Probabilistic & Unsupervised Learning

Sampling Methods

Maneesh Sahani
maneesh@gatsby.ucl.ac.uk

Gatsby Computational Neuroscience Unit, and
MSc ML/CSML, Dept Computer Science
University College London

Term 1, Autumn 2016

Sampling

For inference and learning we need to compute both:

- Posterior distributions (on latents and/or parameters) or predictive distributions.
- Expectations with respect to these distributions.

Sampling

For inference and learning we need to compute both:

- Posterior distributions (on latents and/or parameters) or predictive distributions.
- Expectations with respect to these distributions.

Both are often intractable.

Sampling

For inference and learning we need to compute both:

- Posterior distributions (on latents and/or parameters) or predictive distributions.
- Expectations with respect to these distributions.

Both are often intractable.

Deterministic approximations on distributions (factored variational / mean-field; BP; EP) or expectations (Bethe / Kikuchi methods) provide tractability, at the expense of a fixed approximation penalty.

Sampling

For inference and learning we need to compute both:

- Posterior distributions (on latents and/or parameters) or predictive distributions.
- Expectations with respect to these distributions.

Both are often intractable.

Deterministic approximations on distributions (factored variational / mean-field; BP; EP) or expectations (Bethe / Kikuchi methods) provide tractability, at the expense of a fixed approximation penalty.

An alternative is to represent distributions and compute expectations using randomly generated samples.

Sampling

For inference and learning we need to compute both:

- Posterior distributions (on latents and/or parameters) or predictive distributions.
- Expectations with respect to these distributions.

Both are often intractable.

Deterministic approximations on distributions (factored variational / mean-field; BP; EP) or expectations (Bethe / Kikuchi methods) provide tractability, at the expense of a fixed approximation penalty.

An alternative is to represent distributions and compute expectations using randomly generated samples.

Results are consistent, often unbiased, and precision can generally be improved to an arbitrary degree by increasing the number of samples.

Intractabilities and approximations

- Inference – computational intractability
  - Factored variational approx
  - Loopy BP/EP/Power EP
  - LP relaxations/ convexified BP
  - Gibbs sampling, other MCMC
- Inference – analytic intractability
  - Laplace approximation (global)
  - Parametric variational approx (for special cases).
  - Message approximations (linearised, sigma-point, Laplace)
  - Assumed-density methods and Expectation-Propagation
  - (Sequential) Monte-Carlo methods
- Learning – intractable partition function
  - Sampling parameters
  - Constrastive divergence
  - Score-matching
- Model selection
  - Laplace approximation / BIC
  - Variational Bayes
  - (Annealed) importance sampling
  - Reversible jump MCMC

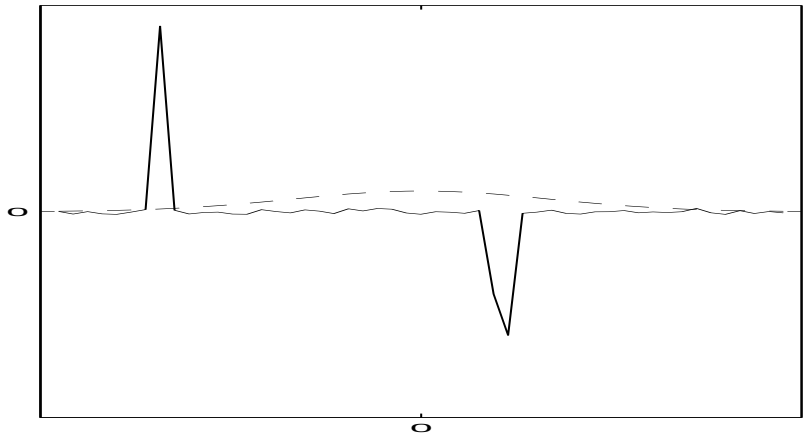
Not a complete list!

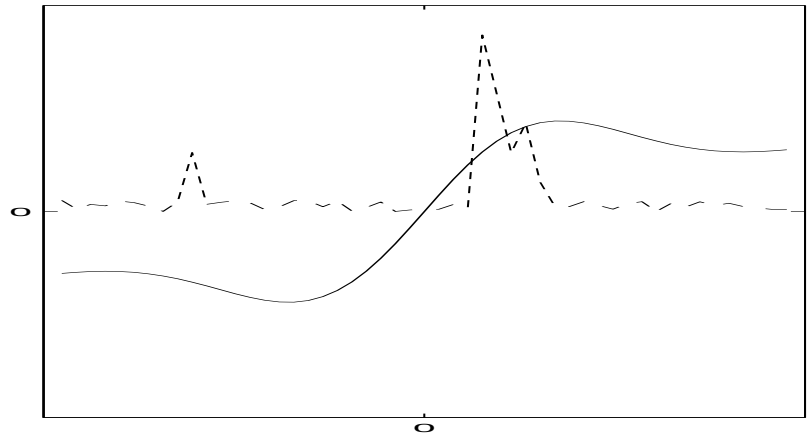
The integration problem

We commonly need to compute expected value integrals of the form:

$$\int F(x)\, p(x) dx,$$

where $F(x)$ is some function of a random variable X which has probability density $p(x)$.





Three typical difficulties:
left panel: full line is some complicated function, dashed is density;
right panel: full line is some function and dashed is complicated density;
not shown: non-analytic integral (or sum) in very many dimensions

Simple Monte-Carlo Integration

Evaluate: $\int dx\, F(x)p(x)$

Idea: Draw samples from $p(x)$, evaluate $F(x)$, average the values.

$$\int F(x)p(x)dx \simeq \frac{1}{T}\sum_{t=1}^{T} F(x^{(t)}),$$

where $x^{(t)}$ are (independent) samples drawn from $p(x)$.

Convergence to integral follows from strong law of large numbers.

Analysis of simple Monte-Carlo

Attractions:

- unbiased:

$$\mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} F(x^{(t)})\right] = \mathbb{E}\left[F(x)\right]$$

- variance falls as $1/T$ independent of dimension:

$$\begin{aligned}\mathbb{V}\left[\frac{1}{T}\sum_{t} F(x^{(t)})\right] &= \mathbb{E}\left[\left(\frac{1}{T}\sum_{t} F(x^{(t)})\right)^2\right] - \mathbb{E}\left[F(x)\right]^2 \\ &= \frac{1}{T^2}\left(T\mathbb{E}\left[F(x)^2\right] + (T^2 - T)\mathbb{E}\left[F(x)\right]^2\right) - \mathbb{E}\left[F(x)\right]^2 \\ &= \frac{1}{T}\left(\mathbb{E}\left[F(x)^2\right] - \mathbb{E}\left[F(x)\right]^2\right)\end{aligned}$$

Problems:

- May be difficult or impossible to obtain the samples directly from $p(x)$.
- Regions of high density $p(x)$ may not correspond to regions where $F(x)$ departs most from it mean value (and thus each $F(x)$ evaluation might have very high variance).

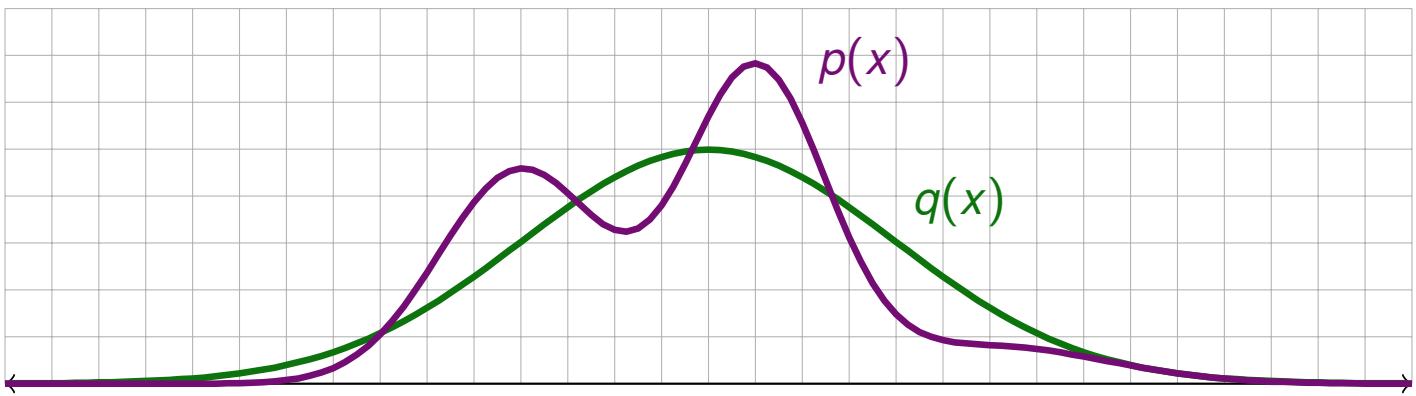
Importance sampling

Idea: Sample from a proposal distribution $q(x)$ and weight those samples by $p(x)/q(x)$.

Samples $x^{(t)} \sim q(x)$:

$$\int F(x)p(x)dx = \int F(x)\frac{p(x)}{q(x)}q(x)dx \simeq \frac{1}{T}\sum_{t=1}^{T} F(x^{(t)})\frac{p(x^{(t)})}{q(x^{(t)})},$$

provided $q(x)$ is non-zero wherever $p(x)$ is; weights $w(x^{(t)}) \equiv p(x^{(t)})/q(x^{(t)})$



- handles cases where $p(x)$ is difficult to sample.
- can direct samples towards high values of integrand $F(x)p(x)$, rather than just high $p(x)$ alone (*e.g.* p prior and F likelihood).

Analysis of importance sampling

Attractions:

- Unbiased: $\mathbb{E}_q\left[F(x)w(x)\right] = \int F(x)\frac{p(x)}{q(x)}q(x)dx = \mathbb{E}_p\left[F(x)\right]$.
- Variance could be smaller than simple Monte Carlo if

$$\mathbb{E}_q\left[\left(F(x)w(x)\right)^2\right] - \mathbb{E}_q\left[F(x)w(x)\right]^2 < \mathbb{E}_p\left[F(x)^2\right] - \mathbb{E}_p\left[F(x)\right]^2$$

  "Optimal" proposal is $q(x) = p(x)F(x)/Z_q$: every sample yields same estimate $F(x)w(x) = F(x)\frac{p(x)}{p(x)F(x)/Z_q} = Z_q$; but normalising requires solving the original problem!

Problems:

- May be hard to construct or sample $q(x)$ to give small variance.
- Variance of weights could be unbounded: $\mathbb{V}\left[w(x)\right] = \mathbb{E}_q\left[w(x)^2\right] - \mathbb{E}_q\left[w(x)\right]^2$

$$\mathbb{E}_q\left[w(x)\right] = \int q(x)w(x)dx = 1$$

$$\mathbb{E}_q\left[w(x)^2\right] = \int \frac{p(x)^2}{q(x)^2}q(x)dx = \int \frac{p(x)^2}{q(x)}dx$$

  e.g. $p(x) = \mathcal{N}(0,1), q(x) = \mathcal{N}(1,.1) \Rightarrow \mathbb{V}\left[w\right] = \int e^{49x^2}$; Monte Carlo average may be dominated by few samples, not even necessarily in region of large integrand.

Importance sampling — unnormalised distributions

Suppose that we only know $p(x)$ and/or $q(x)$ up to constants,

$$p(x) = \tilde{p}(x)/Z_p \qquad\qquad q(x) = \tilde{q}(x)/Z_q$$

where Z_p, Z_q are unknown/too expensive to compute, but that we can nevertheless draw samples from $q(x)$.

- We can still apply importance sampling by estimating the normaliser:

$$\int F(x)p(x)dx \approx \frac{\sum_t F(x^{(t)})w(x^{(t)})}{\sum_t w(x^{(t)})} \qquad\qquad w(x) = \frac{\tilde{p}(x)}{\tilde{q}(x)}$$

- This estimate is only consistent (biased for finite T, converges to true value as $T \to \infty$).
- In particular, we have

$$\frac{1}{T}\sum_t w(x^{(t)}) \to \left\langle \frac{\tilde{p}(x)}{\tilde{q}(x)} \right\rangle_q = \int dx \, \frac{Z_p p(x)}{Z_q q(x)} q(x) = \frac{Z_p}{Z_q}$$

  so with known Z_q we can estimate the partition function of p.
- (Importance sampled integral with $F(x) = 1$.)

Importance sampling — effective sample size

Variance of weights is critical to variance of estimate:

$$\mathbb{V}\left[w(x)\right] = \mathbb{E}_q\left[w(x)^2\right] - \mathbb{E}_q\left[w(x)\right]^2$$

$$\mathbb{E}_q\left[w(x)\right] = \int q(x)w(x)dx = 1$$

$$\mathbb{E}_q\left[w(x)^2\right] = \int \frac{p(x)^2}{q(x)^2}q(x)dx = \int \frac{p(x)^2}{q(x)}dx$$

A small effective sample size may diagnose ineffectiveness of importance sampling. Popular estimate:

$$\left(1 + \mathbb{V}_{\text{sample}}\left[\frac{w(x)}{\mathbb{E}_{\text{sample}}\left[w(x)\right]}\right]\right)^{-1} = \frac{\left(\sum_t w(x^{(t)})\right)^2}{\sum_t w(x^{(t)})^2}$$

However large effective sample size does not prove effectiveness (if no high weight samples found, or if q places little mass where $F(x)$ is large).

Drawing samples

Now, consider the problem of generating samples from an arbitrary distribution $p(x)$.

Standard samplers are available for Uniform$[0, 1]$ and $\mathcal{N}(0, 1)$.

