are these methods used in probabilistic/generative modeling?

Posterior Inference (⇒ inferring a conditional dist.)

- *E.g., Markov chain Monte Carlo (MCMC) Algorithms*
- ⇒ approximate posterior inference (conditional distribution of latent variables given observed variables)... *via sampling from conditional dist.*



- Score-based generative models.
- Bayesian neural networks.
- Energy-based deep generative models.
- Boltzmann Machines (RBMs, DBMs, and DBNs)

Sampling Methods: Monte Carlo and MCMC

Sampling Methods — Overview

Sampling approximate inference methods:

Produce answers by repeatedly generating random numbers from a distribution of interest.

Sampling Methods — Overview

Sampling approximate inference methods:

Produce answers by repeatedly generating random numbers from a distribution of interest.

Some points to note:

- Can be used for both marginal, conditional, and MAP inference queries.

Sampling Methods — Overview

Sampling approximate inference methods:

Produce answers by repeatedly generating random numbers from a distribution of interest.

Some points to note:

- Can be used for both marginal, conditional, and MAP inference queries.
- Can compute various quantities of interesting, such as expectations $\mathbb{E}_{p(x)}\left[g(x)\right]$,
  - Where $g(x)$ is some "test function".

Sampling Methods — Overview

Sampling approximate inference methods:

Produce answers by repeatedly generating random numbers from a distribution of interest.

Some points to note:

- Can be used for both marginal, conditional, and MAP inference queries.
- Can compute various quantities of interesting, such as expectations $\mathbb{E}_{p(x)}\left[g(x)\right]$,
  - Where $g(x)$ is some "test function".
- Sampling methods have historically been most common approximate inference methods.
  - Though over the past ~15 years, variational methods have gained in popularity.
  - With primary advantages being speed, scale, and automation (ease of use).

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Sampling Methods — Overview

We will cover a progression of methods:

- Monte Carlo (MC)
- Markov chain Monte Carlo (MCMC)
  - Markov chains, stationary distributions
  - MCMC: Metropolis-Hastings, Gibbs sampling.
- Gradient-based Markov chain Monte Carlo (gradient-MCMC)
  - Langevin Monte Carlo (LMC), Hamiltonian Monte Carlo (HMC)
  - Also: stochastic-gradient-based Markov chain Monte Carlo (sg-MCMC)

Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

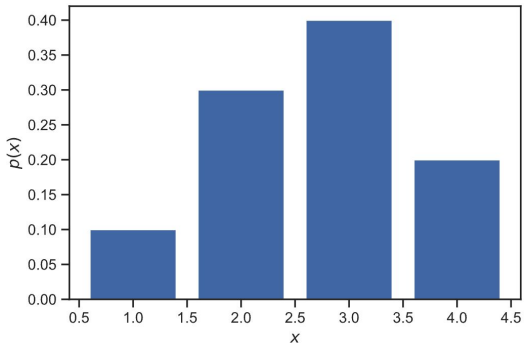
Recall: *categorical distribution*.

- A simple discrete distribution.
- With probability mass function:

$$X = \begin{cases} 1, & \text{with probability } \theta_1, \\ 2, & \text{with probability } \theta_2, \\ \vdots & \\ k, & \text{with probability } \theta_k. \end{cases}$$

... and $\sum_k \theta_k = 1$

For example:



Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

Note: sampling from an arbitrary distribution is not an easy problem!

Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

Note: sampling from an arbitrary distribution is not an easy problem!

- Computers can only generate samples from simple distributions.
- *E.g.*, such as a uniform distribution over $[0, 1]$ ⇒ Even then, these are *pseudorandom!*

Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

Note: sampling from an arbitrary distribution is not an easy problem!

- Computers can only generate samples from simple distributions.
- *E.g.*, such as a uniform distribution over $[0, 1]$ ⇒ Even then, these are *pseudorandom!*

A deterministic sequence with statistical properties (*e.g.*, running averages) similar to that of a true random sequence.

Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

Note: sampling from an arbitrary distribution is not an easy problem!

- Computers can only generate samples from simple distributions.
- *E.g.*, such as a uniform distribution over $[0, 1]$ ⇒ Even then, these are *pseudorandom!*
- ⇒ All (more-complex) sampling techniques involve calling some subroutine multiple times in a clever way.

A deterministic sequence with statistical properties (*e.g.*, running averages) similar to that of a true random sequence.

Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

For sampling from a categorical distribution:

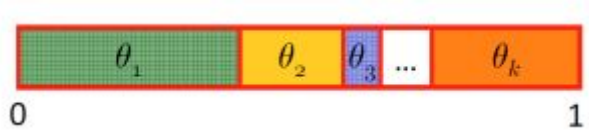
Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

For sampling from a categorical distribution:

- Reduce it to sampling from a uniform distribution.
- Where we subdivide unit interval into k regions, with region i having size θ_i.
- Then we sample uniformly from $[0, 1]$, and return the region in which our sample falls.



Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

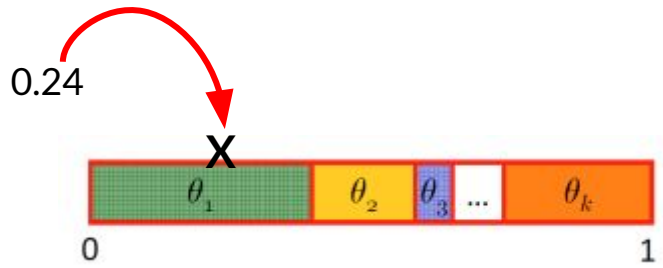
Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

For sampling from a categorical distribution:

- Reduce it to sampling from a uniform distribution.
- Where we subdivide unit interval into k regions, with region i having size θ_k.
- Then we sample uniformly from $[0, 1]$, and return the region in which our sample falls.



Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

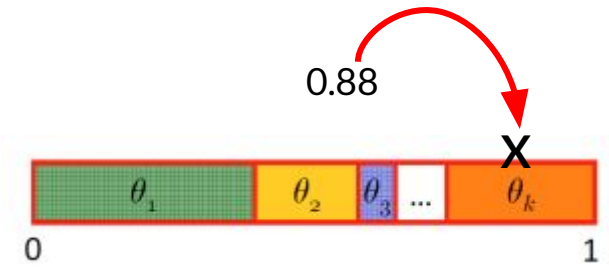
Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

For sampling from a categorical distribution:

- Reduce it to sampling from a uniform distribution.
- Where we subdivide unit interval into k regions, with region i having size θ_k.
- Then we sample uniformly from $[0, 1]$, and return the region in which our sample falls.



Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

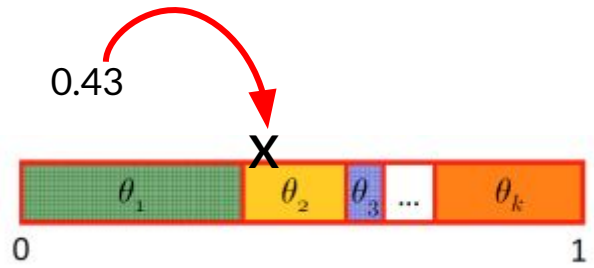
Sampling from a Probability Distribution

As a warmup: consider how we might sample from a *categorical distribution*.

E.g., with k possible outcomes and associated probabilities: $\theta_1, \ldots, \theta_k$.

For sampling from a categorical distribution:

- Reduce it to sampling from a uniform distribution.
- Where we subdivide unit interval into k regions, with region i having size θ_k.
- Then we sample uniformly from $[0, 1]$, and return the region in which our sample falls.



Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Forward Sampling from Joint PDF — For a given a PGM

How about if we have a more-complex, higher dimensional PDF? (*E.g.*, as a PGM).

Suppose we want to sample from this distribution (*i.e.*, joint PDF).

Can we take advantage of the PGM structure?

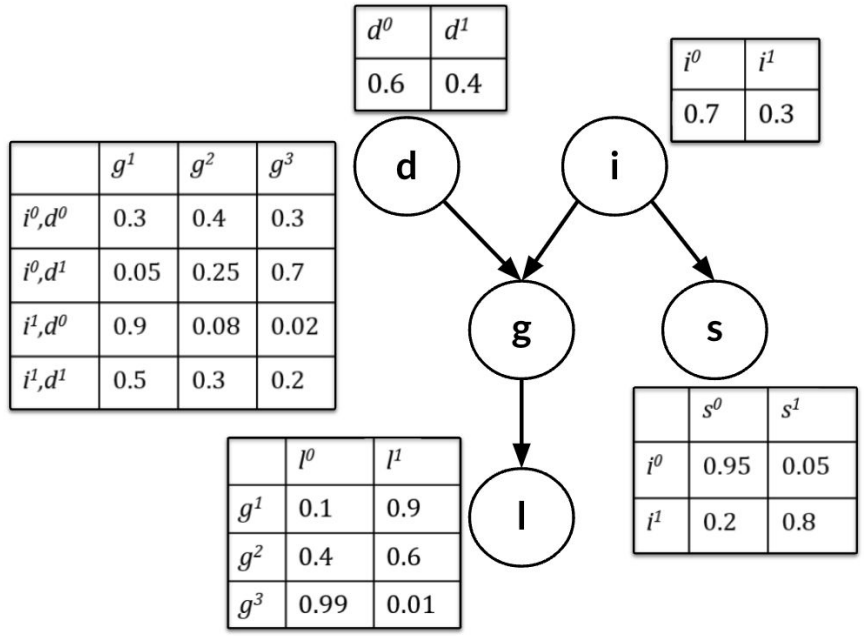
Forward Sampling from Joint PDF — For a given a PGM

Yes! We can sample from the joint PDF of a PGM (*e.g.*, Bayesian network) — often referred to as **forward sampling** — using the following procedure:

Forward Sampling from Joint PDF — For a given a PGM

Yes! We can sample from the joint PDF of a PGM (*e.g.*, Bayesian network) — often referred to as **forward sampling** — using the following procedure:

- Define topological order over nodes in network based on DAG.

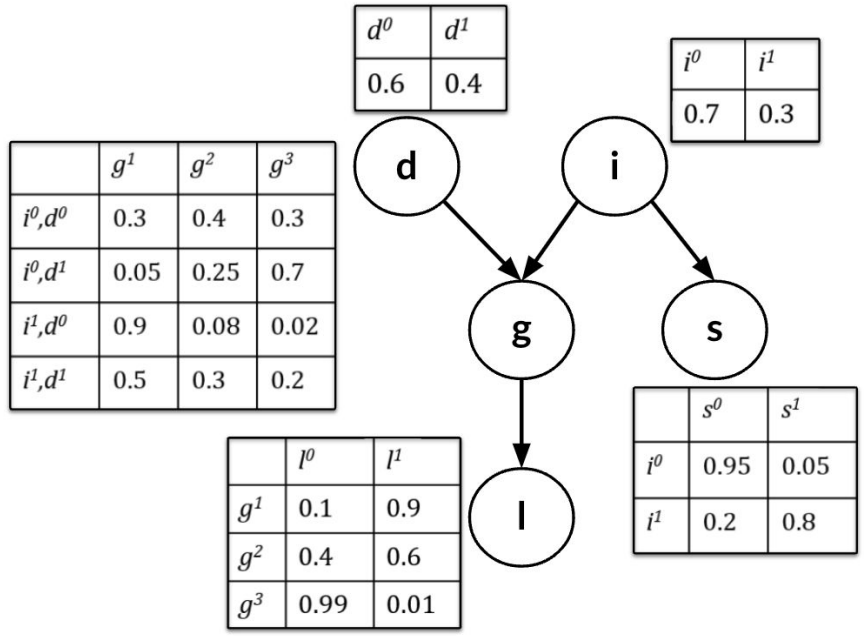


Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Forward Sampling from Joint PDF — For a given a PGM

Yes! We can sample from the joint PDF of a PGM (*e.g.*, Bayesian network) — often referred to as **forward sampling** — using the following procedure:

- Define topological order over nodes in network based on DAG.
- Sample from each conditional probability distribution in turn (*i.e.*, follow generative process).



Joint PDF:

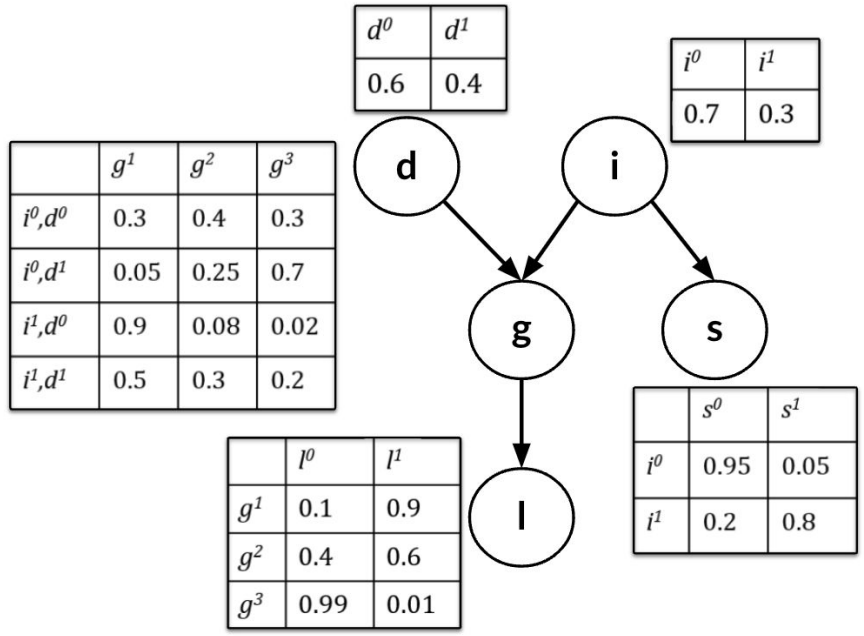
$$p(l, g, i, d, s) = p(d)\ p(i)\ p(s \mid i)\ p(g \mid i, d)\ p(l \mid g)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Forward Sampling from Joint PDF — For a given a PGM

Yes! We can sample from the joint PDF of a PGM (*e.g.*, Bayesian network) — often referred to as **forward sampling** — using the following procedure:

- Define topological order over nodes in network based on DAG.
- Sample from each conditional probability distribution in turn (*i.e.*, follow generative process).
- At each step, conditioning on the sampled values of the previous step.



Joint PDF:

$$p(l, g, i, d, s) = p(d)\ p(i)\ p(s \mid i)\ p(g \mid i, d)\ p(l \mid g)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Monte Carlo (MC) Estimation

Why do we want these samples and what do we typically do with them?

Monte Carlo (MC) Estimation

Samples from a distribution allow us to estimate expectations/integrals of the form:

$$\mathbb{E}_{x \sim p(x)}\left[g(x)\right] = \int_{\mathcal{X}} g(x)p(x)dx$$

$g(x)$ is some arbitrary "test function".

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Monte Carlo (MC) Estimation

Samples from a distribution allow us to estimate expectations/integrals of the form:

$$\mathbb{E}_{x \sim p(x)}\left[g(x)\right] = \int_{\mathcal{X}} g(x)p(x)dx$$

$g(x)$ is some arbitrary "test function".

In general, if $g(x)$ does not have specific form well-suited to $p(x)$ (or matching structure of Bayes net associated with $p(x)$) ⇒ not possible to perform analytically.

Monte Carlo (MC) Estimation

Samples from a distribution allow us to estimate expectations/integrals of the form:

$$\mathbb{E}_{x \sim p(x)}\left[g(x)\right] = \int_{\mathcal{X}} g(x)p(x)dx$$

$g(x)$ is some arbitrary "test function".

In general, if $g(x)$ does not have specific form well-suited to $p(x)$ (or matching structure of Bayes net associated with $p(x)$) ⇒ not possible to perform analytically.

Instead, we can approximate it with a large number of samples from $p(x)$.

⇒ Algorithms such as this are referred to as **Monte Carlo methods.**

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

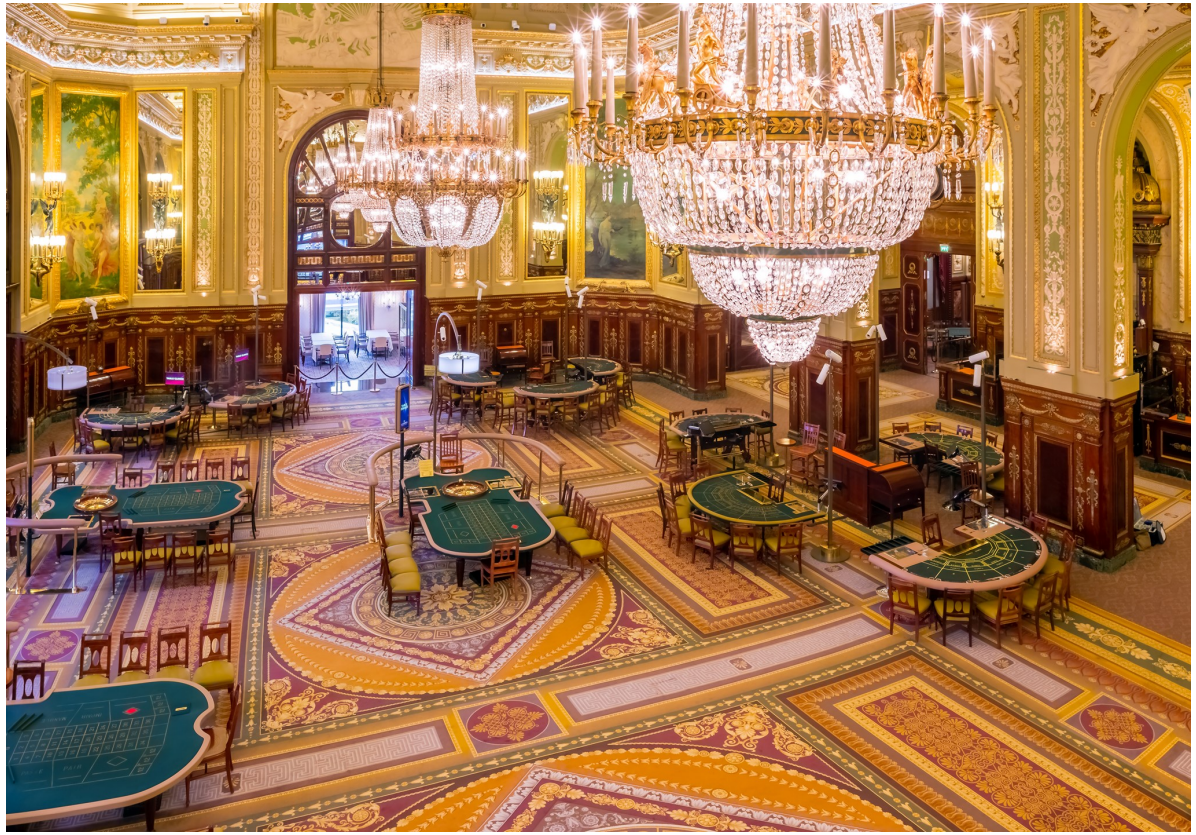
Monte Carlo (MC) Estimation

Backstory —

Source: Wikipedia

Monte Carlo (MC) Estimation

Backstory — the name comes from the Monte Carlo Casino in Monaco.

