1 Sequential Monte Carlo

The sequential Monte Carlo is a strategy for breaking down a really high dimensional problem into a sequence of low dimensional problems. The intuition is that if you do this, then the problem becomes more computationally tractable. In the first step of the strategy, we just perform importance sampling, and the main work steps after that:

\[
X_1^s \sim p(x_1) \\
w_s^i := \frac{p(y_1, x_1)}{p(x_1)}
\]

The idea is to take the samples from the first step and use them as proposals for the second step. Now we have samples for the first two steps, therefore, the subsequent steps are:

\[
\hat{\pi}_s^t(x_{1:t-1}) = \sum_s \tilde{w}_s^t \delta_{x_{1:t}}(x_{1:t-1}) \approx p(x_{1:t-1}|y_{1:t-1}) \\
X_t^s \sim p(x_t|x_{1:t-1}) \\
w_t^i := p(y_t|x_{1:t})
\]

1.1 An Example

It is shown visually in Fig. 1. Suppose that we have a long time series, and now we are going to truncate it. For each of these particles we have a weight and a value as shown in Fig. 1(a). So the idea of sequential Monte Carlo is to construct a sample for the first two time points in the series starting from a sample for the first time point in the series. The idea behind important sampling is that the particles with high weights are kind of good samples for the first time point while the particles with low weights are not so good samples. Let's use this first sample as a basis for the next samples as shown in Fig. 1(b). The idea here is that if it is a good sample, we are going to use it more often as a basis for the next sample. If it is a bad sample, we might not use it at all and so the sample dies out. Formally, what is going on here is the following. We have a series of samples:

\[
(w_1^1, x_1^1), \ldots, (w_1^t, x_1^t)
\]
so we have $w_1$ and $X_1$ for sample 1 through $w_s$ and $X_s$ for sample $s$. We are going to use these things to define a probability distribution in terms of a set of Delta functions. This probability distribution we write as follows (we call it empirical distribution):

$$\hat{\pi}_1(x_1) = \sum_{s=1}^{S} \bar{w}_s \delta_{x_s}(x_1)$$

(4)

where $\bar{w}$ is the normalized weights:

$$\bar{w}_s = \frac{w_s}{\sum_{s'} w_{s'}}$$

(5)

Delta function is something that we can only define relative to an integral. It basically ignores all points of the integral and just pick out the single point at $x_0$ to the integral:

$$\int dx \delta_{x_0}(x)f(x) := f(x_0)$$

(6)

Now that we have the definition for Delta function, we can have the following explanation for Eq. (4): Suppose that $X_1$ was one dimensional, then we could think of this as some sort of distribution where each of particles has height $\bar{w}_1$ and are centered at $X_1$. So once we have defined this, how can we actually sample from this distribution? Suppose that we were trying to sample $X$ from this empirical distribution:

$$X \sim \hat{\pi}_1(x_1)$$

(7)

There is only infinite number of values $X$, we can sample from this distribution. So basically, we have to pick one of the values in this set of weighted values in Eq. (3) and which one we pick depends on the weights. So, first we pick an integer from a discrete distribution which corresponds to each of these values, and that integer we call the ancestor:

$$a \sim \text{Discrete}(\bar{w}_1^s, \ldots, \bar{w}_1^S)$$

(8)

Basically, we pick from the previous set in Eq. (3) whichever index we selected.
So this is what listed below here. We picked an ancestor index based on the weights of the previous set:

$$ a^s_1 \sim \text{Discrete}(\tilde{w}^s_1, \ldots, \tilde{w}^s_1) $$

(9)

Once we picked an ancestor index, then we use this as a starting point and now we sample the next $X$ in the sequence:

$$ X^s_2 \sim p(x_2|x^a_1^s) $$

$$ w^s_2 := p(y_2|x^s_{1:2}) $$

(10)

Now that we have the next two points in the sequence, we can define the following notation, in which we can go backward and think of the ancestor of ancestor:

$$ X^s_{1:t} = \left( X^s_t, X^a_{t-1}^s, X^a_{t-2}^s, \ldots \right) $$

(11)

Now we can repeatedly do this trick as shown in Fig. 1(c).

$$ a^s_t \sim \text{Discrete}(\tilde{w}^s_{t-1}, \ldots, \tilde{w}^s_{t-1}) $$

$$ X^s_t \sim p(x_t|x^a_{t-1}^s) $$

$$ w^s_t := p(y_t|x^s_{1:t-1}) $$

(12)

Basically, like at the first step, we have a sample for the first time point. At the second step, we have samples on the first two time points, and then we pick ancestors of those. Those ones that have relatively high weights may get picked multiple times, and those ones with lower weights may not get selected at all.

