10/21/09

Wednesday 880.05

Handouts:

- Problem set #9.
- Printout of MATLAB codes for stochastic calculations.

Guide to PS#9

- Follow ups to some practical MATLAB issues – random numbers, algorithms for exponentiation.
  - introduce some more MATLAB features we will use, like histograms and displaying (printing to screen) variables.
  - address some questions like seeding of random numbers and how to create a normal distribution with any mean and width.
  - point of emphasis: "It matters how you do it!" In this case, how you calculate e^{A}. Issues are speed, accuracy, scaling, failure for some types of matrix.
- SVM demo program → mostly for cultural enrichment.
- Playing with the code in todays other handout → Metropolis example. Prototype we'll build on for stochastic calculations.

MATLAB stuff for next Tuesday

* • Discrete and continuum calculations of propagators (problems 4 and 5) step by step.

Today:

1st half
- Follow-up and recap to items from Monday.
- Extension to functional integral over fields.

2nd half
- stochastic evaluation – Metropolis et al.

on board

Warm-up functional derivatives $\frac{\delta}{\delta j(x)}\frac{\delta}{\delta j(y)} e^{-\int [a\varphi(x)^2 + j(x)\varphi(x)]dx} = \varphi(y)\varphi(x)e^{-\int \ldots}$

$$\frac{\delta}{\delta f(x)}\int g(y)(f(y))^4 h(y)\,dy = \lim_{\Delta x \to 0}\frac{1}{\Delta x_i}\frac{\partial}{\partial f_i}\sum_j g_j f_j^4 h_j \Delta x_j$$
$$= 4\,g(x)(f(x))^3 h(x)$$

put this on left board

112

10/21/09

Feynman rules for $N=1$ system, $H(p,x;\tau) = \frac{p^2}{2m} + V(x) - x f(\tau)$

Continuum version

$$Z[f] = e^{-\int_0^\beta d\tau \frac{\lambda}{4}\left[\frac{\delta}{\delta f(\tau)}\right]^4} Z_0 \, e^{\frac{1}{2}\int_0^\beta d\tau \int_0^\beta d\tau' f(\tau) A^{-1}(\tau,\tau') f(\tau')}$$

$$= \int \mathcal{D}x(\tau)\, e^{-\int_0^\beta d\tau \left[\frac{m}{2}\left(\frac{dx}{d\tau}\right)^2 + \frac{a}{2}x^2 - x f(\tau)\right]}$$

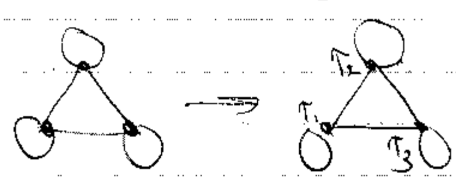
calculate discrete and continuum versions in PS #2

$x(\beta) = x(0)$ ← boundary condition on trajectories

Rules for $\log Z[f]/Z_0$ at order λ^n

1. draw all connected diagrams with n "vertices" •
   (can you predict how many lines? $\Rightarrow (4\times n)/2 = 2n$ ← two ends to lines)
   # lines out ← will be important
2. label each vertex with a different τ variable
3. assign $(-\frac{\lambda}{4}\cdot 4!)$ for each vertex
4. assign $A^{-1}(\tau,\tau')$ for every line connecting τ and τ' vertices.
5. integrate each τ variable from 0 to $\beta\hbar$
6. apply a symmetry factor as in the model partition function

Try

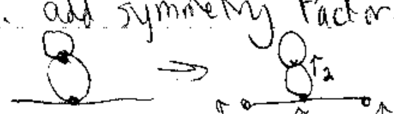


$$(-6\lambda)^3 \left[\left(\tfrac{1}{2}\right)^3 \cdot 1 \cdot \tfrac{1}{3!}\right] \int_0^{\beta\hbar} d\tau_1 d\tau_2 d\tau_3 \, A^{-1}(\tau_2,\tau_2) A^{-1}(\tau_1,\tau_1) A^{-1}(\tau_3,\tau_3) A^{-1}(\tau_1,\tau_2) A^{-1}(\tau_2,\tau_3) A^{-1}(\tau_3,\tau_1)$$

symmetry factor

Rules for $\langle x(\tau_a)\, x(\tau_b)\rangle$ at order λ^n

1. draw all connected diagrams with two external points and n vertices
2. label each vertex with a τ variable and the external points with τ_a, τ_b
3. $(-\frac{\lambda}{4})\cdot 4!$ for each vertex
4. $A^{-1}(\tau,\tau')$ for lines connecting τ and τ' vertices.
5. integrate each internal τ variable from 0 to $\beta\hbar$
6. add symmetry factor

