LATTICE QCD FOR NOVICES

G. PETER LEPAGE

Newman Laboratory of Nuclear Studies
Cornell University, Ithaca, NY 14853
E-mail: gpl@mail.lns.cornell.edu

These lectures are for novices to lattice QCD. They introduce a set of simple ideas
and numerical techniques that can be implemented in a short period of time and
that are capabable of generating nontrivial, nonperturbative results from lattice
QCD. The simplest of these calculations can be completed on a standard worksta-
tion or high-end personal computer.

1 Introduction

These lectures are for novices who are interested in learning how to do lattice
QCD simulations. My intent is to describe in detail everything that one needs
to know in order to create and run a simple lattice QCD simulation. My
focus here is not on lengthy derivations or detailed comparisons of algorithms.
Rather I want to introduce a set of simple ideas and techniques that one
can implement in a relatively short time and that are capable of generating
nontrivial results from lattice QCD.

We begin in Section 2 with simple one-dimensional quantum mechanics.
Most of the simulation techniques can be applied to ordinary quantum me-
chanics, and the simulations require only seconds or minutes of computer time,
rather than the hours or days needed for QCD simulations. Consequently such
applications are ideal for learning the simulation technology. We broaden the
discussion to include quantum field theories in Section 3. Here we discuss the
theoretical techniques needed to obtain accurate results on the relatively coarse
lattices well suited to smaller computers. Finally, in Section 4, we adapt our
techniques for simulations of gluon dynamics in QCD.

2 Numerical Path Integrals

2.1 *Discretizing the Path Integral*

We begin with numerical techniques for evaluating *path integrals*. Recall what
path integrals tell us. In one-dimensional quantum mechanics, for example, the
evolution of a position eigenstate $|x_\mathrm{i}\rangle$ from time t_i to time t_f can be computed
using a path integral[1]:

$$\langle x_\mathrm{f}|\, \mathrm{e}^{-\tilde{H}(t_\mathrm{f}-t_\mathrm{i})}\, |x_\mathrm{i}\rangle = \int \mathcal{D}x(t)\, \mathrm{e}^{-S[x]}. \tag{1}$$

Here the $\int \mathcal{D}x(t)$ designates a sum over all possible particle paths

$$\{x(t) \quad \text{for} \quad t = t_\mathrm{i} \to t_\mathrm{f}\} \tag{2}$$

with

$$x(t_\mathrm{i}) = x_\mathrm{i} \qquad x(t_\mathrm{f}) = x_\mathrm{f}. \tag{3}$$

The hamiltonian is $\tilde{H}$, and $S[x]$ is the classical action,

$$S[x] \equiv \int_{t_\mathrm{i}}^{t_\mathrm{f}} dt\, L(x, \dot{x}) \equiv \int_{t_\mathrm{i}}^{t_\mathrm{f}} dt\, \left[\frac{m\, \dot{x}(t)^2}{2} + V(x(t)) \right], \tag{4}$$

evaluated for each path $x(t)$. There are no i's in these formulas because we are using "euclidean" path integrals. These are derived in the same way as standard path integrals but with $it \to t$. Euclidean path integrals are much better for numerical work since the integrands do not oscillate wildly in sign.

Knowledge of the propagator, Eq. (1), as a function of $x_\mathrm{i}, t_\mathrm{i}, x_\mathrm{f}, t_\mathrm{f}$ gives us complete information about the quantum theory. For example, we can easily determine the groundstate energy and wavefunction. Setting

$$x_\mathrm{i} = x_\mathrm{f} \equiv x \qquad t_\mathrm{f} - t_\mathrm{i} \equiv T, \tag{5}$$

the propagator can be rewritten

$$\langle x|\, \mathrm{e}^{-\tilde{H}T}\, |x\rangle = \sum_n \langle x|\, E_n\rangle\, \mathrm{e}^{-E_n T}\, \langle E_n|\, x\rangle \tag{6}$$

where $|E_n\rangle$ is the energy eigenstate with eigenvalue E_n. The sum is dominated by the lowest-energy states when T is large, because of the exponentials, and in the limit of very large T only the groundstate, $|E_0\rangle$, contributes:

$$\langle x|\, \mathrm{e}^{-\tilde{H}T}\, |x\rangle \overset{T \to \infty}{\longrightarrow} \mathrm{e}^{-E_0 T}\, |\langle x|\, E_0\rangle|^2. \tag{7}$$

We extract the groundstate energy E_0 by integrating over x,

$$\int dx\, \langle x|\, \mathrm{e}^{-\tilde{H}T}\, |x\rangle \overset{T \to \infty}{\longrightarrow} \mathrm{e}^{-E_0 T}, \tag{8}$$

and then, going back to the previous equation, we determine the groundstate wavefunction $\psi_{E_0}(x) \equiv \langle x|\, E_0\rangle$.

Our goal therefore is to develop a numerical procedure for evaluating the propagator using a path integral. There are two issues we must address. First we must find a way to represent an arbitrary particle path $\{x(t),\, t_\mathrm{i} \leq t \leq t_\mathrm{f}\}$

in the computer. A path is specified by a function $x(t)$ which, in principle, can be infinitely complex (and therefore too much for any computer). We approximate this function by specifying $x(t)$ only at the nodes or sites on a discretized t axis:

$$t_j = t_{\mathrm{i}} + j\,a \qquad \text{for } j = 0, 1 \ldots N \tag{9}$$

where a is the grid spacing,

$$a \equiv \frac{t_{\mathrm{f}} - t_{\mathrm{i}}}{N}. \tag{10}$$

Then a path is described by a vector of numbers,

$$x = \{x(t_0), x(t_1) \ldots x(t_N)\}. \tag{11}$$

It is common practice to refer to such a path as a "configuration". The integral over all paths in this approximation becomes an ordinary integral over all possible values for each of the $x(t_j)$'s: that is,

$$\int \mathcal{D}x(t) \rightarrow A \int_{-\infty}^{\infty} dx_1\, dx_2 \ldots dx_{N-1} \tag{12}$$

where we have adopted the notation $x_j \equiv x(t_j)$. We don't integrate over the endpoints since they are held fixed; for example, for boundary conditions (5),

$$x_0 = x_N = x. \tag{13}$$

We won't need the normalization factor A for most of our work, but for our one-dimensional problem it is[1]

$$A \equiv \left(\frac{m}{2\pi a}\right)^{N/2} \tag{14}$$

The second issue we must address concerns the evaluation of the action given only a discretized path $\{x_j\}$. Focusing just on the contribution from $t_j \le t \le t_{j+1}$, the obvious approximation is

$$\int_{t_j}^{t_{j+1}} dt\, L \;\approx\; a \left[\frac{m}{2} \left(\frac{x_{j+1} - x_j}{a} \right)^2 + \frac{1}{2} \left(V(x_{j+1}) + V(x_j) \right) \right] \tag{15}$$

With this approximation, our numerical representation of the path integral is complete, and we have an approximate expression for the quantum mechanical propagator: for example,

$$\langle x | \, \mathrm{e}^{-\tilde{H}T} \, | x \rangle \approx A \int_{-\infty}^{\infty} dx_1 \ldots dx_{N-1}\, \mathrm{e}^{-S_{\mathrm{lat}}[x]} \tag{16}$$

3

where

$$S_{\text{lat}}[x] \equiv \sum_{j=0}^{N-1} \left[\frac{m}{2a}(x_{j+1} - x_j)^2 + aV(x_j) \right], \tag{17}$$

$x_0 = x_N = x$, and $a = T/N$. We have reduced quantum mechanics to a problem in numerical integration.

One might worry about approximating $\dot{x}$ with $(x_{j+1} - x_j)/a$ in our formula for the lattice action $S_{\text{lat}}[x]$. It is not obvious that this is a good approximation given that $x_{j+1} - x_j$ can be arbitrarily large in our path integral; that is, paths can be arbitrarily rough. While not so important for our one-dimensional problem, this becomes a crucial issue for four-dimensional field theories. It is dealt with using renormalization theory, which we discuss in later sections.

