Estimation of Failure Probabilities Using Linked Importance Sampling

Radford M. Neal, University of Toronto

http://www.cs.utoronto.ca/∼radford/

Nonlinearity and Randomness in Complex Systems, SUNY Buffalo, 1 April 2006

The Problem of Estimating Failure Probabilities

How can we estimate the probability of failure for a system designed to make failures rare?

Examples:

- How often will a power grid become unstable and fail? Depends on unknown load patterns, generator failures, etc.

- How likely is radioactive waste to escape from a repository over the next 10000 years? Depends on unknown aspects of rock, weather, etc.

- What is the probability that a codeword sent through a channel will not be decoded correctly? Depends on the particular pattern of noise that occurs.

  I will look at the probability that a message block encoded using a Low Density Parity Check (LDPC) code will not be correctly decoded (by probability propagation) after transmission through an Additive White Gaussian Noise (AWGN) channel.

Modeling the Unknown Quantities

Whether failure occurs depends on unknown quantities. To quantify failure probabilities, we must have a probabilistic model of these unknowns.

Example: The AWGN channel model says that the noise corrupting each bit (represented by -1 or $+1$) is independent from bit to bit, and is Gaussian distributed with mean zero and some standard deviation, σ.

We can estimate the probability of failure by direct simulation from this model, but we need a huge sample to get a reasonable number of failures, if failures are very rare.

The real probability of failure might be dominated by departures of our model from reality — eg, maybe sometimes the channel noise does not have the distribution our model assumes. But I'll assume in this talk that our model is completely correct.

Direct Simulation

Denote the unknown quantities that determine failure by x. Our model assigns a probability density (or mass) $\pi_0(x)$ to each possible x.

I'll assume that it's feasible to compute $\pi_0(x)$ for any x, and to randomly sample independent values from the distribution π_0.

For each x we also can compute $f(x)$, defined to be 1 if failure occurs when x happens, and 0 if not. But computing this may be fairly costly.

We are interested in the failure probability, $\bar{f}$ — the expected value of f under π_0.

In a direct simulation, we draw $x^{(1)}, x^{(2)}, \ldots, x^{(N)}$ independently from π_0, and estimate the failure probability by

$$\hat{f}_{\text{direct}} \quad = \quad (1/N) \sum_{i=1}^{N} f(x^{(i)})$$

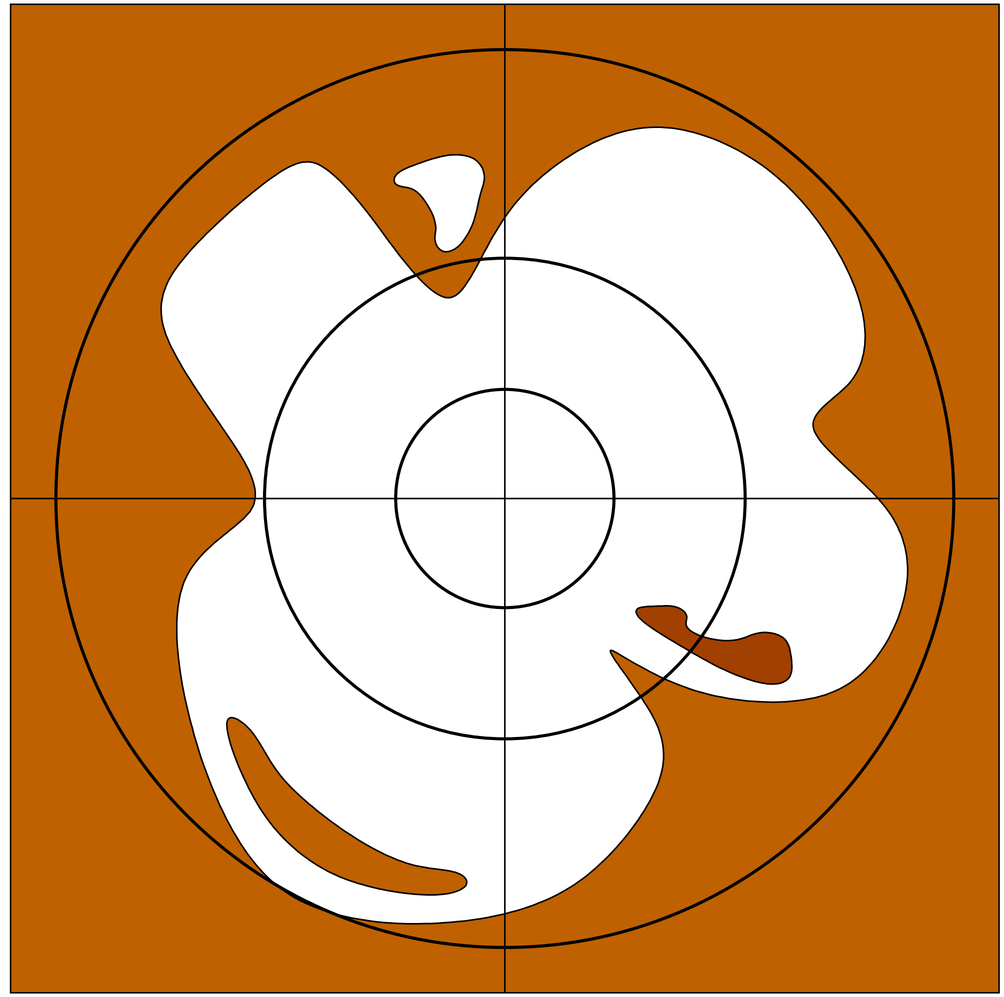
If failures are rare, the sum above will have a Poisson distribution, and one can derive that the standard deviation of $\hat{f}_{\text{direct}}$ is $(\bar{f}/N)^{1/2}$. The relative error is therefore $(\bar{f}N)^{-1/2}$. If $\bar{f}$ is very small, we will need a correspondingly large N.

Schematic Picture of Decoding Failure for Noise Patterns

Axes are amounts of noise for each bit of the codeword.

Circles are contours of the probability density for noise.

The brown region is where decoding fails.



Note that in some directions, increasing the magnitude of the noise can eliminate a failure. This can happen for LDPC codes decoded by probability propagation.

Failure Probabilities as Normalizing Constants

The distribution of x conditional on failure will have a probability density function $\pi_1(x)$ that can be written as

$$\pi_1(x) \quad = \quad \pi_0(x) f(x) \,/\, \bar{f}$$

So the problem of estimating the failure probability, $\bar{f}$, is equivalent to estimating the normalizing constant for the following unnormalized density function:

$$p_1(x) \quad = \quad \pi_0(x) f(x)$$

I'll also write this normalizing constant as $Z_1 = \int p_1(x)\, dx$.