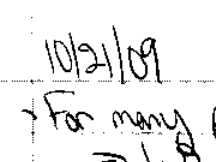


$$(-6\lambda)^2\left(\tfrac{1}{2}\cdot\tfrac{1}{2}\cdot 1\right)\int_0^\beta d\tau_1 d\tau_2 \, A^{-1}(\tau_a,\tau_1) A^{-1}(\tau_1,\tau_b) A^{-1}(\tau_1,\tau_2) A^{-1}(\tau_2,\tau_1) A^{-1}(\tau_2,\tau_2)$$

10/21/09

- For many particles, either symmetrized basis or number basis.
  - $\Rightarrow$ both are in use in "real-life" physics calculations.

- Quick recap of symmetrized states.



Suppose two level and 3 particles (bosons)

Hilbert space is $|a \text{ or } b\rangle \otimes |a \text{ or } b\rangle \otimes |a \text{ or } b\rangle$

particle: 1 2 3

What are the symmetrized possibilities for wave functions?

$|aaa\rangle$,

$|bbb\rangle$

if normalized

$\frac{1}{\sqrt{3}}\left(|aab\rangle + |aba\rangle + |baa\rangle\right)$

$\frac{1}{\sqrt{3}}\left(|abb\rangle + |bab\rangle + |bba\rangle\right)$

eigenstates of $S = \frac{1}{N!}\sum_P P = \frac{1}{6}(P_{123} + P_{132} + \ldots)$

mixing up by any permutation of the states of the 3 particles.

- We designate the coordinate representation states as:

$$|x^{(1)}\, x^{(2)} \cdots x^{(N)}\} = \frac{1}{\sqrt{N!}}\sum_P \xi^P |x^{(P1)}\rangle \ldots |x^{(PN)}\rangle$$

which is a complete set, so when evaluating $\mathrm{tr}\, e^{-\beta \hat{H}}$ we can do the usual splitting into $e^{\epsilon \hat{H}} = :e^{\epsilon \hat{H}}: + O(\epsilon^2)$

So what do we get? $\Rightarrow$ jump to 91ab to show

not N_r, but N = # of particles

$$\{x^{(1)} \cdots x^{(N)} | :e^{-\epsilon \hat{H}}: | y^{(1)} \cdots y^{(N)}\} = \left(\frac{m}{2\pi\epsilon}\right)^{3N/2} \mathrm{Det}\, M\, e^{-\frac{m}{2\epsilon}\sum_i^N (x^{(i)} - y^{(i)})^2 - \frac{\epsilon}{2}\sum_{i \neq j}^N V(y^{(i)} - y^{(j)})}$$

10/21/09

Alternative to inserting $|x^{(1)} \cdots x^{(N)}\}$ states is to insert coherent states built on $|n_1 n_2 \cdots n_\infty\rangle$ states

$$\uparrow a \quad \uparrow b \quad \uparrow c$$

$\Rightarrow$ complete and orthonormal. Use $a_k, a_k^\dagger$ to change numbers.

$$|n_1 \cdots n_\infty\rangle \equiv (a_1^\dagger)^{n_1} \cdots (a_\infty^\dagger)^{n_\infty}|0\rangle$$

$$a_k^\dagger a_k |n_1 \cdots n_\infty\rangle = n_k |n_1 \cdots n_\infty\rangle$$

0, 1 fermions
0, 1, 2, ... bosons

$$\hat{H} = \sum_{\alpha\beta} a_\alpha^\dagger \langle\alpha|T|\beta\rangle a_\beta + \frac{1}{2}\sum_{\alpha\beta\gamma\delta} a_\alpha^\dagger a_\beta^\dagger \langle\alpha\beta|V|\gamma\delta\rangle a_\delta a_\gamma + \ldots$$

· Now $|\alpha\rangle$ states unspecified here. Could be hydrogen wf's, harmonic oscillators, $\vec{k}$ states.

· We know the "wave function" in x space is $|\alpha\rangle = \int dx\, |x\rangle\langle x|\alpha\rangle$ (insert 1)
$= \int dx\, \varphi_\alpha(x) |x\rangle$

We can do this with the operators as well

$$\sum_\alpha \varphi_\alpha(x) a_\alpha = \hat{\psi}(x) \qquad \sum_\alpha a_\alpha^\dagger \varphi_\alpha^*(x) = \hat{\psi}^\dagger(x)$$

weighted sum — wave function $\Rightarrow$ "field operators"

