# Diffusion Math

## ELBO

### Official Formulation

Note: is a distribution determined by the encoding process. It is a Markov process.

is a distribution determined by the decoding process, which has learnable parameters.

The distribution is conditioned on the parameters, but this is not shown.

It is also a Markov process.

**Main Expression**

**Focusing on the Numerator in the log**

Now

is conditionally independent from :

Since is parameterized such that is conditionally independent from :

We can keep going here:

**One way to put this:**

The integrals iterate over all possible values of , almost like a for loop. They are then plugged in for the expression for the joint distribution to get a probability value, which accumulates across the loop. maps a tuple of values to a density value based on the joint distribution.

However, given this same tuple, , we can evaluate the density by evaluating the probability density of the ‘path’ that goes from to , which is this expression:

Which is the probability that the decoder takes that ‘path’ from to .

**Focusing on the denominator in the log**

Now consider

Since the forward process is a Markov Chain:

Once again, since the forward process is a Markov chain:

Continuing:

In other words, one way to evaluate the joint distribution

Is to start with x, and evaluate the probability of the ‘chain of events’ that leads to .

**Now**

The first step seems like a hack – since conditioned on is independent from x, we can add in the extra condition on without worrying.

Thus,

Things cancel from the second term:

**Focusing on the log part**

Putting this together, we have

Assume the last term goes to zero, since the distribution after all the forward diffusion steps should be very similar to . Why is the distribution for Well, we can choose it to be that way, by making the ‘decoder’ evaluate it as such. Or, we can think of as attempting to fit the ‘true’ distribution of the data, which is done best in this case by being

Let us put this in the integral:

**Back to Main Expression**

We can marginalize out a lot of stuff. For instance,

**Single Term**

The stuff in the brackets goes to 1, since all conditional distributions are still distributions, they integrate to 1.

**Back to Main Expression**

Marginalizing the first term (won’t show the whole thing this time)

According to our parametrization, we have

We can actually compute the KL divergence here in closed form:

Which is just proportional to the squared difference between the means.

**Reconstruction Term:**

**Back to Main Expression**

Maximizing this means minimizing this:

Note that depends on and its parameters, although this is not shown.

We can imagine minimizing this term regarding a specific x by sampling from computing the expression, and changing the parameters of . We can see this more clearly by adding stuff to the expectations.

Now using a Monte Carlo estimate:

The loss function minimizes the difference between the estimated of , and the most likely value (mean) it took, given and .

### Mason’s Formulation

**Another way to get some training objective - Mason Stuff:**

Below, we show the objective you can get by sampling and from . That is, sample , then add noise to get . The objective ends up predicting the noise added from to .

However, it turns out that given , , we can directly compute the distribution over . So only one noise-generation step is necessary.

**Focusing on one term in the sum:**

We can marginalize out a lot of stuff. For instance,

For the purposes of optimization, the bottom shouldn’t matter (I think).

**Back to Main Expression**

**Analyzing a term in this expression:**

This is the expectation over of .

is the variance of this decoder step, which we should match to the variance of .

So each term is

Note that we cannot sample and independently, since these are sampling from the two marginal distributions, which is not accurately simulating the joint distribution. and are not truly ‘paired’.

Imagine we wanted to find the probability that it is a hot day and John has a hot dog. You can’t have your random weather generator generate a day, and then your random John simulator simulate eating a hot dog, and then randomly pair the results to do Mone Carlo sampling.

**Back to Main Expression**

Which is basically a sum of weighted MSE, where we simulate one noising step, and get the network to estimate the denoising step.