Exercise: Set $a = 1/2$ and $N = 8$ in approximate formula (16) for the propagator, and integrate the right-hand side numerically. The seven dimensional integral that results is easily evaluated using standard routines, such as vegas[2]. Do this first for the one-dimensional harmonic-oscillator potential

$$V(x) = \frac{x^2}{2} \qquad \text{with} \qquad m = 1. \tag{18}$$

Evaluate the propagator for several values of $x_0 = x_N = x$, and compare your results with those of standard quantum mechanics:

$$\langle x| e^{-\tilde{H}T} |x\rangle \approx |\langle x| E_0\rangle|^2 e^{-E_0 T} \tag{19}$$

where $E_0 = 1/2$ and

$$\langle x| E_0\rangle = \frac{e^{-x^2/2}}{\pi^{1/4}}. \tag{20}$$

Extract the energy and wavefunction from your numerical result. Repeat this exercise for $V(x) = x^4/2$. (If you wish, you may restrict x integrations to the region $-5 \to 5$ rather than $-\infty \to \infty$; this has negligible effect on the results of this exercise).

My results for the harmonic oscillator case are shown in Fig. 1.

2.2 Monte Carlo Evaluation of Path Integrals

Our analysis in the previous section focused on the groundstate. In quantum field theory, where the groundstate is the vacuum, we are generally interested in excited states. To analyze excited states using path integrals, we interrupt the propagation of the groundstate by introducing new operators at intermediate times. Consider, for example, the quantity

$$\langle\langle x(t_2)x(t_1)\rangle\rangle \equiv \frac{\int \mathcal{D}x(t)\, x(t_2)x(t_1)\, e^{-S[x]}}{\int \mathcal{D}x(t)\, e^{-S[x]}} \tag{21}$$
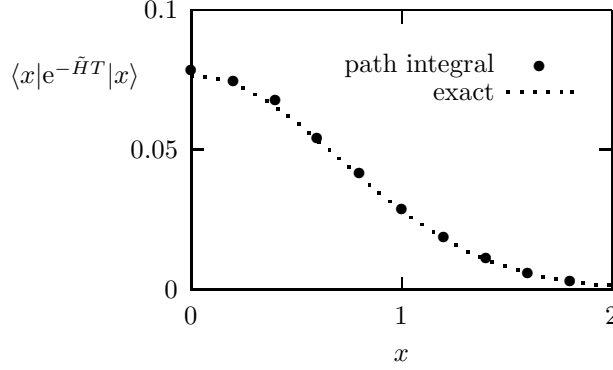
Figure 1: Euclidean harmonic-oscillator propagator at large time ($T = 4$) computed exactly (dotted line), and computed using numerical integration to evaluate the discretized path integral (points). The path integral was approximated by an 8 dimensional integral which was evaluated numerically, using `vegas`, at the points indicated. The exact result is approximated by the square of the ground state wavefunction multiplied by $\exp(-E_0T)$.

where now we integrate over all $x_\mathrm{i} = x_\mathrm{f} = x$ as well as the intermediate $x(t)$'s. This quantity is a weighted average of $x(t_2)x(t_1)$ over all paths, with weight $\exp(-S[x])$. The numerator on the right-hand side equals, in quantum mechanics,

$$\int dx \; \langle x | \, \mathrm{e}^{-\tilde{H}(t_\mathrm{f}-t_2)} \, \tilde{x} \, \mathrm{e}^{-\tilde{H}(t_2-t_1)} \, \tilde{x} \, \mathrm{e}^{-\tilde{H}(t_1-t_\mathrm{i})} \, | x \rangle \,. \tag{22}$$

Setting $T = t_\mathrm{f} - t_\mathrm{i}$ and $t = t_2 - t_1$ we can rewrite the full expression as

$$\langle\langle x(t_2)x(t_1) \rangle\rangle = \frac{\sum \mathrm{e}^{-E_n T} \, \langle E_n | \, \tilde{x} \, \mathrm{e}^{-(\tilde{H}-E_n)t} \, \tilde{x} \, | E_n \rangle}{\sum \mathrm{e}^{-E_n T}} \,. \tag{23}$$

If $T \gg t$ and large, then the groundstate $|E_0\rangle$ dominates the sums and

$$G(t) \equiv \langle\langle x(t_2)x(t_1) \rangle\rangle \to \langle E_0 | \, \tilde{x} \, \mathrm{e}^{-(\tilde{H}-E_0)t} \, \tilde{x} \, | E_0 \rangle \,. \tag{24}$$

In our harmonic oscillator example, the state propagating between the two $\tilde{x}$'s cannot be $|E_0\rangle$ since $\tilde{x}$ switches the parity of the state. Thus if we now make t large (but still $\ll T$)

$$G(t) \xrightarrow{t \text{ large}} |\langle E_0 | \, \tilde{x} \, | E_1 \rangle|^2 \; \mathrm{e}^{-(E_1-E_0)t} \tag{25}$$

5

where $|E_1\rangle$ is the first excited state. Consequently we can extract the first excitation energy from the large-t dependence of $G(t)$,

$$\log(G(t)/G(t+a)) \to (E_1 - E_0)a, \qquad (26)$$

and then, going back to $G(t)$, we can determine the quantum mechanical transition matrix element $\langle E_0| \tilde{x} |E_1\rangle$.

In principle, path integral averages $\langle\langle\Gamma[x]\rangle\rangle$ of arbitrary functionals $\Gamma[x]$ can be used to compute any physical property of the excited states in the quantum theory. Also we note in passing that

$$\langle\langle\Gamma[x]\rangle\rangle = \frac{\sum e^{-E_n T} \langle E_n| \Gamma[\tilde{x}] |E_n\rangle}{\sum e^{-E_n T}} \qquad (27)$$

becomes a (quantum mechanical) thermal average if we replace

$$T \to \beta \equiv 1/k_B\, T_{\text{temp}}. \qquad (28)$$

Thus any computer code designed to compute path integral averages can be used for thermal physics as well. Here we focus on the zero-temperature limit of large T.

We could evaluate the path integrals in $\langle\langle\Gamma[x]\rangle\rangle$ using a standard multidimensional integration code like `vegas`, at least for one-dimensional systems. Here, instead, we employ a more generally useful Monte Carlo procedure. Noting that

$$\langle\langle\Gamma[x]\rangle\rangle = \frac{\int \mathcal{D}x(t)\, \Gamma[x]\, e^{-S[x]}}{\int \mathcal{D}x(t)\, e^{-S[x]}}, \qquad (29)$$

is a weighted average over paths with weight $\exp(-S[x])$, we generate a large number, N_{cf}, of random paths or configurations,

$$x^{(\alpha)} \equiv \{x_0^{(\alpha)} x_1^{(\alpha)} \ldots x_{N-1}^{(\alpha)}\} \qquad \alpha = 1, 2 \ldots N_{\text{cf}}, \qquad (30)$$

on our grid in such a way that the probability $P[x^{(\alpha)}]$ for obtaining any particular path $x^{(\alpha)}$ is

$$P[x^{(\alpha)}] \propto e^{-S[x^{(\alpha)}]}. \qquad (31)$$

Then an unweighted average of $\Gamma[x]$ over this set of paths approximates the weighted average over uniformly distributed paths:

$$\langle\langle\Gamma[x]\rangle\rangle \approx \overline{\Gamma} \equiv \frac{1}{N_{\text{cf}}} \sum_{\alpha=1}^{N_{\text{cf}}} \Gamma[x^{(\alpha)}]. \qquad (32)$$

6

$\overline{\Gamma}$ is our "Monte Carlo estimator" for $\langle\langle\Gamma[x]\rangle\rangle$ on our lattice. Of course the estimate will never be exact since the number of paths N_{cf} will never be infinite. The Monte Carlo uncertainty $\sigma_{\overline{\Gamma}}$ in our estimate is a potential source of error; it is estimated in the usual fashion[3]:

$$\sigma_{\overline{\Gamma}}^2 \approx \frac{1}{N_{\text{cf}}} \left\{ \frac{1}{N_{\text{cf}}} \sum_{\alpha=1}^{N_{\text{cf}}} \Gamma^2[x^{(\alpha)}] - \overline{\Gamma}^2 \right\}. \tag{33}$$

This becomes

$$\sigma_{\overline{\Gamma}}^2 = \frac{\langle\langle\Gamma^2\rangle\rangle - \langle\langle\Gamma\rangle\rangle^2}{N_{\text{cf}}} \tag{34}$$

for large N_{cf}. Since the numerator in this expression is independent of N_{cf} (in principle, it can be determined directly from quantum mechanics), the statistical uncertainties vanish as $1/\sqrt{N_{\text{cf}}}$ when N_{cf} increases.

