Sampled representations:

A crucial, useful fact

\[ x_i \sim q(x) \]

Then

\[ \frac{1}{N} \sum f(x_i) \rightarrow \int f g \, dx \]

This gives a method to represent PDF densities by samples.

\[ \rightarrow \text{it's a representation, because we can compute expectations} \]

Notice this

But we may not get good ests:
$x_i \sim g$ will most likely give a high variance estimate of $\mathbb{E}[f] = \int fg \, dx$

**Importance sampling:**

1. Assume we know $h$, which is "similar" to $fg$ (big when $fg$ is big).

   - $x_i \sim h$

2. \[
   \frac{1}{N} \sum \frac{f(x_i) \cdot g(x_i)}{h(x_i)} \rightarrow \int fg \, dx
   \]

   will be a better estimate (lower variance)
This suggests a representation

\[ \{ (x_i, w_i) \} = \{ (x_i, \frac{g(x_i)}{h(x_i)}) \} \]

Representing a posterior from a prior

we have

\[ p(x/\theta) \]

\[ p(\theta) \]

\[ \text{likelihood} \]

\[ \text{prior} \]

- assume we have \( \theta_i \sim p(\theta) \)

- want to represent

\[ p(\theta/x) = \frac{p(x/\theta) p(\theta)}{p(x)} \]

\[ = \frac{p(x/\theta) p(\theta)}{\int p(x/\theta) p(\theta) d\theta} \]
Now \[ w_i = p(x_i | \theta_i) \]

then

\[ \frac{1}{N} \sum_i w_i \rightarrow \int p(x(\theta_i)) p(\theta) d\theta \]

and

\[ \frac{1}{N} \sum_i f(\theta_i) w_i \rightarrow \int f(\theta) p(x(\theta) | \theta) d\theta \]

So

\[ \frac{\sum_i f(\theta_i) w_i}{\sum_i w_i} \rightarrow \int [f(\theta) p(\theta | x)] d\theta \]

This gives one amazingly simple recipe for representing a posterior!
But: it's often not very good.

![Graph showing probability distributions with p(x|θ) and p(θ) axes.]

In particle filtering, the effect that results is sometimes called "sample impoverishment."

- Many wi are very small, few are big
  \[ \Rightarrow \] very few θ_i contribute to the estimate of the integral in any significant way.

\[ \Rightarrow \text{Test:} \]
look at variance of wi.
How do we get samples?

- Uniform, Normal, Some others:
- there are standard algs
- Finite mixtures of above:
  - Draw a sample from mixture weights
  - Then from component

Rejection sampling:
- Wish to draw a sample from $p(x)$
- Know how to draw from $q(x)$
- $p(x) \leq c q(x)$
Algorithm:
- repeat until accepted
  - draw $x_i \sim q$
  - draw $y \sim U[0, 1]$
  - if $y \leq \frac{p(x_i)}{cq(x_i)}$ accept
  - else go again

1) Why does this work?
\[ P\{\text{sample at } x_i\} = P\{\text{generate sample at } x_i\ \text{and accept}\} \]
\[ = q(x_i) \cdot \frac{p(x_i)}{cq(x_i)} \]
So
\[ P\{\text{sample at } x_i\} \propto p(x_i) \]

2) How efficient is it?
- depends on relationship between $P, q$. 
many reject in this case

Here we have multiple intervals and draw from c.q as a mixture
- Drawing a sample from a discrete distribution

- draw $x \sim \text{U}(0, 1)$ to point location

Or:

- build a binary tree,
do rejection sampling

Or:

- build a binary tree,
use a tertiary biased sample.
Gibbs sampling:

- Assume we want to draw a sample from a multivar distribs. $p(u, v)$

- Assume $p(u | v)$ is easy
  $p(v | u)$ is easy.

- Procedure: Start with $(u_0, v_0)$
  - Draw $u_i \sim p(u | v_{i-1})$
    $v_i \sim p(v | u_i)$
  - Do this many times
    $(u_n, v_n) \sim p(u, v)$

  (If we iterate often enough).
\[ p(u, v) \propto \exp \left[ -\frac{1}{2} \left( \frac{u^2}{\sigma_1^2} + \frac{v^2}{\sigma_2^2} \right) \right] \]

- each stage is a draw from a univariate Gaussian
- not much real advantage, because we could draw \((u, v) \sim N(0, 1)\) then scale.