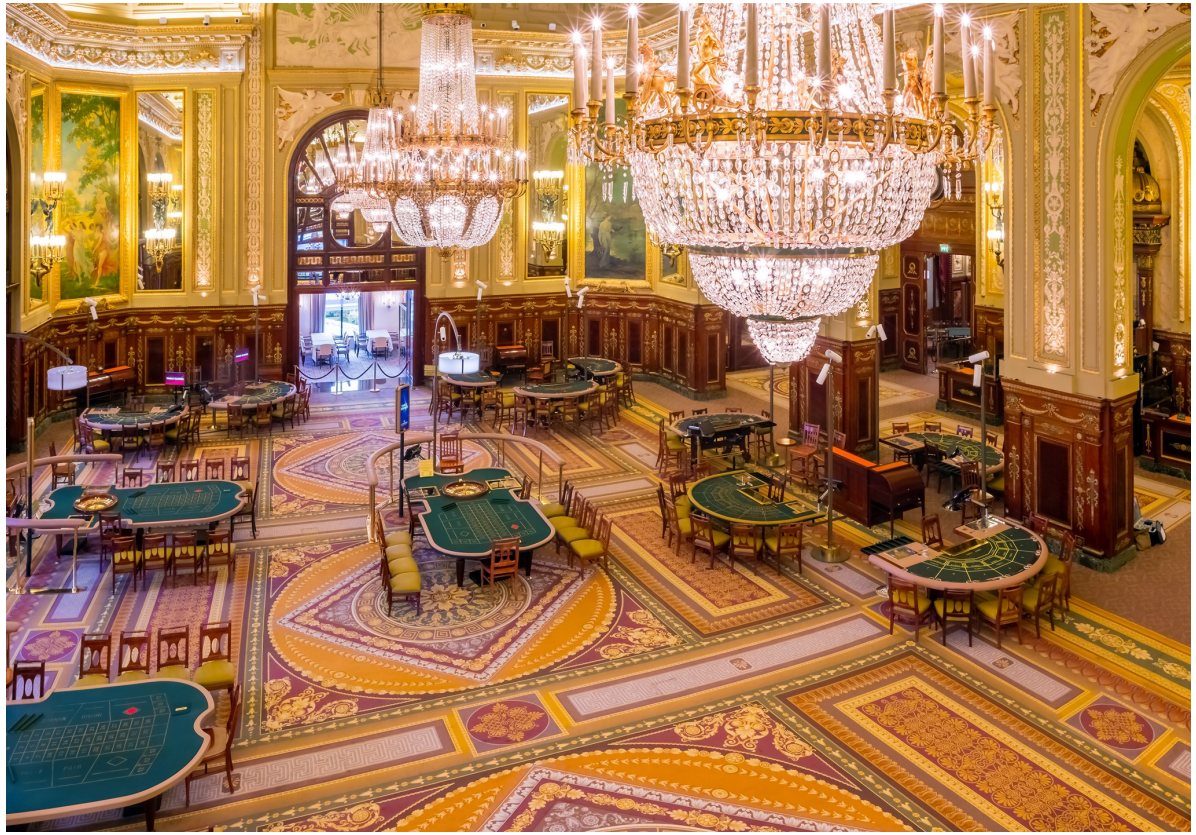


Source: Wikipedia

Monte Carlo (MC) Estimation

Backstory — the name comes from the Monte Carlo Casino in Monaco.

Primary developer of the method was Stanisław Ulam.



Source: Wikipedia

Monte Carlo (MC) Estimation

Backstory — the name comes from the Monte Carlo Casino in Monaco.

Primary developer of the method was Stanisław Ulam.

Developed in 1940s while working on nuclear weapons at Los Alamos.

Inspired by Ulam's uncle's gambling habits.

Along with *John von Neumann* and *Nicholas Metropolis* during early development of the ENIAC computer.

Monte Carlo (MC) Estimation

Backstory — the name comes from the Monte Carlo Casino in Monaco.

Primary developer of the method was Stanisław Ulam.

Developed in 1940s while working on nuclear weapons at Los Alamos.

Inspired by Ulam's uncle's gambling habits.

Along with *John von Neumann* and *Nicholas Metropolis* during early development of the ENIAC computer.

Also a professor at USC! (*briefly!*)

Monte Carlo (MC) Estimation

Going back to this integral:

$$\mathbb{E}_{x \sim p(x)}\left[g(x)\right] = \int_{\mathcal{X}} g(x)p(x)dx$$

Monte Carlo (MC) Estimation

Going back to this integral:

$$\mathbb{E}_{x\sim p(x)}\left[g(x)\right] = \int_{\mathcal{X}} g(x)p(x)dx$$

Monte Carlo sampling approximates this via:

$$\mathbb{E}_{x\sim p(x)}\left[g(x)\right] \approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(1)},\ldots,x^{(T)} \sim p(x)$$

where samples $x^{(1)},\ldots,x^{(T)} \sim p(x)$ are drawn i.i.d. from the distribution $p(x)$.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Monte Carlo (MC) Estimation

Going back to this integral:

$$\mathbb{E}_{x\sim p(x)}\left[g(x)\right] = \int_{\mathcal{X}} g(x)p(x)dx$$

Monte Carlo sampling approximates this via:

A "sample average".

$$\mathbb{E}_{x\sim p(x)}\left[g(x)\right] \approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(1)},\ldots,x^{(T)} \sim p(x)$$

where samples $x^{(1)},\ldots,x^{(T)} \sim p(x)$ are drawn i.i.d. from the distribution $p(x)$.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Monte Carlo (MC) Estimation

A couple of properties — it can be shown that:

Monte Carlo (MC) Estimation

A couple of properties — it can be shown that:

⇒ Unbiased estimate

$$\mathbb{E}_{x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} p(x)} \left[\frac{1}{T} \sum_{t=1}^{T} g(x^{(t)}) \right] = \mathbb{E}_{x \sim p(x)}[g(x)]$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Monte Carlo (MC) Estimation

A couple of properties — it can be shown that:

⇒ Unbiased estimate

$$\mathbb{E}_{x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} p(x)} \left[\frac{1}{T} \sum_{t=1}^{T} g(x^{(t)}) \right] = \mathbb{E}_{x \sim p(x)}[g(x)]$$

$$\text{Var}_{x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} p(x)} \left[\frac{1}{T} \sum_{t=1}^{T} g(x^{(t)}) \right] = \frac{1}{T} \text{Var}_{x \sim p(x)}[g(x)]$$

⇒ Variance can be made arbitrarily small

⇒ the *sample average* approaches the expectation as $T \to \infty$.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Rejection Sampling

How can we sample—or *compute expectations under*—more-complex distributions?

Rejection Sampling

How can we sample—or *compute expectations under*—more-complex distributions?

One strategy: a special case of MC estimation, known as **rejection sampling**.

Rejection Sampling

How can we sample—or *compute expectations under*—more-complex distributions?

One strategy: a special case of MC estimation, known as **rejection sampling**.

Intuition:

Compute the area of a region R by sampling from a larger region with a known area and recording the fraction of samples that falls within R.

Rejection Sampling

How can we sample—or *compute expectations under*—more-complex distributions?

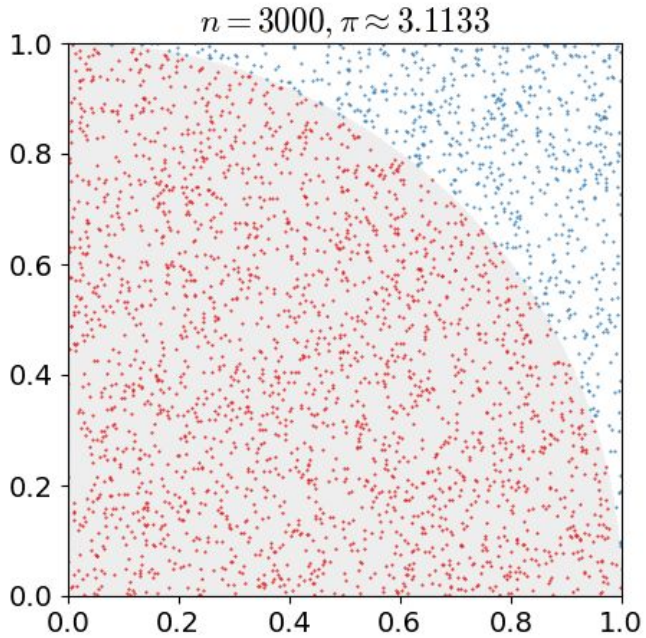
One strategy: a special case of MC estimation, known as **rejection sampling**.

Intuition:

Compute the area of a region R by sampling from a larger region with a known area and recording the fraction of samples that falls within R.

E.g., Approximating π: $(\pi r\text{^}2)/4$ vs $r\text{^}2$ →

Approximating π



Rejection Sampling

More formally:

Rejection Sampling

This method works even if we only know the unnormalized PDF $\tilde{p}(x)$.

More formally:

Suppose we have a (potentially) complex PDF, written $p(x) = \frac{1}{Z}\tilde{p}(x)$.

Rejection Sampling

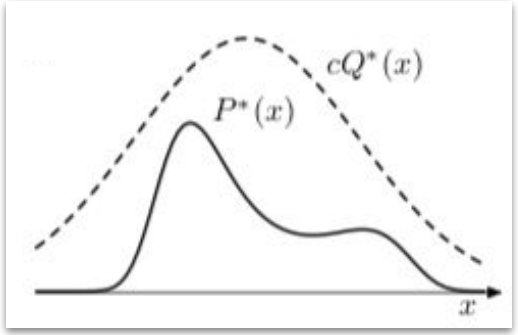
This method works even if we only know the unnormalized PDF $\tilde{p}(x)$.

More formally:

Suppose we have a (potentially) complex PDF, written $p(x) = \frac{1}{Z}\tilde{p}(x)$.

We can propose, and sample from, a (more tractable) *proposal density*, denoted $q(x)$, if we assume that we know a value of a constant c such that:

$$cq(x) > \tilde{p}(x), \quad \forall x$$



Source: Jesse Bettencourt, Probabilistic Machine Learning

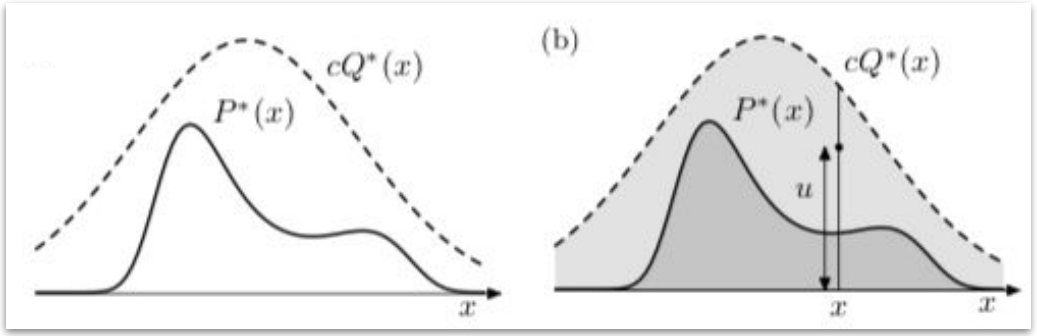
Rejection Sampling

Then you can do the following:

Rejection Sampling

Then you can do the following:

- Draw two random numbers:
  - First: $x \sim q(x)$.
  - Second: $u \sim Uniform([0, cq(x)])$

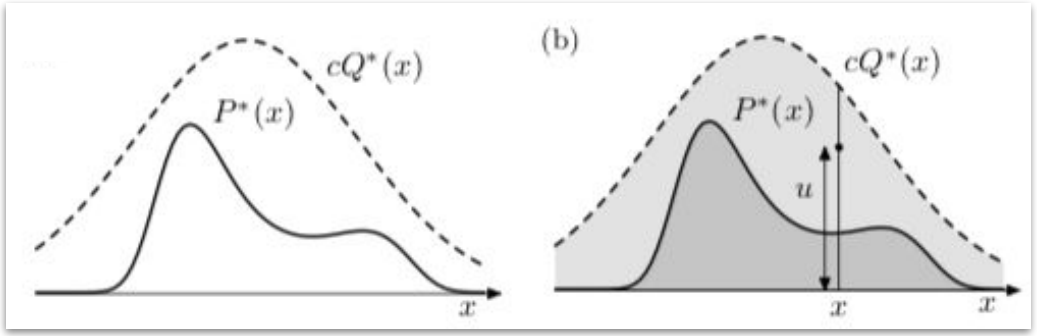


Source: Jesse Bettencourt, Probabilistic Machine Learning

Rejection Sampling

Then you can do the following:

- Draw two random numbers:
  - First: $x \sim q(x)$.
  - Second: $u \sim Uniform([0, cq(x)])$
- If $u \leq \tilde{p}(x)$:
  - Keep x.
- Otherwise, reject x.

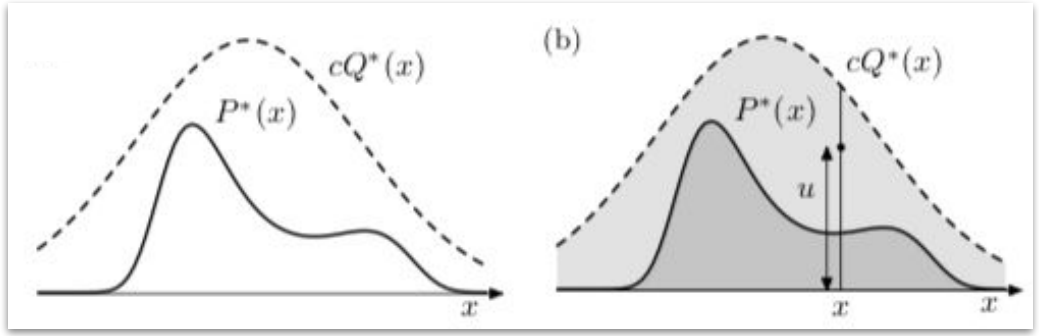


Source: Jesse Bettencourt, Probabilistic Machine Learning

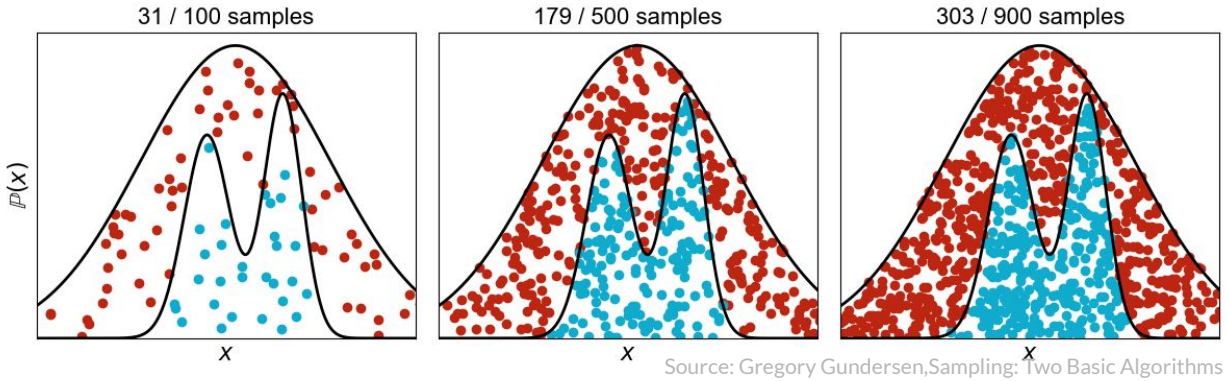
Rejection Sampling

Then you can do the following:

- Draw two random numbers:
  - First: $x \sim q(x)$.
  - Second: $u \sim Uniform([0, cq(x)])$
- If $u \leq \tilde{p}(x)$:
  - Keep x.
- Otherwise, reject x.



Source: Jesse Bettencourt, Probabilistic Machine Learning



Source: Gregory Gundersen,Sampling: Two Basic Algorithms

This samples uniformly over AUC ⇒ x is drawn proportional to PDF!

Rejection Sampling

Another example:

Rejection Sampling

Another example:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We can use rejection sampling to compute marginal probabilities $p(E = e)$.

Rejection Sampling

Another example:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We can use rejection sampling to compute marginal probabilities $p(E = e)$.