We can now just keep repeating this step, and get this kind of natural selection process among some as shown in Fig. 2. We start from the left, and as we keep going forward, more of these particle lineages eventually die out. So by the time we have gotten to the end, we have a bunch of particles that all derive from two common ancestors. So, this is both like a good thing and a bad thing in the sense that when we are sampling this, only the good samples that we drew at the first step survive by the time we get to the end. However, because we might not draw that many good ones, there is not that many kind of different ones that survive, and so we keep trying to replicating the same sample over and over again in order to generate more good samples.
1.2 General Formulation

It turns out that sequential Monte Carlo is a form of importance sampling and in order to see why, we need to do a little bit of algebraic manipulation. In order to do that we are going to take this idea of importance sampling. So we write an importance sampling which is going to take an unnormalized density, and divide it by some proposal:

\[ w_T = \frac{p(y_{1:T}, x_{1:T})}{q(x_{1:T})} \]  

So this is an importance weight. Now we are going to write that importance weights as a product over weights:

\[ w_T = \frac{p(y_{1:T}, x_{1:T})}{q(x_{1:T})} = \frac{p(y_1, x_1)}{q(x_1)} \prod_{t=2}^{T} \frac{p(y_t, x_t|x_{1:t-1})}{q(x_t|x_{1:t-1})} \]  

Let’s see how we could do this. If we want to write the joint probability of a sequence of \( Y \)s, then we can start by writing down the joint probability on the first one, and then we can take a product over \( t \). For each \( t \), we can look at the joint probability of the next observed value and the next unobserved value given everything the came before. In the same way, we can actually define a joint probability over all \( X \)s. So, we write a probability for \( X \) at the first step under the proposal, and then for each step in the proposal we can look at the probability of the next point given the preceding points.

This \( \frac{p(y_1, x_1)}{q(x_1)} \) would be like an importance weight for an importance sampling; and an importance sampler would have a target density \( \gamma_1(x_1) \) which is equal to \( p(y_1, x_1) \). Now, at each step we multiply by additional importance weights, as we recall this by \( w_t \) for \( t > 1 \):

\[ w_1 = \frac{p(y_1, x_1)}{q(x_1)} \]
\[ w_t = \frac{p(y_t, x_t|x_{1:t-1})}{q(x_t|x_{1:t-1})} \text{ for } t > 1 \]  

So, if we keep building up the densities like this, then the final density is the joint probability of all of the data points:

\[ \gamma_1(x_1) = p(y_1, x_1), \ldots, \gamma_T(x_T) = (y_{1:T}, x_{1:T}) \]  

Here in Eq. (14), at each step we are multiplying by the probability of the next point in the sequence given the preceding points in the sequence. So, in order to write that out, we will use a little trick. The probability of the next points given the preceding points, can be written as the joint probability of the first \( t \) points divided by the joint probability of the first \( t - 1 \) points. So, this is by definition equal to:

\[ p(y_t, x_t|x_{1:t-1}) = \frac{p(y_{1:t}, x_{1:t})}{p(y_{1:t-1}, x_{1:t-1})} = \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1})} \]  

4
And so when we substitute Eq. (17) in Eq. (14), then what we can write here is:

\[ w_T = \frac{p(y_{1:T}, x_{1:T})}{q(x_{1:T})} = \frac{\gamma(x_{1:T})}{q(x_{1:T})} = \frac{\gamma_1(x_1)}{q_1(x_1)} \prod_{t=2}^{T} \frac{\gamma_t(x_{1:t})}{\gamma_{t-1}(x_{1:t-1}) q(x_t|x_{1:t-1})} \]  

(18)

By looking at this equation, we see that we have a model like in the Markov model in which there is a sequence of variables. In fact, what we have written down is a version of this algorithm that is a lot more general, because we are really not making any assumptions here. We are just saying that there is a sequence of unnormalized densities, and we do not really say anything about them. Then, we assume that at each point in the sequence, we can generate a sample for a next step starting from samples from the previous step.