This links the problem of estimating failure probabilities to two other extensively researched problems:

- Finding the *free energy* of a simulated physical system, which is related to the normalizing constant for the canonical distribution over states of the system, also called the *partition function*.

- Finding the *marginal likelihood* of a Bayesian statistical model - which is the normalizing constant for the posterior distribution of model parameters.

Estimating Ratios of Normalizing Constants

It's convenient to generalize the problem to that of estimating the *ratio* of normalizing constants for two distributions.

Let the unnormalized probability density functions for these distributions be $p_0(x)$ and $p_1(x)$, with normalizing constants $Z_0 = \int p_0(x)\,dx$ and $Z_1 = \int p_1(x)\,dx$.

We wish to estimate $r = Z_1/Z_0$.

For failure probability estimation, we can use the following:

$$p_0(x) = \pi_0(x) \text{ from our model,} \quad \text{so } Z_0 = 1$$

$$p_1(x) = \pi_0(x) f(x), \quad \text{so } Z_1 = \bar{f}$$

$$\text{Hence } r = Z_1/Z_0 = \bar{f}$$

Estimating Z_1/Z_0 by Simple Importance Sampling

If it is feasible to sample $x^{(1)}, \ldots, x^{(N)}$ independently from $\pi_0(x) = p_0(x)/Z_0$, we can estimate $r = Z_1/Z_0$ by

$$\hat{r}_{\text{SIS}} \quad = \quad (1/N) \sum_{i=1}^{N} \frac{p_1(x^{(i)})}{p_0(x^{(i)})}$$

This estimate is unbiased, as long as $p_0(x)$ is non-zero wherever $p_1(x)$ is non-zero, since

$$E(\hat{r}_{\text{SIS}}) \quad = \quad (1/N) \sum_{i=1}^{N} \int \frac{p_1(x^{(i)})}{p_0(x^{(i)})} \, \frac{p_0(x^{(i)})}{Z_0} \, dx^{(i)} \quad = \quad \frac{Z_1}{Z_0}$$

Unfortunately, the variance of this estimate will be very large if some regions have substantial probability under π_1 but near-zero probability under π_0.

This happens when estimating rare failure probabilities, with $\pi_0(x) = p_0(x)$ being our model probabilities, and $p_1(x) = \pi_0(x)f(x)$. Simple importance sampling then reduces to direct simulation, which works poorly if $\bar{f} = Z_1/Z_0$ is very small.

Can We Find a Better Importance Sampling Distribution?

Simple importance sampling might still work well if we could find a better distribution to sample from.

For the decoding example, suppose we do importance sampling from a noise distribution π_* in which the standard deviation of the Gaussian noise is σ_*, rather than the actual value of σ. If $\sigma_* > \sigma$, we'd (usually) expect more decoding failures.

The importance sampling estimate for the failure probability is then

$$(1/N) \sum_{i=1}^{N} \frac{\pi_0(x^{(i)}) f(x^{(i)})}{\pi_*(x^{(i)})}$$

where $x^{(1)}, \ldots, x^{(N)}$ are independently sampled from π_*.

Unfortunately, this helps only a little — if σ_* is big enough to generate lots of failures, the ratio $\pi_0(x^{(i)})/\pi_*(x^{(i)})$ is very variable, usually being nearly zero, but occasionally being huge. The estimate is still inaccurate unless N is very large.

Introducing Intermediate Distributions

Estimating $r = Z_1/Z_0$ is hard if the distributions π_0 and π_1 are "too far apart".

A general solution: Introduce extra distributions, with unnormalized density functions p_{η_j}, for $j = 1, \ldots, n-1$, where $0 < \eta_1 < \cdots < \eta_{n-1} < 1$. These interpolate between p_0 and p_1. Let $Z_\eta = \int p_\eta(x)\, dx$.

We can then write r as

$$r \;=\; \frac{Z_1}{Z_{\eta_{n-1}}} \, \frac{Z_{\eta_{n-1}}}{Z_{\eta_{n-2}}} \, \cdots \, \frac{Z_{\eta_2}}{Z_{\eta_1}} \, \frac{Z_{\eta_1}}{Z_0}$$

Now, if we can accurately estimate each factor, $Z_{\eta_{j+1}}/Z_{\eta_j}$, we can multiply them to get an accurate estimate of r. And we can hope to estimate them all accurately if we've chosen our sequence of distributions wisely.

Taking the Limit: Thermodynamic Integration

We can write the log of r as follows (letting $\eta_0 = 0$ and $\eta_1 = 1$):

$$\begin{aligned}\log r &= \log \frac{Z_1}{Z_{\eta_{n-1}}} + \log \frac{Z_{\eta_{n-1}}}{Z_{\eta_{n-2}}} + \cdots \log \frac{Z_{\eta_2}}{Z_{\eta_1}} + \log \frac{Z_{\eta_1}}{Z_0} \\ &= \sum_{j=0}^{n-1} \Big(\log Z_{\eta_{j+1}} - \log Z_{\eta_j} \Big)\end{aligned}$$

Taking a limit as $\eta_{j+1} - \eta_j \rightarrow 0$, we get

$$\begin{aligned}\log r &= \int_0^1 \frac{d}{d\eta} \Big(\log Z_\eta \Big) \, d\eta \;=\; \int_0^1 \frac{1}{Z_\eta} \frac{dZ_\eta}{d\eta} \, d\eta \\ &= \int_0^1 \frac{1}{Z_\eta} \int \frac{dp_\eta(x)}{d\eta} \, dx \; d\eta \;=\; \int_0^1 \int \frac{p_\eta(x)}{Z_\eta} \frac{d}{d\eta} \Big(\log p_\eta(x) \Big) \, dx \; d\eta\end{aligned}$$

The inner integral is the expectation of the derivative w.r.t. η of $\log p_\eta(x)$, over the distribution π_η. If we can sample from π_η, we can estimate this by Monte Carlo, and so estimate $\log r$ by numerical integration over η.

Nested Uniform Distributions

A simple example shows the usefulness of intermediate distributions:

Let U_η for any $\eta \in [0, 1]$ be a subset of x values.