We need some sort of specialized random-vector generator to create our set of random paths $x^{(\alpha)}$ with probability (31). Possibly the simplest procedure, though not always the best, is the Metropolis Algorithm[4]. In this procedure, we start with an arbitrary path $x^{(0)}$ and modify it by visiting each of the sites on the lattice, and randomizing the x_j's at those sites, one at a time, in a particular fashion that is described below. In this way we generate a new random path from the old one: $x^{(0)} \rightarrow x^{(1)}$. This is called "updating" the path. Applying the algorithm to $x^{(1)}$ we generate path $x^{(2)}$, and so on until we have N_{cf} random paths. This set of random paths has the correct distribution if N_{cf} is sufficiently large.

The algorithm for randomizing x_j at the jth site is:

- generate a random number ζ, with probability uniformly distributed between $-\epsilon$ and ϵ for some constant ϵ;

- replace $x_j \rightarrow x_j + \zeta$ and compute the change ΔS in the action caused by this replacement (generally only a few terms in the lattice action involve x_j, since lagrangians are local; only these need be examined);

- if $\Delta S < 0$ (the action is reduced) retain the new value for x_j, and proceed to the next site;

- if $\Delta S > 0$ generate a random number η uniformly distributed between 0 and 1; retain the new value for x_j if $\exp(-\Delta S) > \eta$, otherwise restore the old value; proceed to the next site.

An implementation of this algorithm, in the Python computer language[5], is shown in Fig. 2. The code examples and Python are discussed in the Appendix.

```
def update(x):
    for j in range(0,N):
        old_x = x[j]                    # save original value
        old_Sj = S(j,x)
        x[j] = x[j] + uniform(-eps,eps) # update x[j]
        dS = S(j,x) - old_Sj            # change in action
        if dS>0 and exp(-dS)<uniform(0,1):
            x[j] = old_x                # restore old value

def S(j,x):                             # harm. osc. S
    jp = (j+1)%N                        # next site
    jm = (j-1)%N                        # previous site
    return a*x[j]**2/2 + x[j]*(x[j]-x[jp]-x[jm])/a
```

Figure 2: Python code for one Metropolis update of path $\{x_j, \; j = 0 \ldots N-1\}$. The path is stored in array x[j]. Function S(j,x) returns the value of the part of the action that depends on x_j. Function uniform(a,b) returns a random number between a and b. A sample S(j,x) is shown, for a harmonic oscillator with $x_N = x_0$.

There are two important details concerning the tuning and use of this algorithm. First, in general some or many of the x_j's will be the same in two successive random paths. The amount of such overlap is determined by the parameter ϵ: when ϵ is very large, changes in the x_j's are usually large and most will be rejected; when ϵ is very small, changes are small and most are accepted, but the new x_j's will be almost equal to the old ones. Neither extreme is desirable since each leads to very small changes in x, thereby slowing down the numerical exploration of the space of all important paths. Typically ϵ should be tuned so that 40%–60% of the x_j's are changed on each pass (or "sweep") through the lattice. Then ϵ is of order the typical quantum fluctuations expected in the theory. Whatever the ϵ, however, successive paths are going to be quite similar (that is "highly correlated") and so contain rather similar information about the theory. Thus when we accumulate random paths $x^{(\alpha)}$ for our Monte Carlo estimates we should keep only every N_{cor}-th path; the intervening sweeps erase correlations, giving us configurations that are statistically independent. The optimal value for N_{cor} depends upon the theory, and can be found by experimentation. It also depends on the lattice spacing a, going roughly as

$$N_{\mathrm{cor}} \propto \frac{1}{a^2}. \tag{35}$$

Other algorithms exist for which N_{cor} grows only as $1/a$ when a is reduced,

but since our interest is in large a's we will not discuss these further.

The second detail concerns the procedure for starting the algorithm. The very first configuration used to seed the whole process is usually fairly atypical. Consequently we should discard some number of configurations at the beginning, before starting to collect $x^{(\alpha)}$'s. Discarding $5N_{\text{cor}}$ to $10N_{\text{cor}}$ configurations is usually adequate. This is called "thermalizing the lattice."

To summarize, a computer code for a complete Monte Carlo calculation of $\langle\langle\Gamma[x]\rangle\rangle$ for some function $\Gamma[x]$ of a path x consists of the following steps:

- initialize the path, for example, by setting all x_j's to zero;

- update the path $5N_{\text{cor}}$–$10N_{\text{cor}}$ times to thermalize it;

- update the path N_{cor} times, then compute $\Gamma[x]$ and save it; repeat N_{cf} times.

- average the N_{cf} values of $\Gamma[x]$ saved in the previous step to obtain a Monte Carlo estimate $\overline{\Gamma}$ for $\langle\langle\Gamma[x]\rangle\rangle$.

A Python implementation of this procedure is shown in Fig. 3.

Exercise: Write a computer program to implement the Metropolis Monte Carlo algorithm for a one dimensional harmonic oscillator (Eq. (18)), and calculate

$$G(t) = \frac{1}{N}\sum_j \langle\langle x(t_j + t)x(t_j)\rangle\rangle \tag{36}$$

for all $t = 0, a, 2a \ldots (N-1)a$; that is calculate

$$G_n = \frac{1}{N}\sum_j \langle\langle x_{(j+n)\bmod N}\, x_j\rangle\rangle \tag{37}$$

for $n = 0 \ldots N-1$. The $(j+n)\bmod N$ in this last expression reflects the periodic boundary conditions. Try $N = 20$ lattice sites with lattice spacing $a = 1/2$, and set $\epsilon = 1.4$ and $N_{\text{cor}} = 20$. Try N_{cf}'s of 25, 100, 1000 and 10000. Use the results to compute the excitation energy from

$$\Delta E_n \equiv \log(G_n/G_{n+1}) \overset{n \text{ large}}{\longrightarrow} (E_1 - E_0)a \tag{38}$$

Try this for the harmonic oscillator potential and also for anharmonic potentials. Vary the various parameters.

My results for the harmonic oscillator potential with $N = 1000$ configurations are shown in Fig. 4. These results required less than a minute of personal computer time.