This is summarized here. Any sequential Monte Carlo algorithms consists of two steps. The first step is just importance sampling:

\[ X^s_1 \sim q(x_1) \]

\[ w^s_1 := \frac{\gamma(x^s_1)}{q(x^s_1)} \]  

(19)

And then the second step is:

\[ a^s_{t-1} \sim \text{Discrete}(\bar{w}^1_{t-1}, \ldots, \bar{w}^s_{t-1}) \]

\[ X^s_t \sim q(x^s_t|x^s_{1:t-1}), X^s_{1:t} := X^s_t, X^s_{1:t-1} \]

\[ w^s_t := \frac{\gamma_t(x^s_{1:t})}{\gamma_{t-1}(x^s_{1:t-1}) q(x^s_t|x^s_{1:t-1})} \]  

(20)

Here we first generate a sample from the approximate posterior. So basically we are saying:

\[ X_{1:t-1} \sim \hat{\pi}^s_{t-1}(x_{1:t-1}) \]  

(21)

And then we sample \( X_t \) from the following distribution:

\[ X_t \sim q(x_t|x_{1:t-1}) \]  

(22)

And then these importance weights, \( w^s_t \), can be thought of as what we call incremental weights. It shows how different is our new distribution from our old distribution, and how we changed the weights accordingly.

## 2 Markov Chain Monte Carlo

The idea of MCMC for sampling a distribution from \( \pi(x) \) is to construct a Markov Chain with a transition probability \( p(x'|x) \) such that the stationary distribution is \( \pi(x) \). Each \( x^s \) is sampled from \( x^{s-1} \) through the transition \( p(x'|x^{s-1}) \).

As an illustrative example we will consider the case of a Gaussian mixture model on the popular Iris dataset. The generative model is set up as following:

\[ \mu_k, \Sigma_k \sim p(\mu, \Sigma) \]
\[ z \sim \text{Discrete}(\pi_1, \ldots, \pi_K) \]
\[ y_n(z_n = k) \sim \mathcal{N}(\mu_k, \Sigma_k) \]

We will use MCMC, to generate samples from the posterior

\[ z_{1:N}^s, \mu_{1:N}^s, \Sigma_{1:N}^s \sim p(z_{1:N}^s, \mu_{1:N}^s, \Sigma_{1:N}^s | y_{1:N}). \]

Like the HMM in the previous lecture, this is a very high dimensional sampling problem. Intuitively, MCMC algorithms approach this sampling problem by iteratively generating samples. The high level idea here is that if we at each step can use the preceding sample \( x_{s-1} \) to generate a sample \( x_s \) that is at least as good, or preferably better, then we will eventually get to point where our current sample \( x_s \) is representative of the posterior. At that point we can’t generate

**Markov Chains**

A **Markov Chain** is defined as following: a sequence of random variables \( X^1, \ldots, X^S \) is a (discrete-time) Markov Chain with the **Markov property**,\n
\[ x^i | x^{i-1} \perp x^{(s-1)-i} \quad \forall i > 0 \]
\[ p(x^i | x^{1:(i-1)}) = p(x^i | x^{i-1}) \]

A Markov Chain is **homogeneous** when

\[ p(X^i = x' | X^{i-1} = x) = p(X^2 = x' | X^1 = x) = p(x' | x), \]

which means at each sampling step, it uses the same transition probability distribution.

**Detailed Balance**

A Markov Chain **converges** to a target density \( \pi(x) \) when

\[ \lim_{s \to \infty} p(X^s = x) = \pi(X = x) \]

Intuitively, a homogenous Markov Chain defines a random walk, in which the steps are governed by the transition distribution \( p(x^s | x^{s-1}) \). When the Markov chain converges to the target distribution, each state \( x \) will be visited with relative frequency \( \pi(x) \), as long as the walk is long enough. The critical question is now: how do we know if a transition distribution \( p(x' | x) \) converges to the target density \( \pi(x) \)?

In order for a Markov Chain to converge to the target density the transition distribution \( p(x' | x) \) needs to satisfy the following identity

\[ \pi(x') = \int p(x' | x) \pi(x) \, dx \]

This identity states that if \( x \sim \pi(x) \) was a sample from the target density, and we now generate a new sample \( x' \sim p(x' | x) \), then \( x' \) is also a sample from the target density. When this is the case, we say that the transition distribution \( p(x' | x) \) leaves the target density \( \pi(x) \) invariant. Informally this means that the walk doesn't change the distribution \( \pi(x) \).