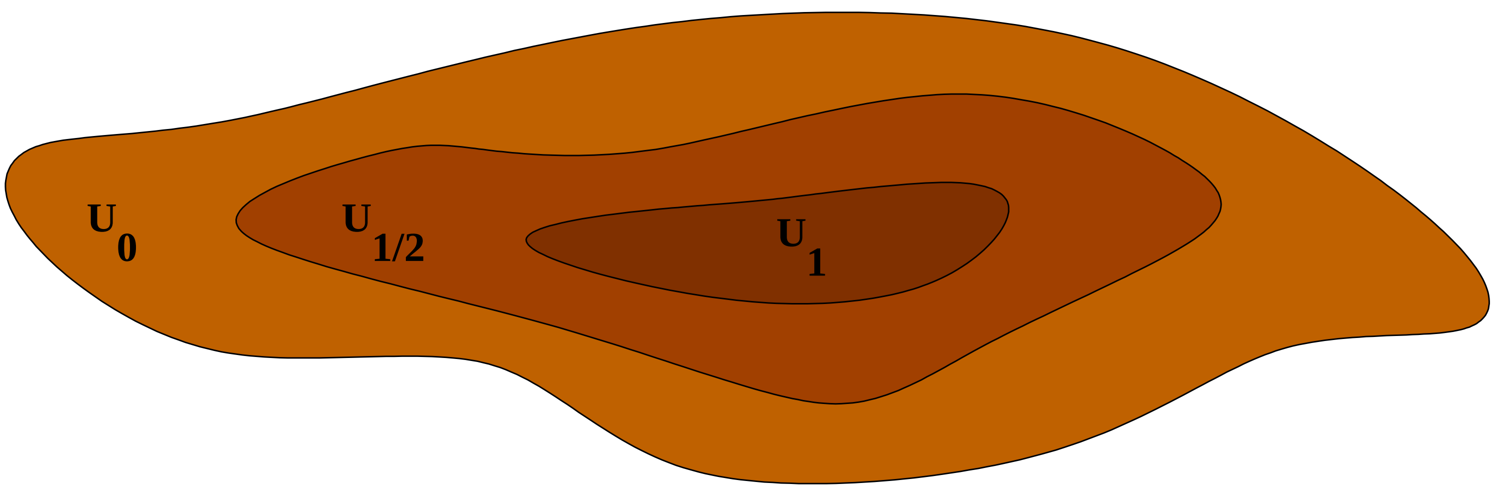
Suppose $U_\eta \subset U_{\eta'}$ whenever $\eta > \eta'$.

Define $p_\eta(x) = I_{U_\eta}(x)$ — the indicator function for U_η.

Then π_η will be the uniform distribution over U_η.

$Z_\eta = \int p_\eta(x)\, dx$ will be the volume of U_η.

Here's an example of nested U_0, $U_{1/2}$, and U_1:



Estimating Z_1/Z_0 for Nested Uniform Distributions

Suppose we can sample independent points from any π_η (ie, uniformly from U_η).

Suppose we wish to estimate $r = Z_1/Z_0$ — the ratio of volumes of U_1 and U_0. We could use simple importance sampling from π_0, but if r is tiny, we'd need a huge sample (more than $1/r$ points).

Introducing intermediate distributions with $0 < \eta_1 < \cdots < \eta_{n-1} < 1$, we estimate each $Z_{\eta_{j+1}}/Z_{\eta_j}$ by sampling uniformly from U_{η_j} and counting what fraction of the points fall in $U_{\eta_{j+1}}$. We can get an accurate estimate with a moderate number of points if the ratio of volumes isn't too small.

Result: The time required falls from order $1/r$ to order $\log(1/r)^2$.

Two Problems

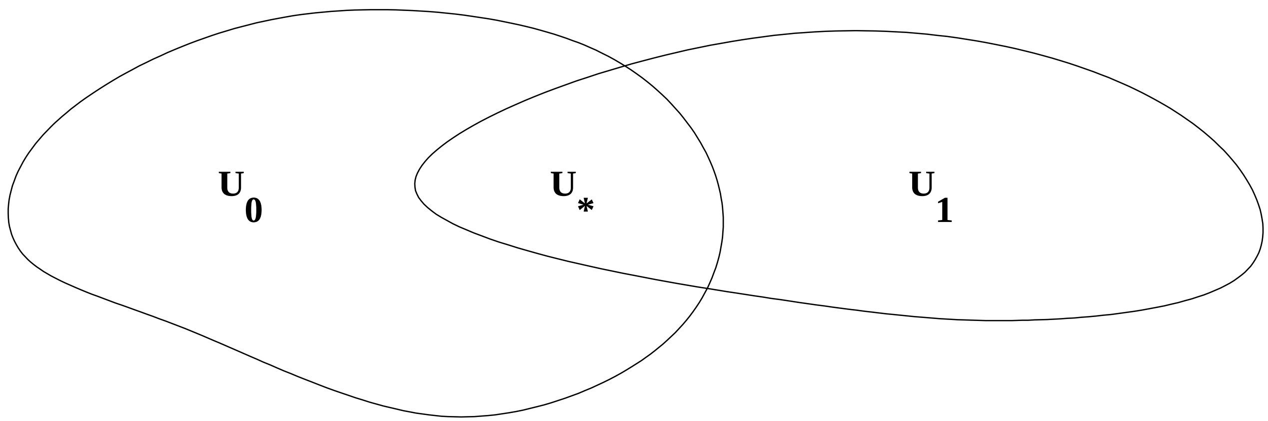
This seems like a great success, but two problems can arise, even for uniform distributions:

- Maybe the distributions aren't nested. If $U_{\eta_{j+1}}$ is not a subset of U_{η_j}, we can't hope to estimate $Z_{\eta_{j+1}}/Z_{\eta_j}$ by sampling just from π_{η_j}.
- Probably we can't easily sample independent points from any π_η. Sometimes we can do this for π_0 but not for the other distributions.

The same issues arise for non-uniform distributions, except that the meaning of "nested" is less clear.

Handling Non-Nested Distributions by Bridging

Consider two uniform distributions over regions U_0 and U_1 that overlap, but aren't nested:



Suppose we have a sample of N_0 points drawn uniformly from U_0. The fraction of these that lie in U_1 will obviously not give a good estimate of Z_1/Z_0. Instead it will be an estimate of Z_*/Z_0, where Z_* is the volume of $U_* = U_0 \cap U_1$.

But suppose we also have a sample of N_1 points drawn uniformly from U_1. The fraction of these that lie in U_0 will be an estimate of Z_*/Z_1.

Taking the ratio of these two estimates gives an estimate of Z_1/Z_0.

Suitable "bridging" distributions, π_*, can also be found for general distributions π_0 and π_1, so that we can more efficiently estimate Z_1/Z_0 as $(Z_*/Z_0)\,/\,(Z_*/Z_1)$.