We can rewrite this marginal as:

$$p(E = e) = \sum_z p(Z = z, E = e) = \sum_x p(x)\mathbb{I}(E = e) = \mathbb{E}_{x \sim p}[\mathbb{I}(E = e)]$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Rejection Sampling

Another example:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We can use rejection sampling to compute marginal probabilities $p(E = e)$.

We can rewrite this marginal as:

$$p(E = e) = \sum_z p(Z = z, E = e) = \sum_x p(x)\mathbb{I}(E = e) = \mathbb{E}_{x \sim p}[\mathbb{I}(E = e)]$$

Definition of marginal

Rejection Sampling

Another example:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We can use rejection sampling to compute marginal probabilities $p(E = e)$.

We can rewrite this marginal as:

$$p(E = e) = \sum_z p(Z = z, E = e) = \sum_x p(x)\mathbb{I}(E = e) = \mathbb{E}_{x \sim p}[\mathbb{I}(E = e)]$$

Definition of marginal

Note: sum over x here

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Rejection Sampling

Another example:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We can use rejection sampling to compute marginal probabilities $p(E = e)$.

We can rewrite this marginal as:

$$p(E = e) = \sum_z p(Z = z, E = e) = \sum_x p(x)\mathbb{I}(E = e) = \mathbb{E}_{x \sim p}[\mathbb{I}(E = e)]$$

Definition of marginal | Note: sum over x here | Rejection sampling.

Rejection Sampling

Another example:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We can use rejection sampling to compute marginal probabilities $p(E = e)$.

We can rewrite this marginal as:

$$p(E = e) = \sum_z p(Z = z, E = e) = \sum_x p(x)\mathbb{I}(E = e) = \mathbb{E}_{x \sim p}[\mathbb{I}(E = e)]$$

Definition of marginal | Note: sum over x here | Rejection sampling.

⇒ draw many samples from p and keep the fraction that are consistent with E=e.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Rejection $\rightarrow$ Importance Sampling

Unfortunately, rejection sampling can be very wasteful (especially in higher dims).

E.g., if $p(E = e)$ equals 1% mass of joint, then we will discard 99% of all samples.

Rejection → Importance Sampling

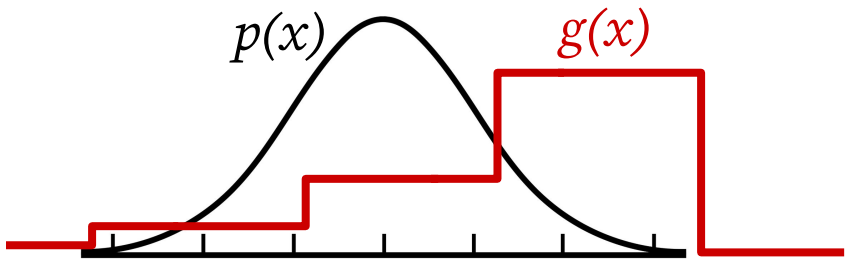
Unfortunately, rejection sampling can be very wasteful (especially in higher dims).

E.g., if $p(E = e)$ equals 1% mass of joint, then we will discard 99% of all samples.

A more efficient method of MC method for expectations ⇒ **importance sampling**.

Main idea:

- 1. Sample from a proposal distribution $q(x)$ (ideally with $q(x)$ roughly proportional to $g(x) \cdot p(x)$.
- 2. Reweight the samples (in a principled way), so that their sum still approximates the desired integral.



Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

More formally, suppose we want to estimate the expectation $\mathbb{E}_{x \sim p(x)}\left[g(x)\right]$.

Importance Sampling

More formally, suppose we want to estimate the expectation $\mathbb{E}_{x \sim p(x)}\left[g(x)\right]$.

We can rewrite this expectation as:

(Assuming discrete RVs, but same argument holds for continuous RVs.)

$$\mathbb{E}_{x \sim p(x)}[g(x)] = \sum_x g(x)p(x)$$

$$= \sum_x g(x)\frac{p(x)}{q(x)}q(x)$$

$$= \mathbb{E}_{x \sim q(x)}[g(x)w(x)], \quad \text{where } w(x) = \frac{p(x)}{q(x)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)})w(x^{(t)}), \quad \text{where } x^{(1)}, \ldots, x^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

More formally, suppose we want to estimate the expectation $\mathbb{E}_{x\sim p(x)}\left[g(x)\right]$.

We can rewrite this expectation as:

$$\mathbb{E}_{x\sim p(x)}[g(x)] = \sum_x g(x)p(x)$$

Definition of expected value

$$= \sum_x g(x)\frac{p(x)}{q(x)}q(x)$$

$$= \mathbb{E}_{x\sim q(x)}[g(x)w(x)], \quad \text{where } w(x) = \frac{p(x)}{q(x)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)})w(x^{(t)}), \quad \text{where } x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

More formally, suppose we want to estimate the expectation $\mathbb{E}_{x\sim p(x)}\left[g(x)\right]$.

We can rewrite this expectation as:

$$\mathbb{E}_{x\sim p(x)}[g(x)] = \sum_x g(x)p(x)$$

$$= \sum_x g(x)\frac{p(x)}{q(x)}q(x)$$

Multiply by *1* = *q(x)/q(x)*

$$= \mathbb{E}_{x\sim q(x)}[g(x)w(x)], \quad \text{where } w(x) = \frac{p(x)}{q(x)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)})w(x^{(t)}), \quad \text{where } x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Importance Sampling

More formally, suppose we want to estimate the expectation $\mathbb{E}_{x\sim p(x)}\left[g(x)\right]$.

We can rewrite this expectation as:

$$\mathbb{E}_{x\sim p(x)}[g(x)] = \sum_x g(x)p(x)$$

$$= \sum_x g(x)\frac{p(x)}{q(x)}q(x)$$

$$= \mathbb{E}_{x\sim q(x)}[g(x)w(x)], \qquad \text{where } w(x) = \frac{p(x)}{q(x)}$$

Rewrite as expectation *w.r.t. q(x)*

$$\approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)})w(x^{(t)}), \quad \text{where } x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

More formally, suppose we want to estimate the expectation $\mathbb{E}_{x\sim p(x)}\left[g(x)\right]$.

We can rewrite this expectation as:

$$\begin{aligned}
\mathbb{E}_{x\sim p(x)}[g(x)] &= \sum_x g(x)p(x) \\
&= \sum_x g(x)\frac{p(x)}{q(x)}q(x) \\
&= \mathbb{E}_{x\sim q(x)}[g(x)w(x)], \qquad \text{where } w(x) = \frac{p(x)}{q(x)} \\
&\approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)})w(x^{(t)}), \quad \text{where } x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} q(x)
\end{aligned}$$

MC Estimate!

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

More formally, suppose we want to estimate the expectation $\mathbb{E}_{x\sim p(x)}\left[g(x)\right]$.

We can rewrite this expectation as:

$$\begin{aligned}
\mathbb{E}_{x\sim p(x)}[g(x)] &= \sum_x g(x)p(x) \\
&= \sum_x g(x)\frac{p(x)}{q(x)}q(x) \\
&= \mathbb{E}_{x\sim q(x)}[g(x)w(x)], \qquad \text{where } w(x) = \frac{p(x)}{q(x)} \\
&\approx \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)})w(x^{(t)}), \quad \text{where } x^{(1)},\ldots,x^{(T)} \overset{\text{iid}}{\sim} q(x)
\end{aligned}$$

Sometimes referred to as an *IS estimate.*

MC Estimate!

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

The variance of this new random variable is:

Importance Sampling

Recall: $\mathrm{Var}\left[X\right] = \mathbb{E}\left[X^2\right] - \mathbb{E}\left[X\right]^2$

The variance of this new random variable is:

$$\mathrm{Var}_{x\sim q(x)}\left[g(x)w(x)\right] = \mathbb{E}_{x\sim q(x)}\left[g^2(x)w^2(x)\right] - \mathbb{E}_{x\sim q(x)}\left[g(x)w(x)\right]^2 \geq 0$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Recall: $\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$

The variance of this new random variable is:

$$\mathrm{Var}_{x\sim q(x)}[g(x)w(x)] = \mathbb{E}_{x\sim q(x)}[g^2(x)w^2(x)] - \mathbb{E}_{x\sim q(x)}[g(x)w(x)]^2 \geq 0$$

Note that this variance goes to 0 if we can set *q(x)* to be:

$$q(x) = \frac{|g(x)|p(x)}{\int |g(x)|p(x)dx}$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Recall: $\mathrm{Var}[X] = \mathbb{E}[X^2] - \mathbb{E}[X]^2$

The variance of this new random variable is:

$$\mathrm{Var}_{x\sim q(x)}[g(x)w(x)] = \mathbb{E}_{x\sim q(x)}[g^2(x)w^2(x)] - \mathbb{E}_{x\sim q(x)}[g(x)w(x)]^2 \geq 0$$

Note that this variance goes to 0 if we can set $q(x)$ to be:

$$q(x) = \frac{|g(x)|p(x)}{\int |g(x)|p(x)dx}$$

In practice: this is not constructive → the denominator is the quantity we want!

However, it gives intuition for what we want: high where both $g(x)$ and $p(x)$ are high.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Take our previous example from rejection sampling:

Importance Sampling

Take our previous example from rejection sampling:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We used rejection sampling to compute marginal probabilities $p(E = e)$.

Importance Sampling

Take our previous example from rejection sampling:

Suppose we have a Bayes Net over the set of (discrete) variables $X = Z \cup E$.

We used rejection sampling to compute marginal probabilities $p(E = e)$.

→ Recall that in rejection sampling, we simply throw away all samples from $p(x)$ where the variables in E did not take a value of e.

$$p(E = e) = \mathbb{E}_{x \sim p}[\mathbb{I}(E = e)]$$

"draw many samples from p and keep the fraction that are consistent with the value of the marginal."

Importance Sampling

Now let's do an *importance sampling* estimate for $p(E = e)$:

Importance Sampling

Now let's do an *importance sampling* estimate for $p(E = e)$:

Suppose we draw samples from $q(x)$, which we take to be a Uniform distribution.

$$\Rightarrow \quad p(e) = \int_z p(e \mid z)p(z)dz = \mathbb{E}_{z\sim p(z)}\left[p(e \mid z)\right]$$

$$= \mathbb{E}_{z\sim q(z)}\left[p(e \mid z)\frac{p(z)}{q(z)}\right] = \mathbb{E}_{z\sim q(z)}\left[\frac{p(e,z)}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q(z)}[w_e(z)], \qquad \text{where } w_e(z) = \frac{p(e,z)}{q(z)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} w_e(z^{(t)}), \quad \text{where } z^{(1)}, \ldots, z^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Now let's do an *importance sampling* estimate for $p(E = e)$:

Suppose we draw samples from $q(x)$, which we take to be a Uniform distribution.

$$\Rightarrow \quad p(e) = \int_z p(e \mid z)p(z)dz = \mathbb{E}_{z\sim p(z)}\left[p(e \mid z)\right]$$

Definition of joint/conditional

$$= \mathbb{E}_{z\sim q(z)}\left[p(e \mid z)\frac{p(z)}{q(z)}\right] = \mathbb{E}_{z\sim q(z)}\left[\frac{p(e,z)}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q(z)}[w_e(z)], \qquad \text{where } w_e(z) = \frac{p(e,z)}{q(z)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} w_e(z^{(t)}), \quad \text{where } z^{(1)}, \ldots, z^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Now let's do an *importance sampling* estimate for $p(E = e)$:

Suppose we draw samples from $q(x)$, which we take to be a Uniform distribution.

$$\Rightarrow \quad p(e) = \int_z p(e \mid z)p(z)dz = \mathbb{E}_{z\sim p(z)}\left[p(e \mid z)\right]$$

Multiply by 1, switch arguments in expectation (same as in RS)

$$= \mathbb{E}_{z\sim q(z)}\left[p(e \mid z)\frac{p(z)}{q(z)}\right] = \mathbb{E}_{z\sim q(z)}\left[\frac{p(e,z)}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q(z)}[w_e(z)], \quad \text{where } w_e(z) = \frac{p(e,z)}{q(z)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} w_e(z^{(t)}), \quad \text{where } z^{(1)}, \ldots, z^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Now let's do an *importance sampling* estimate for $p(E = e)$:

Suppose we draw samples from $q(x)$, which we take to be a Uniform distribution.

$$\Rightarrow \quad p(e) = \int_z p(e \mid z)p(z)dz = \mathbb{E}_{z\sim p(z)}\left[p(e \mid z)\right]$$

$$= \mathbb{E}_{z\sim q(z)}\left[p(e \mid z)\frac{p(z)}{q(z)}\right] = \mathbb{E}_{z\sim q(z)}\left[\frac{p(e,z)}{q(z)}\right]$$

Define weights w

$$= \mathbb{E}_{z\sim q(z)}[w_e(z)], \qquad \text{where } w_e(z) = \frac{p(e,z)}{q(z)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} w_e(z^{(t)}), \quad \text{where } z^{(1)}, \ldots, z^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Now let's do an *importance sampling* estimate for $p(E = e)$:

Suppose we draw samples from $q(x)$, which we take to be a Uniform distribution.

$$\Rightarrow \quad p(e) = \int_z p(e \mid z)p(z)dz = \mathbb{E}_{z\sim p(z)}\left[p(e \mid z)\right]$$

$$= \mathbb{E}_{z\sim q(z)}\left[p(e \mid z)\frac{p(z)}{q(z)}\right] = \mathbb{E}_{z\sim q(z)}\left[\frac{p(e,z)}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q(z)}[w_e(z)], \qquad \text{where } w_e(z) = \frac{p(e,z)}{q(z)}$$

Monte Carlo estimate!

$$\approx \frac{1}{T}\sum_{t=1}^{T} w_e(z^{(t)}), \quad \text{where } z^{(1)}, \ldots, z^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Importance Sampling

Now let's do an *importance sampling* estimate for $p(E = e)$:

Suppose we draw samples from $q(x)$, which we take to be a Uniform distribution.

$$\Rightarrow \quad p(e) = \int_z p(e \mid z)p(z)dz = \mathbb{E}_{z\sim p(z)}\left[p(e \mid z)\right]$$

$$= \mathbb{E}_{z\sim q(z)}\left[p(e \mid z)\frac{p(z)}{q(z)}\right] = \mathbb{E}_{z\sim q(z)}\left[\frac{p(e,z)}{q(z)}\right]$$

$$= \mathbb{E}_{z\sim q(z)}[w_e(z)], \qquad \text{where } w_e(z) = \frac{p(e,z)}{q(z)}$$

$$\approx \frac{1}{T}\sum_{t=1}^{T} w_e(z^{(t)}), \quad \text{where } z^{(1)}, \ldots, z^{(T)} \overset{\text{iid}}{\sim} q(x)$$

Unlike rejection sampling, this uses all of the samples when computing the estimate!

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC)

Markov Chain Monte Carlo (MCMC)

Suppose we want to sample from more complex distribution!

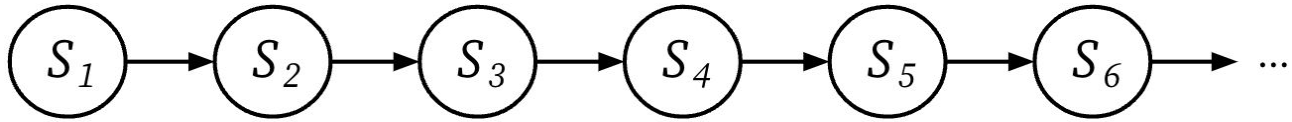
Markov chain Monte Carlo (**MCMC**) gives us a way to do this.

Markov Chain Monte Carlo (MCMC) — Markov Chains

Suppose we want to sample from more complex distribution!

Markov chain Monte Carlo (**MCMC**) gives us a way to do this.

Review on Markov chains:



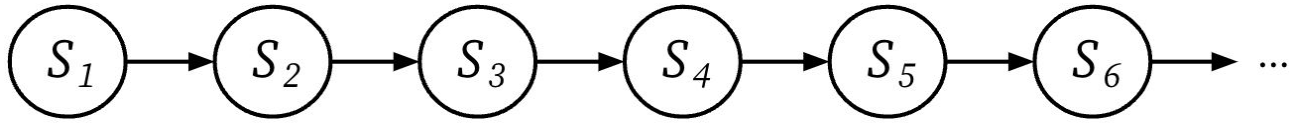
Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Suppose we want to sample from more complex distribution!

Markov chain Monte Carlo (**MCMC**) gives us a way to do this.

Review on Markov chains:



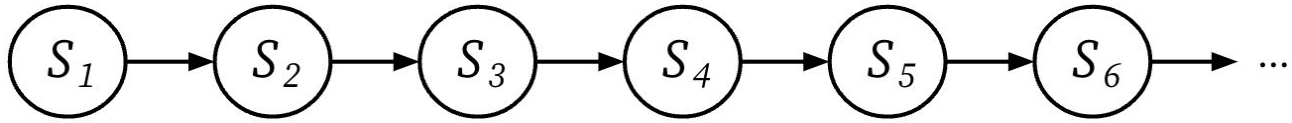
- A *(discrete-time) Markov chain* is a sequence of random variables:

Markov Chain Monte Carlo (MCMC) — Markov Chains

Suppose we want to sample from more complex distribution!

Markov chain Monte Carlo (**MCMC**) gives us a way to do this.

Review on Markov chains:



- A *(discrete-time) Markov chain* is a sequence of random variables:
- The initial state is distributed according to a probability $p(S_1)$.