One way to satisfy the invariance condition above is to define a transition distribution \( p(x' | x) \) satisfies **detailed balance**,\n
\[ p(x' | x) \pi(x) = p(x | x') \pi(x'). \]
In order to achieve the converged distribution \( \pi \) through MCMC, we have to define the correct walk \( p(x^t | x^{t-1}) \). However, finding such \( p(x^t | x^{t-1}) \) could be challenging. The \textbf{Metropolis-Hastings} strategy allows us to construct it through a proposal distribution \( q(x' | x) \) of our choice. At each step, we sample a proposal \( x' \sim q(x' | x^{t-1}) \). We then define the next sample \( x^t \) according to the rule:

\[
x^t = \begin{cases} 
  x' \sim q(x | x^{t-1}) & \text{with probability } \alpha(x' | x^{t-1}) \\
  x^{t-1} & \text{with probability } 1 - \alpha(x' | x^{t-1})
\end{cases}
\]

where the acceptance ratio \( \alpha(x' | x^t) \) is defined as

\[
\alpha(x' | x^t) = \min \left[ 1, \frac{\pi(x')q(x | x')}{\pi(x^t)q(x^t | x^{t-1})} \right]
\]

Together the proposal and acceptance ratio define the transition distribution

\[
p(X^t = x' | X^{t-1} = x) = \begin{cases} 
  \alpha(x' | x)q(x' | x), & x' \neq x \\
  q(x | x) + \int dx'(1 - \alpha(x' | x))q(x' | x), & x' = x
\end{cases}
\]

Note that when \( \alpha(x' | x) = 1 \) when the proposal distribution \( q(x' | x) \) satisfies detailed balance, since this implies that \( \pi(x')q(x | x') = \pi(x)q(x' | x) \). Intuitively we can think of the acceptance ratio as a means of correcting for the fact that our proposal may not quite satisfy detailed balance. When this is the case, the acceptance ratio ensures that

- When a transition \( x \rightarrow x' \) is underrepresented by the proposal, which is to say that
  \[
  \pi(x)q(x | x') < \pi(x')q(x | x'),
  \]
  then the acceptance ratio \( \alpha(x' | x) = 1 \). Intuitively this means that when a transition \( x \rightarrow x' \) is not proposed often enough, we always accept it.

- When a transition \( x \rightarrow x' \) is overrepresented by the proposal, which is to say that
  \[
  \pi(x)q(x | x') > \pi(x')q(x | x'),
  \]
  then we accept with some probability \( \alpha(x' | x) < 1 \), which corrects for the degree of overrepresentation.
3.1 Proof of Detailed Balance

The Metropolis-Hastings acceptance ratio ensures that detailed balance is maintained, which guarantees the convergence to the target density $\pi(x)$. To see this, we note that detailed balance is trivially satisfied for the case $x' = x$. For the case $x' \neq x$ we can write

$$\pi(x)p(x' | x) = \pi(x)\alpha(x' | x)q(x' | x),$$

$$= \pi(x)\min \left[ 1, \frac{\pi(x')q(x | x')}{\pi(x)q(x' | x)} \right]q(x' | x),$$

$$= \min \left[ \pi(x)q(x' | x), \pi(x')q(x | x') \right],$$

$$= \min \left[ \frac{\pi(x)q(x' | x)}{\pi(x')q(x | x')}, 1 \right]p(x' | x) = \pi(x')p(x | x').$$

3.2 Unnormalized Densities

A challenge in many inference problems is that we are not able to point-wise evaluate the target density $\pi(x)$ (i.e. the posterior $p(x | y)$ in the case of a Bayesian network). The core problem is that we can often evaluate an unnormalized density $\gamma(x)$ (i.e. the joint $p(y, x)$ in the case of Bayesian network), but we are not able to compute the normalizing constant $Z = \int \gamma(x)dx$ (i.e. the marginal likelihood $p(y)$), since this requires a high dimensional integral.