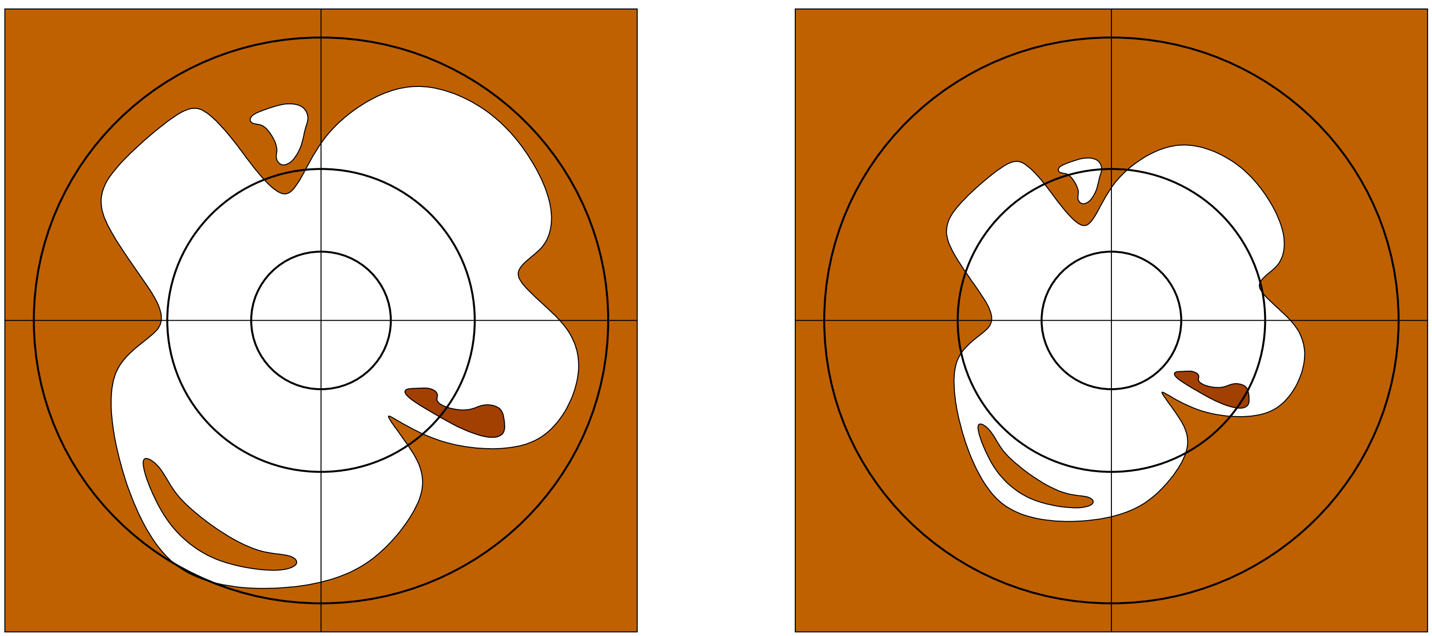
Distributions for Estimating Decoding Failure Probability

To help estimate the probability that noise results in an LDPC decoding failure, we can define a set of distributions $\pi_\eta(x) = p_\eta(x)/Z_\eta$ as follows:

$$p_\eta(x) \quad = \quad \pi_0(x) f(x/\eta)$$

where $\pi_0(x) = p_0(x)$ is the probability density of the noise vector, and $f(x)$ is the indicator of failure if the noise is x.

Here are schematic pictures of π_1 and $\pi_{3/4}$:



They may not be nested if decoding failure isn't monotonic in the noise magnitude.

Sampling by Simulating a Markov Chain

For the LDPC decoding problem, with low failure probability, there's no easy way to sample independently from π_η unless η is zero (or close to zero). This is typical of interesting problems.

Instead, we have to use Markov chain Monte Carlo (MCMC) methods. For each η, we define a Markov chain whose equilibrium distribution is π_η. Simulating this for some time, and discarding the early portion, provides dependent points sampled approximately from π_η.

Problems:

- How do we find such a Markov chain? Not hard, actually...
- Does the chain reach equilibrium in a reasonable amount of time? Sometimes, but often not.
- How can we tell that the chain has reached equilibrium? Even if it has, it's hard to tell for sure.

Metropolis-Hastings Updates

Here's a general method for constructing a Markov chain with equilibrium probability density $\pi_\eta(x) = p_\eta(x)/Z_\eta$, dating back over fifty years.

If the current state of the chain is x, move to the state x' as follows:

1) Propose a candidate state, x^*, according to some distribution with probability density $S_\eta(x^*|x)$.

2) Compute the acceptance ratio

$$a(x, x^*) \quad = \quad \frac{p_\eta(x^*)\, S_\eta(x|x^*)}{p_\eta(x)\, S_\eta(x^*|x)}$$

3) If $a(x, x^*) \geq 1$, set $x' = x^*$.
   If $a(x, x^*) < 1$, set $x' = x^*$ with probability $a(x, x^*)$, and otherwise set $x' = x$.

One can show that such updates leave π_η invariant, and hence the chain will eventually converge to π_η as its equilibrium distribution, *provided* it cannot be trapped in some subset of the state space.

Sampling for Noise Patterns That Cause Decoding Failure

To find the probability of decoding failure for LDPC codes with an AWGN channel, we need to sample from distributions of the form $\pi_\eta(x) = p_\eta(x)/Z_\eta$, where

$$p_\eta(x) \;\;=\;\; \pi_0(x) f(x/\eta)$$

Here $f(x)$ indicates decoding failure when the noise vector is x, and $\pi_0(x)$ is the Gaussian distribution for x where components are independent with mean zero and standard deviation σ.

I use a Metropolis-Hastings method in which the proposal, x^*, is found as follows:

$$x^* \;\;=\;\; (1-\omega_\eta^2)^{1/2}\, x \;+\; \omega_\eta\, \sigma\, n$$

where n is a vector of standard normal random variates.

One can show that for any $\omega_\eta \in [0,2]$, the density, $S_\eta(x^*|x)$, corresponding to this proposal satisfies $\pi_0(x)S_\eta(x^*|x) \;=\; \pi_0(x^*)S_\eta(x|x^*)$. It follows that the acceptance ratio for these proposals is

$$a(x,x^*) \;\;=\;\; \frac{p_\eta(x^*)\, S_\eta(x|x^*)}{p_\eta(x)\, S_\eta(x^*|x)} \;\;=\;\; \frac{\pi_0(x^*)\, f(x^*/\eta)\, S_\eta(x|x^*)}{\pi_0(x)\, f(x/\eta)\, S_\eta(x^*|x)} \;\;=\;\; f(x^*/\eta)$$

(Note that $f(x/\eta)$ should be one, if we're starting at a possible point.)