9

```
def compute_G(x,n):
    g = 0
    for j in range(0,N):
        g = g + x[j]*x[(j+n)%N]
    return g/N

def MCaverage(x,G):
    for j in range(0,N):                    # initialize x
        x[j] = 0
    for j in range(0,5*N_cor):              # thermalize x
        update(x)
    for alpha in range(0,N_cf):             # loop on random paths
        for j in range(0,N_cor):
            update(x)
        for n in range(0,N):
            G[alpha][n] = compute_G(x,n)
    for n in range(0,N):                    # compute MC averages
        avg_G = 0
        for alpha in range(0,N_cf):
            avg_G = avg_G + G[alpha][n]
        avg_G = avg_G/N_cf
        print "G(%d) = %g" % (n,avg_G)
```

Figure 3: Sample Python code for a Monte Carlo evaluation of of $G(t)$ (Eq. (36)). Function `computeG(x,t)` computes $G(t)$ for a given path x. Function `MCaverage(x,G)` computes the Monte Carlo average over random paths x. The results for path $x^{(\alpha)}$ are stored in the array `G[alpha][t]`, and the averages are computed and printed. Function `update(x)` does one Metropolis sweep through the lattice (see Fig. 2).
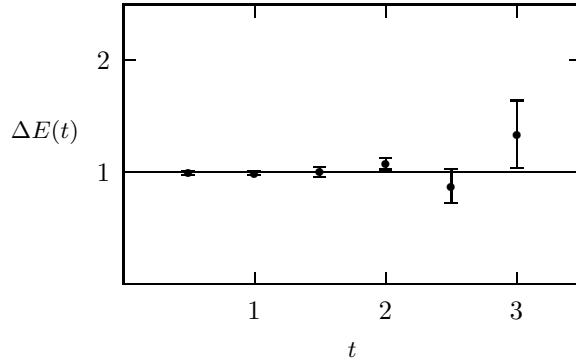
Figure 4: Monte Carlo values $\Delta E(t) \equiv \log(G(t)/G(t+a))/a$ plotted versus t for an harmonic oscillator. The exact asymptotic result, $\Delta E(\infty) = 1$, is indicated by a line. Results are for a one dimensional lattice with $N = 20$ sites, lattice spacing $a = 1/2$, and $N_{\mathrm{cf}} = 1000$ configurations, keeping configurations only every $N_{\mathrm{cor}} = 20$ sweeps. The Metropolis step size ϵ was 1.4, resulting in a Metropolis acceptance ratio of 0.5.

Exercise: Redo the previous exercise but propagator

$$G(t) = \frac{1}{N} \sum_j \langle\langle x^3(t_j + t) x^3(t_j) \rangle\rangle. \tag{39}$$

Here we use x^3 rather than x to create and destroy the excited stated; that is, we use x^3 rather than x as the "source" and the "sink". Note that $\Delta E(t)$ converges to the same result, but only at much larger t's than before. Different sources and sinks often lead to different asymptotic behavior. Choices that result in fast convergence as t increases are usually preferable because statistical errors are smaller at smaller t's. Compare your best estimate of the asymptotic value obtained using x^3 with that obtained using x.

Note also that $\Delta E(t)$ approaches its its asymptotic value from above. Prove that this must be true in general, provided source and sink are the same operator, using Eq. (23). This result is useful because it implies that each $\Delta E(t)$ gives a rigorous upper bound on the asymptotic value, even at small t's before convergence.

My results with an x^3 source and sink are shown in Fig. 5.

2.3 Statistical Errors

A Monte Carlo estimate $\overline{\Gamma}$ of some expectation value $\langle\langle\overline{\Gamma}\rangle\rangle$ is never exact; there are always statistical errors that vanish only in the limit where infinitely many
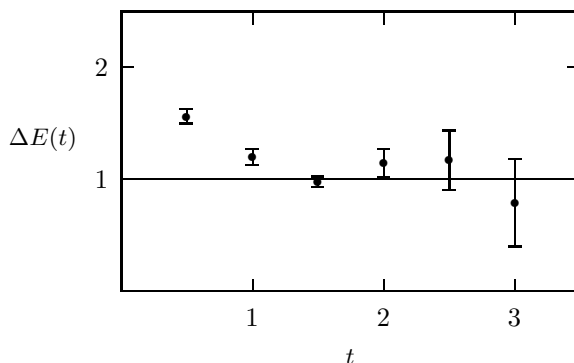
11

Figure 5: Monte Carlo values $\Delta E(t) \equiv \log(G(t)/G(t+a))/a$ plotted versus t for an harmonic oscillator using x^3 as the source and sink. Parameters are the same as used to generate Fig. 4. The energies take longer to reach their asymptotic value.

configurations are employed ($N_{\rm cf} \to \infty$). An important part of any Monte Carlo analysis is the estimation of these statistical errors. There is a simple but very powerful method, called the "statistical bootstrap," for making such estimates.

In the previous exercises, for example, we assemble an "ensemble" of measurements of the propagator $G^{(\alpha)}$, one for each configuration $x^{(\alpha)}$. These are averaged to obtain $\overline{G}$, and, from it, an estimate for ΔE_n (Eq. (38)). An obvious way to check the statistical errors on this estimate for ΔE_n is to redo the whole calculation, say, 100 times, each time with different random numbers to generate different random paths. With 100 copies of the entire calculation, we could analyze the distribution of the 100 random ΔE_n's obtained, and deduce the statistical uncertainty in our original estimate. This, however, is exceedingly expensive in computer time. The bootstrap procedure provides new, almost *zero-cost* random ensembles of measurements by synthesizing them from the original ensemble of $N_{\rm cf}$ measurements.

Given an ensemble $\{G^{(\alpha)}, \alpha = 1 \dots N_{\rm cf}\}$ of Monte Carlo measurements, we assemble a "bootstrap copy" of that ensemble by selecting $G^{(\alpha)}$'s at random from the original ensemble, taking $N_{\rm cf}$ in all while allowing duplications and omissions. The resulting ensemble of G's might have two or three copies of some $G^{(\alpha)}$'s, and no copies of others. This new ensemble can be averaged and a new estimate obtained for ΔE_n. This procedure can be repeated to generated as many bootstrap copies of the original ensemble as we wish, and from each

```
def bootstrap(G):
    N_cf = len(G)
    G_bootstrap = []                    # new ensemble
    for i in range(0,N_cf):
        alpha = int(uniform(0,N_cf))    # choose random config
        G_bootstrap.append(G[alpha])    # keep G[alpha]
    return G_bootstrap
```

Figure 6: Sample Python code for producing a bootstrap copy of an ensemble of measurements G. The original ensemble consists of individual measurements G[alpha], one for each configuration. The function bootstrap(G) returns a single bootstrap copy of ensemble G, consisting of N_cf measurements. Function uniform(a,b) returns a random number between a and b.

we can generate a new estimate for ΔE_n. The distribution of these ΔE_n's approximates the distribution of ΔE_n's that would have been obtained from the original Monte Carlo, and so can be used to estimate the statistical error in our original estimate. A Python implementation of this procedure is shown in Fig. 6.

Another useful procedure related to statistical errors is "binning." At the end of a large simulation we might have 100's or even 100,000's of configurations $x^{(\alpha)}$, and for each a set of measurements like $G^{(\alpha)}$, our propagator. The measurements will inevitably be averaged, but we want to save the separate $G^{(\alpha)}$'s for making bootstrap error estimates and the like. We can save a lot of disk space, RAM, and CPU time by partially averaging or binning the measurements: For example, instead of storing each of

$$ G^{(1)} \quad G^{(2)} \quad G^{(3)} \quad G^{(4)} \quad G^{(5)} \dots \tag{40} $$

we might instead store

$$ \begin{aligned} \overline{G}^{(1)} &\equiv \frac{G^{(1)} + G^{(2)} + G^{(3)} + G^{(4)}}{4} \\ \overline{G}^{(2)} &\equiv \frac{G^{(5)} + G^{(6)} + G^{(7)} + G^{(8)}}{4} \\ &\dots \end{aligned} \tag{41} $$

The $\overline{G}^{(\beta)}$'s are far less numerous but have the same average, standard deviation, and other statistical properties as the original set. Typically the bin size is adjusted so that there are only 50–100 $\overline{G}^{(\beta)}$'s. A Python implementation of this procedure is shown in Fig. 7.

```
def bin(G,binsize):
    G_binned = []                        # binned ensemble
    for i in range(0,len(G),binsize):    # loop on bins
        G_avg = 0
        for j in range(0,binsize):       # loop on bin elements
            G_avg = G_avg + G[i+j]
        G_binned.append(G_avg/binsize)   # keep bin avg
    return G_binned
```

Figure 7: Sample Python code for producing a binned copy of an ensemble of measurements `G`. The original ensemble consists of individual measurements `G[alpha]`, one for each configuration. The function `bin(G,binsize)` bins the ensemble into bins of size `binsize`, averages the `G`'s within each bin, and returns an ensemble consisting of the averages.