Metropolis-Hastings has the nice property that the acceptance ratio, or $\alpha$, can be computed without needing to normalize the density $\pi$. We can compute $\alpha$ just as well with the un-normalized density $\gamma$, since

$$\frac{\pi(x')}{\pi(x)} = \frac{\gamma(x')/Z}{\gamma(x)/Z} = \frac{\gamma(x')}{\gamma(x)}.$$ 

This means that the acceptance ratio can also be evaluated in terms of unnormalized densities

$$\alpha(x' | x) = \min \left[ 1, \frac{\pi(x')q(x | x')}{\pi(x)q(x' | x)} \right] = \min \left[ 1, \frac{\gamma(x')q(x | x')}{\gamma(x)q(x' | x)} \right].$$

3.3 Choosing a proposal

The MH strategy is guaranteed to converge but we don’t know about the rate of convergence. The choice of proposal distribution does have affect on both the quality of our sample and the rate of convergence. Let's pick a proposal of Normal distribution $q(x' | x) \sim N(x, \sigma^2)$. As we see, with a low $\sigma$, the acceptance rate is high, but we end up exploring locally, which leads to high correlation between samples. This is considered a bias in sampling, which suggests low quality in sampling. Increasing $\sigma$, we reduce this bias, but the acceptance rate will be low, which increases the sampling time and convergence rate. Tuning the proposal is the key to balance these trade-offs.

3.4 Independent Metropolis-Hastings

Suppose that our model of interest is a Bayesian network with with posterior $\pi(x) = p(x | y)$. In this case, a simple distribution to use as the proposal is the prior. For this choice of proposal the acceptance
ratio becomes
\[
\alpha(x | x') = \min \left[ 1, \frac{\pi(x')q(x' | x)}{\pi(x)q(x | x')} \right] = \min \left[ 1, \frac{p(y, x')p(x)}{p(y, x)p(x')} \right],
\]
\[
= \min \left[ 1, \frac{p(y | x')p(x')p(x)}{p(y | x)p(x)p(x')} \right] = \min \left[ 1, \frac{p(y | x')}{p(y | x)} \right].
\]

In other words, the acceptance ratio is simply the ratio of the likelihoods \(p(y | x')\) and \(p(y | x)\). We refer to this type of sampling strategy as an instance of \textit{independent Metropolis-Hastings}, since the the probability for a proposal \(x'\) does not depend on the probability of the current sample \(x\). In general we can perform independent Metropolis-Hastings with any proposal distribution \(q(x)\), in which case the acceptance ratio becomes
\[
\alpha(x' | x) = \min \left[ 1, \frac{\pi(x')q(x)}{\pi(x)q(x')} \right],
\]

Note that this acceptance ratio is basically a ratio of importance weights.

4 Gibbs Sampling

If Metropolis-Hastings is compared to stochastic gradient descent, then Gibbs Sampling borrows its idea from stochastic coordinate descent. Instead of transition from \(x'\) to \(x\), Gibbs Sampling defines transitions on a single variable at a time, keeping all others fixed. If \(x_i\) is the \(i\)th coordinate of \(x\), define \(x_{-i}\) as all without \(x_i\). A Gibbs sampler then loops over a series of updates
\[
x_i' \sim p(x_i | y, x_{-i}).
\]

The motivation of using Gibbs is when then the dimension of \(x\) is high. Sampling over 1 variable at a time could be more efficient. Let’s take the case of a 2-dimensional distribution as an example
\[
g(x) = p(y|x_1, x_2)p(x_1, x_2).
\]

The Gibbs updates are then defined as
\[
x_1' \sim p(x_1 | y, x_2),
\]
\[
x_2' \sim p(x_2 | y, x_1').
\]

The Metropolis-Hastings acceptance ratio for each of these steps is 1. To see this, let us write out the acceptance ratio for the update on \(x_1\)
\[
\alpha = \min \left[ 1, \frac{p(y, x_1', x_2)p(x_1' | y, x_2)}{p(y, x_1, x_2)p(x_1 | y, x_2)} \right],
\]
\[
= \min \left[ 1, \frac{p(y, x_1', x_2)p(y, x_1, x_2)p(y, x_2)}{p(y, x_1, x_2)p(y, x_2)p(y, x_1', x_2)} \right] = 1.
\]
Example: Gaussian Mixture Model

The key requirement in Gibbs sampling is that we are able to generate samples from the conditional distributions. This is often possible, but can require some algebraic manipulations. As an example of this, we will consider the Gaussian Mixture Model. We will where consider the case where we want to jointly sample \( p(z, \mu, \Sigma \mid y) \) using Gibbs updates

\[
\begin{align*}
  z_n \mid y, \mu, \Sigma &\sim p(z_n \mid y, \mu, \Sigma) \\
  \mu_k, \Sigma_k \mid y, z &\sim p(\mu_k, \Sigma_k \mid y, z) \quad k = 1, \ldots, K
\end{align*}
\]