Sampling Noise Patterns (Continued)

Unless η is close to zero (lots of noise), we expect that only a small fraction of noise patterns lead to decoding failure.

Proposals that try to change x by a lot are therefore unlikely to be accepted. So we should use a value for ω_η that is not too far from zero, so that the candidate state, $x^* \;=\; (1-\omega_\eta^2)^{1/2}\,x \;+\; \omega_\eta\,\sigma\,n$, isn't too far from x.

We probably want ω_η to be smaller for larger η.

With this scheme, the Gaussian nature of the noise is accounted for entirely by the proposal distribution. The end result is much like sampling from a uniform distribution over $\{\,x\,:\,f(x/\eta) = 1\,\}$, except "uniform" has been adjusted to be "Gaussian".

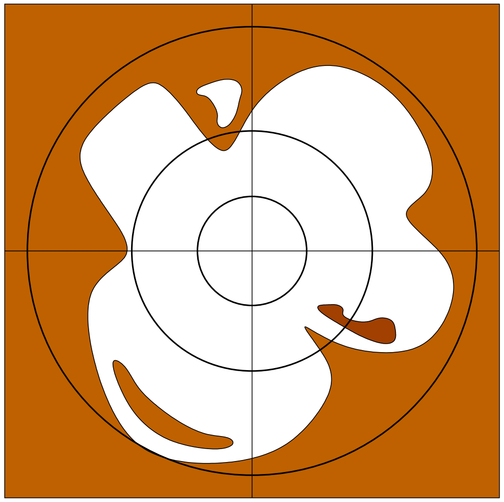
The Problem of Isolated Regions

Using a Markov chain to sample from each distribution, π_{η_j}, we could try to estimate all the ratios, $Z_{\eta_{j+1}}/Z_{\eta_j}$, then multiply them to get an estimate of Z_1/Z_0.

Unfortunately, this probably won't work.

The problem is that Metropolis-Hastings updates of the state for each π_{η_j} probably don't converge in any reasonable amount of time, because there are isolated regions of the space that are hard to move between.

Consider again the schematic picture of the region of noise space where there are decoding failures:



If the noise is likely to lie within the middle contour, the likely regions of decoding failure will be isolated with respect to small updates. And in high dimensions, a large update is very unlikely to land in a region of decoding failure.

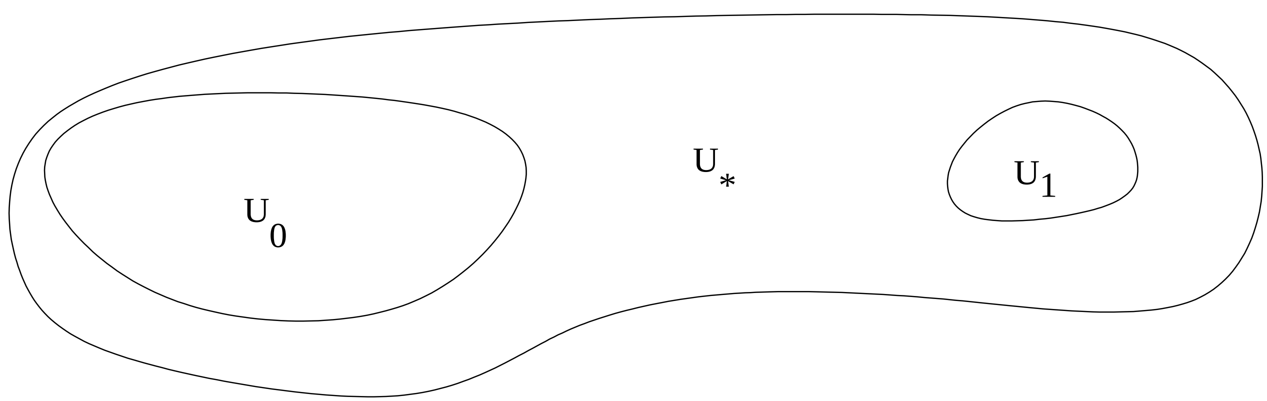
Umbrella Sampling Methods

One solution is sort of the opposite of bridging…

Define an "umbrella" distribution π_*, that overlaps both π_0 and π_1 — ie, all regions of high probability under either π_0 or π_1 have reasonably high probability under π_*.

If we can sample from π_*, we can then use the points we get to estimate both Z_0/Z_* and Z_1/Z_*. Dividing gets us an estimate of Z_1/Z_0.

Here's a picture, for distributions that are uniform over a region:



Practicalities of Umbrella Sampling

In practice, Markov chain sampling from an umbrella distribution, π_*, will work well only if it also gives reasonably high probability to a path of states connecting π_0 and π_1.

One approach is to define π_* in terms of a sequence of intermediate distributions.

We typically need to find suitable weighting factors relating to the values of Z_{η_j}, so that π_* gives roughly equal probabilities to regions of high probability under each π_{η_j}, even though Z_1/Z_0 may be tiny. Essentially, we need an approximation to the answer in order to get an accurate answer.

Variations on this theme go under the names "simulated tempering" and the "multicanonical method".

Parallel Tempering

Another approach is to sample from each π_{η_j} in parallel, using points from distributions with small η to help in sampling from distributions with large η (where movement is more difficult).

The state of the Markov chain used for this consists of $x_0, \ldots, x_n$. In each transition of the Markov chain we do the following for $j = 0, \ldots, n$:

1) Perform some update (such as a Metropolis-Hasting update) on x_j that leaves π_{η_j} invariant.

2) Pick some $j' \neq j$, and then attempt to swap x_j with $x_{j'}$. This is a Metropolis-Hastings proposal, with the acceptance ratio being

$$\frac{\pi_{\eta_j}(x_{j'})\,\pi_{\eta_{j'}}(x_j)}{\pi_{\eta_j}(x_j)\,\pi_{\eta_{j'}}(x_{j'})}$$

If this works well, we can use the results to estimate each $Z_{\eta_{j+1}}/Z_{\eta_j}$ using bridging, and multiply them to get an estimate of Z_1/Z_0.

Annealed Importance Sampling

In parallel tempering, information moves back and forth among the π_{η_j}. But maybe it's enough for it to move one way — from distributions with small η to those with bigger η. This is the heuristic behind "simulated annealing".

We start with a point, x_0, drawn from π_0, and then generate each x_{j+1} from x_j by applying some Markov chain transition that leaves $\pi_{\eta_{j+1}}$ invariant. This gets us a single point associated with each π_{η_j}.