- Other univariate distributions:

  $u \sim \text{Uniform}[0, 1]$

  $x = G^{-1}(u)$ with $G(x) = \int_{-\infty}^{x} p(x')dx'$ the target CDF

- Multivariate normal with covariance C:

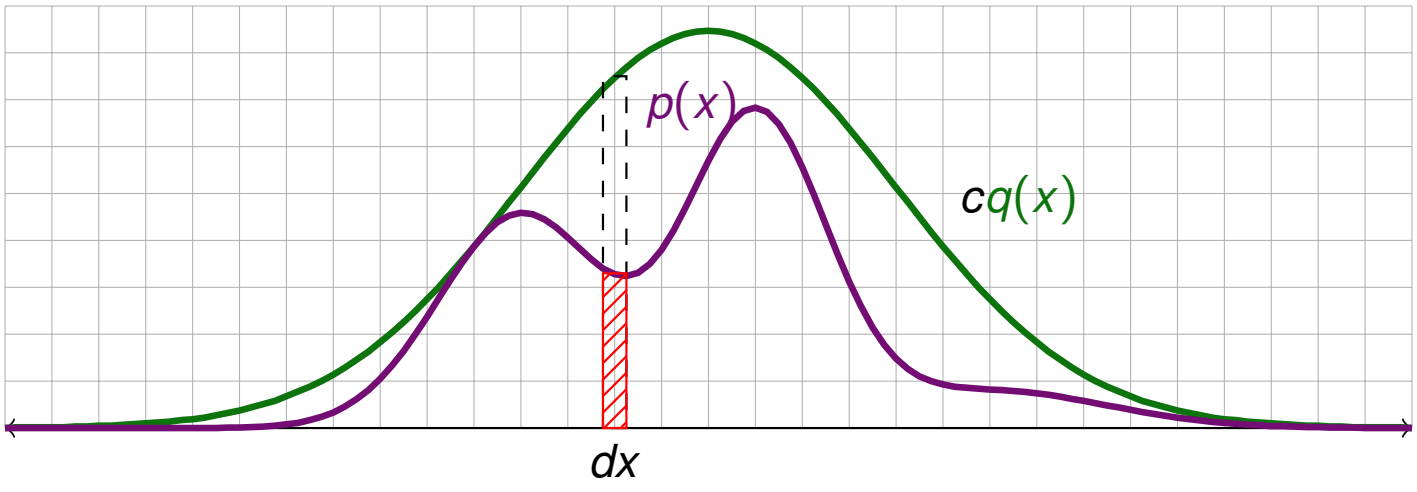
  $r_i \sim \mathcal{N}(0, 1)$

  $\mathbf{x} = C^{\frac{1}{2}}\mathbf{r}$ $[\Rightarrow \langle \mathbf{x}\mathbf{x}^{\mathsf{T}} \rangle = C^{\frac{1}{2}} \langle \mathbf{r}\mathbf{r}^{\mathsf{T}} \rangle C^{\frac{1}{2}} = C]$

Rejection Sampling

Idea: sample from an upper bound on $p(x)$, rejecting some samples.

- Find a distribution $q(x)$ and a constant c such that $\forall x,\ p(x) \leq cq(x)$
- Sample x^* from $q(x)$ and accept x^* with probability $p(x^*)/(cq(x^*))$.
- Reject the rest.



Let $y^* \sim \text{Uniform}[0, cq(x^*)]$; then the joint proposal (x^*, y^*) is a point uniformly drawn from the area under the $cq(x)$ curve.

The proposal is accepted if $y^* \leq p(x^*)$ (i.e. proposal falls in red box). The probability of this is $= q(x)dx * p(x)/cq(x) = p(x)/c\ dx$.

Thus accepted $x^* \sim p(x)$ (with average probability of acceptance $1/c$).

Rejection Sampling

Attractions:

- Unbiased; accepted x^* is true sample from $p(x)$.
- Diagnostics easier than (say) importance sampling: number of accepted samples is true sample size.

Problem:

- It may be difficult to find a $q(x)$ with a small $c \Rightarrow$ lots of wasted area.
  Examples:
  - Compute $p(X_i = b | X_j = a)$ in a directed graphical model: sample from $P(X)$, reject if $X_j \neq a$, averaging the indicator function $I(X_i = b)$
  - Compute $\mathbb{E}\left[x^2 | x > 4\right]$ for $x \sim \mathcal{N}(0, 1)$

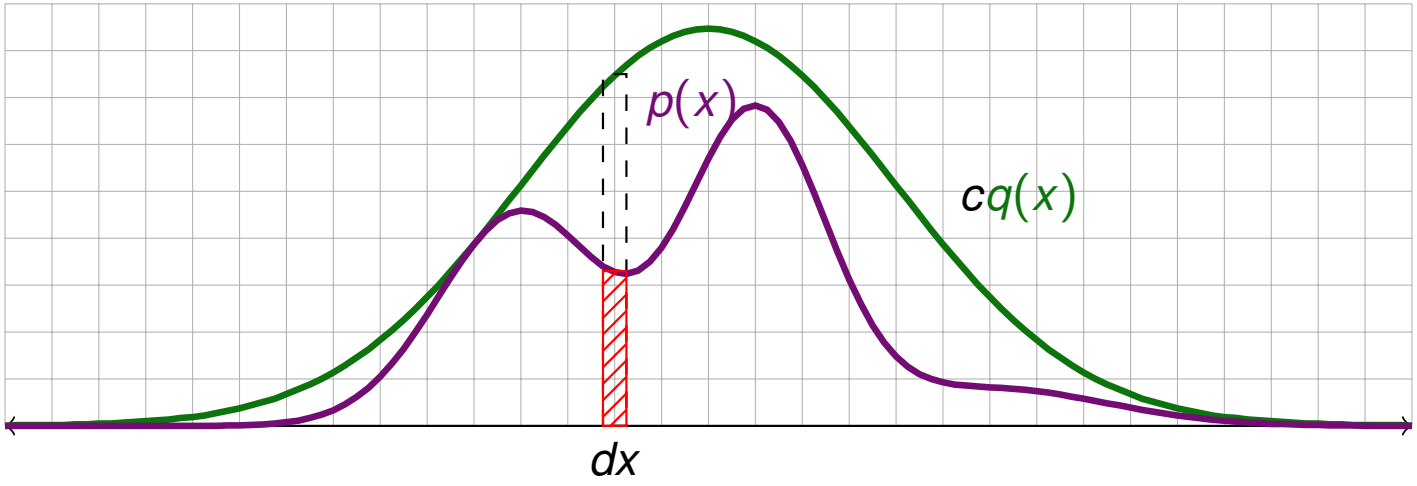
Unnormalized Distributions: say we only know $p(x), q(x)$ up to a constant,

$$p(x) = \tilde{p}(x)/Z_p \qquad\qquad q(x) = \tilde{q}(x)/Z_q$$

where Z_p, Z_q are unknown/too expensive to compute, but we can still sample from $q(x)$.

We can still apply rejection sampling if using c with $\tilde{p}(x) \leq c\tilde{q}(x)$. Still unbiased.

Relationship between importance and rejection sampling



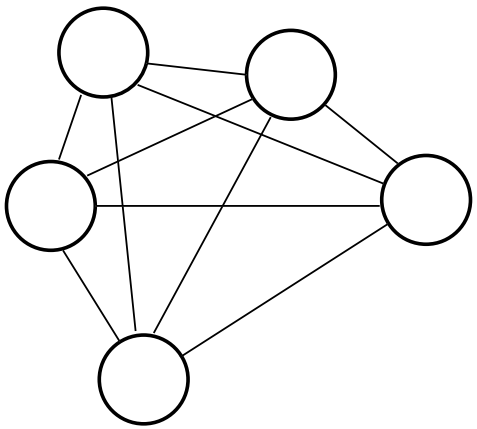
If we have c for which $q(x)$ is an upper bound on $p(x)$, then importance weights are upper bounded:

$$\frac{p(x)}{q(x)} \leq c$$

So importance weights have finite variance and importance sampling is well-behaved.

Upper bound condition makes both rejection sampling work and importance sampling well-behaved.

Learning in Boltzmann Machines



$$\log p(\mathbf{s}^V \mathbf{s}^H | W, \mathbf{b}) = \sum_{ij} W_{ij} s_i s_j - \sum_i b_i s_i - \log Z$$

with $Z = \sum_{\mathbf{s}} e^{\sum_{ij} W_{ij} s_i s_j - \sum_i b_i s_i}$.

Generalised (gradient M-step) EM requires parameter step

$$\Delta W_{ij} \propto \frac{\partial}{\partial W_{ij}} \left\langle \log p(\mathbf{s}^V \mathbf{s}^H | W, \mathbf{b}) \right\rangle_{p(\mathbf{s}^H | \mathbf{s}^V)}$$

Write $\langle \rangle_c$ (clamped) for expectations under $p(\mathbf{s}^H | \mathbf{s}^V)$. Then

$$\nabla W_{ij} = \langle s_i s_j \rangle_c - \langle s_i s_j \rangle_u$$

with $\langle \rangle_u$ (unclamped) an expectation under the current joint distribution $p(\mathbf{s}^H, \mathbf{s}^V)$.

Computing expectations

How do we find the required expectations?

- **Junction tree** is generally intractable in all but the sparsest nets (triangulation of loops makes cliques grow very large).
- **Rejection and Importance sampling** require good proposal distributions, which are difficult to find.
- **Loopy belief propagation** fails in nets with strong correlations.
- **Mean-field methods** are possible, but approximate (and pretty inaccurate).

Easy to compute conditional samples: given settings of nodes in the Markov blanket of s_i can compute and normalise the scalar $p(s_i)$ and toss a (virtual) coin.

This suggests an iterative approach:

- Choose variable settings randomly (set any clamped nodes to clamped values).
- Cycle through (unclamped) s_i, choosing $s_i \sim p(s_i|\mathbf{s}_{\setminus i})$.

After enough samples, we might expect to reach a sample from the correct distribution.

This is an example of Gibbs Sampling. Also called the heat bath or Glauber dynamics.

Markov chain Monte Carlo (MCMC) methods

Suppose we seek samples from a distribution $p^*(x)$.

Let us construct a Markov chain:

$$x_0 \to x_1 \to x_2 \to x_3 \to x_4 \to x_5 \ldots$$

where $x_0 \sim p_0(x)$ and $T(x \to x') = p(X_t = x' | X_{t-1} = x)$ is the Markov chain transition probability from x to x', and we can easily sample from each of these.

Then the marginal distributions in the chain are $x_t \sim p_t(x)$ with the property that:

$$p_t(x') = \sum_x p_{t-1}(x) T(x \to x')$$

Under some conditions, these marginals converge to an invariant/stationary/equilibrium distribution characterised by T with:

$$p_\infty(x') = \sum_x p_\infty(x) T(x \to x') \qquad \forall x'$$

If we can choose a $T(x \to x')$ so as to ensure $p_\infty = p^*$ and sample from the Markov chain for long enough, we can obtain samples from distributions arbitrarily close to p^*.

Constructing a Markov chain for MCMC

When does the Markov chain $x_0 \to x_1 \to x_2 \to x_3 \to \ldots$ with marginals $x_0 \sim p_0(x)$ and

$$p_t(x') = \sum_x p_{t-1}(x) T(x \to x')$$

according to transition probability $T(x \to x')$ have the right invariant distribution?

- First we need convergence to a unique stationary distribution regardless of initial state $x_0 \sim p_0(x)$: this is a form of ergodicity.

  $$\lim_{t \to \infty} p_t(x) = p_\infty(x)$$

  A sufficient condition for the Markov chain to be ergodic is that

  $$T^k(x \to x') > 0 \text{ for all } x, x' \in \mathcal{X} \text{ and some } k. \qquad (*)$$

  That is: it is possible to reach any state from any other state in exactly k steps.

- A useful sufficient condition for $p^*(x)$ being invariant is detailed balance:

  $$p^*(x') T(x' \to x) = p^*(x) T(x \to x') \qquad (**)$$

  If T and p^* satisfy both (*) and (**) then the marginal of the chain defined by T will converge to p^*.

Gibbs Sampling

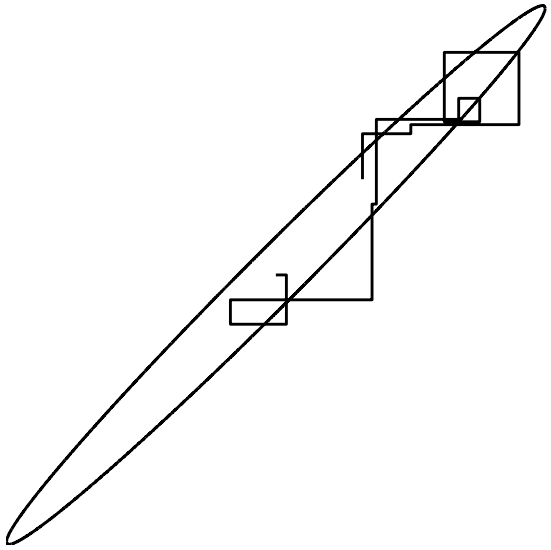
A method for sampling from a multivariate distribution, $p(\mathbf{x})$
Idea: sample from the conditional of each variable given the settings of the other variables.

Repeatedly:
1) pick i (either at random or in turn)
2) replace x_i by a sample from the conditional distribution

$$p(x_i|\mathbf{x}_{\setminus i}) = p(x_i|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots x_n)$$

Gibbs sampling is feasible if it is easy to sample from the conditional probabilities.
This creates a Markov chain

$$\mathbf{x}^{(1)} \rightarrow \mathbf{x}^{(2)} \rightarrow \mathbf{x}^{(3)} \rightarrow \ldots$$



Example: 20 (half-) iterations of Gibbs sampling on a bivariate Gaussian

Under some (mild) conditions, the **equilibium distribution**, i.e. $p(\mathbf{x}^{(\infty)})$, of this Markov chain is $p(\mathbf{x})$.

Detailed balance for Gibbs sampling

We can show that Gibbs sampling has the right stationary distribution $p(\mathbf{x})$ by showing that the detailed balance condition is met.

The transition probabilities are given by:

$$T(\mathbf{x} \to \mathbf{x}') = \pi_i p(x_i' | \mathbf{x}_{\setminus i})$$

where π_i is the probability of choosing to update the ith variable (to handle rotation updates instead of random ones, we need to consider transitions due to one full sweep).