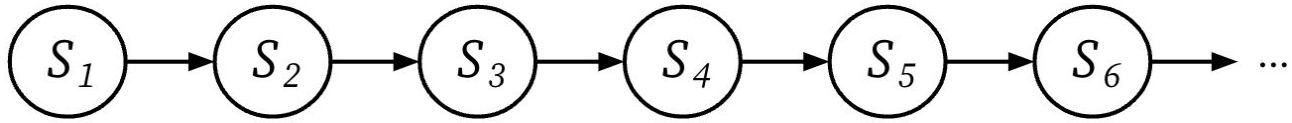
Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Suppose we want to sample from more complex distribution!

Markov chain Monte Carlo (**MCMC**) gives us a way to do this.

Review on Markov chains:



- A *(discrete-time) Markov chain* is a sequence of random variables:
- The initial state is distributed according to a probability $p(S_1)$.
- All subsequent states are conditionally distributed according to: $p(S_i \mid S_{i-1})$.
- Note that the conditional probability is same for each i …

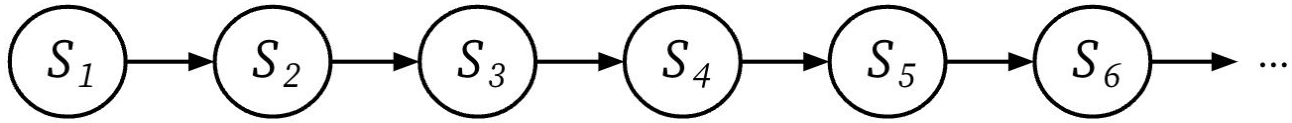
Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Suppose we want to sample from more complex distribution!

Markov chain Monte Carlo (**MCMC**) gives us a way to do this.

Review on Markov chains:



- A *(discrete-time) Markov chain* is a sequence of random variables:
- The initial state is distributed according to a probability $p(S_1)$.
- All subsequent states are conditionally distributed according to: $p(S_i \mid S_{i-1})$.
- Note that the conditional probability is same for each i …
- ⇒ the transition probabilities at any time depend only on the given state and not on the history of how we got there (“Markov assumption”).

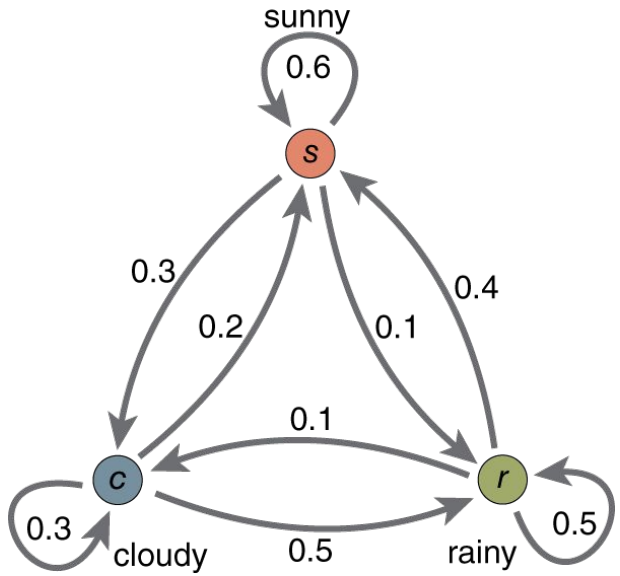
“Given the present, the future does not depend on the past”

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Example: Markov chain with discrete random variables.

⇒ Each random variable $S_i \in \{1, 2, \ldots, d\}$ taking one of d possible values.



Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

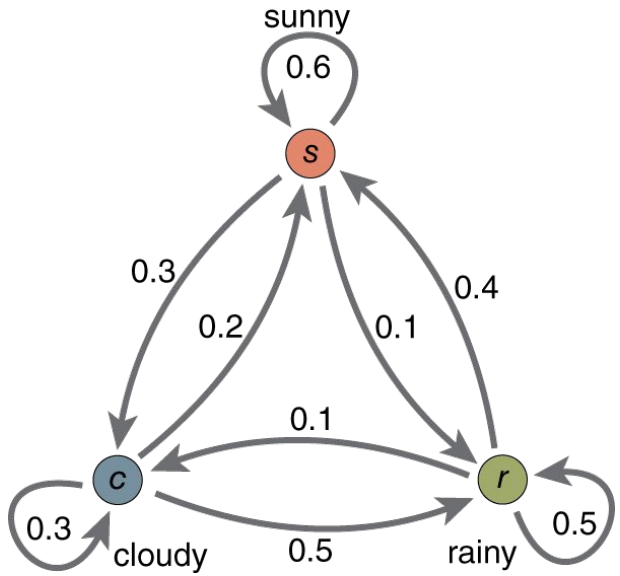
Markov Chain Monte Carlo (MCMC) — Markov Chains

Example: Markov chain with discrete random variables.

⇒ Each random variable $S_i \in \{1, 2, \ldots, d\}$ taking one of d possible values.

It is convenient to represent the conditional (*transition)* distribution as a d x d matrix:

$$T_{ij} = p(S_i = i \mid S_{i-1} = j)$$

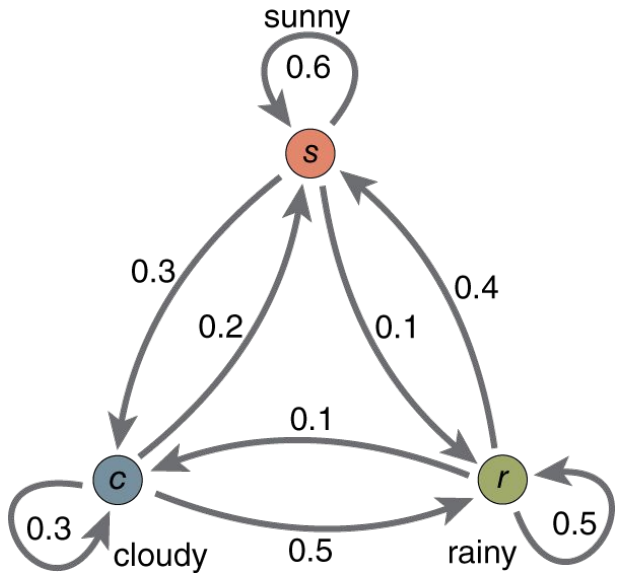


Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Example: Markov chain with discrete random variables.

⇒ Each random variable $S_i \in \{1, 2, \ldots, d\}$ taking one of d possible values.



It is convenient to represent the conditional (*transition)* distribution as a d x d matrix:

$$T_{ij} = p(S_i = i \mid S_{i-1} = j)$$

Suppose we write the initial state distribution $p(S_0)$ as a vector p_0.

⇒ We can write the probability p_t of ending up in a given state as:

$$p_t = T^t p_0$$

T^t is a matrix power (*i.e.*, apply the matrix operator t times)

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Stationary Distributions

Markov Chain Monte Carlo (MCMC) — Markov Chains

Stationary Distributions

The limit $\pi = \lim_{t \to \infty} p_t$ (when it exists) is called a *stationary distribution* of the MC.

Markov Chain Monte Carlo (MCMC) — Markov Chains

Stationary Distributions

The limit $\pi = \lim_{t \to \infty} p_t$ (when it exists) is called a *stationary distribution* of the MC.

Below we will construct Markov chains with the same stationary distribution for any initial distribution p_0.

⇒ So we often refer to it as ***the*** stationary distribution of the Markov chain.

Markov Chain Monte Carlo (MCMC) — Markov Chains

Detailed Balance

Markov Chain Monte Carlo (MCMC) — Markov Chains

Detailed Balance

(Part of) a sufficient condition for π to be a stationary distribution is a property called *detailed balance*:

$$\pi_i T_{ij} = \pi_j T_{ji} \quad \text{for all } i, j$$

Markov Chain Monte Carlo (MCMC) — Markov Chains

Detailed Balance

(Part of) a sufficient condition for π to be a stationary distribution is a property called *detailed balance*:

$$\pi_i T_{ij} = \pi_j T_{ji} \quad \text{for all } i, j$$

"The flow of probability between any two states is balanced"
(after sampling from stationary dist.)

Markov Chain Monte Carlo (MCMC) — Markov Chains

Detailed Balance

(Part of) a sufficient condition for π to be a stationary distribution is a property called *detailed balance*:

$$\pi_i T_{ij} = \pi_j T_{ji} \quad \text{for all } i, j$$

"The flow of probability between any two states is balanced"
(after sampling from stationary dist.)

(Note that inverse may not hold… *e.g.*, it is possible to derive valid Markov chains for MCMC algorithms without detailed balance).

Markov Chain Monte Carlo (MCMC) — Markov Chains

Also holds for **continuous** random variable MCs (w/ *transition distributions*)

Detailed Balance

(Part of) a sufficient condition for $\pi(x)$ to be a stationary distribution is a property called *detailed balance*:

$$\pi(x')T(x \mid x') = \pi(x)T(x' \mid x) \quad \text{for all } x$$

(Note that inverse may not hold… *e.g.*, it is possible to derive valid Markov chains for MCMC algorithms without detailed balance).

Markov Chain Monte Carlo (MCMC) — Markov Chains

Ergodicity

Additional sufficient conditions for existence of such a Markov chain:

Markov Chain Monte Carlo (MCMC) — Markov Chains

Ergodicity

Additional sufficient conditions for existence of such a Markov chain:

- *Irreducibility*:
  - It is possible to get from any state to any other state with probability > 0, in a finite number of steps.
- *Aperiodicity*:
  - It is possible to return to any (previously sampled) state at any time.

In continuous variable MC's these conditions are often referred* to as *ergodicity*.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Ergodicity

Additional sufficient conditions for existence of such a Markov chain:

- *Irreducibility*:

  "It's possible to transition from any state to any other state"

  - It is possible to get from any state to any other state with probability > 0, in a finite number of steps.

- *Aperiodicity*:
  - It is possible to return to any (previously sampled) state at any time.

In continuous variable MC's these conditions are often referred* to as *ergodicity*.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Markov Chains

Ergodicity

Additional sufficient conditions for existence of such a Markov chain:

- *Irreducibility*: "It's possible to transition from any state to any other state"
  - It is possible to get from any state to any other state with probability > 0, in a finite number of steps.
- *Aperiodicity*: "Avoids transitioning between sets of states in a cycle"
  - It is possible to return to any (previously sampled) state at any time.

In continuous variable MC's these conditions are often referred* to as *ergodicity*.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Overview

High level idea of MCMC:

Markov Chain Monte Carlo (MCMC) — Overview

High level idea of MCMC:

- Given a probability distribution of interest, $p(x)$.

E.g., posterior distribution or marginal distribution of a probability model

Markov Chain Monte Carlo (MCMC) — Overview

High level idea of MCMC:

- Given a probability distribution of interest, $p(x)$.
- Construct a Markov chain.

E.g., posterior distribution or marginal distribution of a probability model

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Overview

High level idea of MCMC:

- Given a probability distribution of interest, $p(x)$.
- Construct a Markov chain.
- Whose states are assignments of all variables x (jointly) in the probability distribution.

E.g., posterior distribution or marginal distribution of a probability model

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Overview

High level idea of MCMC:

E.g., posterior distribution or marginal distribution of a probability model

- Given a probability distribution of interest, $p(x)$.
- Construct a Markov chain.
- Whose states are assignments of all variables x (jointly) in the probability distribution.
- And whose stationary distribution equals the probability distribution $p(x)$.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Overview

High level idea of MCMC:

- Given a probability distribution of interest, $p(x)$.
- Construct a Markov chain.
- Whose states are assignments of all variables x (jointly) in the probability distribution.
- And whose stationary distribution equals the probability distribution $p(x)$.

E.g., posterior distribution or marginal distribution of a probability model

⇒ then we can sample from this Markov chain, and iteratively get closer to samples from the probability distribution (as we approach stationary distribution).

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Overview

Thus, in an MCMC algorithm you typically perform the following two steps:

(Given as input: a transition matrix/distribution T, and an initial assignment of variables, x_0).

Markov Chain Monte Carlo (MCMC) — Overview

Thus, in an MCMC algorithm you typically perform the following two steps:

(Given as input: a transition matrix/distribution T, and an initial assignment of variables, x_0).

- 1. Sample from the Markov chain, starting with x_0, for B *burn-in* steps.
- 2. Sample from the Markov chain for N *sampling steps*, and record each of these samples.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Overview

Thus, in an MCMC algorithm you typically perform the following two steps:

(Given as input: a transition matrix/distribution T, and an initial assignment of variables, x_0).

- 1. Sample from the Markov chain, starting with x_0, for B *burn-in* steps.
- 2. Sample from the Markov chain for N *sampling steps*, and record each of these samples.

Assuming that B is sufficiently large, the latter set of N samples will approximately be drawn from $p(x)$.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Markov Chain Monte Carlo (MCMC) — Algorithms

We'll review through a few different MCMC algorithms following these steps:

Markov Chain Monte Carlo (MCMC) — Algorithms

We'll review through a few different MCMC algorithms following these steps:

- Metropolis-Hastings (MH) — a "core" MCMC algorithm
- Gibbs Sampling — can be viewed as a special case of MH.
- Langevin Monte Carlo — can be viewed as a *gradient-based* case of MH.
- Hamiltonian Monte Carlo — can be viewed as a *better gradient-based* case of MH.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Metropolis-Hastings (MH)

Metropolis-Hastings (MH)

The **Metropolis Hastings Algorithm** constructs a transition distribution $T(x' \mid x)$ using two components:

- A user-specified *transition kernel* $Q(x' \mid x)$.
- An *acceptance probability* for moves proposed by Q, specified by the algorithm as:

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right).$$

Often called the *acceptance ratio.*

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Metropolis-Hastings (MH)

The **Metropolis Hastings Algorithm** constructs a transition distribution $T(x' \mid x)$ using two components:

- A user-specified *transition kernel* $Q(x' \mid x)$.
- An *acceptance probability* for moves proposed by Q, specified by the algorithm as:

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right).$$

Often called the *acceptance ratio.*

At each step of the MH, we sample $x' \sim Q(x' \mid x)$ and then we either *accept* it with probability $\alpha = A(x' \mid x)$ or reject it and remain at our previous state.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Metropolis-Hastings (MH)

The **Metropolis Hastings Algorithm** constructs a transition distribution $T(x' \mid x)$ using two components:

- A user-specified *transition kernel* $Q(x' \mid x)$.
- An *acceptance probability* for moves proposed by Q, specified by the algorithm as:

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right).$$

Often called the *acceptance ratio.*

Important: $p(x)$, $p(x')$ can be unnormalized PDF! ⇒ easier to apply.

At each step of the MH, we sample $x' \sim Q(x' \mid x)$ and then we either *accept* it with probability $\alpha = A(x' \mid x)$ or reject it and remain at our previous state.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Metropolis-Hastings (MH)

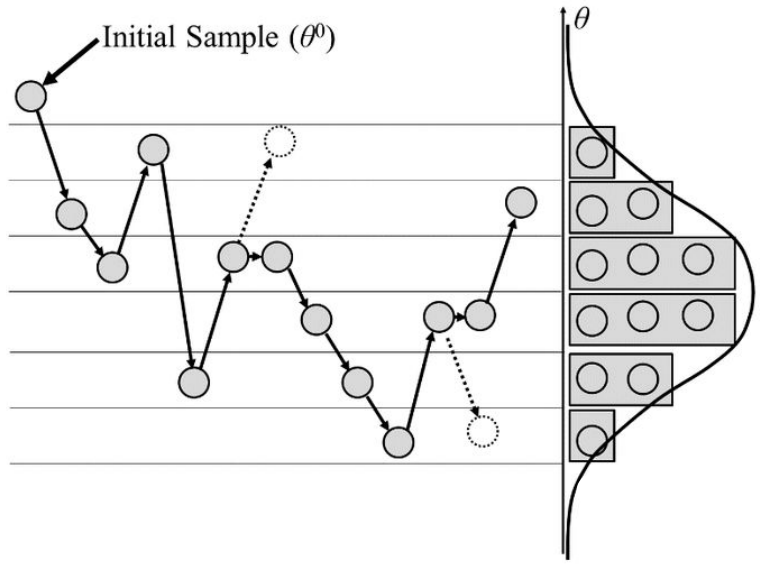
Intuition:

Metropolis-Hastings (MH)

Intuition:

The acceptance probability encourages us to move towards more-likely (higher-mass) regions of the distribution $p(x)$.

When Q suggests we move into a low-probability region, we follow that move only a fraction of the time.



Source: Wikipedia, "Metropolis-Hastings Algorithm"

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Metropolis-Hastings (MH)

Intuition:

In practice, the distribution $Q(x' \mid x)$ is often taken to be something simple, like a Gaussian centered at x (for continuous variables).

Metropolis-Hastings (MH)

Intuition:

In practice, the distribution $Q(x' \mid x)$ is often taken to be something simple, like a Gaussian centered at x (for continuous variables).

Note that when $Q(x' \mid x) = Q(x \mid x')$, a symmetric distribution (*e.g.*, Gaussians), then the acceptance ratio simplifies to:

$$A(x' \mid x) = \min\left(1, \frac{p(x')}{p(x)}\right).$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Metropolis-Hastings (MH)

Intuition:

In practice, the distribution $Q(x' \mid x)$ is often taken to be something simple, like a Gaussian centered at x (for continuous variables).

Note that when $Q(x' \mid x) = Q(x \mid x')$, a symmetric distribution (*e.g.*, Gaussians), then the acceptance ratio simplifies to:

$$A(x' \mid x) = \min\left(1, \frac{p(x')}{p(x)}\right).$$

⇒ *Metropolis* Algorithm (symmetric proposals)

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Metropolis-Hastings (MH)

Backstory:

Metropolis-Hastings (MH)

Backstory:

This variant of the algorithm (*Metropolis Algorithm*) originally published in 1953!