From these points we can produce simple importance sampling estimates for each $Z_{\eta_{j+1}}/Z_{\eta_j}$. Multiplying them, we get an estimate for $r = Z_1/Z_0$:

$$\hat{r}_{\text{AIS}} \quad = \quad \prod_{j=0}^{n-1} \frac{p_{\eta_{j+1}}(x_j)}{p_{\eta_j}(x_j)}$$

We might not expect this estimate to be very good, since each factor is based on a single point, but remarkably, it is exactly unbiased, *even when the Markov chain updates have not reached equilibrium*. So we can average M such estimates to obtain a better estimate.

This result was first obtained by C. Jarzynski, and later independently by myself.

A Difficulty with Annealed Importance Sampling

Unfortunately, Annealed Importance Sampling won't work for the problem of estimating failure probabilities.

Each factor in $\hat{r}_{\text{AIS}}$, corresponding to some $Z_{\eta_{j+1}}/Z_{\eta_j}$, is a simple importance sampling estimate, and so will work well only if no region with substantial probability under π_{η_j} has zero (or very small) probability under $\pi_{\eta_{j+1}}$.

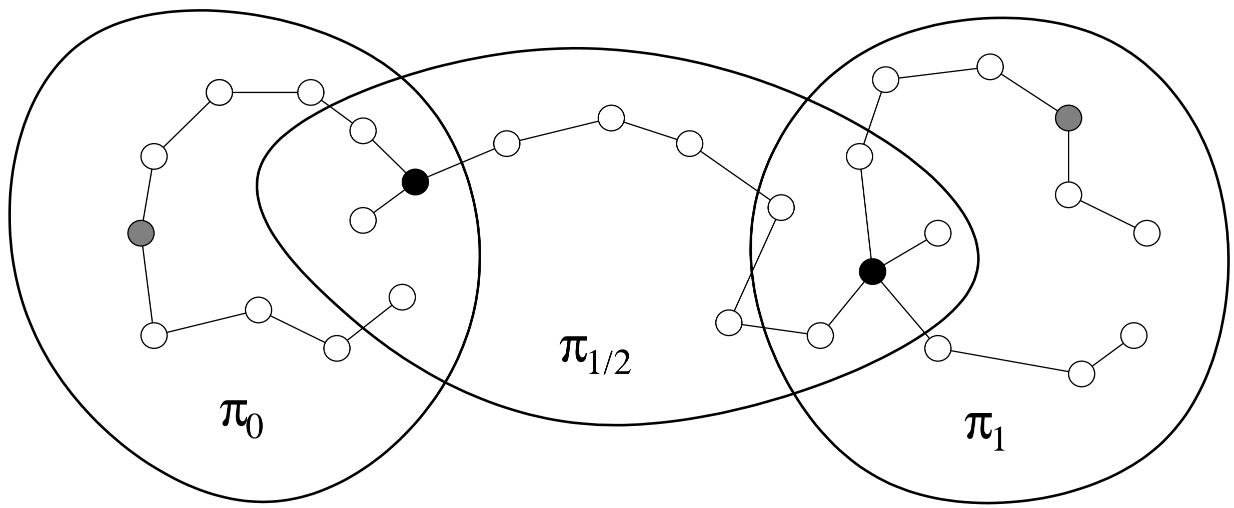
This is often not a problem, since with sufficiently many intermediate distributions, $\pi_{\eta_{j+1}}$ will be similar to π_{η_j}. But for distributions with sharp boundaries — eg, dividing decoding failure and decoding success — introducing intermediate distributions doesn't help.

We've seen how bridging can solve such problems. Can it be used here, while still preserving the attractive property that Annealed Importance Sampling is exactly unbiased?

Linked Importance Sampling

I recently devised a scheme that does this, called Linked Importance Sampling.

It's illustrated below for three uniform distributions, π_0, $\pi_{1/2}$, and π_1:



The procedure starts by drawing the gray point on the left from π_0. A Markov chain leaving π_0 invariant is then simulated (both forward and backward) to produce a sequence of states. One of them — the black "link" state — is selected from a bridging distribution (here the intersection of the regions for π_0 and π_1). The procedure continues in this way from this state, until we reach π_1.

We then use all these states to estimate Z_1/Z_0, using bridging for each factor of $Z_{\eta_{j+1}}/Z_{\eta_j}$. The result is exactly unbiased, so we can average over many such runs.

Experiments on LDPC Codes

I've tried using Linked Importance Sampling to estimate the probability of decoding failure of Low Density Parity Check codes for an AWGN channel.

Codewords were 2000 bits long, derived from source blocks of 1000 bits (ie, the rate was 1/2). The 1000×2000 sparse parity check matrix was randomly constructed with 2 1's in 20% of the columns, 3 1's in 70% of the columns, and 7 1's in 10% of the columns. The number of 1's in each row was kept as even as possible. Cycles of length 4 were eliminated by moving bits within columns.

This procedure produces a reasonably good code, but it isn't especially optimized.

Decoding by Probability Propagation

Decoding was done by probability propagation (also known as loopy belief propagation and the sum-product algorithm). The decoder uses an AWGN channel model, and knows the correct noise standard deviation.

Up to 25 iterations of probability propagation were done. Decoding was stopped after fewer iterations if thresholding the probabilities at that point produced a valid codeword.

A failure occurs when either a valid codeword is not found within 25 iterations, or the codeword found is not the one that was transmitted.

Note: The noise level assumed by the decoder was fixed for all distributions used during a Linked Importance Sampling run. It was *not* changed with η.

Details of Runs with Noise Level of $\sigma = 0.6$

Setting the noise standard deviation to $\sigma = 0.6$, I simulated transmission and decoding of 10,000,000 codewords. Two of these failed to decode correctly. This gives an estimate for the failure rate of 2×10^{-7}, but it's not very accurate.

I used Linked Importance Sampling with $n = 5$ to see whether a good estimate could be obtained with fewer simulated transmissions.