Then we have:

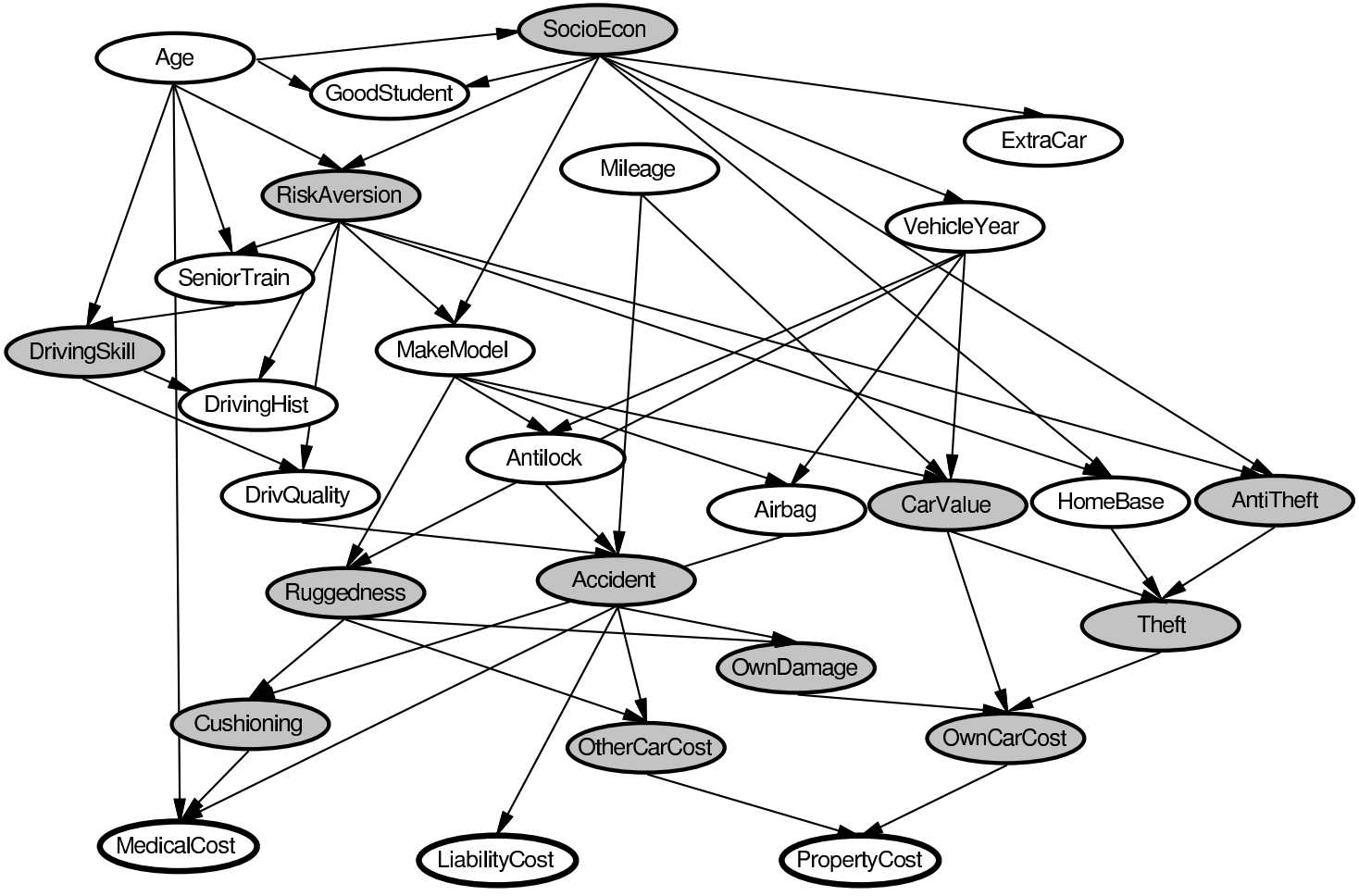
$$T(\mathbf{x} \to \mathbf{x}')p(\mathbf{x}) = \pi_i p(x_i' | \mathbf{x}_{\setminus i}) \underbrace{p(x_i | \mathbf{x}_{\setminus i}) p(\mathbf{x}_{\setminus i})}_{p(\mathbf{x})}$$

and

$$T(\mathbf{x}' \to \mathbf{x})p(\mathbf{x}') = \pi_i p(x_i | \mathbf{x}'_{\setminus i}) \underbrace{p(x_i' | \mathbf{x}'_{\setminus i}) p(\mathbf{x}'_{\setminus i})}_{p(\mathbf{x}')}$$

But $\mathbf{x}'_{\setminus i} = \mathbf{x}_{\setminus i}$ so detailed balance holds.

Gibbs Sampling in Graphical Models



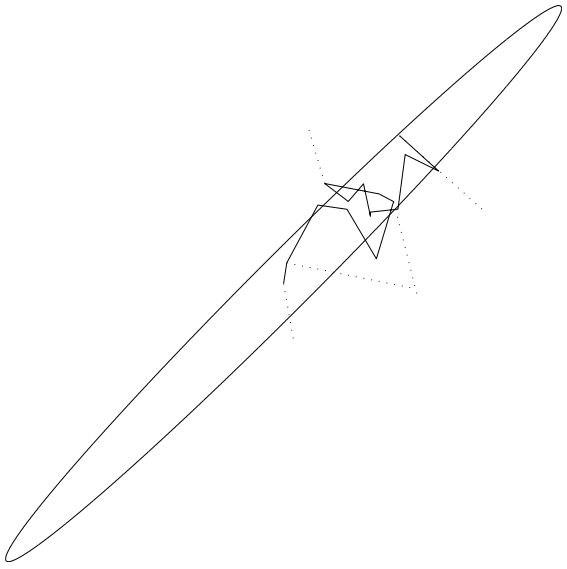
Initialize all variables to some settings. Sample each variable conditional on other variables (equivalently, conditional on its Markov blanket).

The BUGS software implements this algorithm for very general probabilistic models (but not too big ones).

The Metropolis-Hastings algorithm

Gibbs sampling can be slow ($p(x_i)$ may be well determined by $\mathbf{x}_{\setminus i}$), and conditionals may be intractable. Global transition might be better.

Idea: Propose a change to current state; accept or reject.
(A kind of rejection sampling)



Each step: Starting from the current state $\mathbf{x}$,

1. Propose a new state $\mathbf{x}'$ using a proposal distribution
   $$S(\mathbf{x}'|\mathbf{x}) = S(\mathbf{x} \to \mathbf{x}').$$
2. Accept the new state with probability:
   $$\min\left(1, p(\mathbf{x}')S(\mathbf{x}' \to \mathbf{x})/p(\mathbf{x})S(\mathbf{x} \to \mathbf{x}')\right);$$
3. Otherwise **retain the old state**.

Example: 20 iterations of global metropolis sampling from bivariate Gaussian; rejected proposals are dotted.

- Metropolis algorithm was symmetric $S(\mathbf{x}'|\mathbf{x}) = S(\mathbf{x}|\mathbf{x}')$. Hastings generalised.
- **Local** (changing one or few x_i's) vs **global** (changing all $\mathbf{x}$) proposal distributions.
- Efficiency dictated by balancing between high acceptance rate and large step size.
- May adapt $S(\mathbf{x} \to \mathbf{x}')$ to balance these, but stationarity only holds once S is fixed.
- Note, we need only to compute ratios of probabilities (no normalizing constants).

Detailed balance for Metropolis-Hastings

The transition kernel is:

$$T(\mathbf{x} \to \mathbf{x}') = S(\mathbf{x} \to \mathbf{x}') \min\left(1, \frac{p(\mathbf{x}')S(\mathbf{x}' \to \mathbf{x})}{p(\mathbf{x})S(\mathbf{x} \to \mathbf{x}')}\right)$$

with $T(\mathbf{x} \to \mathbf{x})$ the expected rejection probability.

Without loss of generality we assume $p(\mathbf{x}')S(\mathbf{x}' \to \mathbf{x}) \leq p(\mathbf{x})S(\mathbf{x} \to \mathbf{x}')$.

Then

$$\begin{aligned} p(\mathbf{x})T(\mathbf{x} \to \mathbf{x}') &= p(\mathbf{x})S(\mathbf{x} \to \mathbf{x}') \cdot \frac{p(\mathbf{x}')S(\mathbf{x}' \to \mathbf{x})}{p(\mathbf{x})S(\mathbf{x} \to \mathbf{x}')} \\ &= p(\mathbf{x}')S(\mathbf{x}' \to \mathbf{x}) \end{aligned}$$

and

$$\begin{aligned} p(\mathbf{x}')T(\mathbf{x}' \to \mathbf{x}) &= p(\mathbf{x}')S(\mathbf{x}' \to \mathbf{x}) \cdot 1 \\ &= p(\mathbf{x}')S(\mathbf{x}' \to \mathbf{x}) \end{aligned}$$

Practical MCMC

Markov chain theory guarantees that

$$\frac{1}{T}\sum_{t=1}^{T} F(x_t) \to \mathbb{E}\left[F(x)\right] \text{ as } T \to \infty.$$

But given finite computational resources we have to compromise...

- Convergence diagnosis is hard. Usually plot various useful statistics, e.g. log probability, clusters obtained, factor loadings, and eye-ball convergence.
- Control runs: initial runs of the Markov chain used to set parameters like step size etc for good convergence. These are discarded.
- Burn-in: discard first samples from Markov chain before convergence.
- Collecting samples: usually run Markov chain for a number of iterations between collected samples to reduce dependence between samples.
- Number of runs: for the same amount of computation, we can either run one long Markov chain (best chance of convergence), lots of short chains (wasted burn-ins, but chains are independent), or in between.

Practical MCMC

- Multiple transition kernels: different transition kernels have different convergence properties and it may often be a good idea to use multiple kernels.
  - Mixing MCMC transitions in a way that depends on the last sample ("adaptive transition") risks breaking detailed balance with respect to the target and creating a different invariant distribution.
  - Can sometimes be rescued by treating mixed transitions as a mixture proposal distribution and introducing Hastings accept/reject step.
- Integrated autocorrelation time: uses empirical autocorrelation to estimate the number of transitions needed for samples $F(x_t)$ to be effectively "independent" (no guarantees, probably underestimates true dependence). Assume wlog $\mathbb{E}[F(x)] = 0$.

$$\mathbb{V}\left[\frac{1}{T}\sum_{t=1}^{T} F(x_t)\right] = \mathbb{E}\left[\left(\frac{1}{T}\sum_{t=1}^{T} F(x_t)\right)^2\right] = \frac{\mathbb{V}[F(x)]}{T}\left(1 + 2\sum_{\tau=1}^{T-1}\left(1 - \frac{\tau}{T}\right)\frac{C_\tau}{C_0}\right)$$

where $C_\tau = \mathbb{E}[F(x_t)F(x_{t+\tau})]$ can be estimated empirically. The integrated autocorrelation (term in parentheses) gives the number of correlated samples we need to match one iid sample in terms of variance. As $T \to \infty$, this is:

$$1 + 2\sum_{\tau=1}^{\infty}\frac{C_\tau}{C_0}$$

Annealing

Very often, need to sample from distribution defined by unnormalised energy E:

$$p(\mathbf{x}) = \frac{1}{Z} e^{-E(x)}$$

MCMC sampling works well in this setting [MH: no change needed; Gibbs: usually possible to normalise conditional] but may mix slowly.

Often useful to introduce temperature $1/\beta$:

$$p_\beta(\mathbf{x}) = \frac{1}{Z_\beta} e^{-\beta E(x)}$$

When $\beta \to 0$ (temperature $\to \infty$) all states are equally likely: easy to sample and mix.
As $\beta \to 1$, $p_\beta \to p$.

Simulated annealing: start chain with β small and gradually increase to 1. Can be used for optimisation by taking $\beta \to \infty$ (provably finds global mode, albeit using exponentially slow annealing schedule).

Annealed importance sampling: use importance sampling to correct for fact that chain need not have converged at each β. Equivalently, use annealing to construct a good proposal for importance sampling.

Auxilliary variable methods

Many practical MCMC methods are based on the principle of auxilliary variables:

- Want to sample from $p^*(x)$, but suitable Markov chain is either difficult to design, or does not mix well.
- Introduce a new variable y, defined using conditional $p(y|x)$.
- Sample from joint $p^*(x)p(y|x)$ (often by Gibbs).
- Discard the ys. The xs are drawn from the target p^*.

Hybrid/Hamiltonian Monte Carlo: overview

- The transition processes of the Metropolis-Hastings and Gibbs algorithms are essentially shaped random walks.
- The typical distance traveled by a random walk in n steps is proportional to $\sqrt{n}$. Thus these methods explore the distribution slowly. We would like to seek regions of high probability while avoiding random walk behavior.
- Let the target distribution be $p(\mathbf{x})$, and suppose that we can compute the gradient of p with respect to $\mathbf{x}$.
- Can we use the gradient information to shape a proposal distribution without breaking detailed balance?
- Hybrid or Hamiltonian Monte Carlo: introduce a fictitious physical system describing a particle with position $\mathbf{x}$ and momentum $\mathbf{v}$ (an auxilliary variable).
- Make proposals by simulating motion in the dynamical system so that the marginal distribution over position corresponds to the target.
- MH acceptance step corrects for errors of discretised simulation.

Hamiltonian Monte Carlo: the dynamical system

In the physical system, "positions" $\mathbf{x}$ corresponding to the random variables of interest are augmented by momentum variables $\mathbf{v}$:

$$p(\mathbf{x}, \mathbf{v}) \propto \exp(-H(\mathbf{x}, \mathbf{v})) \qquad E(\mathbf{x}) = -\log p(\mathbf{x})$$

$$H(\mathbf{x}, \mathbf{v}) = E(\mathbf{x}) + K(\mathbf{v}) \qquad K(\mathbf{v}) = \frac{1}{2}\sum_i v_i^2$$

With these definitions $\int p(\mathbf{x}, \mathbf{v}) d\mathbf{v} = p(\mathbf{x})$ (the desired distribution) and $p(\mathbf{v}) = N(0, I)$.