THE JOURNAL OF CHEMICAL PHYSICS VOLUME 21, NUMBER 6 JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

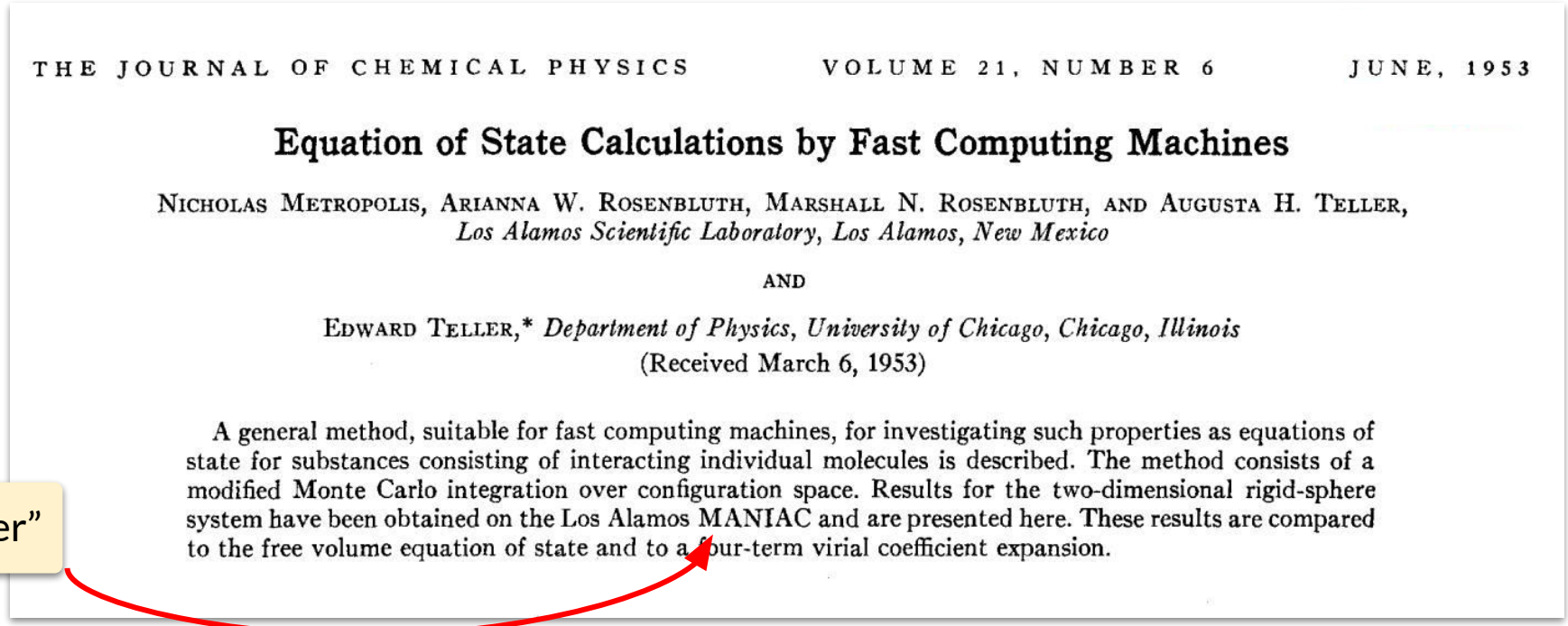
(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

Metropolis-Hastings (MH)

Backstory:

This variant of the algorithm (*Metropolis Algorithm*) originally published in 1953!



THE JOURNAL OF CHEMICAL PHYSICS VOLUME 21, NUMBER 6 JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

(Received March 6, 1953)

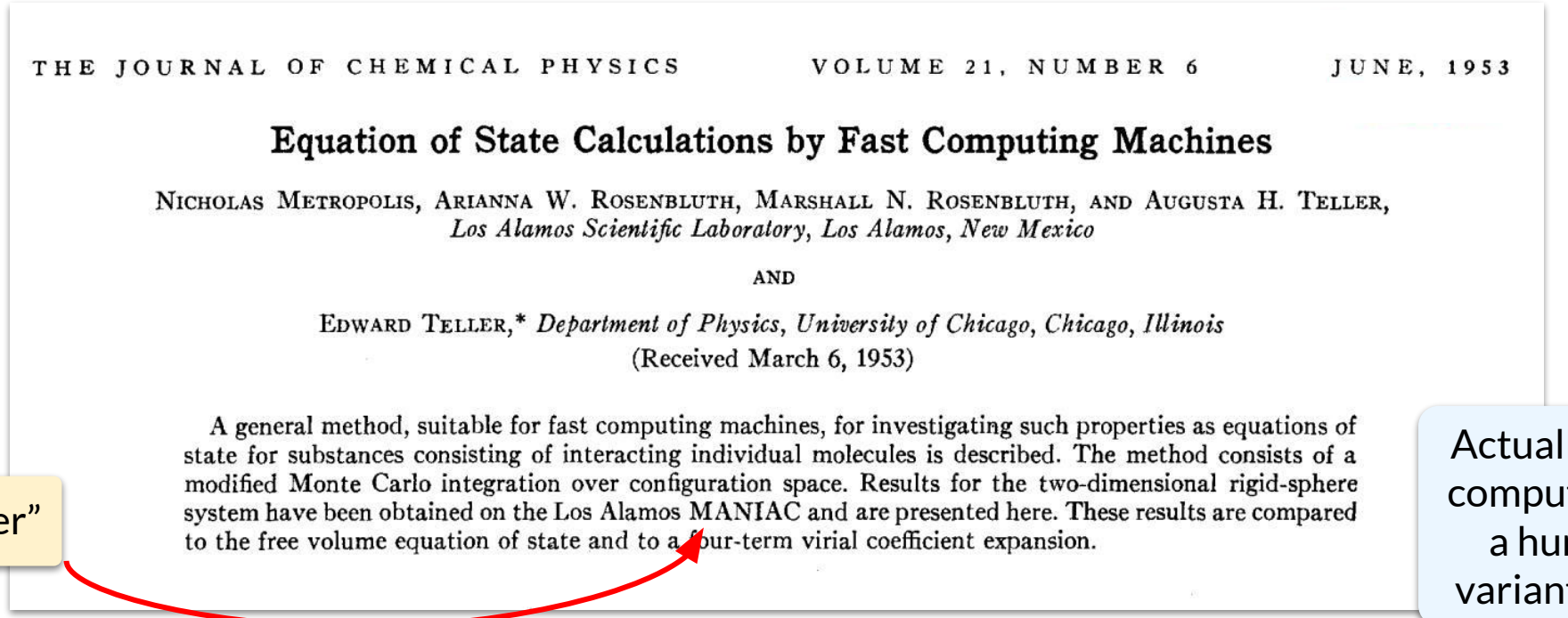
A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

"Fast computer"

Metropolis-Hastings (MH)

Backstory:

This variant of the algorithm (*Metropolis Algorithm*) originally published in 1953!



THE JOURNAL OF CHEMICAL PHYSICS VOLUME 21, NUMBER 6 JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

"Fast computer"

Actually, the first computer to beat a human at a variant of chess!

Metropolis-Hastings (MH)

Animation of Metropolis Hastings Algorithm:

Source: Stata Blog

Metropolis-Hastings (MH)

Animation of Metropolis Hastings Algorithm:

Uses a Gaussian proposal distribution to sample from a more complex distribution!

(In this case, a Beta(5, 7) distribution).

Source: Stata Blog

Metropolis-Hastings (MH)

Animation of Metropolis Hastings Algorithm:

Uses a Gaussian proposal distribution to sample from a more complex distribution!

(In this case, a Beta(5, 7) distribution).

Notice that subsequent samples are (slightly) correlated.

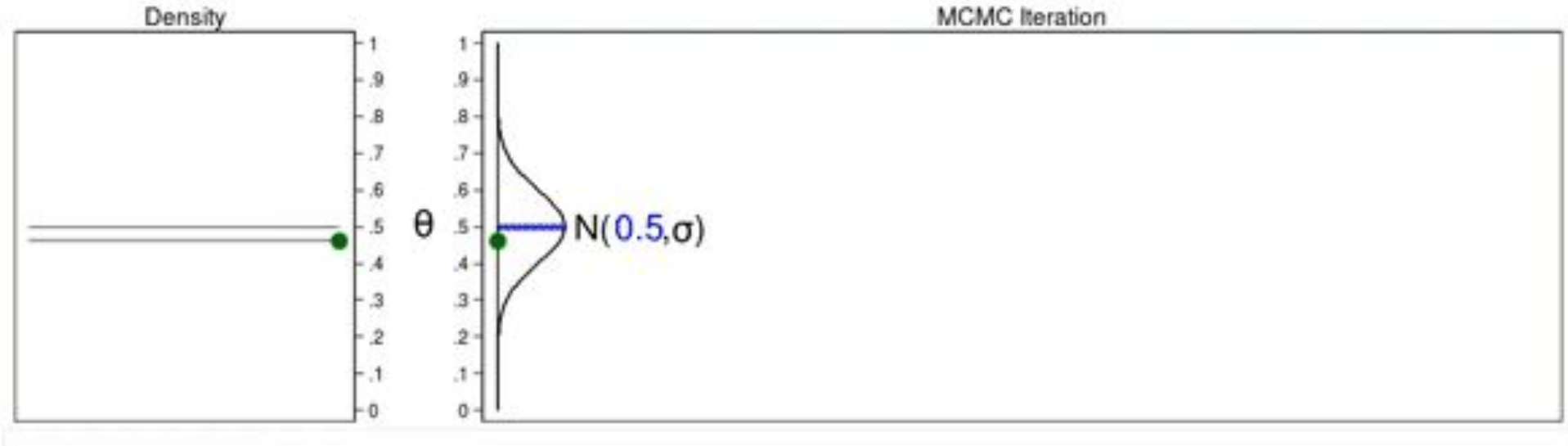
Source: Stata Blog

Metropolis-Hastings (MH)

Animation of ~~Metropolis Hastings Algorithm~~:
regular sampling from a simple distribution.

Compare that with exact sampling.

E.g., from a Gaussian distribution.



Here, all samples are independent.

Source: Stata Blog

Quick Note on “Posterior Sampling”

MCMC is often synonymous with “posterior sampling” (drawing samples from a posterior distribution).

However: methods so far have not really been posterior-distribution specific!

Quick Note on “Posterior Sampling”

MCMC is often synonymous with “posterior sampling” (drawing samples from a posterior distribution).

However: methods so far have not really been posterior-distribution specific!

⇒ They just require: access to (unnormalized) PDF from which we want to sample.

⇒ Which is well suited to posterior sampling (but also to other inference as well).

Gibbs Sampling

Gibbs Sampling

A widely-used special case of MH is known as **Gibbs Sampling**.

Think of it as an *"axially aligned, carefully designed, transition function that always yields an acceptance ratio of 1"*.

Gibbs Sampling

A widely-used special case of MH is known as **Gibbs Sampling**.

Think of it as an *"axially aligned, carefully designed, transition function that always yields an acceptance ratio of 1"*.

Why is it useful if the acceptance ratio is high?

- Less wasted samples!
- In high dimensional space, a bad (or even just simple) transition distribution can lead to very low acceptance ratio…

⇒ bad numerical properties!

Gibbs Sampling — Algorithm

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

For each sample…

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

Set buffer to previous sample

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

For each variable (*dimension*)

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

Sample from conditional of $p(x)$
($x_{i\text{-}1}$ denotes "*all other variables*")

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

Replace variable (*dimension*) with sample

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

Set next sample to buffer

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling — Algorithm

Suppose we have an ordered set of variables in our model: $x_1, \ldots, x_n$

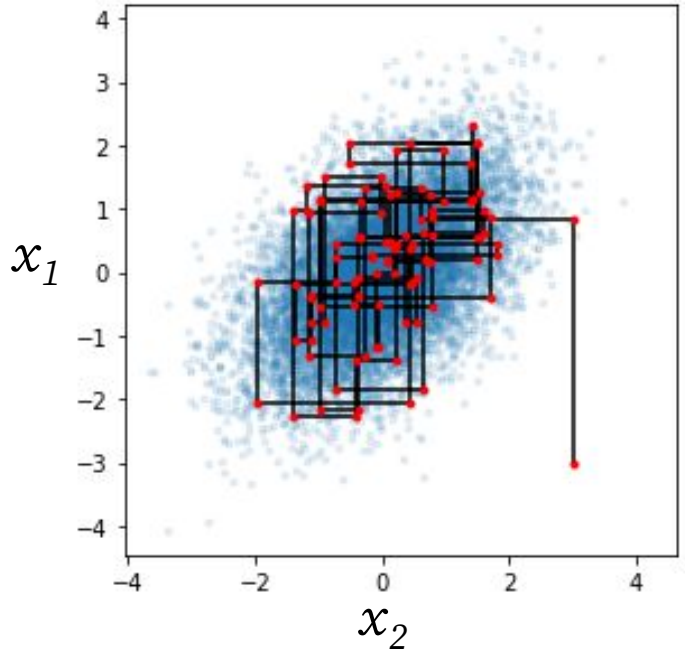
And an initial configuration: $x^0 = (x_1^0, \ldots, x_n^0)$

Then carry out the following procedure (to generate samples from $p(x)$):

Repeat until convergence for $t = 1, 2, 3, \ldots$

- Set $x \leftarrow x^{t-1}$
- For each variable $x_i, \quad i = 1, \ldots, n$
  - Sample $x' \sim p(x_i \mid x_{i-1})$
  - Update $x \leftarrow (x_1, \ldots, x_i', \ldots, x_n)$
- Set $x^t \leftarrow x$

Looks like this:



Source: Jessica Stringham, Gibbs Sampling in Python

Gibbs Sampling

But if this is a special case of MH, where is the acceptance ratio?

Gibbs Sampling

But if this is a special case of MH, where is the acceptance ratio?

⇒ you can view this as a special case of MH with proposal:

$$Q(x' \mid x) = Q(x'_i, x_{-i} \mid x_i, x_{-i}) = p(x'_i \mid x_{-i})$$

Gibbs Sampling

But if this is a special case of MH, where is the acceptance ratio?

⇒ you can view this as a special case of MH with proposal:

$$Q(x' \mid x) = Q(x'_i, x_{-i} \mid x_i, x_{-i}) = p(x'_i \mid x_{-i})$$

If you plug this into the acceptance ratio formula, you can see that the numerator and denominator cancel out!

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling

But if this is a special case of MH, where is the acceptance ratio?

⇒ you can view this as a special case of MH with proposal:

$$Q(x' \mid x) = Q(x'_i, x_{-i} \mid x_i, x_{-i}) = p(x'_i \mid x_{-i})$$

If you plug this into the acceptance ratio formula, you can see that the numerator and denominator cancel out!

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right)$$

$$\Rightarrow \quad A(x' \mid x) = \min\left(1, \frac{p(x'_i \mid x'_{-i})p(x'_{-i})\ p(x_i \mid x'_{-i})}{p(x_i \mid x_{-i})p(x_{-i})\ p(x'_i \mid x_{-i})}\right)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling

But if this is a special case of MH, where is the acceptance ratio?

⇒ you can view this as a special case of MH with proposal:

$$Q(x' \mid x) = Q(x'_i, x_{-i} \mid x_i, x_{-i}) = p(x'_i \mid x_{-i})$$

If you plug this into the acceptance ratio formula, you can see that the numerator and denominator cancel out!

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right)$$

$$\Rightarrow \quad A(x' \mid x) = \min\left(1, \frac{\cancel{p(x'_i \mid x'_{-i})}p(x'_{-i})\; p(x_i \mid x'_{-i})}{p(x_i \mid x_{-i})p(x_{-i})\cancel{p(x'_i \mid x_{-i})}}\right)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling

But if this is a special case of MH, where is the acceptance ratio?

⇒ you can view this as a special case of MH with proposal:

$$Q(x' \mid x) = Q(x'_i, x_{-i} \mid x_i, x_{-i}) = p(x'_i \mid x_{-i})$$

If you plug this into the acceptance ratio formula, you can see that the numerator and denominator cancel out!

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right)$$

$$\Rightarrow A(x' \mid x) = \min\left(1, \frac{\cancel{p(x'_i \mid x'_{-i})}p(x'_{-i})\cancel{p(x_i \mid x'_{-i})}}{\cancel{p(x_i \mid x_{-i})}p(x_{-i})\cancel{p(x'_i \mid x_{-i})}}\right)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling

But if this is a special case of MH, where is the acceptance ratio?

⇒ you can view this as a special case of MH with proposal:

$$Q(x' \mid x) = Q(x'_i, x_{-i} \mid x_i, x_{-i}) = p(x'_i \mid x_{-i})$$

If you plug this into the acceptance ratio formula, you can see that the numerator and denominator cancel out!

$$A(x' \mid x) = \min\left(1, \frac{p(x')Q(x \mid x')}{p(x)Q(x' \mid x)}\right)$$

$$\Rightarrow A(x' \mid x) = \min\left(1, \frac{\cancel{p(x'_i \mid x'_{-i})p(x'_{-i})}\ \cancel{p(x_i \mid x'_{-i})}}{\cancel{p(x_i \mid x_{-i})p(x_{-i})}\ \cancel{p(x'_i \mid x_{-i})}}\right)$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Gibbs Sampling

One important question: how to sample from the conditional $x' \sim p(x_i \mid x_{i-1})$?

Gibbs Sampling

One important question: how to sample from the conditional $x' \sim p(x_i \mid x_{i-1})$?

⇒ Gibbs sampling does not provide a general-purpose way to sample this!

⇒ *"It's up to you to figure out"* :D

Gibbs Sampling

One important question: how to sample from the conditional $x' \sim p(x_i \mid x_{i-1})$?

⇒ Gibbs sampling does not provide a general-purpose way to sample this!