The values of η used for these distributions were as follows:

$$\eta_0 = 0, \quad \eta_1 = 1/0.81, \quad \eta_2 = 1/0.775, \quad \eta_3 = 1/0.72, \eta_4 = 1/0.66, \quad \eta_5 = 1/0.6$$

The values of ω used for Metropolis-Hastings updates (6 for each transition) were

$$\omega_1 = 0.56, \quad \omega_2 = 0.45, \quad \omega_3 = 0.37, \quad \omega_4 = 0.29, \quad \omega_5 = 0.22$$

The sample sizes at each stage were given by

$$K_0 = 700, \quad K_1 = 700, \quad K_2 = 700, \quad K_3 = 700, \quad K_4 = 700, \quad K_5 = 200$$

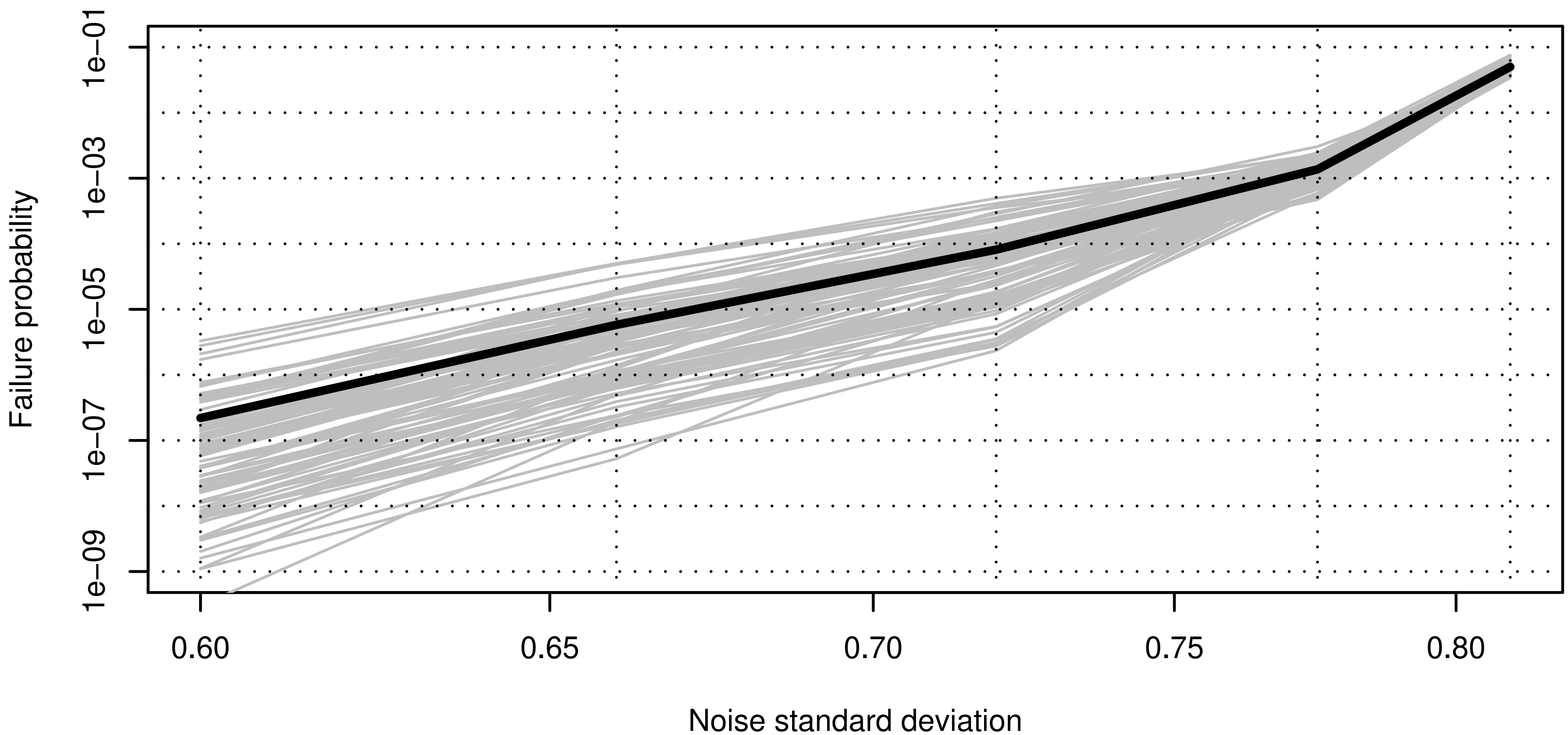
$M = 100$ runs were done with these parameters, and the results used to obtain an estimate of $r = Z_1/Z_0 = \bar{f}$ and its standard error.

Results of Runs with Noise Level of $\sigma = 0.6$

A total of 2,380,500 codewords were transmitted and decoded in the 100 Linked Importance Sampling runs.

The estimate of the failure rate was $2.2 \times 10^{-7} \pm 0.5 \times 10^{-7}$.

Here is a plot of the single-run estimates for Z_η / Z_0 for each of the 100 runs, along with their average:



Details of Runs with Noise Level of $\sigma = 0.48$

Setting the noise standard deviation to $\sigma = 0.48$ reduces the failure rate to a value one could not hope to estimate by direct simulation.

I used Linked Importance Sampling with $n = 7$ to see whether a good estimate could be obtained.

The values of η used for these distributions were as follows:

$$\eta_0 = 0, \quad \eta_1 = 1/0.80, \quad \eta_2 = 1/0.77, \quad \eta_3 = 1/0.71,$$
$$\eta_4 = 1/0.65, \quad \eta_5 = 1/0.59, \quad \eta_6 = 0.535, \quad \eta_7 = 0.48$$

The values of ω used for Metropolis-Hastings updates (6 for each transition) were

$$\omega_1 = 0.62, \quad \omega_2 = 0.45, \quad \omega_3 = 0.37, \quad \omega_4 = 0.29, \quad \omega_5 = 0.22, \quad \omega_6 = 0.18, \quad \omega_7 = 0.15$$

The sample sizes at each stage were given by

$$K_0 = 700, \quad K_1 = 700, \quad K_2 = 700, \quad K_3 = 700,$$
$$K_4 = 700, \quad K_5 = 700, \quad K_6 = 700, \quad K_7 = 200$$