We think of $E(\mathbf{x})$ as the potential energy of being in state $\mathbf{x}$, and $K(\mathbf{v})$ as the kinetic energy associated with momentum $\mathbf{v}$. We assume "mass" $= 1$, so momentum $=$ velocity.

The physical system evolves at constant total energy H according to Hamiltonian dynamics:

$$\frac{dx_i}{d\tau} = \frac{\partial H}{\partial v_i} = v_i \qquad \frac{dv_i}{d\tau} = -\frac{\partial H}{\partial x_i} = \frac{\partial E}{\partial x_i}.$$

The first equation says derivative of position is velocity. The second equation says that the system accelerates in the direction that decreases potential energy.

Think of a ball rolling on a frictionless hilly surface.

Hamiltonian Monte Carlo: how to simulate the dynamical system

We can simulate the above differential equations by discretising time and iterating over finite differences on a computer. This introduces small (we hope) errors. (The errors we care about are errors which change the total energy—we will correct for these by occasionally rejecting moves that change the energy.)

A good simulation scheme for HMC is given by leapfrog simulation. We take L discrete steps of size ϵ to simulate the system evolving for $L\epsilon$ time:

$$
\begin{aligned}
\hat{v}_i(\tau + \tfrac{\epsilon}{2}) &= \hat{v}_i(\tau) - \frac{\epsilon}{2} \frac{\partial E(\hat{x}(\tau))}{\partial x_i} \\
\hat{x}_i(\tau + \epsilon) &= \hat{x}_i(\tau) + \epsilon \frac{\hat{v}_i(\tau + \frac{\epsilon}{2})}{m_i} \\
\hat{v}_i(\tau + \epsilon) &= \hat{v}_i(\tau + \tfrac{\epsilon}{2}) - \frac{\epsilon}{2} \frac{\partial E(\hat{x}(\tau + \epsilon))}{\partial x_i}
\end{aligned}
$$

Hamiltonian Monte Carlo: properties of the dynamical system

Hamiltonian dynamics has the following important properties:

- preserves total energy, H,
- is reversible in time
- preserves phase space volumes (Liouville's theorem)

The leapfrog discretisation only approximately preserves the total energy H, but

- is reversible in time
- preserves phase space volume

The dynamical system is simulated using the leapfrog discretisation and the new state is used as a proposal in the Metropolis algorithm to eliminate errors introduced by the leapfrog approximation

Stochastic dynamics

Changes in the total energy, H, are introduced by interleaving the deterministic leapfrog transitions with stochastic updates of the momentum variables.

Since the distribution of the momenta are independent Gaussian, this is easily done by a trivial Gibbs sampling step (which doesn't depend on $\mathbf{x}$).

In practice, it is often useful to introduce persistence in momentum to further suppress random walks:

$$v_i = \alpha v_i + \sqrt{1 - \alpha^2}\varepsilon,$$

where $\varepsilon \sim \mathcal{N}(0, 1)$ and the persistence parameter $0 \leq \alpha < 1$.

Hamiltonian Monte Carlo Algorithm

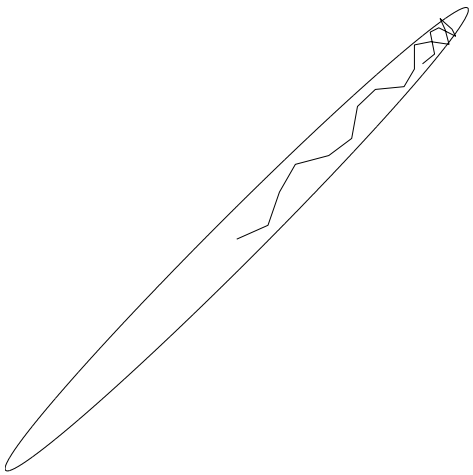
1. A new state is proposed by deterministically simulating a trajectory with L discrete steps from $(\mathbf{x}, \mathbf{v})$ to $(\mathbf{x}^*, \mathbf{v}^*)$. To compensate for errors of discretisation, the new state $(\mathbf{x}^*, \mathbf{v}^*)$ is accepted with probability:

$$\min\left(1, \frac{p(\mathbf{v}^*, \mathbf{x}^*)}{p(\mathbf{v}, \mathbf{x})}\right) = \min(1, e^{-(H(\mathbf{v}^*, \mathbf{x}^*) - H(\mathbf{v}, \mathbf{x}))})$$

   otherwise the state remains the same. [Formally, Metropolis step with proposal defined by choosing direction of momentum uniformly at random and simulating L (reversible) leapfrog steps.]

2. Resample the momentum vector (by a trivial Gibbs sampling step or with persistence)

$$\mathbf{v} \sim p(\mathbf{v}|\mathbf{x}) = p(\mathbf{v}) = \mathcal{N}(0, I)$$



Example: $L = 20$ leapfrog iterations when sampling from a bivariate Gaussian

Langevin Monte Carlo

If we take exactly one leapfrog step, then the HMC updates reduce to:

$$\mathbf{v} \sim \mathcal{N}(0, I)$$

$$x_i^* = x_i + \epsilon \hat{v}_i(\tau + \tfrac{\epsilon}{2}) \qquad = x_i - \frac{\epsilon^2}{2}\frac{\partial E(x_i)}{\partial x_i} + \epsilon v_i$$

$$v_i^* = \hat{v}_i(\tau + \tfrac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i} \qquad = v_i - \frac{\epsilon}{2}\frac{\partial E(x_i)}{\partial x_i} - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i}$$

$$p_{\text{accept}} = \min\left(1, \frac{p(\mathbf{x}^*)}{p(\mathbf{x})} e^{-\frac{1}{2}(\|\mathbf{v}^*\|^2 - \|\mathbf{v}\|^2)}\right)$$

Langevin Monte Carlo

If we take exactly one leapfrog step, then the HMC updates reduce to:

$$\mathbf{v} \sim \mathcal{N}(0, I)$$

$$x_i^* = x_i + \epsilon \hat{v}_i(\tau + \tfrac{\epsilon}{2}) \qquad = x_i - \frac{\epsilon^2}{2}\frac{\partial E(x_i)}{\partial x_i} + \epsilon v_i$$

$$v_i^* = \hat{v}_i(\tau + \tfrac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i} \qquad = v_i - \frac{\epsilon}{2}\frac{\partial E(x_i)}{\partial x_i} - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i}$$

$$p_{\text{accept}} = \min\left(1, \frac{p(\mathbf{x}^*)}{p(\mathbf{x})} e^{-\frac{1}{2}(\|\mathbf{v}^*\|^2 - \|\mathbf{v}\|^2)}\right)$$

Note that the proposal for $\mathbf{x}^*$ looks like a step up the gradient of $\log p(\mathbf{x})$ plus Gaussian noise. The relative scales of the step and noise are adjusted to keep Hamiltonian energy constant.

Langevin Monte Carlo

If we take exactly one leapfrog step, then the HMC updates reduce to:

$$\mathbf{v} \sim \mathcal{N}(0, I)$$

$$x_i^* = x_i + \epsilon \hat{v}_i(\tau + \tfrac{\epsilon}{2}) \qquad = x_i - \frac{\epsilon^2}{2}\frac{\partial E(x_i)}{\partial x_i} + \epsilon v_i$$

$$v_i^* = \hat{v}_i(\tau + \tfrac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i} \qquad = v_i - \frac{\epsilon}{2}\frac{\partial E(x_i)}{\partial x_i} - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i}$$

$$p_{\text{accept}} = \min\left(1, \frac{p(\mathbf{x}^*)}{p(\mathbf{x})} e^{-\frac{1}{2}(\|\mathbf{v}^*\|^2 - \|\mathbf{v}\|^2)}\right)$$

Note that the proposal for $\mathbf{x}^*$ looks like a step up the gradient of $\log p(\mathbf{x})$ plus Gaussian noise. The relative scales of the step and noise are adjusted to keep Hamiltonian energy constant.

- Possible to rewrite acceptance probability in terms of $(\mathbf{x}, \mathbf{x}^*)$ alone: equivalent to Hastings acceptance rule taking asymmetric proposal into account.

Langevin Monte Carlo

If we take exactly one leapfrog step, then the HMC updates reduce to:

$$\mathbf{v} \sim \mathcal{N}(0, I)$$

$$x_i^* = x_i + \epsilon \hat{v}_i(\tau + \tfrac{\epsilon}{2}) \qquad = x_i - \frac{\epsilon^2}{2}\frac{\partial E(x_i)}{\partial x_i} + \epsilon v_i$$

$$v_i^* = \hat{v}_i(\tau + \tfrac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i} \qquad = v_i - \frac{\epsilon}{2}\frac{\partial E(x_i)}{\partial x_i} - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i}$$

$$p_{\text{accept}} = \min\left(1, \frac{p(\mathbf{x}^*)}{p(\mathbf{x})} e^{-\frac{1}{2}(\|\mathbf{v}^*\|^2 - \|\mathbf{v}\|^2)}\right)$$

Note that the proposal for $\mathbf{x}^*$ looks like a step up the gradient of $\log p(\mathbf{x})$ plus Gaussian noise. The relative scales of the step and noise are adjusted to keep Hamiltonian energy constant.

- Possible to rewrite acceptance probability in terms of $(\mathbf{x}, \mathbf{x}^*)$ alone: equivalent to Hastings acceptance rule taking asymmetric proposal into account.
- In practice, with small ϵ, a single leapfrog step introduces very small discretisation errors in energy so $p_{\text{accept}} \approx 1$.

Langevin Monte Carlo

If we take exactly one leapfrog step, then the HMC updates reduce to:

$$\mathbf{v} \sim \mathcal{N}(0, I)$$

$$x_i^* = x_i + \epsilon \hat{v}_i(\tau + \tfrac{\epsilon}{2}) \qquad = x_i - \frac{\epsilon^2}{2}\frac{\partial E(x_i)}{\partial x_i} + \epsilon v_i$$

$$v_i^* = \hat{v}_i(\tau + \tfrac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i} \qquad = v_i - \frac{\epsilon}{2}\frac{\partial E(x_i)}{\partial x_i} - \frac{\epsilon}{2}\frac{\partial E(x_i^*)}{\partial x_i}$$

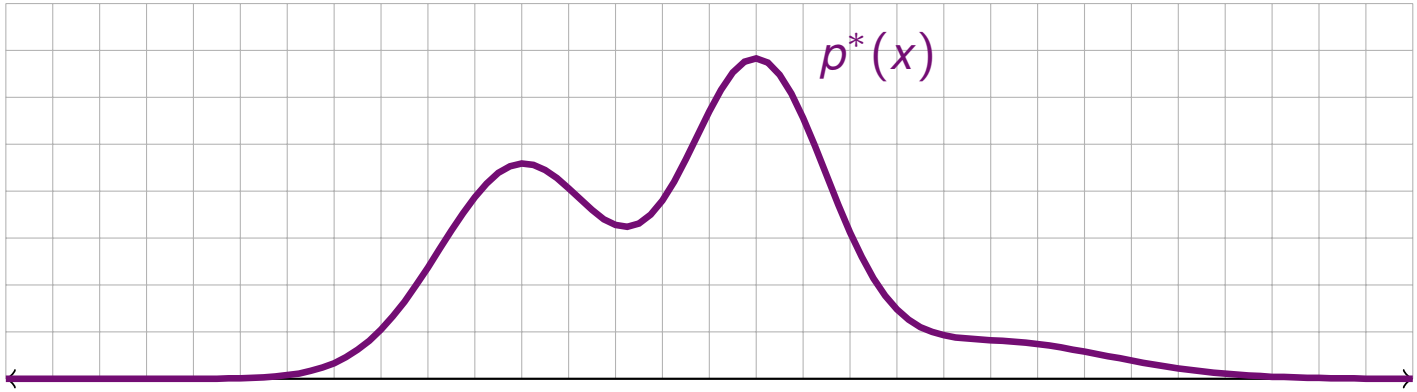
$$p_{\text{accept}} = \min\left(1, \frac{p(\mathbf{x}^*)}{p(\mathbf{x})} e^{-\frac{1}{2}(\|\mathbf{v}^*\|^2 - \|\mathbf{v}\|^2)}\right)$$

Note that the proposal for $\mathbf{x}^*$ looks like a step up the gradient of $\log p(\mathbf{x})$ plus Gaussian noise. The relative scales of the step and noise are adjusted to keep Hamiltonian energy constant.

- Possible to rewrite acceptance probability in terms of $(\mathbf{x}, \mathbf{x}^*)$ alone: equivalent to Hastings acceptance rule taking asymmetric proposal into account.
- In practice, with small ϵ, a single leapfrog step introduces very small discretisation errors in energy so $p_{\text{accept}} \approx 1$.
- Thus, acceptance step is often neglected: Langevin sampling

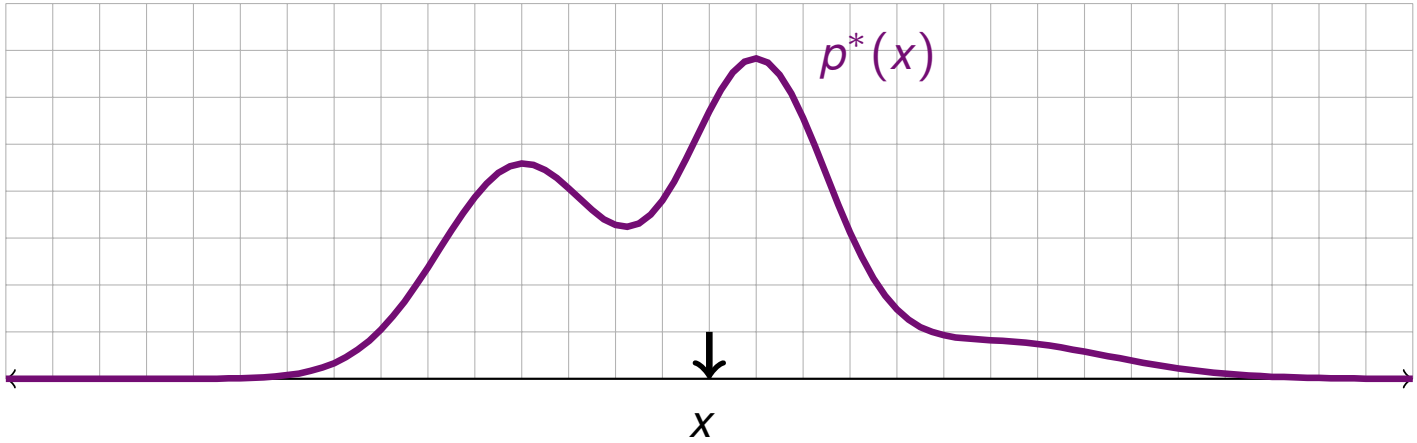
Slice sampling

- Efficient auxilliary-variable Gibbs sampler for low-dimensional distributions: can be used as an internal component for a high-D Gibbs or M-H chain.