Binning has an important side effect: it reduces or can even remove the effects of correlations between different configurations. If, when generating configurations, N_{cor} is too small, successive Monte Carlo estimates are statistically correlated. This leads to error estimates, using Eq. (33), that can be much smaller than the true errors—a very bad situation. If, however, the Monte Carlo estimates are binned with sufficiently large bins, the majority of estimates in one bin will be uncorrelated from the majority in adjacent bins. Consequently the bin averages will be uncorrelated, and standard statistical formulas, like Eq. (33), are reliable. To determine the bin size required to remove correlations, first bin the measurements and compute the errors. Then rebin with double the bin size and recompute the errors. There are no correlations if the statistical errors are roughly independent of bin size; if, on the other hand, the statistical errors grow substantially with bin size (for example, proportional to the square root of the bin size), then there are strong correlations between bins. Continuing doubling the bin size until the statistical errors stop growing. Note that measurements of different physical quantities decorrelate at different rates; different things may require different bin sizes.

Exercise: Rerun your Metropolis simulation of the harmonic oscillator with $N_{\text{cor}} = 1$. Do several different runs and compare your results. Do they agree within statistical errors? Try binning the results from each of the runs in bins of 20 and recompute the statistical errors. Verify that different runs now agree within the errors computed from the binned results.

In Fig. 8 I show results from $N = 1000$ configurations using $N_{\text{cor}} = 1$. Compare this with results in Fig. 4, which come from the same number of configurations but with $N_{\text{cor}} = 20$. The errorbars in the $N_{\text{cor}} = 1$ plot are obviously unreliable.
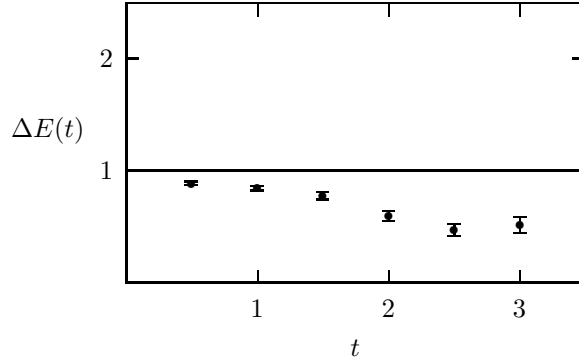
Figure 8: Monte Carlo values $\Delta E(t) \equiv \log(G(t)/G(t+a))/a$ plotted versus t for an harmonic oscillator, as in Fig. 4 but with $N_{\mathrm{cor}} = 1$. The errorbars are unreliable.

3   Field Theory on a Lattice

3.1   From Quantum Mechanics to Field Theory

Field theories of the sort we are interested in have lagrangian formulations and so can be quantized immediately using path integrals. The procedure is precisely analogous to what we do in the previous section when quantizing the harmonic oscillator. The analogues of the coordinates $x(t)$ in quantum mechanics are just the fields $\phi(x)$ or $A_\mu(x)$ where $x = (t, \vec{x})$ is a space-time point. Indeed our quantum mechanical examples can be thought of as field theory examples in 0 spatial and 1 temporal dimension: $x(t) \to \phi(t) \to \phi(x)$. The analogue of the ground state in quantum field theory is the vacuum state, $|0\rangle$, while the analogues of the excited states, created when $\phi(x)$ or ϕ^3 or . . . acts on $|0\rangle$, correspond to states with one or more particles create in the vacuum.

In the lattice approximation both space and time are discrete:

15
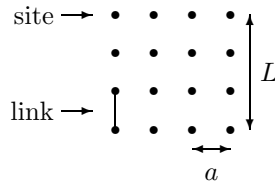
The nodes or "sites" of the grid are separated by lattice spacing a, and the length of a side of the grid is L; the lines joining adjacent sites are called "links." The quantum field is specified by its values at the grid sites: a configuration is describe by the set of numbers $\{\phi(x_j), \; \forall\, x_j \,\epsilon\, \text{grid}\}$. The path integral generalizes in the obvious fashion:

$$\langle\langle \Gamma[\phi] \rangle\rangle \equiv \frac{1}{\mathcal{Z}} \int \mathrm{e}^{-S[\phi]}\, \Gamma[\phi] \prod_{x_j\,\epsilon\,\text{grid}} d\phi(x_j) \tag{42}$$

where

$$\mathcal{Z} \equiv \int \mathrm{e}^{-S[\phi]} \prod_{x_j} d\phi(x_j). \tag{43}$$

Here the action $S[\phi]$ is the continuum action with spatial and temporal derivatives replaced by differences between field values at the grid sites. We study excitations of the field theory using operators like

$$\Gamma(t) \equiv \frac{1}{\sqrt{N}} \sum_{\vec{x}_j} \phi(\vec{x}_j, t) \tag{44}$$

where the sum over the N spatial $\vec{x}_j$'s enforces zero three-momentum. Since the excitations correspond to particle creation, their energies are the energies of particles: for example,

$$\langle\langle \Gamma(t)\Gamma(0) \rangle\rangle \; \overset{t\,\text{large}}{\longrightarrow} \; |\langle 0|\,\Gamma(0)\,|\phi : \vec{p} = 0\rangle|^2\, \mathrm{e}^{-m_\phi t}, \tag{45}$$

where $|\phi : \vec{p} = 0\rangle$ is a one ϕ-particle state with zero three momentum, and m_ϕ is the mass of the ϕ particle.

Exercise: Show that

$$\langle 0|\,\Gamma(0)\,|\phi : \vec{p} = 0\rangle = \frac{Z_2}{2m_\phi} \tag{46}$$

where Z_2 is the wavefunction renormalization parameter for the ϕ field.

3.2 Coarse Lattices

We have, through the lattice approximation to the path integral, turned the problem of solving a nonperturbative relativistic quantum field theory, once again, into a problem of numerical integration. This is a major development for theories, like QCD, where perturbation theory doesn't suffice (at low energies).

Early enthusiasm for such an approach to QCD, back when QCD was first invented, quickly gave way to the grim realization that very large computers

would be needed to numerically integrate the path integral. In recent years, however, two developments have made QCD simulations far more accessible. One is that small computers have become much faster; the other is that QCD simulations have become much faster — 10^3 to 10^6 times faster. These developments imply that the simplest QCD simulations can be done using no more than a single personal computer or even a laptop.

What has changed to make QCD simulations faster? The cost of a QCD calculation is given roughly by the formula

$$\text{cost} \approx \left(\frac{L}{a}\right)^4 \frac{1}{a} \frac{1}{m_\pi^2 a}, \tag{47}$$

where the first factor is the number of lattice sites, while the second and third factors are due to "critical-slowing-down" of the algorithms used for the simulation. From this formula, the single most important determinant of the cost is the lattice spacing: the cost is proportional to $1/a^6$. This means that one wants to keep a as large as possible. Until recently it was thought that $a < 0.05$–0.1 fm would be essential for accurate QCD simulations. As we shall see $a \approx 0.3$–0.4 fm works quite well. Given that the cost varies as $1/a^6$, the coarser lattices should be 10^3 to 10^6 times cheaper to simulate.

The size of the lattice spacing is limited by discretization errors. The challenge is to make the lattice spacing as large as possible while keeping the discretization errors of order, say, a few percent or less. These errors have two sources: first, the lattice forces us to use approximate derivatives, and, second, it imposes an ultraviolet cutoff. We consider each in turn.