$\hat{\psi}^\dagger(x)$ creates and $\hat{\psi}(x)$ destroys a particle at x. Spin: $\hat{\psi}(\vec{x}) \to \hat{\psi}_\alpha(\vec{x})$

$$\hat{H} \to \int d^3x\, \hat{\psi}_\alpha^\dagger(\vec{x})\left(-\frac{\nabla^2}{2m}\right)\hat{\psi}_\alpha(\vec{x}) + \int d^3x\, d^3x'\, \hat{\psi}_\alpha^\dagger(\vec{x})\hat{\psi}_\beta^\dagger(\vec{x}')V(\vec{x}-\vec{x}')\hat{\psi}_\beta(\vec{x}')\hat{\psi}_\alpha(\vec{x})$$

$$\xrightarrow{V(\vec{x}-\vec{x}') = \lambda\delta(\vec{x}-\vec{x}')} \int d^3x\, \hat{\psi}_\lambda^\dagger(x)\hat{\psi}_\beta^\dagger(\vec{x})\hat{\psi}_\beta(\vec{x})\hat{\psi}_\alpha(\vec{x})$$

Looks like what we've been doing!

10/21/09

Revisit coherent states and then continue with functional integral

- Some more on how to interpret bosonic coherent states. (From Negele + Orland, problem 1.5).

- What is the wave function of a (classical) pendulum? Answer: a coherent state!

- Consider a (1 dof) harmonic oscillator:

$$\hat{H} = \frac{1}{2}\hat{p}^2 + \frac{1}{2}\omega^2\hat{x}^2 = \hbar\omega(a^\dagger a + \tfrac{1}{2}) \quad (\text{mass } m=1 \text{ here})$$

$$\Rightarrow \hat{x} = \left(\frac{\hbar}{2\omega}\right)^{1/2}(a^\dagger + a) \qquad \hat{p} = i\left(\frac{\hbar\omega}{2}\right)^{1/2}(a^\dagger - a)$$

- We can look at the coherent state

In our earlier discussion of this was called z ⇒ complex!

$$|\phi\rangle = e^{a^\dagger\phi}|0\rangle = \sum_n \frac{\phi^n}{\sqrt{n!}}|n\rangle \xrightarrow[\text{derive and solve equation.}]{\text{coordinate space}} \langle x|\phi\rangle = \phi(x) = C e^{-\left[\left(\frac{\omega}{2\hbar}\right)^{1/2}x - \phi\right]^2}$$

This is Heisenberg rep. Schrödinger time dependent wf is

ϕ labels both the wave function and the value. Confusing!

$$\text{claim: } |\phi_{sch}(x,t)|^2 = \sqrt{\frac{\omega}{\pi\hbar}}\, e^{-2\left[\sqrt{\frac{\omega}{2\hbar}}x - |\phi|\cos\omega t\right]^2}$$

This is the same as a "minimum uncertainty wave packet". [Solution to S-eqn with $\Delta x \Delta p$ minimum]

- We can calculate $\langle\hat{x}\rangle = \left(\frac{2\hbar}{\omega}\right)^{1/2}|\phi|\cos\omega t$, $\langle\hat{p}\rangle = (2\hbar\omega)^{1/2}|\phi|\sin\omega t$ and $\langle\hat{H}\rangle = \frac{1}{2}(\langle\hat{p}^2\rangle + \omega^2\langle\hat{x}^2\rangle) = \hbar\omega(|\phi|^2 + 1/2)$
so $\langle x\rangle$ and $\langle\hat{p}\rangle$ satisfy in coherent state the classical equations of motion.

- amplitude is proportional to $|\phi|$!

- $|\phi\rangle$ mixes all $|n\rangle$'s. Most probable has $n = |\phi|^2$ and $E_n = \hbar\omega(n + 1/2) = \hbar\omega(|\phi|^2 + \tfrac{1}{2}) = \langle H\rangle$

- Sampling strategies

Goal: find the best (most efficient) way to obtain $N_{samples}$ independent configurations of the trajectory $\{x(\tau)\}$, distributed according to the probability $P[x(\tau)] = \frac{1}{Z} \exp\{-S_E[x(\tau)]\}$

Four basic strategies

① Heat bath algorithm → Useful when you can sample directly using a readily available random number generator (RNG).

Note: • People use this all the time in lattice QCD to generate "pseudofermions". More on this later!

• This method can be used for certain simple pure gauge theories (see Refs. in lecture 6)

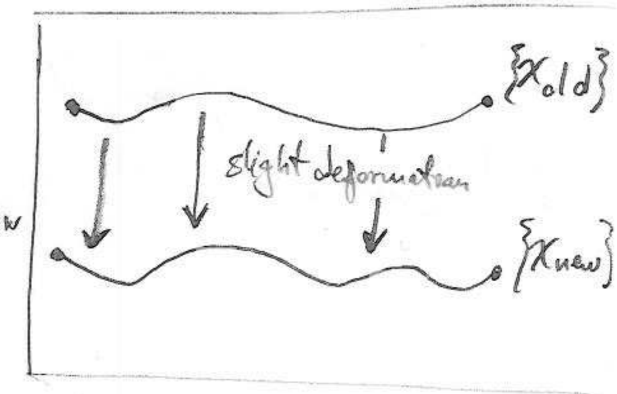
In general the probability is a very complicated function, so we need more general strategies.