Slice sampling

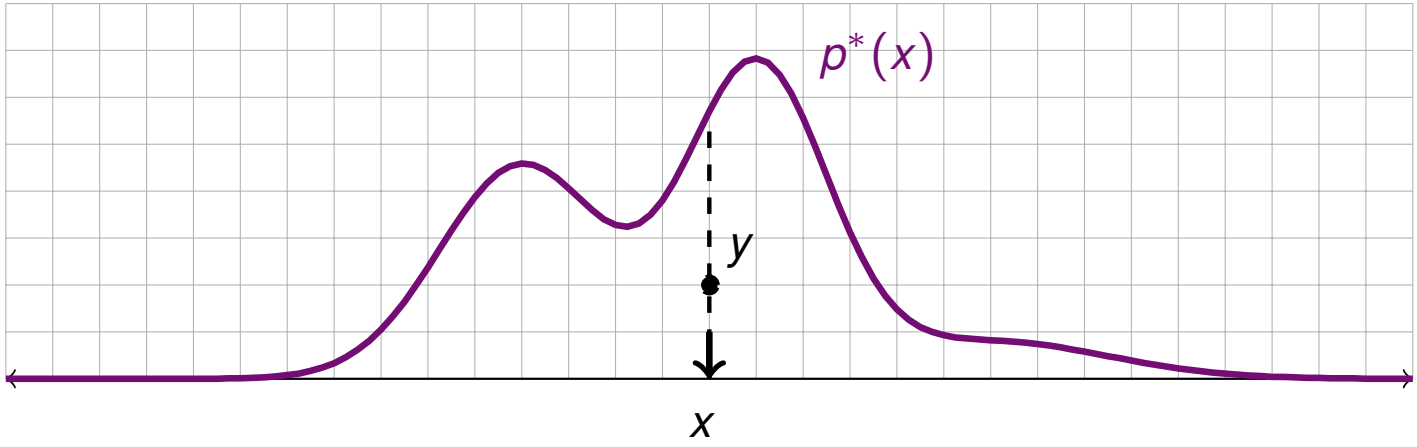
- Efficient auxilliary-variable Gibbs sampler for low-dimensional distributions: can be used as an internal component for a high-D Gibbs or M-H chain.



- Start with sample x.

Slice sampling

- Efficient auxilliary-variable Gibbs sampler for low-dimensional distributions: can be used as an internal component for a high-D Gibbs or M-H chain.

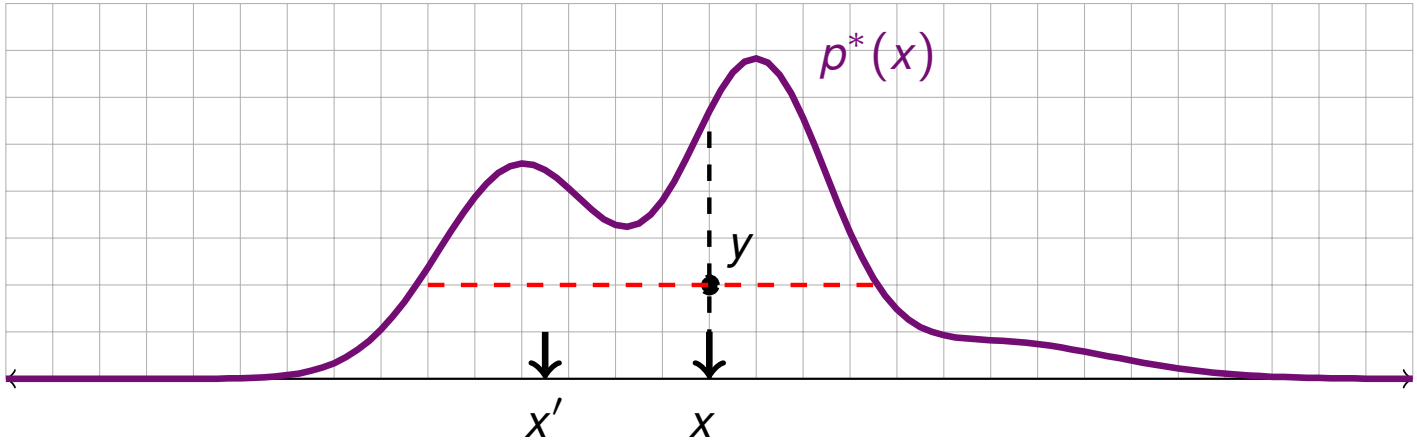


- Start with sample x.
- Sample auxilliary variable $y|x \sim \text{Uniform}[0, p^*(x)]$

$$p(x,y) = p^*(x)p(y|x) = \begin{cases} p^*(x)\frac{1}{p^*(x)} = 1 & \text{if } 0 \leq y \leq p^*(x) \\ 0 & \text{otherwise} \end{cases}$$

Slice sampling

- Efficient auxilliary-variable Gibbs sampler for low-dimensional distributions: can be used as an internal component for a high-D Gibbs or M-H chain.



- Start with sample x.
- Sample auxilliary variable $y|x \sim \text{Uniform}[0, p^*(x)]$

$$p(x,y) = p^*(x)p(y|x) = \begin{cases} p^*(x)\frac{1}{p^*(x)} = 1 & \text{if } 0 \le y \le p^*(x) \\ 0 & \text{otherwise} \end{cases}$$

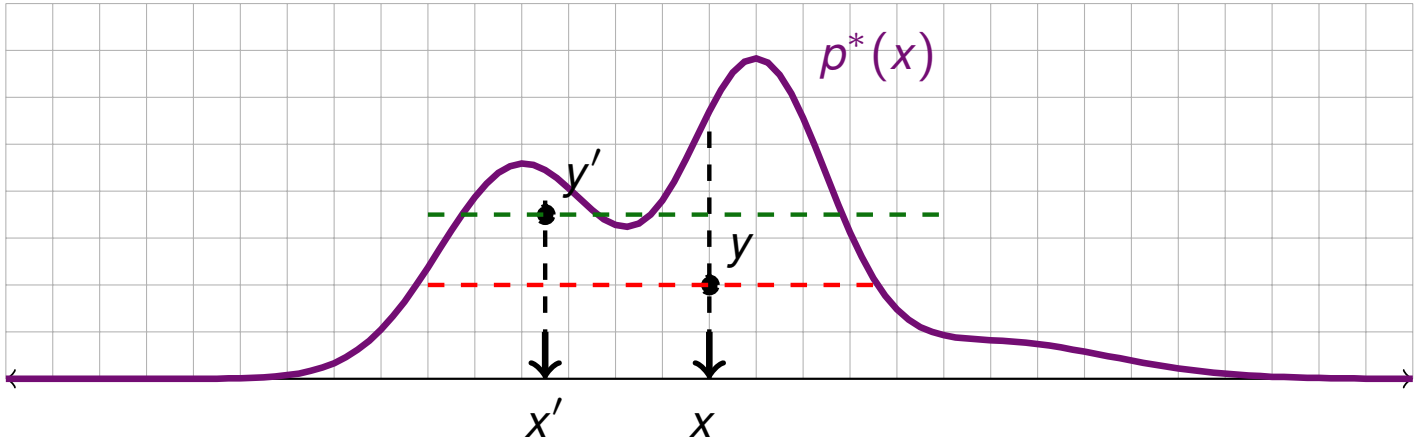
- Sample x' from $p(x'|y)$

$$p(x'|y) \propto p(x',y) = \begin{cases} 1 & \text{if } p^*(x') \ge y \\ 0 & \text{otherwise} \end{cases}$$

So $x' \sim \text{Uniform}[\{x : p^*(x) \ge y\}]$.

Slice sampling

- Efficient auxilliary-variable Gibbs sampler for low-dimensional distributions: can be used as an internal component for a high-D Gibbs or M-H chain.



- Start with sample x.
- Sample auxilliary variable $y|x \sim \text{Uniform}[0, p^*(x)]$

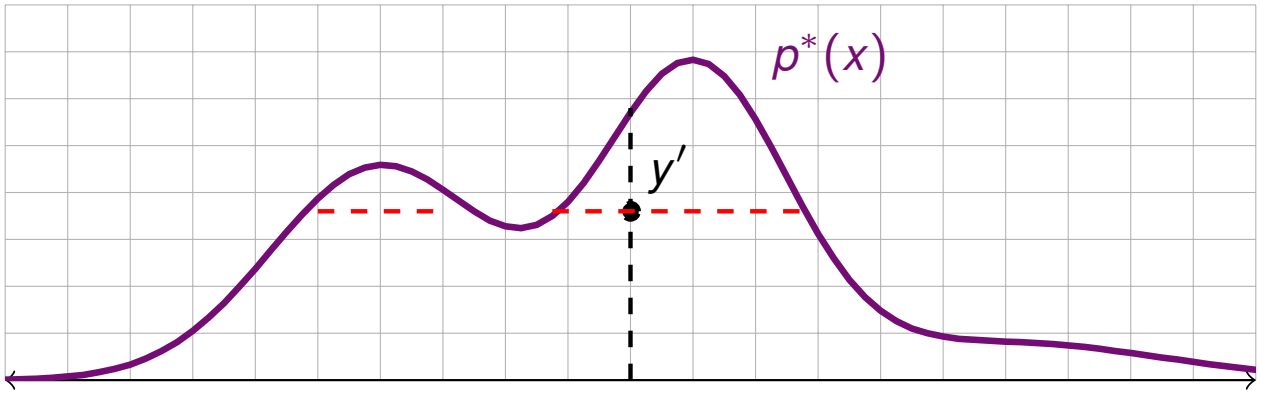
$$p(x,y) = p^*(x)p(y|x) = \begin{cases} p^*(x)\frac{1}{p^*(x)} = 1 & \text{if } 0 \le y \le p^*(x) \\ 0 & \text{otherwise} \end{cases}$$

- Sample x' from $p(x'|y)$

$$p(x'|y) \propto p(x',y) = \begin{cases} 1 & \text{if } p^*(x') \ge y \\ 0 & \text{otherwise} \end{cases}$$

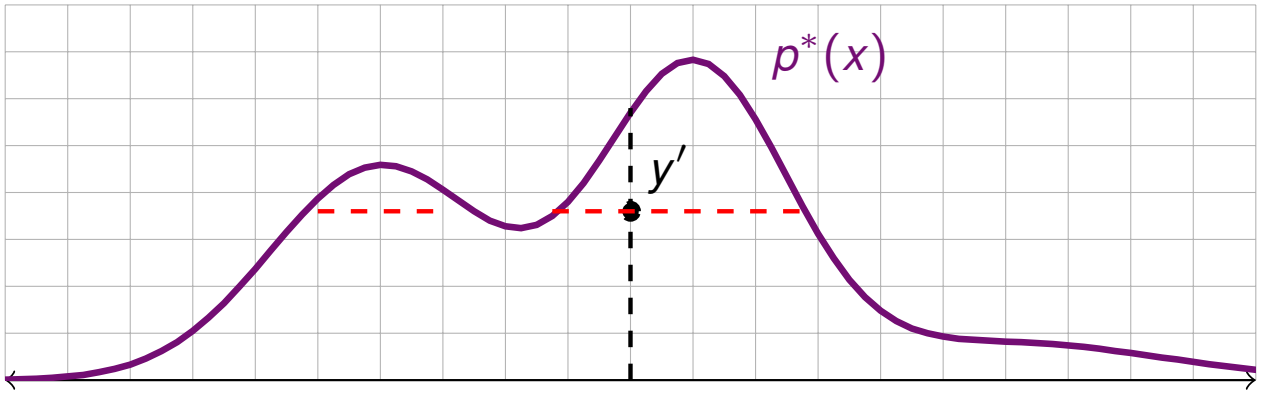
  So $x' \sim \text{Uniform}[\{x : p^*(x) \ge y\}]$.
- Defining $\{x : p^*(x) \ge y\}$ often difficult – rejection sample in (adaptive) y-level "slice" around old x extending outside at least local mode.

Slice sampling – defining the y-level slices



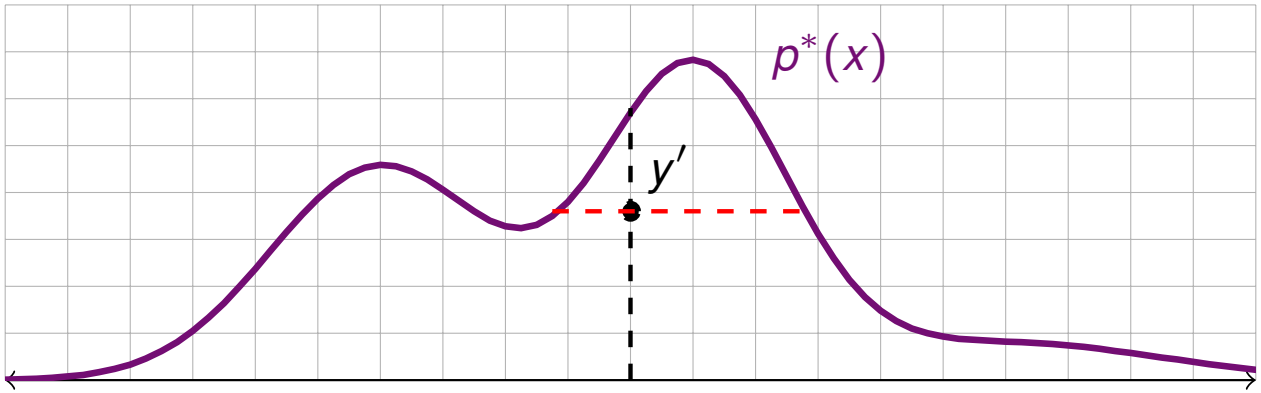
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.