In the lattice approximation, we only know the fields at the lattice sites. Thus all derivatives in field equations, the action, and the like must be converted to differences. For example, the second derivative of field ϕ at some point x_j on the lattice is given approximately by

$$\frac{\partial^2 \phi(x_j)}{\partial x^2} = \Delta_x^{(2)} \phi(x_j) + \mathcal{O}(a^2) \tag{48}$$

where

$$\Delta_x^{(2)} \phi(x) \equiv \frac{\phi(x+a) - 2\phi(x) + \phi(x-a)}{a^2}. \tag{49}$$

We generally want more accurate approximations for work on coarse lattices: for example, the approximation

$$\frac{\partial^2 \phi(x_j)}{\partial x^2} = \Delta_x^{(2)} \phi(x_j) - \frac{a^2}{12} (\Delta_x^{(2)})^2 \phi(x_j) + \mathcal{O}(a^4) \tag{50}$$

17

is accurate to within a few percent even when acting on structures in $\phi(x)$ that are only four or five lattice spacings across. With such precision one might expect that lattice spacings as large as a quarter the diameter of hadron, or about 0.4 fm, would still be quite useful. Our theories are quantum theories, however, and therefore there is a second important consideration.

The shortest wavelength oscillation that can be modeled on a lattice is one with wavelength $\lambda_{\min} = 2a$; for example, the function $\phi(x) = +1, -1, +1 \ldots$ for $x = 0, a, 2a \ldots$ oscillates with this wavelength. Thus gluons and quarks with momenta $p = 2\pi/\lambda$ larger than π/a are excluded from the lattice theory by the lattice; that is, the lattice functions as an ultraviolet cutoff. In simple classical field theories this is often irrelevant: short-wavelength ultraviolet modes are either unexcited or decouple from the long-wavelength infrared modes of interest. However, in a noisy nonlinear theory, like an interacting quantum field theory, ultraviolet modes strongly affect infrared modes. Thus we cannot simply discard all particles with momenta larger than π/a; we must somehow mimic their effects on infrared states. This is done by changing or "renormalizing" the parameters in our discretized theory and by adding new local interactions.

The new interactions complicate the improved discretizations discussed above. For example, an interacting scalar theory on the lattice would have a discretized kinetic lagrangian

$$\sum_\mu \tfrac{1}{2}\phi^\dagger \partial_\mu^2 \phi \rightarrow \sum_\mu \tfrac{1}{2}\left(\phi^\dagger \Delta_\mu^{(2)}\phi + a^2 c\, \phi^\dagger (\Delta_\mu^{(2)})^2\phi\right) \tag{51}$$

where parameter c has two parts: $-1/12$ from numerical analysis (Eq. (50)), and an additional renormalization due to the cutoff. Typically the renormalization is completely context dependent — for example, it is different for QED and QCD, or for particles of different spin, and so on. It cannot be looked up in a numerical analysis book; rather, it must be computed using quantum field theory. In QCD these renormalizations can be computed using (weak-coupling) perturbation theory, since the renormalizations are due to QCD physics at large momenta, $p > \pi/a$, where the theory is perturbative:

$$c = -\frac{1}{12} + c_1 \alpha_s(\pi/a) + c_2 \alpha_s^2(\pi/a) + \cdots. \tag{52}$$

This is true, that is, provided the lattice spacing is small enough that momentum π/a is perturbative. Work in continuum QCD suggests that lattice spacings of 0.5 fm or smaller should suffice, but, until recently, lattice simulations seemed to suggest that perturbation theory only started to work for lattice spacings smaller than 0.05–0.1 fm.

18

Exercise: Our action for one-dimensional quantum mechanics, Eq. (4), can be rewritten

$$S[x] \equiv \int_{t_i}^{t_f} dt \left[-\tfrac{1}{2} m\, x(t)\ddot{x}(t) + V(x(t)) \right],$$

(53)

by integrating by parts (assuming $x(t_i) = x(t_f) = x$). To discretize we replace

$$\ddot{x}(t_j) \to \Delta^{(2)} x_j \equiv \frac{x_{j+1} - 2x_j + x_{j-1}}{a^2},$$

(54)

where the x_j's are periodic ($x_0 = x_N$, $x_{-1} = x_{N-1}$, and so on); this gives the same lattice action we used earlier. We can improve the discretization, however, by using the corrected approximation, Eq. (50), for the second derivative:

$$S_{\text{imp}}[x] \equiv \sum_{j=0}^{N-1} a \left[-\tfrac{1}{2} m\, x_j \left(\Delta^{(2)} - a^2 (\Delta^{(2)})^2/12 \right) x_j + aV(x_j) \right]$$

(55)

Modify your Monte Carlo code for the harmonic oscillator to include the correction term. Run high-statistics comparisons ($N_{\text{cf}} = 10^4$ or 10^5) with and without the correction term.

Exercise: Discretized, euclidean classical equations of motion can be derived from the actions in the previous exercise by setting

$$\frac{\partial S[x]}{\partial x_j} = 0.$$

(56)

Using the improved action, for example, we obtain

$$m \left(\Delta^{(2)} - a^2 (\Delta^{(2)})^2/12 \right) x_j = \frac{dV(x_j)}{dx_j}.$$

(57)

Setting

$$V(x) = \tfrac{1}{2} m\, \omega_0^2 x^2,$$

(58)

find solutions of the form $x_j = \exp(-\omega t_j)$, the euclidean-time version of an oscillatory solution with frequency ω. Show that the frequency is given by

$$\omega^2 = \omega_0^2 \left[1 - \frac{(a\omega_0)^2}{12} + \mathcal{O}((a\omega)^4) \right]$$

(59)

for the unimproved action. The $(a\omega_0)^2$ correction is the error caused by the finite lattice spacing.

Repeat the exercise for the improved action and show that it has two solutions. One,

$$\omega^2 = \omega_0^2 \left[1 + \frac{(a\omega_0)^4}{90} + \mathcal{O}((a\omega)^6) \right],$$
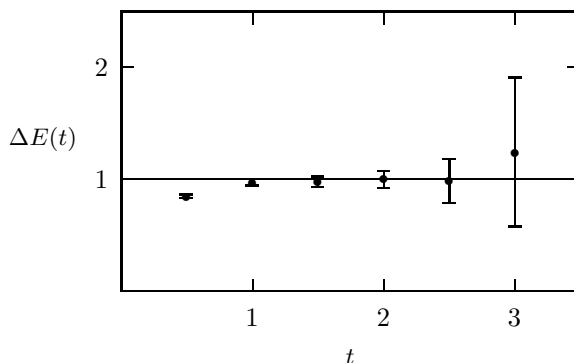
(60)

19

Figure 9: Monte Carlo values $\Delta E(t) \equiv \log(G(t)/G(t+a))/a$ plotted versus t for an harmonic oscillator, as in Fig. 4 but with an improved action. The energies approach their asymptotic value from below.

is an improved version of the previous result; its errors are fourth order in $a\omega_0$ rather than second order. The other solution, however, is

$$\omega^2 \approx \left(\frac{2.6}{a}\right)^2. \tag{61}$$

It corresponds to a new oscillation mode that does not appear in the continuum; it is an artifact of the improved lattice theory. This new mode is sometimes called a "numerical ghost." In a quantum field theory it would be a new, very massive particle ($m \propto 1/a$).

Our lattice theory was designed to be accurate for low-energies, and so we should not be surprised when unphysical modes appear at high energies. These ghost modes, being high-energy, typically decouple from low-energy physics and so can usually be ignored. However, they can have one unfortunate effect on the numerical analysis. Returning to the previous exercise, note that the ΔE_n's for the improved action are below the asymptotic result when n is small (see Fig. 9), in contradiction of the general result discussed in Section 2. The general result ignored the possibility that spurious states might be induced by the numerical analysis. Here these states have negative norm (impossible for real quantum states), which is why the energies rise from below. This is incovenient because it means that the ΔE_n's cannot be used to rigorously bound the true answer — they may be either above or below it — unlike the case for the unimproved action, where they must always be above.