② Metropolis algorithm → This is so generic that almost every non-heat-bath method involves this algorithm.

If you've done an Ising model simulation, then you surely know about this.

Before going into the details of why this algorithm works, let's try to understand the recipe (i.e. how it works).

- Metropolis recipe (I mean... algorithm!)



0. Pick a random configuration x_{old}

1. Pick a tentative new configuration x_{new}

2. Compute

$$q = \exp\left\{-\left(S_E[x_{new}] - S_E[x_{old}]\right)\right\}$$

3. Pick a uniform random number $\xi \in [0,1]$.

4. If $\xi < q \rightarrow$ set $x_{old} = x_{new}$ (updating!)

   If $\xi > q \rightarrow$ retain x_{old}, discard x_{new}.

5. Go back to step 1.

- Claim: • After a large enough # of iterations the "current" x_{old} configuration will be distributed according to $P = \frac{e^{-S_E}}{Z}$.

- Comments: (Ask first!)
  - When will x_{new} be accepted regardless of the value of ξ?

    Answer: when $S_E(\text{new}) < S_E(\text{old})$
  - What happens if S_E is not bounded from below? What are the consequences for P in that case?

    Answer: • the simulation will "run away"; it becomes unstable
    • P is not bounded from above in that case.

- Why does this algorithm work in practice?

It works because it creates a Markov chain that is

- Ergodic: a priori there is no prefered or forbidden configurations, so all of config. space will be explored if we wait long enough.

- Reversible:

$$\frac{e^{-S_E[x_{old}]}}{Z} \cdot W(x_{old} \to x_{new}) = \frac{e^{-S_E[x_{new}]}}{Z} W(x_{new} \to x_{old})$$

With these two conditions one can show that e^{-S_E} is an equilibrium distribution and that the equilibrium is stable. Let's prove the first.

- Equilibrium:

$$P[x_{new}] = \int \mathcal{D}x_{old} \frac{e^{-S_E[x_{old}]}}{Z} W(x_{old} \to x_{new}) \overset{\text{Reversibility}}{=} \frac{e^{-S_E[x_{new}]}}{Z} \underbrace{\int \mathcal{D}x_{old} W(x_{new} \to x_{old})}_{1} = \frac{e^{-S_E[x_{new}]}}{Z}$$

probability that we get configuration $\{x_{new}(\tau)\}$

assuming we are already sampling according to $\frac{e^{-S_E[x]}}{Z}$

by ergodicity

We obtain the same distribution!

- Is Metropolis reversible?

  - If $S_E[y] < S_E[x]$, then the move $x \to y$ is accepted, and $y \to x$ is accepted w/ probability $q = e^{-(S_E[x] - S_E[y])}$ (Notice we are looking at $y \to x$, not $x \to y$!)

$$\therefore \quad \frac{W(x \to y)}{W(y \to x)} = \frac{1}{q} = \frac{e^{-S_E[y]}}{e^{-S_E[x]}} \qquad \text{QED.}$$

• Using the Metropolis algorithm

• Always remember, we need to wait for a # of steps so that we reach

- Equilibration (a.k.a. thermalization, before taking the first sample)

- Decorrelation (i.e. wait between samples so they are independent).

• How do we decide on the changes when going from $x_{old} \to x_{new}$?

• Big changes $\to$ Big changes in S_E $\longrightarrow$ Better decorrelation, but acceptance rate drops !

• Small changes $\longrightarrow$ Better acceptance rate, but decorrelation time increases ... :(

Compromise: Make changes such that acc. rate $\simeq 1/2$.

- Problems of using Metropolis alone.

Perhaps the most important disadvantage of using the Metropolis algorithm without any extra concept is the following.

Actions tend to be complicated functions of the field variables, especially when fermions are present, in which case the action is <u>very</u> non-linear and <u>very</u> non-local.

As a consequence, only small localized changes are possible (if we do this randomly), or the acceptance rate will drop dramatically.

→ <u>We need algorithms that can perform global changes.</u>
More specifically, we need an updating strategy that does not destroy the acceptance rate, but makes changes everywhere.

(i.e. there is nothing wrong with Metropolis itself,
rather the updating strategy $x_{old} \to x_{new}$ is the problem)