Slice sampling – defining the y-level slices



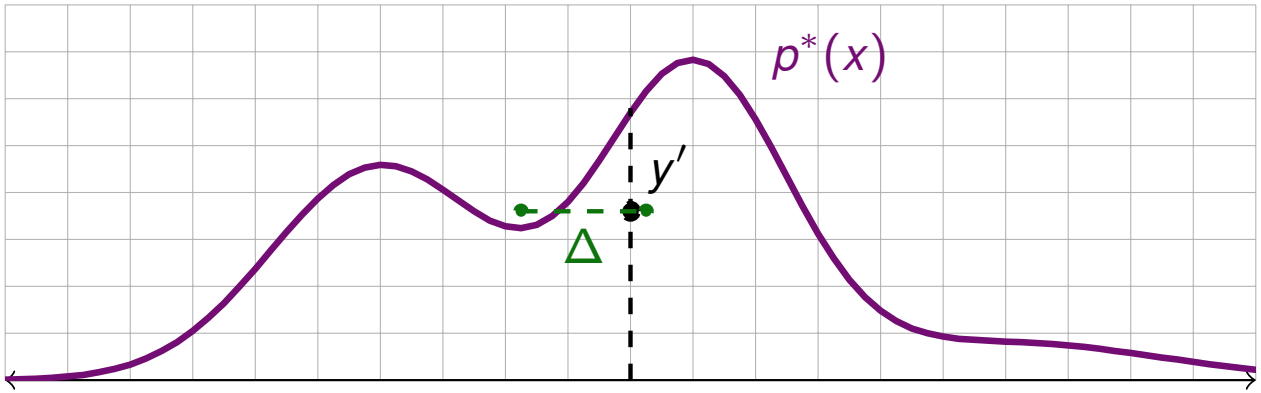
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.

Slice sampling – defining the y-level slices



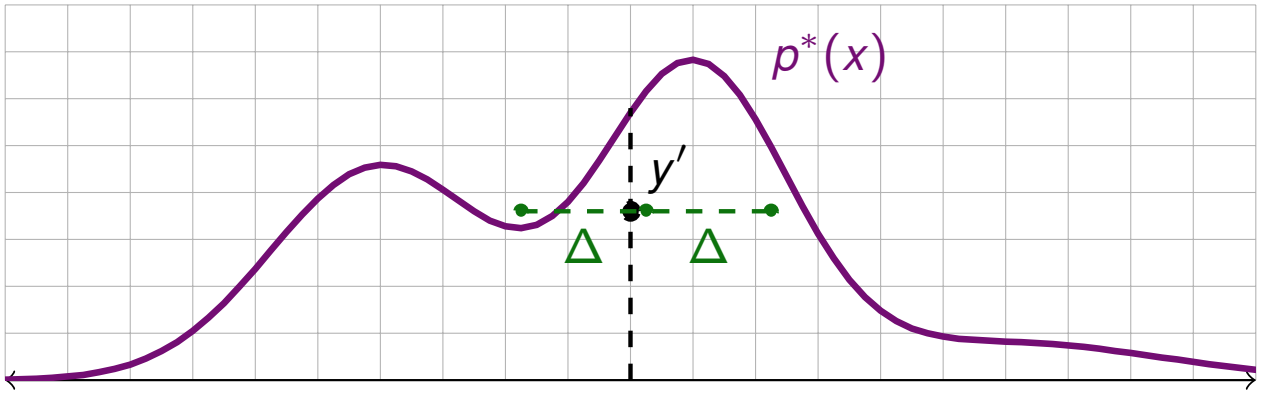
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).

Slice sampling – defining the y-level slices



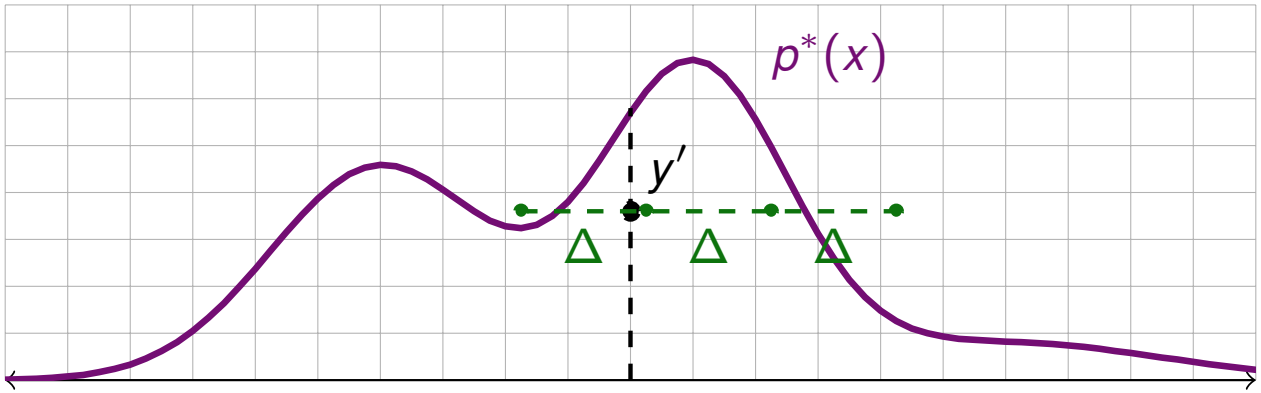
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density;

Slice sampling – defining the y-level slices



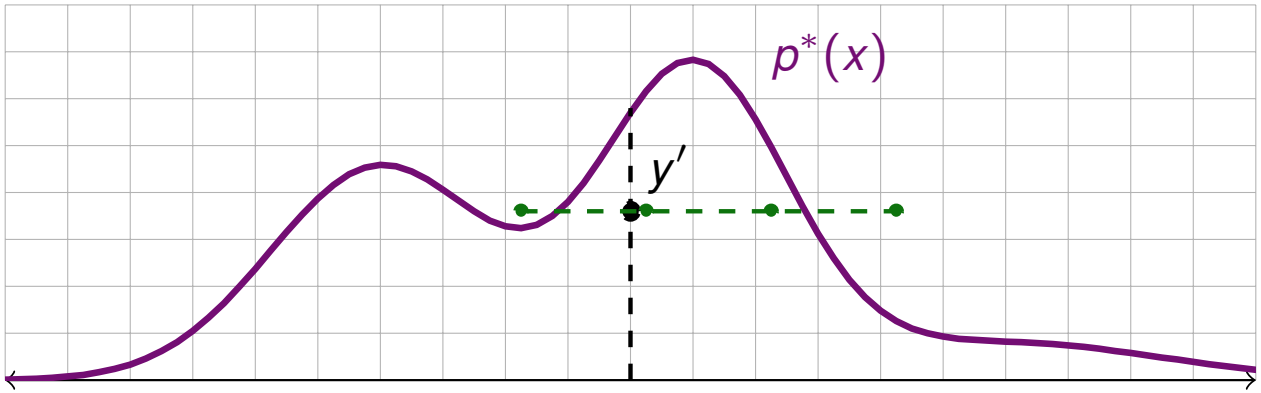
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density;

Slice sampling – defining the y-level slices



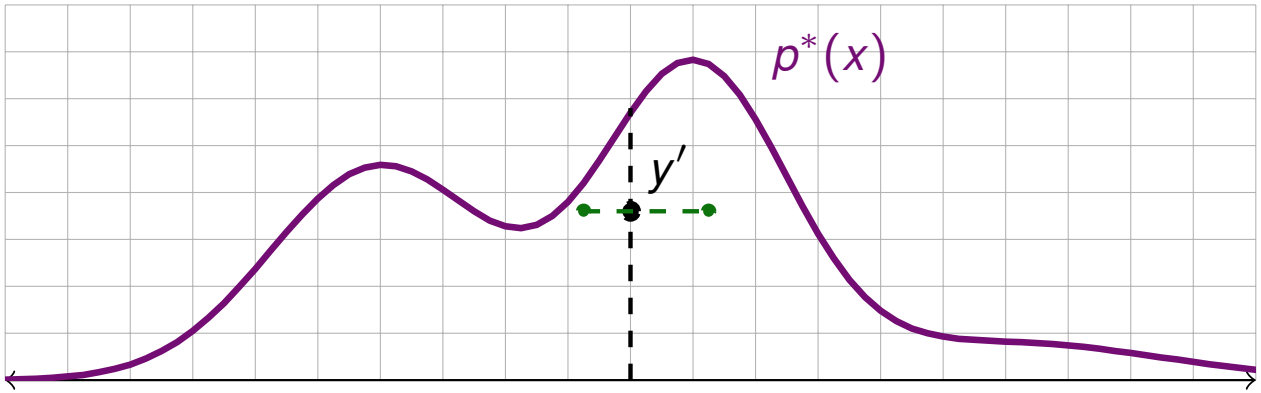
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density;

Slice sampling – defining the y-level slices



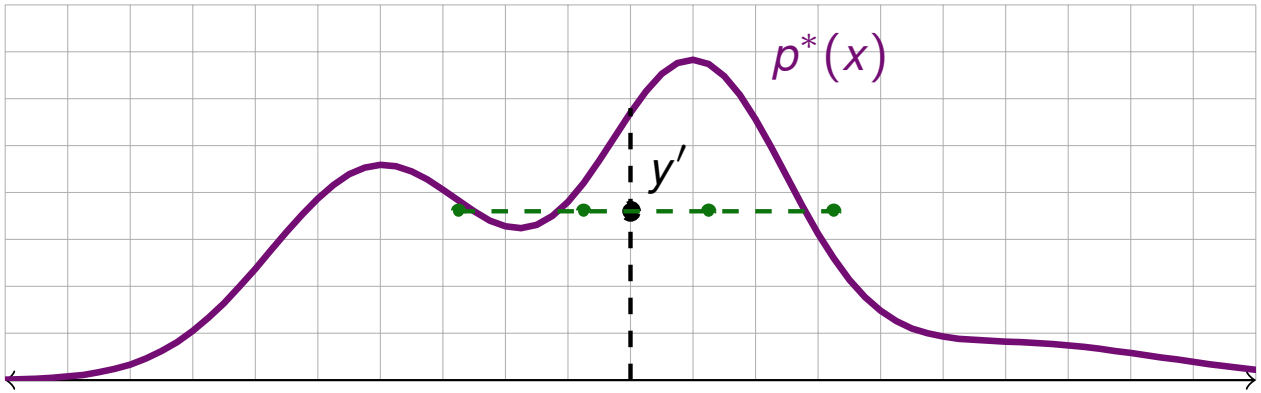
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.

Slice sampling – defining the y-level slices



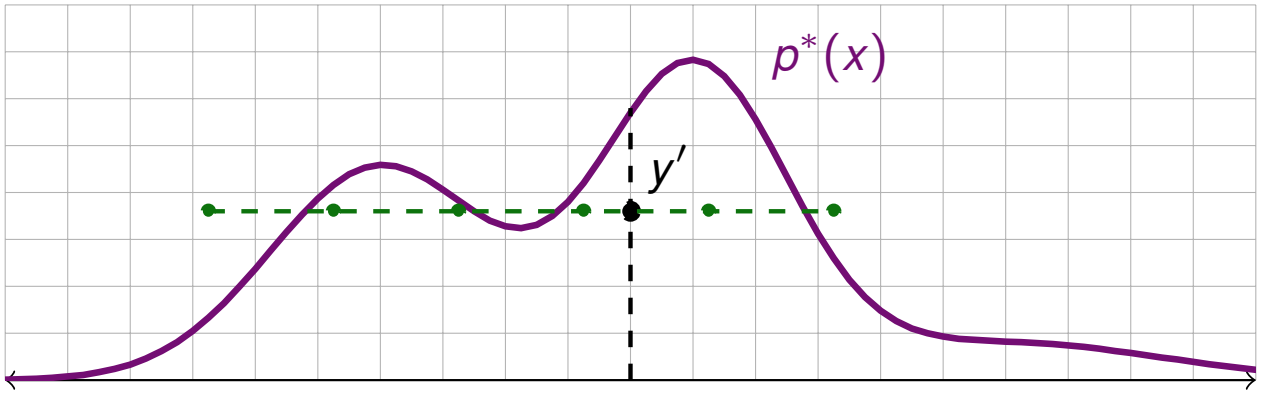
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.

Slice sampling – defining the y-level slices



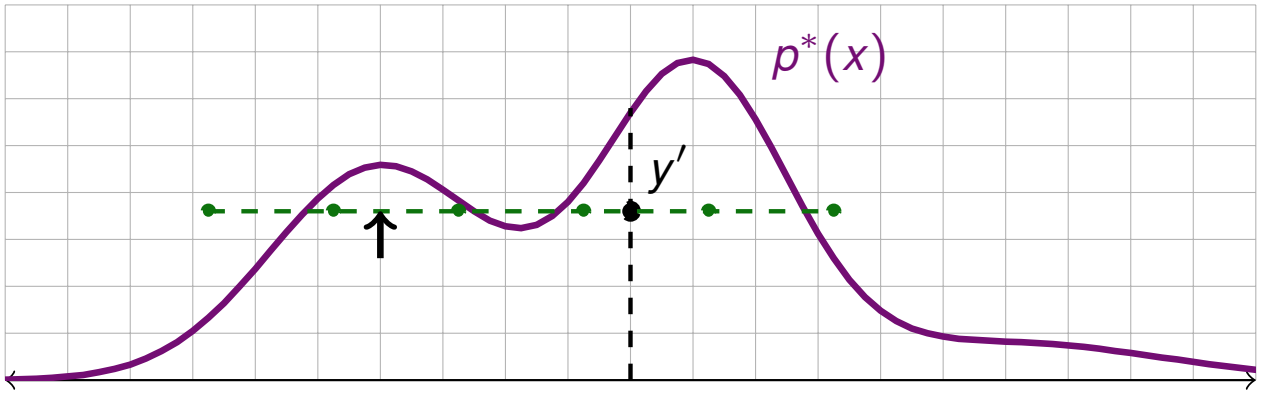
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes.