Ghost modes always arise when improved discretizations are used for temporal derivatives. There is a trick, however, for correcting temporal difference oper-

20

ators that avoids extra states. This is to change integration variables in the path integral: for the harmonic oscillator we replace $x_j \to \tilde{x}_j$ where

$$x_j = \tilde{x}_j + \delta\tilde{x}_j \tag{62}$$

and

$$\delta\tilde{x}_j \equiv \xi_1 \, a^2 \Delta^{(2)} \tilde{x}_j + \xi_2 \, a^2 \omega_0^2 \tilde{x}_j. \tag{63}$$

Substituting this into the action we obtain

$$
\begin{aligned}
S[x] &= S[\tilde{x} + \delta\tilde{x}] \\
&= S[\tilde{x}] + \sum_j \delta\tilde{x}_j \frac{\partial S[\tilde{x}]}{\partial \tilde{x}_j} + \mathcal{O}(a^4) \\
&\equiv \tilde{S}[\tilde{x}]
\end{aligned}
\tag{64}
$$

Find values for the ξ_i such that the improved harmonic-oscillator action, in terms of $\tilde{x}$, is

$$\tilde{S}_{\mathrm{imp}}[\tilde{x}] = \sum_{j=0}^{N-1} a \left[-\tfrac{1}{2} m \, \tilde{x}_j \Delta^{(2)} \tilde{x}_j + \tilde{V}_{\mathrm{imp}}(x_j) \right] \tag{65}$$

where

$$\tilde{V}_{\mathrm{imp}}(\tilde{x}) \equiv \tfrac{1}{2} m \omega_0^2 \tilde{x}_j^2 \left(1 + \frac{(a\omega_0)^2}{12} \right) \tag{66}$$

and all $\mathcal{O}(a^4)$ terms are ignored. This action has no a^2 errors, but also has no ghosts. The value of the path integral is not changed by a simple change of variables (provided that the jacobian is included — show that in this case it has no effect on expectation values). Rerun your numerical tests from the last exercise using this action.

Exercise: The field transformation trick in the previous exercise is particularly simple for the harmonic oscillator. Generalize the trick for the case of an anharmonic oscillator with, for example,

$$V(x) = \tfrac{1}{2} m \, \omega_0^2 x^2 (1 + cm\omega_0 x^2). \tag{67}$$

where c is a dimensionless parameter. Try a variable change with

$$\delta\tilde{x}_j \equiv \xi_1 \, a^2 \Delta^{(2)} \tilde{x}_j + \xi_2 \, a^2 \omega_0^2 \tilde{x}_j + \xi_3 \, a^2 m\omega_0^3 \tilde{x}_j^3. \tag{68}$$

The resulting action is as above but with a new $\tilde{V}_{\mathrm{imp}}$:

$$
\begin{aligned}
\tilde{V}_{\mathrm{imp}}(\tilde{x}) &= \frac{m\omega_0^2}{2} \tilde{x}^2 (1 + cm\omega_0 \tilde{x}^2) + \frac{a^2 m\omega_0^4}{24} \left(\tilde{x} + 2cm\omega_0 \tilde{x}^3 \right)^2 \\
&\quad - a\delta v(\tilde{x}) + \frac{a^3}{2} \delta v(\tilde{x})^2
\end{aligned}
\tag{69}
$$

21

where the terms involving

$$\delta v(\tilde{x}) \equiv cm\omega_0^3 \tilde{x}^2/4 \tag{70}$$

are due to the jacobian (from the change of integration variables; the jacobian matters here because the change is nonlinear). Test this improved lattice action against the original unimproved action using $m = \omega_0 = 1$ and $c = 2$. (The asymptotic value for ΔE is 1.933 with these parameters.)

One might expect errors of order a^4 with the improved action. However renormalization effects arise when interactions are anharmonic. In particular the coefficients of the x^2 and x^4 interactions are renormalized away from their naive values. Such effects enter at the same order as the corrections from the jacobian. In our one-dimensional theory, unlike in QCD, these corrections vanish like powers of a when $a \to 0$. The leading such correction is an $\mathcal{O}(a)$ shift in the coefficient of the x^2 potential. Run high-precision simulations at $a = 1/2$ and $a = 1/4$ to compute the $\mathcal{O}(a)$ error due to renormalization. Try adjusting the coefficient of x^2 to remove this error. (Alternatively one could try using perturbation theory to compute the shift needed to eliminate the error.)

3.3 *Perturbation Theory and Tadpole Improvement*[a]

Improved discretizations and large lattice spacings are old ideas, pioneered by Wilson, Symanzik and others[7]. However, perturbation theory is essential; the lattice spacing a must be small enough so that $p \approx \pi/a$ QCD is perturbative. This was the requirement that drove lattice QCD towards very costly simulations with tiny lattice spacings. Traditional perturbation theory for lattice QCD begins to fail at distances of order $1/20$ to $1/10$ fm, and therefore lattice spacings must be at least this small before improved actions are useful. This seems very odd since phenomenological applications of continuum perturbative QCD suggest that perturbation theory works well down to energies of order $1\,\text{GeV}$, which corresponds to a lattice spacing of $0.6\,\text{fm}$. The breakthrough, in the early 1990's, was the discovery of a trivial modification of lattice QCD, called "tadpole improvement," that allows perturbation theory to work even at distances as large as $1/2\,\text{fm}$[6,8,9].

One can readily derive Feynman diagram rules for lattice QCD using the same techniques as in the continuum, but applied to the lattice lagrangian[11]. The particle propagators and interaction vertices are usually complicated functions of the momenta that become identical to their continuum analogues in the low-momentum limit. All loop momenta are cut off at $p_\mu = \pm\pi/a$.

Testing perturbation theory is also straightforward. One designs short-distance quantities that can be computed easily in a simulation (i.e., in a

Monte Carlo evaluation of the lattice path integral). The Monte Carlo gives the exact value which can then be compared with the perturbative expansion for the same quantity. An example of such a quantity is the expectation value of the Wilson loop operator,

$$W(\mathcal{C}) \equiv \langle 0|\tfrac{1}{3}\operatorname{Re}\operatorname{Tr}\mathcal{P}e^{-ig\oint_{\mathcal{C}}A\cdot dx}|0\rangle, \tag{71}$$

where A is the QCD vector potential, $\mathcal{P}$ denotes path ordering, and $\mathcal{C}$ is any small, closed path or loop on the lattice. $W(\mathcal{C})$ is perturbative for sufficiently small loops $\mathcal{C}$. We can test the utility of perturbation theory over any range of distances by varying the loop size while comparing numerical Monte Carlo results for $W(\mathcal{C})$ with perturbation theory.

Fig. 10 illustrates the highly unsatisfactory state of traditional lattice-QCD perturbation theory. It shows the "Creutz ratio" of $2a \times 2a$, $2a \times a$ and $a \times a$ Wilson loops,

$$\chi_{2,2} \equiv -\ln\left(\frac{W(2a \times 2a)\,W(a \times a)}{W^2(2a \times a)}\right), \tag{72}$$

plotted versus the size $2a$ of the largest loop. Traditional perturbation theory (dotted lines) underestimates the exact result by factors of three or four for loops of order $1/2\,\mathrm{fm}$; only when the loops are smaller than $1/20\,\mathrm{fm}$ does perturbation theory begin to give accurate results.