Sampling \( z_n \) is relatively straightforward here. We can exploit the conditional independence in the Gaussian mixture model

\[
z_n \perp z_{-n} \mid \mu, \Sigma, y.
\]

This means that we can compute marginal distribution separately for each data point by explicitly normalizing the joint \( p(y_n, z_n \mid \mu, \Sigma) \) with respect to \( z_n \),

\[
p(z_n = k \mid y, \mu, \Sigma) = \frac{p(z_n = k, y, \mu, \Sigma)}{p(y, \mu, \Sigma)} = \frac{p(y \mid z_n = k, \mu, \Sigma)p(z = k)}{\sum_{l=1}^{K} p(y \mid z_n = l, \mu, \Sigma)p(z = l)}.
\]

In the case of the conditional distribution \( \mu \) and \( \Sigma \) we need to exploit the properties of conjugate priors. We will here consider the normal inverse-Wishart prior

\[
\begin{align*}
  \Sigma_k &\sim \text{InvWishart}(\nu_0, \Sigma_0), \\
  \mu_k &\sim \mathcal{N}(\mu_0, \Sigma_k / \lambda_0).
\end{align*}
\]

For this prior we can exploit the fact that \( p(\mu, \Sigma) \) is conjugate to \( p(y, z \mid \mu, \Sigma) \) in order to define the conditional posterior

\[
\begin{align*}
  \Sigma_k \mid y, z &\sim \text{InvWishart}(\nu_k^N, \Sigma_k^N) \\
  \mu_k \mid y, z &\sim \mathcal{N}(\mu_k^N, \Sigma_k / \lambda_k^N)
\end{align*}
\]

where,

\[
\begin{align*}
  \lambda_k^N &= \lambda_0 + N_k, \\
  \nu_k^N &= \nu_0 + N_k, \\
  \mu_k^N &= \frac{\lambda_0 \mu_0 + N_k \bar{y}}{\lambda_0 + N_k}, \\
  \Sigma_k^N &= \Sigma_0 + \frac{\lambda_0 N_k}{\lambda_0 + N_k} (\bar{y} - \mu_0)(\bar{y} - \mu_0)^T + \sum_{n=1}^{N} I[z_n = k](y_n - \bar{y}_k)(y_n - \bar{y}_k)^T.
\end{align*}
\]

**Metropolis within Gibbs**

In certain cases it may not be possible, or convenient, to derive closed form conditional distributions for each of the updates. In such cases, we can can replace Gibbs updates with Metropolis-Hastings updates for single coordinates. This type of sampling strategy is known as Metropolis within Gibbs.
As an example, suppose that we wish to avoid using a normal Inverse-Wishart prior in the Gaussian mixture model, and instead consider a Gaussian priors

\[ \mu_{k,d} \sim \text{Normal}(m, s^2), \]

\[ \sigma_{k,d} \sim \text{Gamma}(a, b), \]

where the covariance matrix is defined as the diagonal matrix

\[ \Sigma_k = \begin{bmatrix}
\sigma_{k,1}^2 \\
\sigma_{k,2}^2 \\
\vdots \\
\sigma_{k,d}^2
\end{bmatrix}. \]

we could then define Gaussian proposals

\[ q(\mu'_{k,d} | \mu_{k,d}) = \text{Normal}(\mu_{k,d}; \Delta^2_{\mu}) \]

\[ q(\sigma'_{k,d} | \sigma_{k,d}) = \text{Normal}(\sigma_{k,d}; \Delta^2_{\sigma}) \]

Since these proposal are each reversible, this leads to the updates

\[ \alpha(\mu'_{k,d} | \mu_{k,d}) = \min \left( 1, \frac{p(y, z^s, \mu'_{k,d}, \mu_{k,-d}, \sigma)}{p(y, z^s, \mu_{k,d}, \mu_{k,-d}, \sigma)} \right), \]

\[ \alpha(\sigma'_{k,d} | \sigma_{k,d}) = \min \left( 1, \frac{p(y, z^s, \mu, \sigma'_{k,d}, \sigma_{k,-d})}{p(y, z^s, \mu, \sigma_{k,d}, \sigma_{k,-d})} \right). \]