Slice sampling – defining the y-level slices



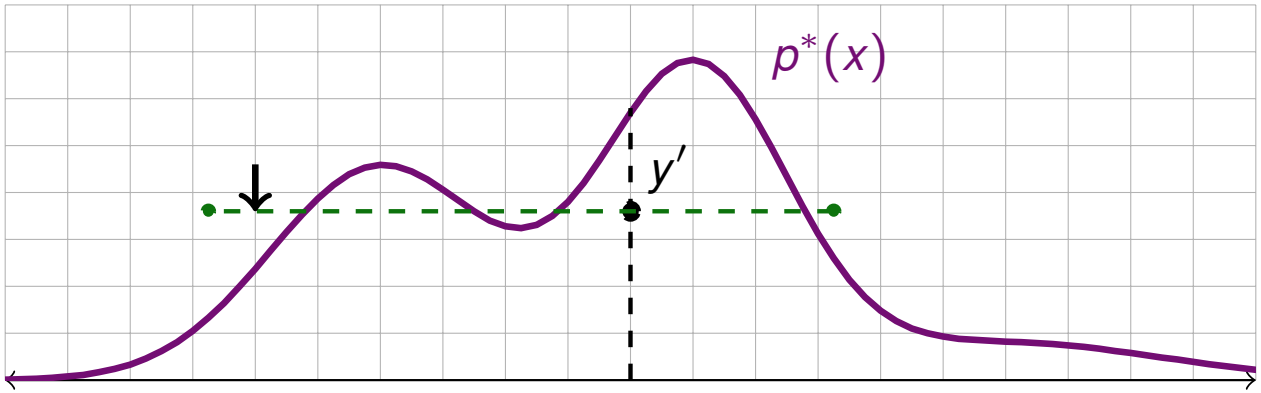
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes. If so, keep extending.

Slice sampling – defining the y-level slices



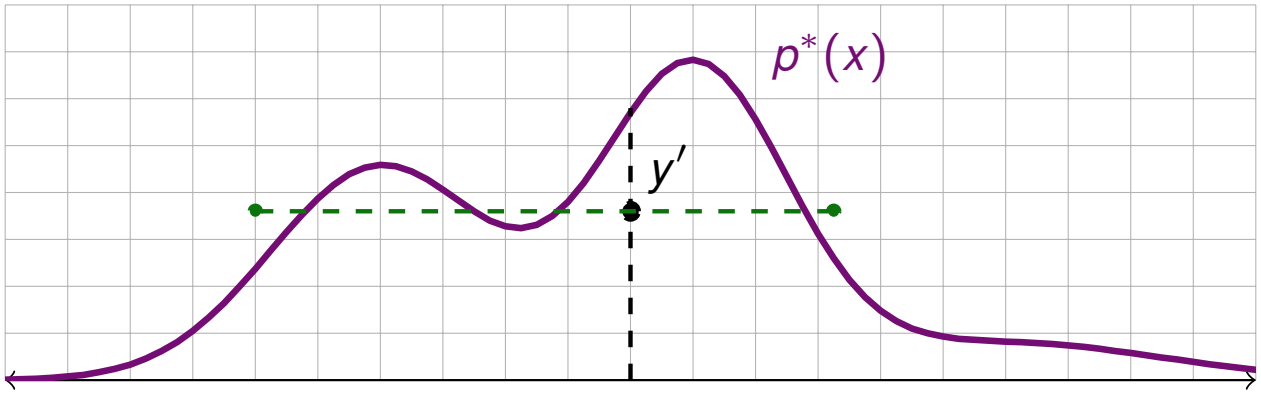
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes. If so, keep extending. Same process started at sample in next mode can cross back into initial mode, preserving reversibility.

Slice sampling – defining the y-level slices



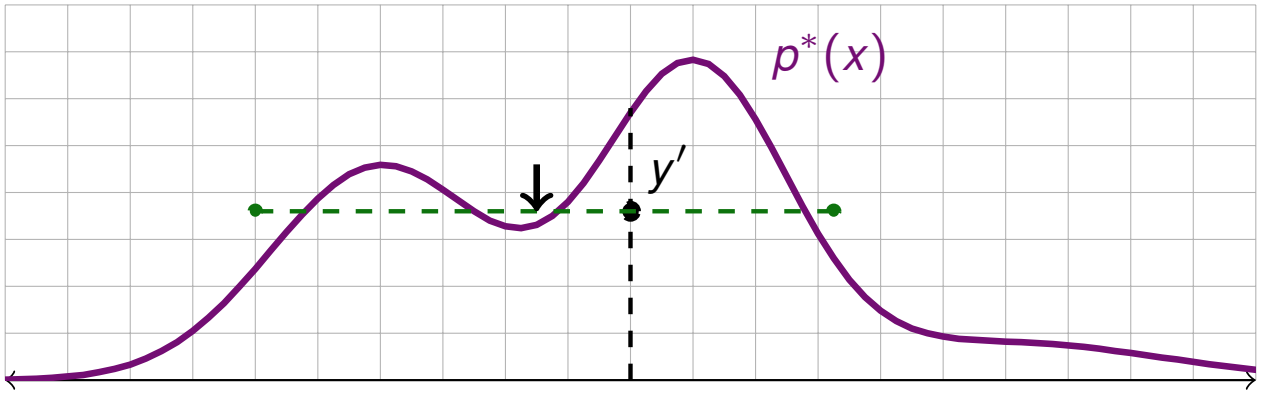
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x,y) \to (x',y')) = T((x',y') \to (x,y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes. If so, keep extending. Same process started at sample in next mode can cross back into initial mode, preserving reversibility.
- When rejection sampling, any samples that fall above density can be used to restrict proposal slice (adaptive RS).

Slice sampling – defining the y-level slices



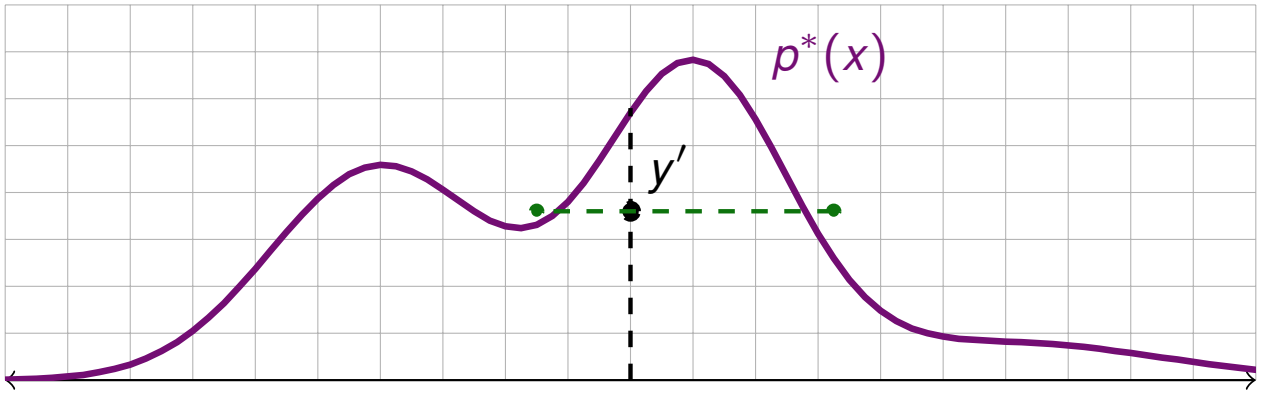
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes. If so, keep extending. Same process started at sample in next mode can cross back into initial mode, preserving reversibility.
- When rejection sampling, any samples that fall above density can be used to restrict proposal slice (adaptive RS).

Slice sampling – defining the y-level slices



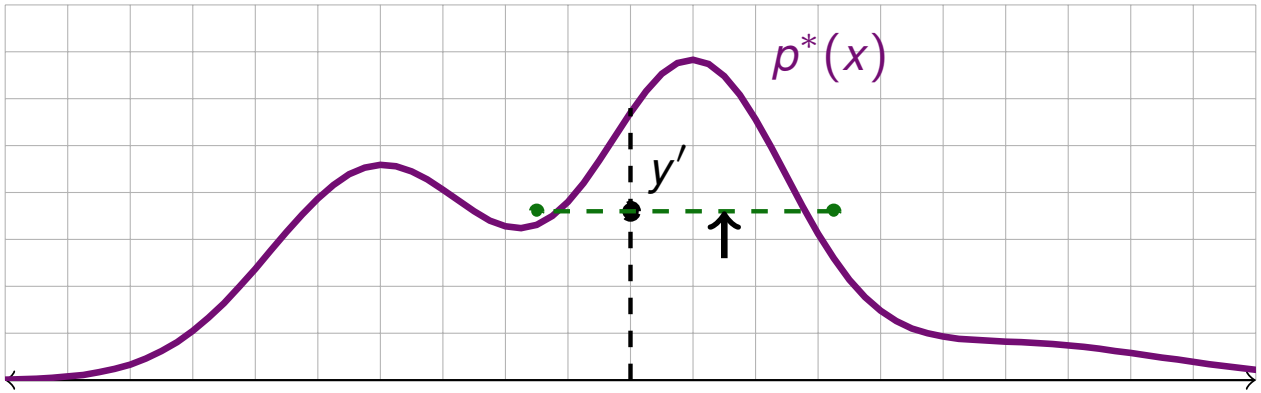
- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes. If so, keep extending. Same process started at sample in next mode can cross back into initial mode, preserving reversibility.
- When rejection sampling, any samples that fall above density can be used to restrict proposal slice (adaptive RS).

Slice sampling – defining the y-level slices



- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x,y) \to (x',y')) = T((x',y') \to (x,y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes. If so, keep extending. Same process started at sample in next mode can cross back into initial mode, preserving reversibility.
- When rejection sampling, any samples that fall above density can be used to restrict proposal slice (adaptive RS).

Slice sampling – defining the y-level slices



- Finding the boundaries of $\{x : p^*(x) \geq y\}$ requires inverting the density function: often very difficult.
- Target stationary distribution is uniform under $p^*(x)$ density curve. So to preserve stationarity it is sufficient to have:
  - ergodicity
  - detailed balance $\Rightarrow T((x, y) \to (x', y')) = T((x', y') \to (x, y))$.
- $\Rightarrow$ sufficient to transition uniformly within local mode (assuming contiguous support).
- Grow slice in steps of size Δ until both ends are above density; rejection sample.
  - First step randomly positioned around current sample.
  - Slice may cross over neighbouring modes. If so, keep extending. Same process started at sample in next mode can cross back into initial mode, preserving reversibility.
- When rejection sampling, any samples that fall above density can be used to restrict proposal slice (adaptive RS).

Evaluating the evidence — Annealed Importance Sampling (AIS)

- Bayesian learning often depends on evaluating the marginal likelihood

$$p(\mathcal{D}) = \int d\theta \, p(\mathcal{D}|\theta) p(\theta)$$

Evaluating the evidence — Annealed Importance Sampling (AIS)

- Bayesian learning often depends on evaluating the marginal likelihood

$$p(\mathcal{D}) = \int d\theta \, p(\mathcal{D}|\theta) p(\theta)$$

- Prior mass and likelihood may not agree, so simple Monte-Carlo estimate (with samples drawn from $p(\theta)$) may have high variance (particularly in high dimensions).

Evaluating the evidence — Annealed Importance Sampling (AIS)

- Bayesian learning often depends on evaluating the marginal likelihood

$$p(\mathcal{D}) = \int d\theta \, p(\mathcal{D}|\theta)p(\theta)$$

- Prior mass and likelihood may not agree, so simple Monte-Carlo estimate (with samples drawn from $p(\theta)$) may have high variance (particularly in high dimensions).
- Samples from unnormalised posterior $p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$ can often be found by MCMC; but samples alone do not provide estimate of $p(\mathcal{D})$.

Evaluating the evidence — Annealed Importance Sampling (AIS)

- Bayesian learning often depends on evaluating the marginal likelihood

$$p(\mathcal{D}) = \int d\theta \, p(\mathcal{D}|\theta)p(\theta)$$

- Prior mass and likelihood may not agree, so simple Monte-Carlo estimate (with samples drawn from $p(\theta)$) may have high variance (particularly in high dimensions).
- Samples from unnormalised posterior $p(\theta|\mathcal{D}) \propto p(\mathcal{D}|\theta)p(\theta)$ can often be found by MCMC; but samples alone do not provide estimate of $p(\mathcal{D})$.
- Idea: use MCMC transitions from known distribution to form proposal for importance sampling.

Annealed importance sampling

- Consider a sequence of densities $p_1 \ldots p_n$:

$$p_j(x) = \frac{1}{Z_j} p_*(x)^{\beta_j} p_\bullet(x)^{1-\beta_j} \qquad 1 = \beta_1 > \cdots > \beta_n = 0$$

  where $p_1 = p_*$ is the target density and $p_n = p_\bullet$ is easy to sample and normalise (perhaps a prior).

- Let $T_j(x \to x')$ be an MCMC transition rule that leaves p_j invariant.

- Draw samples from $q(x_1 \ldots x_{n-1}) = p_n T_{n-1} \ldots T_2$:

$$x_{n-1}^{(t)} \sim p_n; \quad x_{n-2}^{(t)} \sim T_{n-1}(x_{n-1}^{(t)} \to \cdot); \quad \ldots \quad ; x_1^{(t)} \sim T_2(x_2^{(t)} \to \cdot)$$

- Importance weight samples with

$$w^{(t)} = \left(\frac{p_*(x_1^{(t)})}{p_\bullet(x_1^{(t)})}\right)^{\beta_1-\beta_2} \left(\frac{p_*(x_2^{(t)})}{p_\bullet(x_2^{(t)})}\right)^{\beta_2-\beta_3} \cdots \left(\frac{p_*(x_{n-1}^{(t)})}{p_\bullet(x_{n-1}^{(t)})}\right)^{\beta_{n-1}-\beta_n}$$

- Then:

$$\frac{1}{T}\sum_t w^{(t)} \to \frac{Z_*}{Z_\bullet}$$

Annealed importance weights

- Define the reversed transition probability:

$$\overleftarrow{T}_j(x' \to x) = T_j(x \to x') \frac{p_j(x)}{p_j(x')}$$

(only a different function if detailed balance doesn't hold).