⇒ *"It's up to you to figure out"* :D

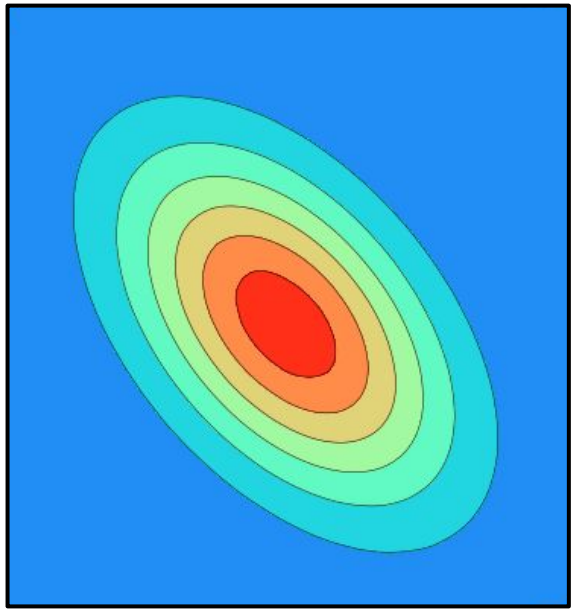
In some cases:

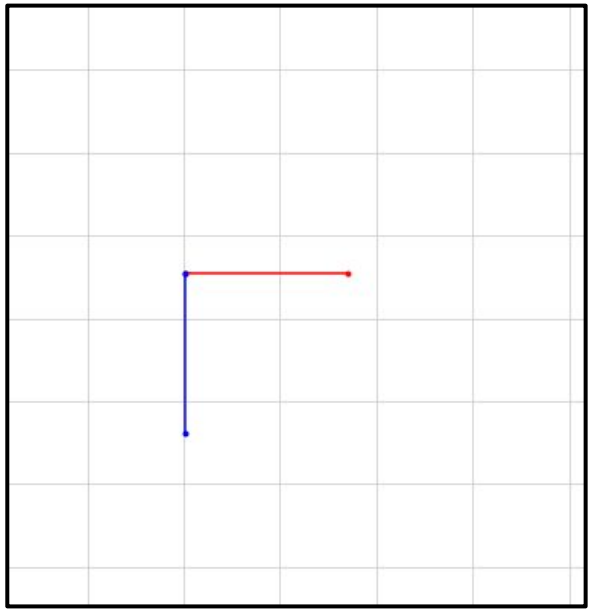
- You can sample from this conditional in closed form (*in certain models*, *e.g.*, *LDA*).
- You run a subroutine to sample from this conditional.
  - As an example: Metropolis-Hastings algorithm!
  - Sometimes this "fixes" low acceptance rate issues, since MH is now over only *one variable* (much lower dimensionality).

Gibbs Sampling

Animation:

Gibbs sampling in practice!





Source: Sandipan Dey, Bayesian Machine Learning

Gradient Based MCMC

Gradient Based MCMC

One final strategy to avoid low acceptance rate in Metropolis-Hastings (particularly in high dimensions) — using gradients ✨.

Gradient Based MCMC

One final strategy to avoid low acceptance rate in Metropolis-Hastings (particularly in high dimensions) — using gradients ✨.

Background — the issue with simple proposals in MH:

Gradient Based MCMC

One final strategy to avoid low acceptance rate in Metropolis-Hastings (particularly in high dimensions) — using gradients ✨.

Background — the issue with simple proposals in MH:

- You could achieve a higher acceptance rate by just taking small steps (*e.g.*, use a symmetric distribution with small variance centered at previous sample).
- *However*, small steps ⇒ bad mixing and very slow progress of the algorithm (*i.e.*, high correlation ⇒ need many more samples for same statistical properties).

Gradient Based MCMC

A few popular MCMC methods leverage the *gradient of the PDF* in order to propose a sample in a high mass region from potentially far away in space.

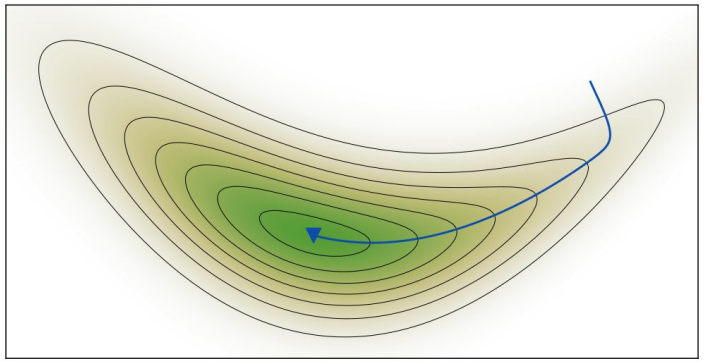
(These methods original came from the field of quantum chromodynamics)

Gradient Based MCMC

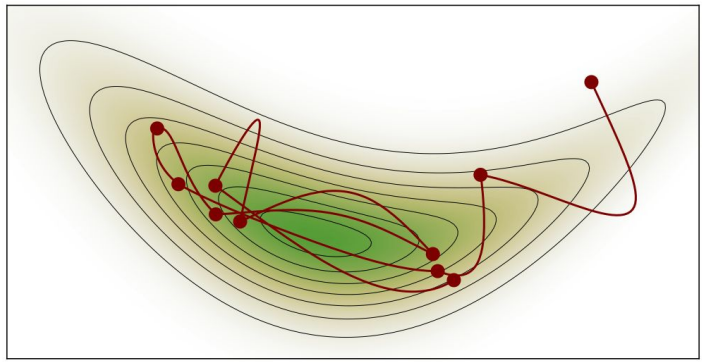
A few popular MCMC methods leverage the *gradient of the PDF* in order to propose a sample in a high mass region from potentially far away in space.

(These methods original came from the field of quantum chromodynamics)

They are called: **Langevin Monte Carlo** (LMC) and **Hamiltonian Monte Carlo** (HMC).



Gradient Descent Path



Hamiltonian Monte Carlo Samples

Gradient Based MCMC

We could do a full lesson on these methods alone — for now will keep it brief!

(And provide more resources for those who are interested to learn more).

Gradient Based MCMC

Suppose we have a given PDF $p(x), \quad x \in \mathbb{R}^n$.

Consider an objective function: $f(x) = -\log p(x)\,.$

Gradient Based MCMC

Suppose we have a given PDF $p(x), \quad x \in \mathbb{R}^n$.

Consider an objective function: $f(x) = -\log p(x)\,.$

The gradient of this objective is:

$$\nabla_x f(x) = \left[\frac{d}{dx_1} f(x), \ldots, \frac{d}{dx_n} f(x)\right]^\top$$

Gradient Based MCMC

If we wanted to perform *optimization* (*e.g.*, to find a minimum of the objective), we could perform gradient descent:

$$x_t = x_{t-1} - \eta_t \nabla_x f(x_{t-1})$$

where η_t is the learning rate/step size at iteration t.

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Langevin Monte Carlo (LMC) is very similar!

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Langevin Monte Carlo (LMC) is very similar!

Each step in LMC can be written:

$$x_t = x_{t-1} - \frac{\epsilon_t}{2}\nabla_x f(x_{t-1}) + \sqrt{\epsilon_t}\mathcal{N}(0, I_n)$$

Gradient Based MCMC — Langevin Monte Carlo (LMC)

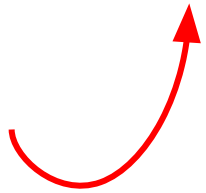
Langevin Monte Carlo (LMC) is very similar!

Each step in LMC can be written:

$$x_t = x_{t-1} - \frac{\epsilon_t}{2}\nabla_x f(x_{t-1}) + \sqrt{\epsilon_t}\mathcal{N}(0, I_n)$$

$$x_t = x_{t-1} - \eta_t \nabla_x f(x_{t-1})$$

Compare with gradient descent



Gradient Based MCMC — Langevin Monte Carlo (LMC)

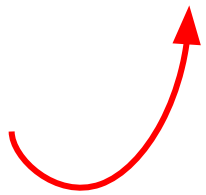
Langevin Monte Carlo (LMC) is very similar!

Each step in LMC can be written:

$$x_t = x_{t-1} - \frac{\epsilon_t}{2}\nabla_x f(x_{t-1}) + \sqrt{\epsilon_t}\mathcal{N}(0, I_n)$$

$$x_t = x_{t-1} - \eta_t \nabla_x f(x_{t-1})$$

Compare with gradient descent



Main difference is an extra "stochastic term"

Can view this as "*gradient descent plus noise*"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Langevin Monte Carlo (LMC) is very similar!

Each step in LMC can be written:

$$x_t = x_{t-1} - \frac{\epsilon_t}{2}\nabla_x f(x_{t-1}) + \sqrt{\epsilon_t}\mathcal{N}(0, I_n)$$

Can also, equivalently, write this LMC update as a sample (from a Markov chain):

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Langevin Monte Carlo (LMC) is very similar!

Each step in LMC can be written:

$$x_t = x_{t-1} - \frac{\epsilon_t}{2}\nabla_x f(x_{t-1}) + \sqrt{\epsilon_t}\mathcal{N}(0, I_n)$$

Can also, equivalently, write this LMC update as a sample (from a Markov chain):

$$x_t \sim p(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t \mid \underbrace{x_{t-1} - \frac{\epsilon_t}{2}\nabla_x f(x_{t-1})}_{\text{Mean}}, \underbrace{\epsilon_t I_n}_{\text{Variance}}\right)$$

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Intuitively:

You are moving toward higher-mass regions of the PDF (in expectation).

Source: Roy Friedman, “A Simplified Overview of Langevin Dynamics”

Source: Roy Friedman, “A Simplified Overview of Langevin Dynamics”

Gradient Based MCMC — Langevin Monte Carlo (LMC)

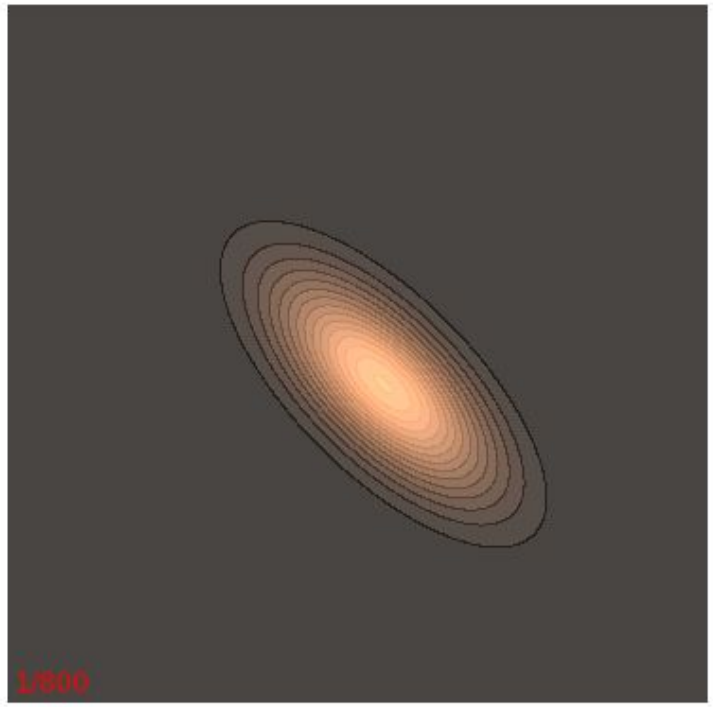
Intuitively:

You are moving toward higher-mass regions of the PDF (in expectation).

But also: adding noise such you have some probability of visiting lower-mass regions.

⇒ balanced so that the probability of visiting any region is proportional to the density in that region.

(Unlike gradient descent, where you only move towards higher-mass regions.)



Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

So does it work?

Source: Roy Friedman, “A Simplified Overview of Langevin Dynamics”

Gradient Based MCMC — Langevin Monte Carlo (LMC)

So does it work?

You can prove that the stationary distribution of this Markov chain is equal to $p(x)$, in the limit, as $T \to \infty$ and $\epsilon_t \to 0$.

Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

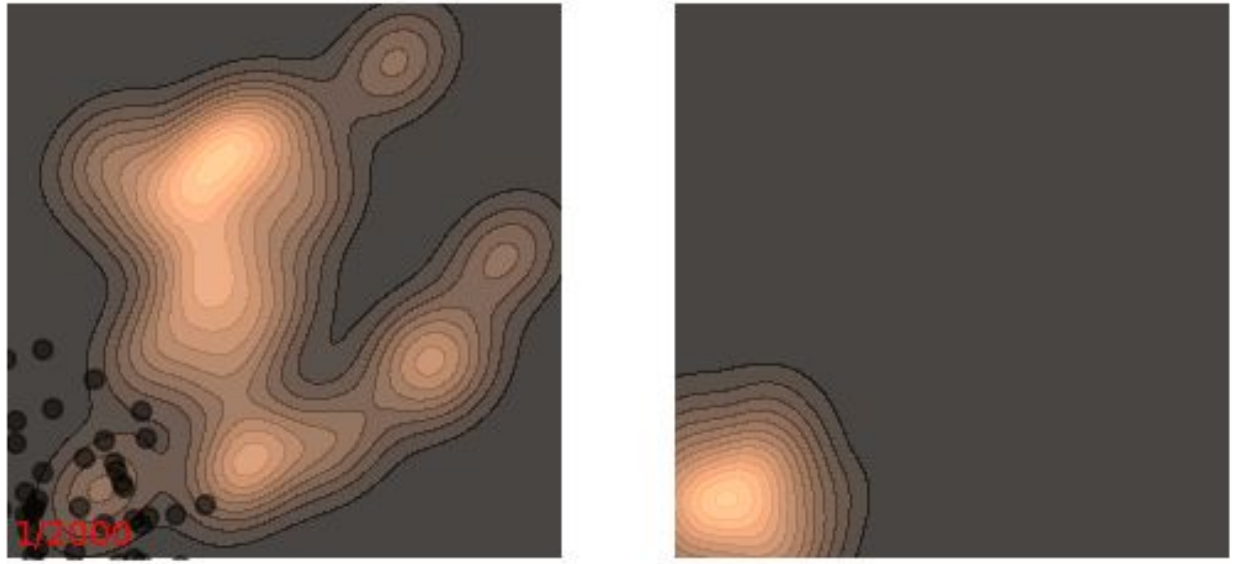
Gradient Based MCMC — Langevin Monte Carlo (LMC)

So does it work?

You can prove that the stationary distribution of this Markov chain is equal to $p(x)$, in the limit, as $T \to \infty$ and $\epsilon_t \to 0$.

For an empirical illustration:

(using a small ϵ_t).



Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

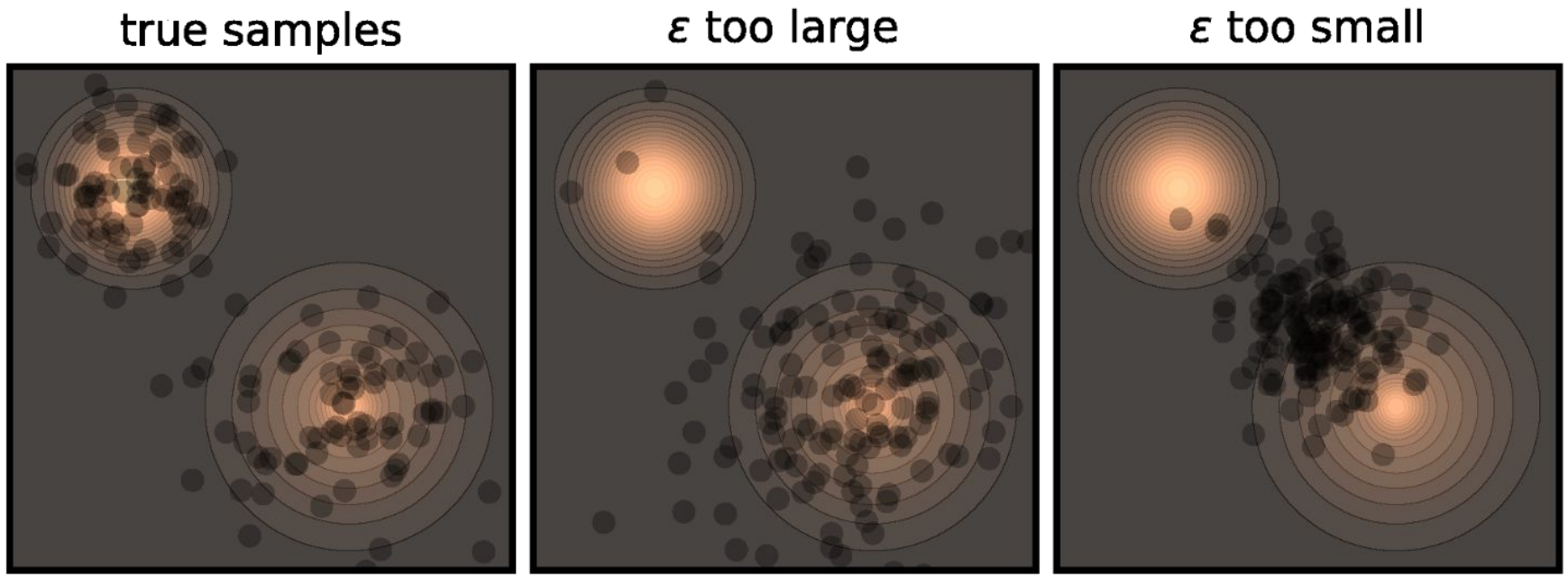
Issues with this:

Source: Roy Friedman, “A Simplified Overview of Langevin Dynamics”

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Issues with this:

- If step size is too large ⇒ samples will be biased (incorrect stationary distribution).
- If step size is too small ⇒ takes much longer (bad *mixing* properties).