\text{E.g.: strongly correlated } u, v

\text{Here it will take a long time for } u_n, v_n \text{ to have high prob.}
Markov chain

sequence of random vars,

\[ X_i \ldots \text{ such that} \]

\[ P(X_{i+1} = x \mid X_i = \ldots X_0 =) = p(X_{i+1} = x \mid X_i = \ldots) \]

And this transition is stationary.

Discrete case:

Write \( p(X_{i+1} = x_u \mid X_i = x_v) = P_{uv} \)

some matrix

Now, start with \( p(X_0 = x_w) = \pi_w \)

\[ P(X_1 = x_3) = \sum_{x_w} P(X_1 = x_3 \mid X_0 = x_w) p(X_0 = x_w) \]

\[ = \sum_w P_{uw} \pi_w = P \pi \]
Similarly:

\[ P(\ x_i = x_s) \]

\[ = \sum_{w_i - 1} P_{w_i} P_{w_{i-1}} \cdots P_{w_2} P_1 \Pi_{w_0} \]

\[ = P \]

\[ = P^k \Pi \]

- Now \( P \) has at least 1 unit eigenvalue
  (because \( P^T \) does, \( P^T \mathbf{1} = \mathbf{1} \))

- Assume only 1; all others < 1

- Then \( P^k \mathbf{v} \to \Pi_s \) such that \( P \Pi_s = \Pi_s \)

Stationary Distribution
This yields following procedure to sample from discrete \( \text{Dist} \), \( T_i \):

1. Build \( MC \) with \( T_i \) as stationary dist.
2. "Run the chain."

Ex: to sample from \( \frac{1}{2} \) \( \frac{1}{2} \)

\[
\begin{bmatrix}
1 - \frac{3}{2} & \frac{1}{2} \\
\frac{3}{2} & 1 - \frac{3}{2}
\end{bmatrix}
\]

Something seems like a problem here \( \rightarrow \)

No guarantee that \( \overline{x_i} \), \( \overline{x_{i-1}} \)!
Q: How do we build such a chain?

- Design requirement

\[ \sum_{x_u} P(X_{i+1} = x_u | X_i = x_\omega) \prod_i (X_i = x_\omega) = \prod_{x_u} \prod_i (X_i = x_u) \]

\[ = \prod (X_i = x_u) \]

or

\[ \sum P_{u\omega} \prod \omega = \prod_{u} \]

This is a weak condition, rather hard to achieve by choice of \( P \)

- Detailed balance (stronger condition)

\[ P_{u\omega} \prod \omega = P_{u} \prod \omega \]

For each \( u, \omega \)

Notice this implies above.
Now we can build an MC in a continuous domain with stationary dist \( \Pi(x) \).

1) obtain proposal process \( P(x \rightarrow y) \) (this is \( P(X_{i+1} = y \mid X_i = x) \))

   - \( P(x \rightarrow y) \) should satisfy some support conditions.
   
   - Good enough if:
     \[
     \Pi(y) > 0 \Rightarrow P(x \rightarrow y) > 0 \\
     P(x \rightarrow y) > 0 \Rightarrow P(y \rightarrow x) > 0
     \]

2) Let the achieve detailed balance by a form of rejection sampling.
Algorithm

given $x_i$

- draw $x_{i+1} \sim P(x_i \rightarrow y)$
- compute

$$\alpha = \min \left( 1, \frac{P(x_{i+1} \rightarrow x_i) \Pi(x_{i+1})}{P(x_i \rightarrow x_{i+1}) \Pi(x_i)} \right)$$

$$x_{i+1} = \begin{cases} 
\text{draw } x_{i+1} \text{ with prob } \alpha \\
x_i & \text{with prob } 1 - \alpha.
\end{cases}$$

Thus: for large enough $N$,

$$x_N \sim \Pi$$

whatever $x_0$
Amazingly powerful algorithm

+ • can draw samples from unnormalized dists

+ • easy; general

+ • can sometimes prove that chains are "fast mixing" (i.e. forget start point quickly)

? • • Samples are correlated

- • Correlation may be hard to est.

- • Can be hard to tell if sampler is "sticky"

- • "Sticky" samplers look better.