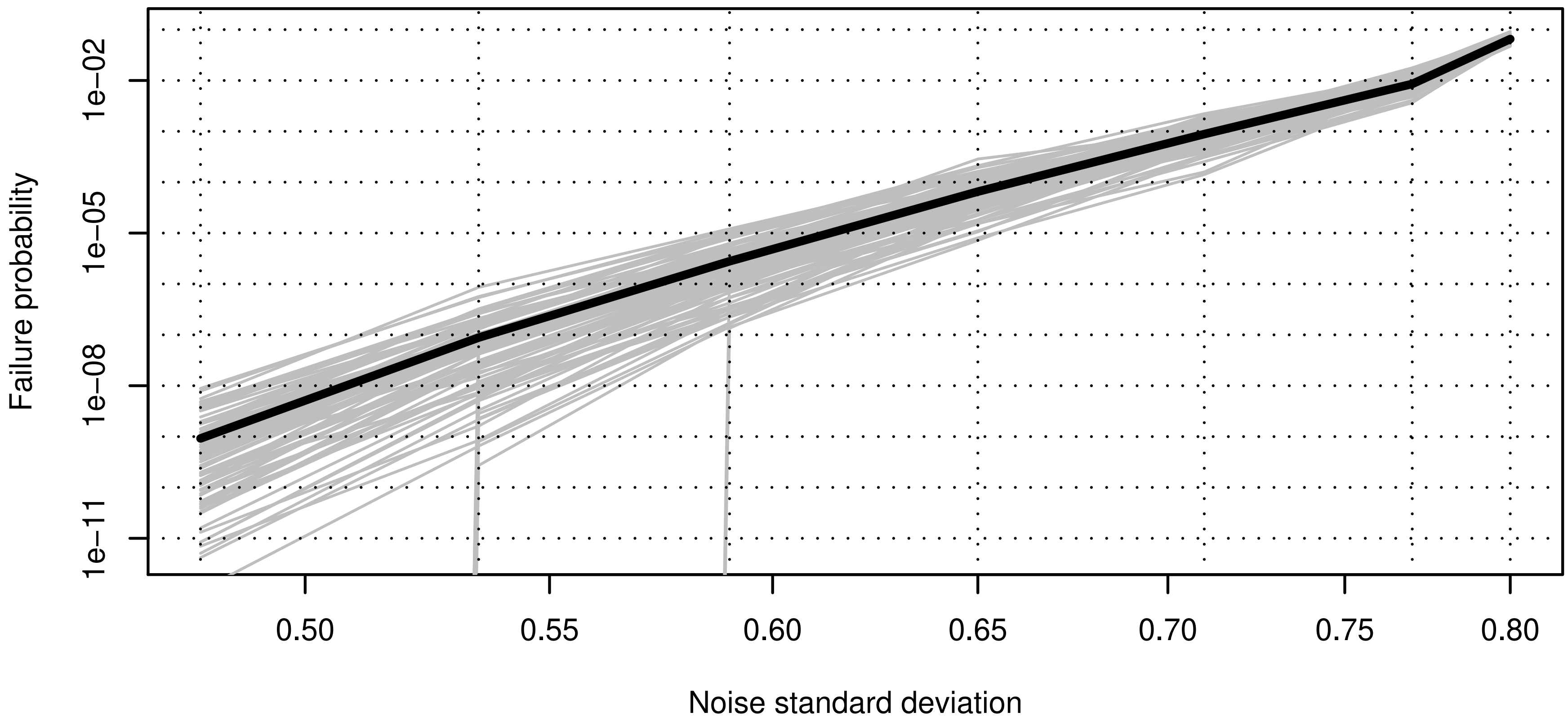
$M = 100$ runs were done with these parameters, and the results used to obtain an estimate of $r = Z_1/Z_0 = \bar{f}$ and its standard error.

Results of Runs with Noise Level of $\sigma = 0.48$

A total of 3,500,700 codewords were transmitted and decoded in the 100 Linked Importance Sampling runs.

The estimate of the failure rate was $9.2 \times 10^{-10} \pm 1.6 \times 10^{-10}$.

Here is a plot of the single-run estimates for Z_η/Z_0 for each of the 100 runs, along with their average:



Discussion

- The results aren't quite as good as they seem, because the number of decoding iterations is smaller when there is no failure.

  For $\sigma = 0.6$, the 10,000,000 codewords sent in the direct simulation required 33,000,000 iterations of probability propagation. The 2,380,500 codewords sent during the Linked Importance Sampling runs required 36,000,000 iterations of probability propagation.

- Some fiddling was needed to find suitable values for η_j, ω_j, etc.

- Detailed examination of the results shows that the π_η distributions are not nested — 10% or more of the states produced by the Markov chain for $\pi_{\eta_{j+1}}$ have zero probability under π_{η_j}.

- It is necessary for correctness that Metropolis-Hastings updates starting at a point from π_1 have a high probability of reaching a point with non-zero probability under $\pi_{\eta_{n-1}}$, and so forth back through the distributions.

- It's conceivable that some regions of decoding failure are not accessible, given the Metropolis-Hastings updates used, and the sequence of π_η.

In view of the last two points, it's hard to be completely sure of the results, but the same may be true of any other method.

Future Research

- A different sequence of intermediate distributions could be defined by varying the number of iterations allowed for probability propagation. These distributions would necessarily be nested.
- With nested distributions, one could try a "sequential Monte Carlo" method.
- The variance of Linked Importance Sampling estimates could be improved by letting the number of Metropolis-Hastings updates increase as needed to ensure that an adequate sample is obtained. It's tricky, but possible, to do this while still preserving the proof of correctness.
- It would be interesting to compare with parallel tempering, which is perhaps the most promising of the other methods.

References

J. S. Liu (2001) *Monte Carlo Strategies in Scientific Computing*, Springer-Verlag.

D. J. C. MacKay (2003) *Information Theory, Inference, and Learning Algorithms.* `http://wol.ra.phy.cam.ac.uk/mackay/itila/book.html`

R. M. Neal (1993) *Probabilistic Inference Using Markov Chain Monte Carlo Methods.* `http://www.cs.utoronto.ca/~radford/review.abstract.html`

Bridge Sampling:

Bennett, C. H. (1976) Efficient estimation of free energy differences from Monte Carlo data, *Journal of Computational Physics*, vol. 22, pp. 245-268.

A. Gelman, X.-L. Meng (1998) Simulating normalizing constants: From importance sampling to bridge sampling to path sampling, *Statistical Science*, **13**, 163-185.

Annealed Importance Sampling:

Jarzynski, C. (1997) Nonequilibrium equality for free energy differences, *Physical Review Letters*, vol. 78, pp. 2690-2693.

R. M. Neal (2001) Annealed Importance Sampling, *Statistics and Computing*, **11**, 125-139.

Linked Importance Sampling:

R. M. Neal (2005) Estimating ratios of normalizing constants using Linked Importance Sampling, Technical Report No. 0511, Dept. of Statistics, University of Toronto.

References (Continued)

Umbrella/Tempering/Multicanonical Methods:

Berg, B. A. and Celik, T. (1992) New approach to spin-glass simulations, *Physical Review Letters*, vol. 69, pp. 2292-2295.

Marinari, E. and Parisi, G. (1992) "Simulated tempering: A new Monte Carlo Scheme", *Europhysics Letters*, vol. 19, pp. 451-458.

Torrie, G. M. and Valleau, J. P. (1977) "Nonphysical sampling distributions in Monte Carlo free-energy estimation: Umbrella sampling“, *Journal of Computational Physics*, vol. 23, pp. 187-199.

Parallel Tempering:

Geyer, C. J. (1991) "Markov chain Monte Carlo maximum likelihood", in E. M. Keramidas (editor), *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*, pp. 156-163, Interface Foundation.