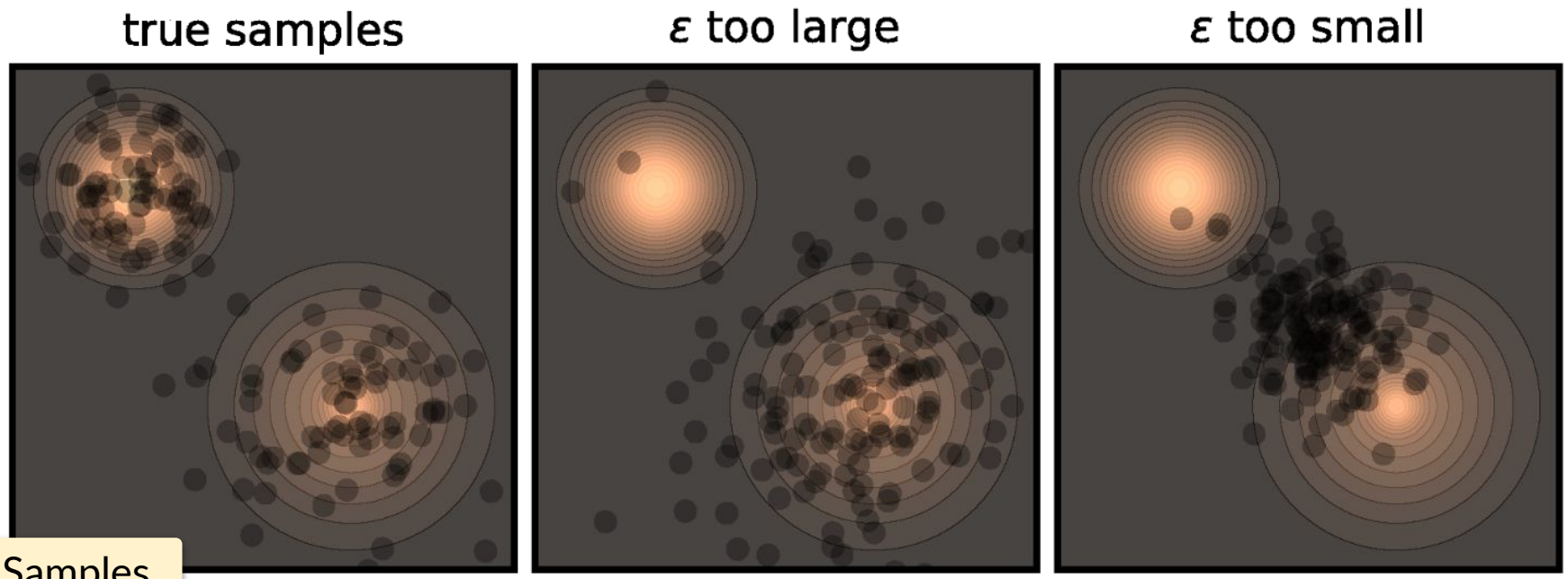


Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Issues with this:

- If step size is too large ⇒ samples will be biased (incorrect stationary distribution).
- If step size is too small ⇒ takes much longer (bad *mixing* properties).



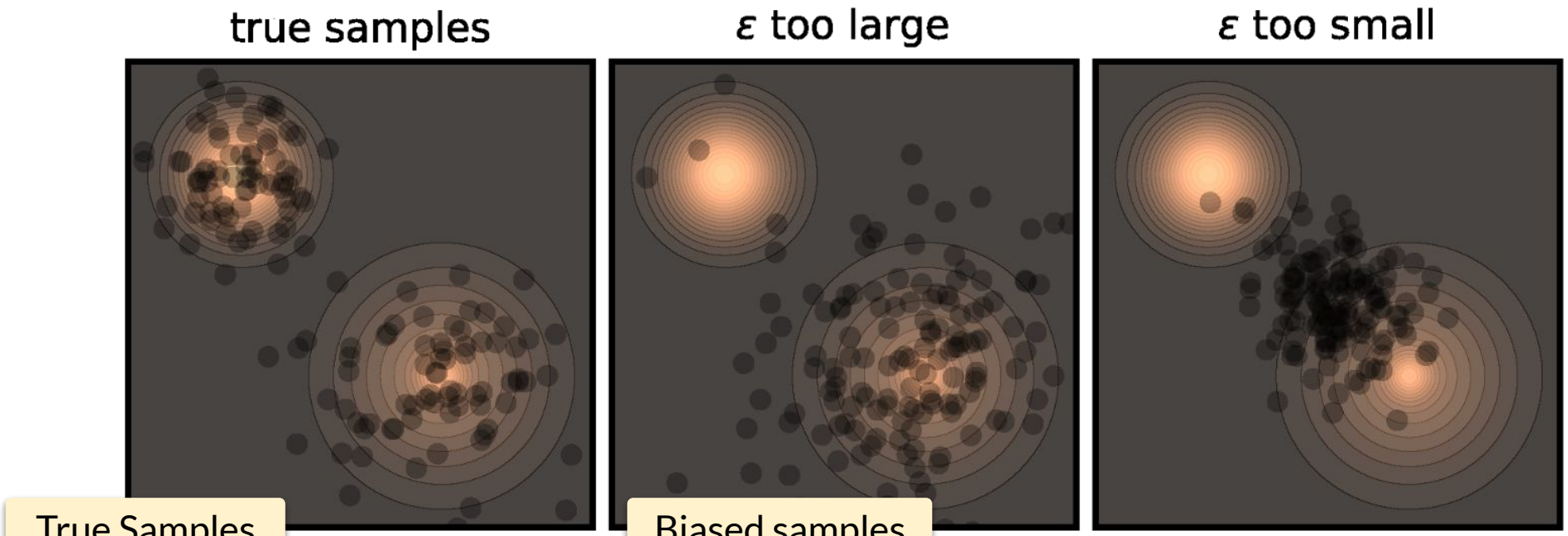
True Samples

Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Issues with this:

- If step size is too large ⇒ samples will be biased (incorrect stationary distribution).
- If step size is too small ⇒ takes much longer (bad *mixing* properties).

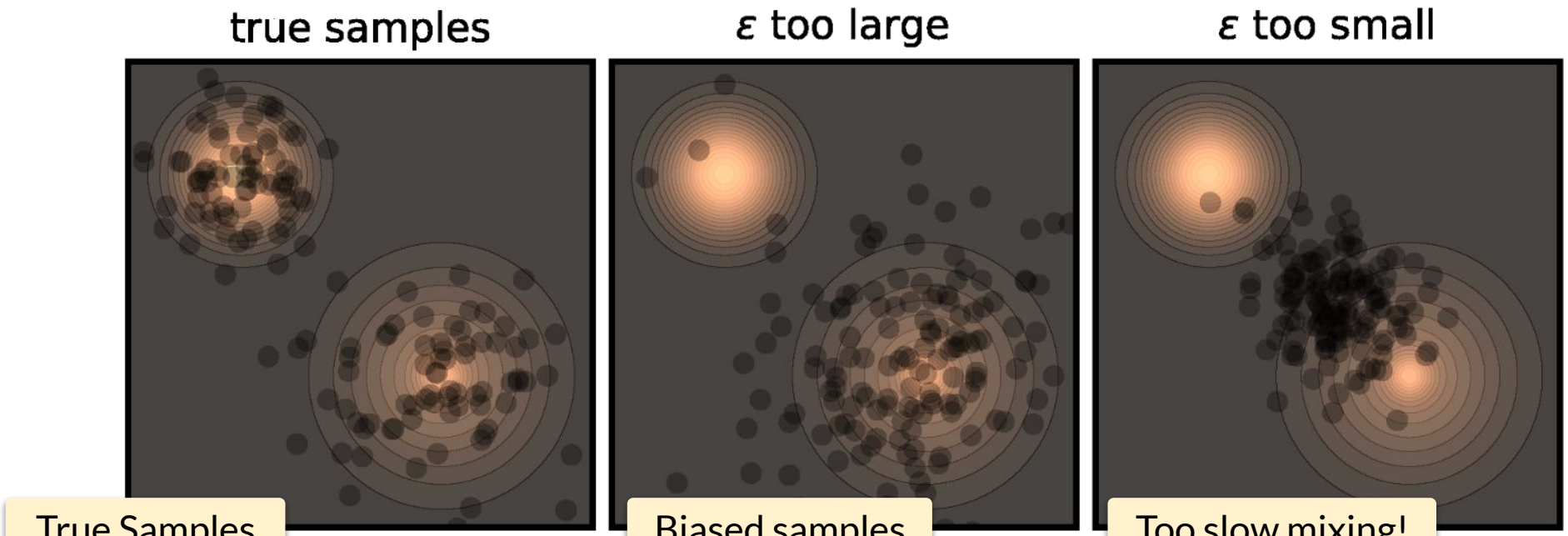


Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

Issues with this:

- If step size is too large ⇒ samples will be biased (incorrect stationary distribution).
- If step size is too small ⇒ takes much longer (bad *mixing* properties).



Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

A few solutions in practice:

Gradient Based MCMC — Langevin Monte Carlo (LMC)

A few solutions in practice:

1. You can combine LMC with Metropolis-Hastings.

Gradient Based MCMC — Langevin Monte Carlo (LMC)

A few solutions in practice:

1. You can combine LMC with Metropolis-Hastings.

In particular:

- Run LMC with a large step size (biased samples).
- Then accept/reject as you would typically do in Metropolis-Hasting.
- ⇒ Good mixing, and reasonable acceptance rate, even in high dimensions.

Gradient Based MCMC — Langevin Monte Carlo (LMC)

A few solutions in practice:

1. You can combine LMC with Metropolis-Hastings.

In particular:

- Run LMC with a large step size (biased samples).
- Then accept/reject as you would typically do in Metropolis-Hasting.
- ⇒ Good mixing, and reasonable acceptance rate, even in high dimensions.

Note: this is often called MALA - Metropolis adjusted Langevin algorithm.

Gradient Based MCMC — Langevin Monte Carlo (LMC)

A few solutions in practice:

2. You can anneal the step-size.

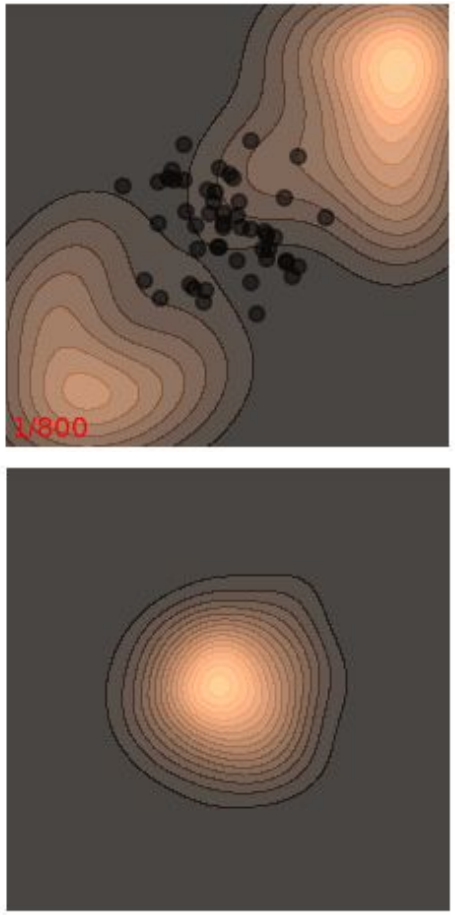
Gradient Based MCMC — Langevin Monte Carlo (LMC)

A few solutions in practice:

2. You can anneal the step-size.

In particular:

- Start LMC with a large step size (biased samples).
- Then anneal the step size to a small value as you proceed with sampling.
- ⇒ You mix well at the start, and capture details of high-mass regions at the end.



Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Langevin Monte Carlo (LMC)

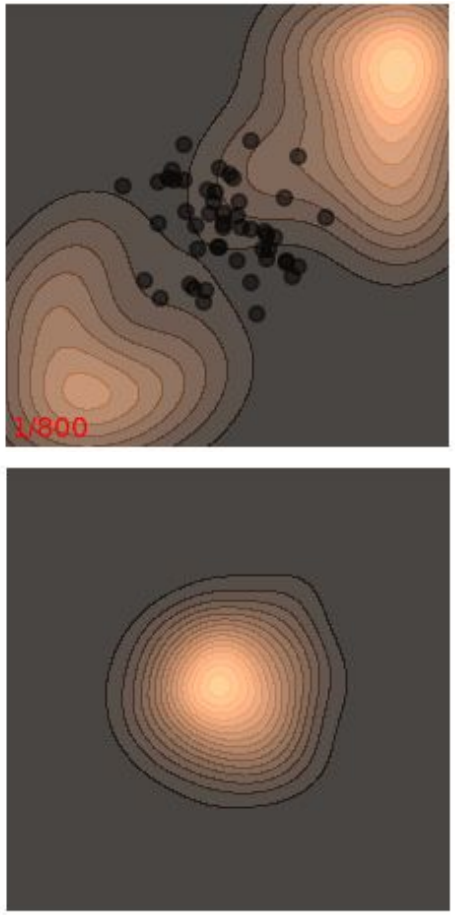
A few solutions in practice:

2. You can anneal the step-size.

In particular:

- Start LMC with a large step size (biased samples).
- Then anneal the step size to a small value as you proceed with sampling.
- ⇒ You mix well at the start, and capture details of high-mass regions at the end.

We will see that this strategy is used in variants of diffusion models, such as score-based generative models.



Source: Roy Friedman, "A Simplified Overview of Langevin Dynamics"

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

A few solutions in practice:

3. Hamiltonian Monte Carlo (HMC)!

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

A few solutions in practice:

3. Hamiltonian Monte Carlo (HMC)!

In particular:

- Take a sequence of gradient steps (in a very specific way).
- Such that you can travel far away from your previous sample…
- … but stay at a same / similar PDF value.

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

A few solutions in practice:

3. Hamiltonian Monte Carlo (HMC)!

In particular:

- Take a sequence of gradient steps (in a very specific way).
- Such that you can travel far away from your previous sample…
- … but stay at a same / similar PDF value.
- (And then still use Metropolis-Hastings to accept/or reject final proposal).

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

A few solutions in practice:

3. Hamiltonian Monte Carlo (HMC)!

In particular:

- Take a sequence of gradient steps (in a very specific way).
- Such that you can travel far away from your previous sample…
- … but stay at a same / similar PDF value.
- (And then still use Metropolis-Hastings to accept/or reject final proposal).
- ⇒ The potential for: even better mixing, and good acceptance rate, correct samples.

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

HMC **Intuition** (high level):

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

HMC **Intuition** (high level):

The goal of MCMC is to yield samples for Monte Carlo expectations of the form:

$$\mathbb{E}_{p(x)}\left[g(x)\right] = \int p(x)g(x)dx$$

$$= \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(t)} \sim p(x)$$

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

HMC **Intuition** (high level):

The goal of MCMC is to yield samples for Monte Carlo expectations of the form:

Expected value

$$\mathbb{E}_{p(x)}\left[g(x)\right] = \int p(x)g(x)dx$$

$$= \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(t)} \sim p(x)$$

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

HMC **Intuition** (high level):

The goal of MCMC is to yield samples for Monte Carlo expectations of the form:

$$\mathbb{E}_{p(x)}\left[g(x)\right] = \int p(x)g(x)dx$$

Expected value

$$= \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(t)} \sim p(x)$$

Monte Carlo estimate, given samples

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

But that you could always introduce *more variables* (*i.e.*, a probability distribution over more variables whose marginal is equal to the original probability model).

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

Called: ***auxiliary variables***

But that you could always introduce *more variables* (*i.e.*, a probability distribution over more variables whose marginal is equal to the original probability model).

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

But that you could always introduce *more variables* (*i.e.*, a probability distribution over more variables whose marginal is equal to the original probability model).

This might look like:

$$\mathbb{E}_{p(x)}\left[g(x)\right] = \int p(x)g(x)dx = \int p(x,v)g(x)dx\,dv$$

$$= \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(t)}, v^{(t)} \sim p(x,v)$$

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

But that you could always introduce *more variables* (*i.e.*, a probability distribution over more variables whose marginal is equal to the original probability model).

This might look like:

Expected value

$$\mathbb{E}_{p(x)}\left[g(x)\right] = \int p(x)g(x)dx = \int p(x,v)g(x)dx\,dv$$

$$= \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(t)}, v^{(t)} \sim p(x,v)$$

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

But that you could always introduce *more variables* (*i.e.*, a probability distribution over more variables whose marginal is equal to the original probability model).

This might look like:

Expected value

Expected value given *auxiliary variables*

$$\mathbb{E}_{p(x)}\left[g(x)\right] = \int p(x)g(x)dx = \int p(x,v)g(x)dx\,dv$$

$$= \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(t)}, v^{(t)} \sim p(x,v)$$

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

But that you could always introduce *more variables* (*i.e.*, a probability distribution over more variables whose marginal is equal to the original probability model).

This might look like:

Expected value

Expected value given *auxiliary variables*

$$\mathbb{E}_{p(x)}\left[g(x)\right] = \int p(x)g(x)dx = \int p(x,v)g(x)dx\,dv$$

$$= \frac{1}{T}\sum_{t=1}^{T} g(x^{(t)}), \quad x^{(t)}, v^{(t)} \sim p(x,v)$$

Monte Carlo estimate (only uses x samples!)

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

In HMC, we define auxiliary variables such that (roughly speaking)...

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

In HMC, we define auxiliary variables such that (roughly speaking)...

