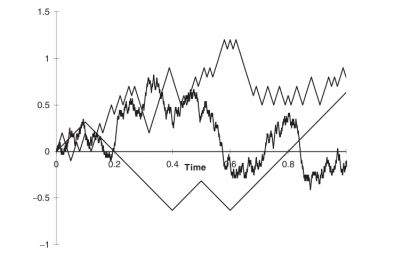
**Path Integrals**

Say we have a process (· means Ito):



(if X0 ≠ 0, then have to do some contortions – not sure why – have to check out that file) The w(t) = dW(t)/dt variables takes on many different paths, as illustrated below,



And consequently X(t) will take many different paths of course. And we’d like to calculate two things. First I wanna look at the probability (density) of taking a particular path P(Xpath(t)). The path integral apparatus would provide a natural framework within which to consider this question. And then after that, I’d like to go back and apply this to P(X(t)).

**Probability of Particular Path from X(0) = 0 to X(t) = X: P(ξ(t))**

And we’d like to consider the probability density for a particular path. We can write this in terms of a path integral. First consider the probability of going from X0 to X in time t *along a particular path* X(t) = [ξ1, ξ 2, …, ξN = X] in discreet space. The process equations and (conditional) probabilities are, discreetly:





And so, the probability of the whole path is just the product of probabilities….



Filling in the expression for the white noise probability distribution, and also a Fourier representation of the delta function, we have:



We can do the ΔW integrals…



Letting ξ(t) denote our path, we have something like.



(don’t confuse differential D with diffusion D) Now we could do the k(t) integral technically. We would find (might check out Path Integrals folder for more on this, and the prefactor):



Moving on,

**Probability of P(X(t))**

Anyway, we want the absolute probability of going from x0 to x in time t, not the probability of doing so along a particular path. And this is given by summing over all paths going from x0 to x:



If we do the k(t) integral, as above, then we’d be left with [might wonder where the prefactor P(0) that we had in the last section went, but I don’t think it shows up or gets absorbed into the measure, whatever; the same thing happens in QM/Time Dependent/Path Integrals file when integrating out the momentum coordinate to get a path integral purely in terms of X(t)] :



We can do a saddle point approximation on this, and adapting our conclusions from the QM Path Integral files, I think we’d find:



(S depends on x0 and x because path is constrainted to go from former to latter from t = 0 to t = t) And then the general rules for the Feynman diagrams would be to first, McLaurin expand Δa(ξ,t) and b(ξ,t) in powers of ξ I suppose, to turn them into polynomials. And then every ξm·kn term would be represented as a vertex with m - ξ legs, and n - k legs. But I think we need to express these terms as m+n integrals, and so would need to introduce some delta functions. Seems like we’d probably want to switch to the Fourier basis. Perhaps we could use MFA, etc., techniques to work out the result.

**Example**

What is the path-integral representation of P[**W**(t)]?



and I guess we could write this as:



**Example**

Let’s consider the following process:



What’s the path probability expression?



And what is P[X(t)]? Well,



Can we evaluate this integral? Should be able to. Saddle point approximation should be exact.



Let’s find the saddle point.



Boundary conditions are that it equals x0 (say) at s = 0, and x at s = t.



So now we have:



Filling this in to evaluate the saddle point action,



Final steps,



And this is indeed the correct answer.

**Example**

Let’s consider the following process:



What’s the path probability expression?



And what is P[X(t)]? Well,



Can we evaluate this integral? Should be able to. Ssaddle point approximation should be exact, again.



Let’s find the saddle point. We’ll just use the Euler-Lagrange Equation:



Solution is:



Boundary conditions are that it equals x0 (say) at s = 0, and x at s = t. The first requires A = x0. The second requires,



So the solution is:



Filling this in to evaluate the saddle point action,



Eeeew.

**Example**

Let’s consider the following process:



What’s the path probability expression?



And what is P[X(t)]?



What would the saddle point approximation say?



Let’s find the saddle point.