- We drew samples from joint $q(x_1 \ldots x_{n-1})$. Need a joint target:

$$p(x_1 \ldots x_{n-1}) = p_1(x_1)\overleftarrow{T}_2(x_1 \to x_2)\overleftarrow{T}_3(x_2 \to x_3)\ldots\overleftarrow{T}_{n-1}(x_{n-2} \to x_{n-1})$$

(Note that x_1 is drawn from the target distribution p_*).

- Importance weights are:

$$\begin{aligned}
w^{(t)} = \frac{p}{q} &= \frac{p_1(x_1^{(t)})\overleftarrow{T}_2(x_1^{(t)} \to x_2^{(t)})\overleftarrow{T}_3(x_2^{(t)} \to x_3^{(t)})\ldots\overleftarrow{T}_{n-1}(x_{n-2}^{(t)} \to x_{n-1}^{(t)})}{T_2(x_2^{(t)} \to x_1^{(t)})T_3(x_3^{(t)} \to x_2^{(t)})\ldots T_{n-1}(x_{n-1}^{(t)} \to x_{n-2}^{(t)})p_n(x_{n-1}^{(t)})} \\
&= p_1(x_1^{(t)})\frac{p_2(x_2^{(t)})}{p_2(x_1^{(t)})}\frac{p_3(x_3^{(t)})}{p_3(x_2^{(t)})}\cdots\frac{p_{n-1}(x_{n-1}^{(t)})}{p_{n-1}(x_{n-2}^{(t)})}\Big/p_n(x_{n-1}^{(t)}) \\
&= \frac{p_*(x_1^{(t)})^{\beta_1}p_\bullet(x_1^{(t)})^{1-\beta_1}}{p_*(x_1^{(t)})^{\beta_2}p_\bullet(x_1^{(t)})^{1-\beta_2}}\frac{p_*(x_2^{(t)})^{\beta_2}p_\bullet(x_2^{(t)})^{1-\beta_2}}{p_*(x_2^{(t)})^{\beta_3}p_\bullet(x_2^{(t)})^{1-\beta_3}}\cdots\frac{p_*(x_{n-1}^{(t)})^{\beta_{n-1}}p_\bullet(x_{n-1}^{(t)})^{1-\beta_{n-1}}}{p_*(x_{n-1}^{(t)})^{\beta_n}p_\bullet(x_{n-1}^{(t)})^{1-\beta_n}} \\
&= \left(\frac{p_*(x_1^{(t)})}{p_\bullet(x_1^{(t)})}\right)^{\beta_1-\beta_2}\left(\frac{p_*(x_2^{(t)})}{p_\bullet(x_2^{(t)})}\right)^{\beta_2-\beta_3}\cdots\left(\frac{p_*(x_{n-1}^{(t)})}{p_\bullet(x_{n-1}^{(t)})}\right)^{\beta_{n-1}-\beta_n}
\end{aligned}$$

Other Ideas in MCMC

- Rao-Blackwellisation or collapsing: integrate out variables that are tractable to lower variance or improve convergence.
- Exact sampling: yield **exact** samples from the equilibrium distribution of a Markov chain, making use of the idea of coupling from the past—if two Markov chains use the same set of pseudo-random numbers, then even if they started in different states, once they transition to the same state they will stay in the same state.
- Adaptive rejection sampling: during rejection sampling, if sample rejected use it to improve the proposal distribution.
- Nested sampling: another, quite different, Monte-Carlo integration method.
- ...

Sampling – Importance Resampling (SIR)

Consider message passing in an analytically intractable graphical model (*e.g.* a NLSSM), where we represent messages using samples.

- MCMC requires burn-in to generate accurate samples. Unsuited to online settings.
- Proposal-based methods (*e.g.* rejection sampling) generate samples immediately.
- SIR is another proposal-based approach to generating samples from an unnormalised distribution
  - Sample $\xi^{(s)} \sim q(x)$, and calculate importance weights $w^{(s)} = p(\xi^{(s)})/q(\xi^{(s)})$.
  - Define $\tilde{q}(x) = \sum_{s=1}^{S} w^{(s)} \delta(x - \xi^{(s)}) \Big/ \sum_{s=1}^{S} w^{(s)}$.
  - Resample $x^{(t)} \sim \tilde{q}(x)$.

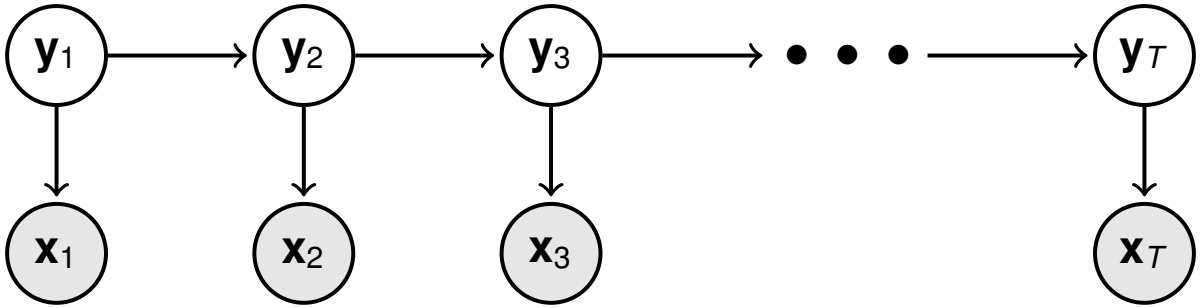
  Resampled points yield consistent expectations (as in Importance Sampling),

$$E_x[F(x)] = \int dx\ F(x)\tilde{q}(x) = \int dx\ F(x) \frac{\sum_{s=1}^{S} w^{(s)} \delta(x - \xi^{(s)})}{\sum_{s=1}^{S} w^{(s)}}$$

$$= \frac{\sum_{s=1}^{S} w^{(s)} F(\xi^{(s)})}{\sum_{s=1}^{S} w^{(s)}}$$

  but are not unbiased for $S < \infty$ even if all distributions are normalised.
- For integration, SIR introduces added sampling noise relative to IS (although trades off with weight variance). But for message passing unweighted samples are redrawn towards bulk of distribution.

Sequential Monte Carlo — particle filtering



Suppose we want to compute $p(\mathbf{y}_t|\mathbf{x}_1 \dots \mathbf{x}_t)$ in an NLSSM. We have

$$p(\mathbf{y}_t|\mathbf{x}_1 \dots \mathbf{x}_t) \propto \int p(\mathbf{y}_t \mathbf{y}_{t-1} \mathbf{x}_t|\mathbf{x}_1 \dots \mathbf{x}_{t-1})\, d\mathbf{y}_{t-1}$$
$$= \int p(\mathbf{x}_t|\mathbf{y}_t) p(\mathbf{y}_t|\mathbf{y}_{t-1}) p(\mathbf{y}_{t-1}|\mathbf{x}_1 \dots \mathbf{x}_{t-1})\, d\mathbf{y}_{t-1}$$

If we have samples $\mathbf{y}_{t-1}^{(s)} \sim p(\mathbf{y}_{t-1}|\mathbf{x}_1 \dots \mathbf{x}_{t-1})$ we can recurse (approximately):

- draw $\mathbf{y}_t^{(s)} \sim p(\mathbf{y}_t|\mathbf{y}_{t-1}^{(s)})$
- calculate (unnormalised) weights $w_t^{(s)} = p(\mathbf{x}_t|\mathbf{y}_t^{(s)})$.
- resample $\mathbf{y}_t^{(s')} \sim \sum_{s=1}^{S} w_t^{(s)} \delta(\mathbf{y} - \mathbf{y}_t^{(s)}) \Big/ \sum_{s=1}^{S} w_t^{(s)}$

This is called Particle Filtering (this version, with $q = p(\mathbf{y}_t|\mathbf{y}_{t-1}^{(s)})$ is also called a "bootstrap filter" or "condensation" algorithm).

Particle filtering

- Technically, the entire trajectory $\mathbf{y}_1^{(s)}, \mathbf{y}_2^{(s)}, \ldots, \mathbf{y}_t^{(s)}$ is resampled. As $S \to \infty$, surviving trajectories are drawn from full posterior (but not for practical sample sizes).

Particle filtering

- Technically, the entire trajectory $\mathbf{y}_1^{(s)}, \mathbf{y}_2^{(s)}, \ldots, \mathbf{y}_t^{(s)}$ is resampled. As $S \to \infty$, surviving trajectories are drawn from full posterior (but not for practical sample sizes).
- Resampling can be avoided by propagating weights instead. However variance in weights accumulates. Resampling helps eliminate unlikely particles, reducing variance.

Particle filtering

- Technically, the entire trajectory $\mathbf{y}_1^{(s)}, \mathbf{y}_2^{(s)}, \ldots, \mathbf{y}_t^{(s)}$ is resampled. As $S \to \infty$, surviving trajectories are drawn from full posterior (but not for practical sample sizes).
- Resampling can be avoided by propagating weights instead. However variance in weights accumulates. Resampling helps eliminate unlikely particles, reducing variance.
- Possible to trigger resample events conditioned on variance – “stratified resampling”.

Particle filtering

- Technically, the entire trajectory $\mathbf{y}_1^{(s)}, \mathbf{y}_2^{(s)}, \ldots, \mathbf{y}_t^{(s)}$ is resampled. As $S \to \infty$, surviving trajectories are drawn from full posterior (but not for practical sample sizes).
- Resampling can be avoided by propagating weights instead. However variance in weights accumulates. Resampling helps eliminate unlikely particles, reducing variance.
- Possible to trigger resample events conditioned on variance – "stratified resampling".
- Better proposal distributions (including, ideally, $p(\mathbf{y}_t|\mathbf{y}_{t-1}^{(s)}, \mathbf{x}_t)$ if available) improve performance.

Particle filtering

- Technically, the entire trajectory $\mathbf{y}_1^{(s)}, \mathbf{y}_2^{(s)}, \ldots, \mathbf{y}_t^{(s)}$ is resampled. As $S \to \infty$, surviving trajectories are drawn from full posterior (but not for practical sample sizes).
- Resampling can be avoided by propagating weights instead. However variance in weights accumulates. Resampling helps eliminate unlikely particles, reducing variance.
- Possible to trigger resample events conditioned on variance – "stratified resampling".
- Better proposal distributions (including, ideally, $p(\mathbf{y}_t|\mathbf{y}_{t-1}^{(s)}, \mathbf{x}_t)$ if available) improve performance.
- Particle *smoothing* is possible, but often has high variance. Difficult to create a good proposal.

Particle filtering

- Technically, the entire trajectory $\mathbf{y}_1^{(s)}, \mathbf{y}_2^{(s)}, \ldots, \mathbf{y}_t^{(s)}$ is resampled. As $S \to \infty$, surviving trajectories are drawn from full posterior (but not for practical sample sizes).
- Resampling can be avoided by propagating weights instead. However variance in weights accumulates. Resampling helps eliminate unlikely particles, reducing variance.
- Possible to trigger resample events conditioned on variance – "stratified resampling".
- Better proposal distributions (including, ideally, $p(\mathbf{y}_t|\mathbf{y}_{t-1}^{(s)}, \mathbf{x}_t)$ if available) improve performance.
- Particle *smoothing* is possible, but often has high variance. Difficult to create a good proposal.
- Widely used in engineering tracking applications, where filtering is most appropriate.

Particle filtering

- Technically, the entire trajectory $\mathbf{y}_1^{(s)}, \mathbf{y}_2^{(s)}, \ldots, \mathbf{y}_t^{(s)}$ is resampled. As $S \to \infty$, surviving trajectories are drawn from full posterior (but not for practical sample sizes).
- Resampling can be avoided by propagating weights instead. However variance in weights accumulates. Resampling helps eliminate unlikely particles, reducing variance.
- Possible to trigger resample events conditioned on variance – "stratified resampling".
- Better proposal distributions (including, ideally, $p(\mathbf{y}_t|\mathbf{y}_{t-1}^{(s)}, \mathbf{x}_t)$ if available) improve performance.
- Particle *smoothing* is possible, but often has high variance. Difficult to create a good proposal.
- Widely used in engineering tracking applications, where filtering is most appropriate.
- Many variants ...

References

- Excellent introductions to MCMC: Radford Neal (1993) Probabilistic Inference Using Markov Chain Monte Carlo Methods, Tech report CRG-TR-93-1 Toronto Computer Science; and David MacKay, Introduction to Monte Carlo Methods.
- Radford Neal (1998) Annealed Importance Sampling, Tech report 9805 Toronto Statistics, and Statistics and Computing (2001) 11:125-139.
- Radford Neal (2003) Slice Sampling, 31:705-767.
- W. R. Gilks and P. Wild (1992) Adaptive Rejection Sampling for Gibbs Sampling, Applied Statistics 41:337-348.
- James G. Propp and David B. Wilson (1996) Exact sampling with coupled Markov chains and applications to statistical mechanics, Random Structures and Algorithms, 9:223-252.
- A. Doucet, N. de Freitas and N. Gordon (2001) Sequential Monte Carlo Methods in Practice, Springer.
- P. Del Moral, A. Doucet, A. Jasra (2006) Sequential Monte Carlo Samplers, Journal of the Royal Statistical Society B, 68:411-436.
- P. Fearnhead (1998) Sequential Monte Carlo Methods in Filter Theory, Ph.D. Thesis, Merton College, University of Oxford.
- M. Isard and A. Blake (1996) Contour tracking by stochastic propagation of conditional density, European Conference on Computer Vision.