You can write out the joint PDF in terms of a *Hamiltonian* (from physics/mechanics):

$$
\begin{aligned}
p(x, v) \propto p(x)p(v) &= e^{\log p(x)} e^{-\frac{1}{2}v^\top v} \\
&= e^{-E(x)} e^{-K(v)} \\
&= e^{-E(x)-K(v)} \\
&= e^{-H(x,v)}
\end{aligned}
$$

Auxiliary variable v is typically Gaussian, is independent from x, and is viewed as *momentum*.

We call H the *Hamiltonian*

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

In HMC, we define auxiliary variables such that (roughly speaking)...

⇒ Then you can simulate *Hamiltonian Dynamics*.

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

In HMC, we define auxiliary variables such that (roughly speaking)...

⇒ Then you can simulate *Hamiltonian Dynamics*.

⇒ (*i.e.*, solve a differential equation to preserve the Hamiltonian energy…equivalent to keeping the same PDF value under the joint model $p(x, v)$.)

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

In HMC, we define auxiliary variables such that (roughly speaking)...

⇒ Then you can simulate *Hamiltonian Dynamics*.

⇒ (*i.e.*, solve a differential equation to preserve the Hamiltonian energy…equivalent to keeping the same PDF value under the joint model $p(x, v)$.)

⇒ In practice: iteratively update x and v according to some update rules… *e.g.*:

Called the *leapfrog method*.

$$v \leftarrow v + \frac{\epsilon}{2}\nabla_x \log p(x)$$
$$x \leftarrow x + \epsilon v$$
$$v \leftarrow v + \frac{\epsilon}{2}\nabla_x \log p(x)$$

Not a perfect solution to Hamiltonian dynamics (a numerical approximation).

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

So to take a step (generate next sample in MCMC):

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

So to take a step (generate next sample in MCMC):

- First you perturb the auxiliary variable v .
  - Sample from a Gaussian distribution centered on current v.

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

So to take a step (generate next sample in MCMC):

- First you perturb the auxiliary variable v .
  - Sample from a Gaussian distribution centered on current v.
- Then you simulate Hamiltonian dynamics from this new point.
  - Following updates on previous slide → yields an updated (x, v) with similar PDF value.

Gradient Based MCMC — Hamiltonian Monte Carlo (HMC)

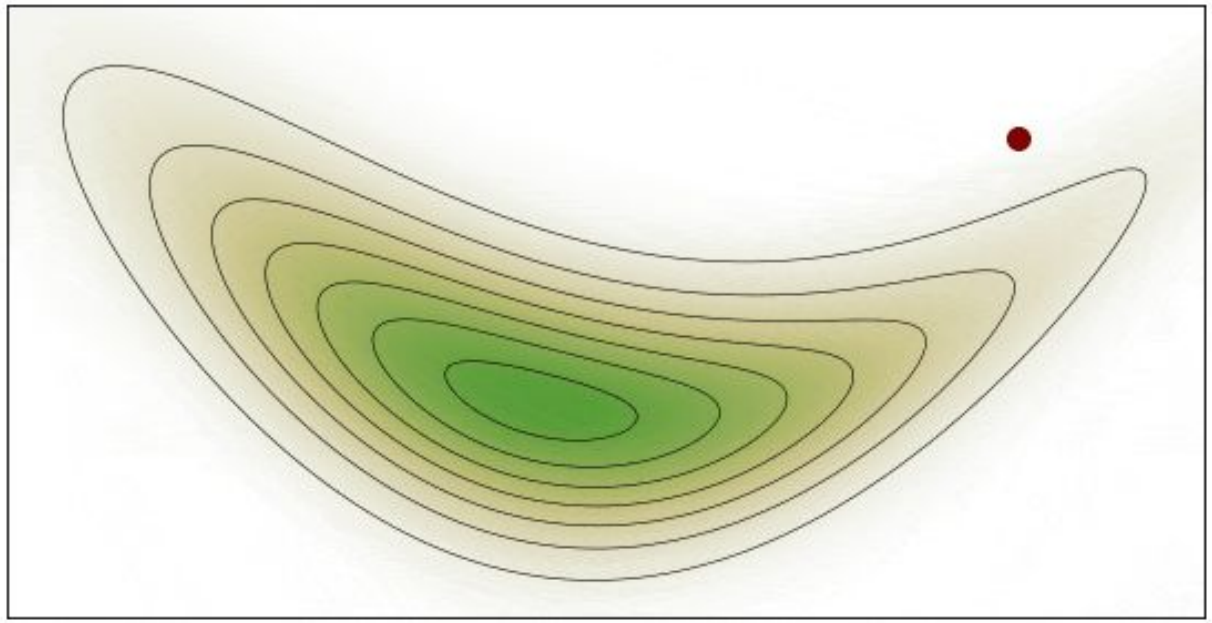
So to take a step (generate next sample in MCMC):

- First you perturb the auxiliary variable v .
  - Sample from a Gaussian distribution centered on current v.
- Then you simulate Hamiltonian dynamics from this new point.
  - Following updates on previous slide → yields an updated (x, v) with similar PDF value.
- Then you do an accept/reject step (Metropolis-Hastings) to convert this into a valid MCMC algorithm (asymptotically exact samples).

Gradient Based MCMC

Animation:

HMC in practice!



MCMC Demo Website

Check out this MCMC Demo by Chi Feng

- https://chi-feng.github.io/mcmc-demo/app.html
- https://github.com/chi-feng/mcmc-demo

Variational Inference

Variational Inference — Overview

To begin, a few potential shortcomings of sampling-based approximate inference:

1. Although they are guaranteed to perform *correct inference* given enough time, it is difficult to tell the *approximation quality* after running for a finite amount of time.

2. MCMC methods may require choosing an appropriate proposal distribution (or various hyperparameters). This choice can be difficult.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Variational Inference — Overview

Instead, we could take the strategy/philosophy of *inference as optimization*.

This method of approximate inference is known as **variational inference (VI)**.

The main idea:

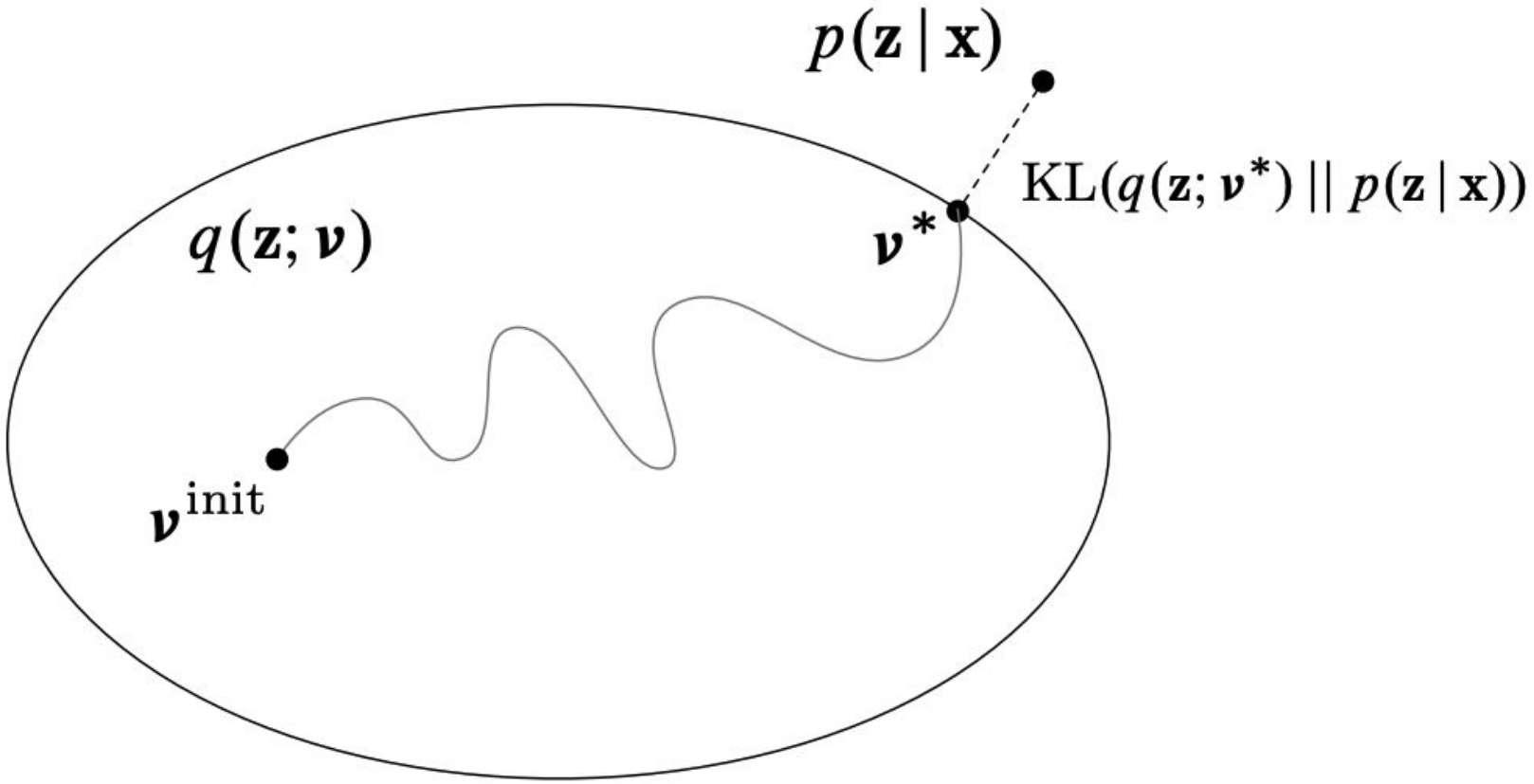
- Suppose we are given an intractable (*complex*) probability distribution $p(x)$.
- VI aims to solve an optimization problem over a set of tractable distributions Q.
- And, after optimization, return a $q(x) \in Q$ that is *most similar* to $p(x)$.
- We can then make an inference query on $q(x)$ instead of $p(x)$.c

Typically: in terms of KL divergence

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Variational Inference — Overview

From Dave Blei's Variational Inference Tutorial:

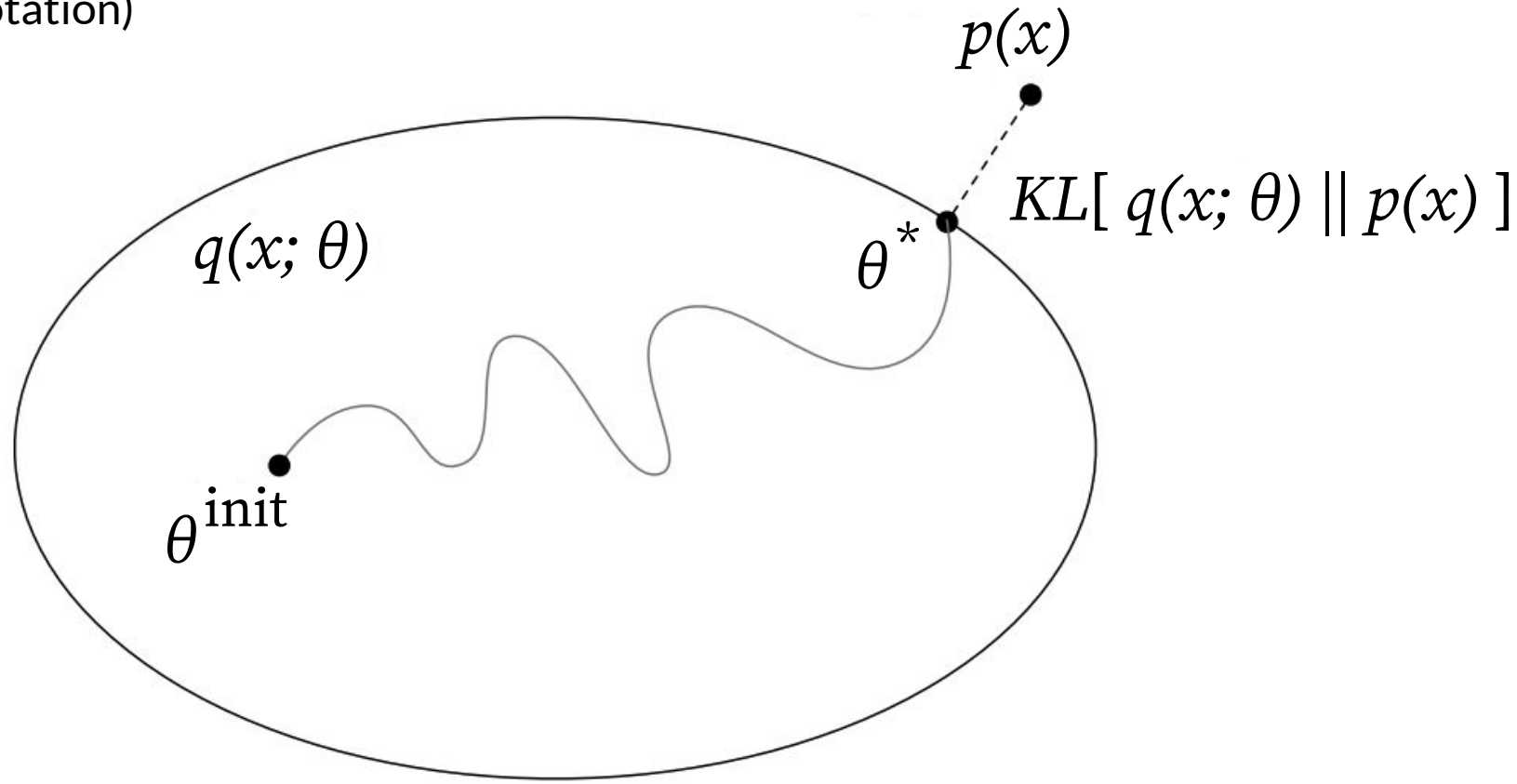


Source: Dave Blei, "Variational Inference"

Variational Inference — Overview

From Dave Blei's Variational Inference Tutorial:

(replaced with our notation)



Source: Dave Blei, "Variational Inference"

Variational Inference — Overview

Main differences with sampling/MCMC methods:

- Unlike sampling, variational methods will almost never find the exact/globally optimal solution (even in the limit as $T \to \infty$).
- However, it is easy to determine if they have converged (at least to a local optima).
- In practice, variational inference methods often scale better.
- ⇒ Amenable to techniques like stochastic gradient optimization, parallelization over multiple processors, and acceleration using GPUs.
    - (While it's easier for VI, this is also a popular field of study for MCMC).

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Kullback-Leibler Divergence

Quick background on Kullback-Leibler (KL) Divergence.

To optimize for an optimal approximation $q(x)$, we must choose an approximating family Q, and an objective $J(q)$.

The objective should capture the similarity between $q(x)$ and $p(x)$.

The field of information theory provides a tool: **Kullback-Leibler (KL) divergence**.

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Kullback-Leibler Divergence

More formally:

The KL divergence between distributions q and p with *discrete support* is:

$$\mathrm{KL}\left[q \parallel p\right] = \sum_{x} q(x) \log\left(\frac{q(x)}{p(x)}\right)$$

And the KL divergence between distributions q and p with *continuous support* is:

$$\mathrm{KL}\left[q \parallel p\right] = \int_{x \in \mathcal{X}} q(x) \log\left(\frac{q(x)}{p(x)}\right) dx$$

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

Kullback-Leibler Divergence

The KL divergence has a couple of properties that make it useful in this setting:

- $\mathrm{KL}\left[q \parallel p\right] \geq 0$ for all q, p.
- $\mathrm{KL}\left[q \parallel p\right] = 0$ if and only if $q = p$.

Importantly, note that $\mathrm{KL}\left[q \parallel p\right] \neq \mathrm{KL}\left[p \parallel q\right]$.

⇒ *i.e.*, KL divergence is **not symmetric**.

⇒ (this is why we say that it is a “divergence” and not a “distance”).

Source: Stefano Ermon, Probabilistic Graphical Models (CS228) Class

To be continued…

In the next lecture!

Paper Presentations

Paper Presentations — Goal

- During the semester, each student gives **one 20-minute presentation** on a paper relevant to probabilistic and generative models.
- Ideally from a modern machine learning conference (e.g., NeurIPS, ICML, ICLR, AAAI, etc)
  - Could be a “classic” (older) paper relevant to modern models, as well.
- This will accomplish a few things:
  - Gives the class a chance to see a broad set of interesting probabilistic/generative modeling papers.
  - Gives each student (more) experience with distilling key content from a paper & presenting it.

Paper Presentations — Schedule

Starting at the **end of February** we will go through a few (~4) presentations per class (2nd half of class).

Paper Presentations — Schedule

Starting at the **end of February** we will go through a few (~4) presentations per class (2nd half of class).

In the spreadsheet, which I will share, students will sign up for a time slot during the semester.

To make it fair, students who sign up for presentations on the first two dates will get slightly more-lenient grading (a point of extra credit on this assignment).

A	B	C	D	E	F	G	H
Date	Presentation ID	Presenter Name **(sign up!)**	Paper Title	Link/url to paper	Discussion Lead 1 **(sign up!)**	Discussion Lead 2 **(sign up!)**	Scribe **(sign up!)**
February 28							
	1						
	2						
	3						
	4						
March 7							
	5						

In-Class Participation and Discussion

There is a grade for in-class participation/discussion (during student presentations).
Students will sign up for one of three roles (again in the spreadsheet).
Each student has to do each role once during semester.

In-Class Participation and Discussion

There is a grade for in-class participation/discussion (during student presentations).

Students will sign up for one of three roles (again in the spreadsheet).

Each student has to do each role once during semester.

Roles 1 and 2 — Discussion Leads

- Read the paper before the class; Responsible for formulating a short list (~5 questions) for discussion; bring up a couple of these during class; submit all after class.

Role 3 — Scribe

- Read the paper before the class; take notes on paper during presentation; write up a ~1 page summary of the presentation.