The problem with traditional lattice-QCD perturbation theory is that the coupling it uses is much too small. The standard practice was to express perturbative expansions of short-distance lattice quantities in terms of the bare coupling α_{lat} used in the lattice lagrangian. This practice followed from the notion that the bare coupling in a cutoff theory is approximately equal to the running coupling evaluated at the cutoff scale, here $\alpha_s(\pi/a)$, and therefore that it is the appropriate coupling for quantities dominated by momenta near the cutoff. In fact the bare coupling in traditional lattice QCD is much smaller than true effective coupling at large lattice spacings: for example,

$$\begin{aligned}\alpha_{\mathrm{lat}} &= \alpha_V(\pi/a) - 4.7\,\alpha_V^2 + \cdots \tag{73}\\ &\leq \tfrac{1}{2}\alpha_V(\pi/a) \qquad \text{for } a > .1\,\mathrm{fm} \tag{74}\end{aligned}$$

where $\alpha_V(q)$ is a continuum coupling defined by the static-quark potential,

$$V_{\mathrm{Q}\overline{\mathrm{Q}}}(q) \equiv -4\,\pi\,C_{\mathrm{F}}\frac{\alpha_V(q)}{q^2}. \tag{75}$$

Consequently α_{lat} expansions, though formally correct, badly underestimate perturbative effects, and converge poorly.
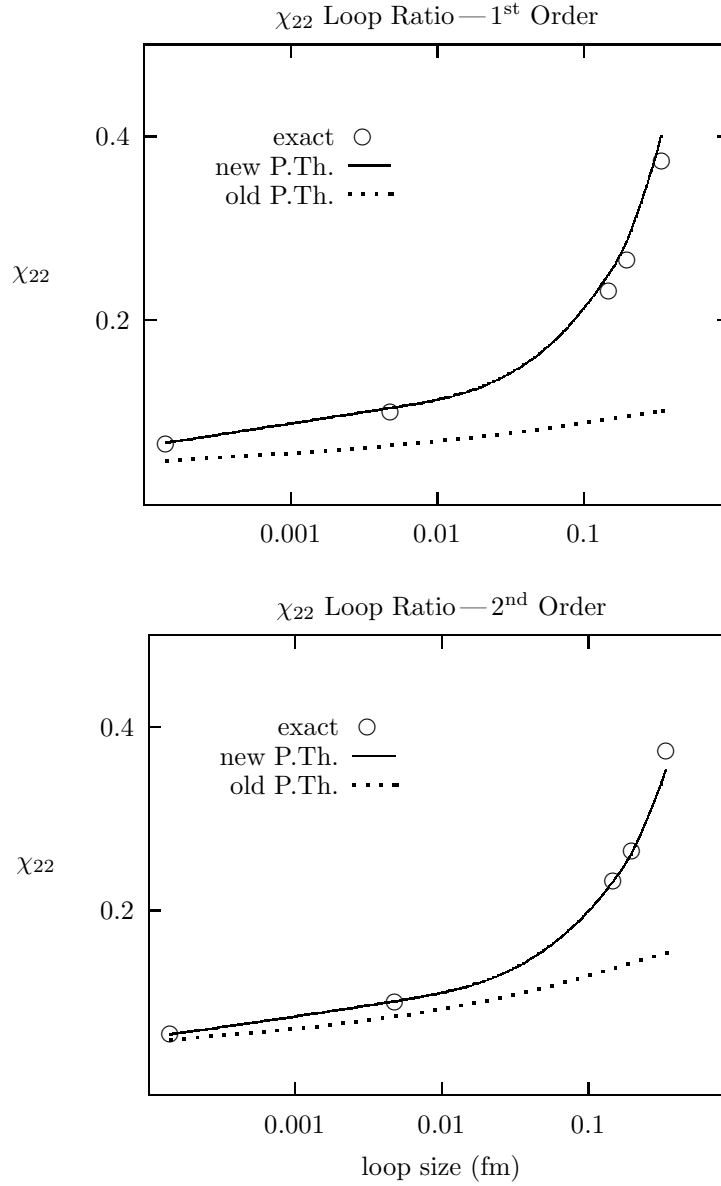
Figure 10: The χ_{22} Creutz ratio of Wilson loops versus loop size. Results from Monte Carlo simulations (exact), and from tadpole-improved (new) and traditional (old) lattice perturbation theory are shown.

24

The anomalously small bare coupling in the traditional lattice theory is a symptom of the "tadpole problem". As we discuss later, all gluonic operators in lattice QCD are built from the link operator

$$U_\mu(x) \equiv \mathcal{P}\mathrm{e}^{-i\int_x^{x+a\hat{\mu}} gA\cdot dx} \approx \mathrm{e}^{-iagA_\mu} \tag{76}$$

rather than from the vector potential A_μ. Thus, for example, the leading term in the lagrangian that couples quarks and gluons is $\overline{\psi}U_\mu\gamma_\mu\psi/a$. Such a term contains the usual $\overline{\psi}gA\cdot\gamma\psi$ vertex, but, in addition, it contains vertices with any number of additional powers of agA_μ. These extra vertices are irrelevant for classical fields since they are suppressed by powers of the lattice spacing. For quantum fields, however, the situation is quite different since pairs of A_μ's, if contracted with each other, generate ultraviolet divergent factors of $1/a^2$ that precisely cancel the extra a's. Consequently the contributions generated by the extra vertices are suppressed by powers of g^2 (not a), and turn out to be uncomfortably large. These are the tadpole contributions.

The tadpoles result in large renormalizations — often as large as a factor of two or three — that spoil naive perturbation theory, and with it our intuition about the connection between lattice operators and the continuum. However tadpole contributions are generically process independent and so it is possible to measure their contribution in one quantity and then correct for them in all other quantities.

The simplest way to do this is to cancel them out. The mean value u_0 of $\frac{1}{3}\operatorname{Re}\operatorname{Tr}U_\mu$ consists of only tadpoles and so we can largely cancel the tadpole contributions by dividing every link operator by u_0. That is, in every lattice operator we replace

$$U_\mu(x) \to \frac{U_\mu(x)}{u_0} \tag{77}$$

where u_0 is computed numerically in a simulation.

The u_0's cancel tadpole contributions, making lattice operators and perturbation theory far more continuum-like in their behavior. Thus, for example, the only change in the standard gluon action when it is tadpole-improved is that the new bare coupling α_{TI} is enhanced by a factor of $1/u_0^4$ relative to the coupling α_{lat} in the unimproved theory:

$$\alpha_{\mathrm{TI}} = \frac{\alpha_{\mathrm{lat}}}{u_0^4}. \tag{78}$$

Since $u_0^4 < .6$ when $a > .1\,\mathrm{fm}$, the tadpole-improved coupling is typically more than twice as large for coarse lattices. Expressing α_{TI} in terms of the continuum

coupling α_V, we find that now our intuition is satisfied:

$$\alpha_{\text{TI}} = \alpha_V(\pi/a) - .5\,\alpha_V^2 + \cdots \qquad (79)$$
$$\approx \alpha_V(\pi/a). \qquad (80)$$

Perturbation theory for the Creutz ratio Eq. (72) converges rapidly to the correct answer when it is reexpressed in terms of α_{TI}. An even better result is obtained if the expansion is reexpressed as a series in a coupling constant defined in terms of a physical quantity, like the static-quark potential, where that coupling constant is measured in a simulation. By measuring the coupling we automatically include any large renormalizations of the coupling due to tadpoles. It is important that the scale q^* at which the running coupling constant is evaluated be chosen appropriately for the quantity being studied[6,8]. When these refinements are added, perturbation theory is dramatically improved, and, as illustrated in Fig. 10, is still quite accurate for loops as large as $1/2$ fm.

This same conclusion follows from Fig. 11 which shows the value of the bare quark mass needed to obtain zero-mass pions using Wilson's lattice action for quarks. This quantity diverges linearly as the lattice spacing vanishes, and so should be quite perturbative. Here we see dramatic improvements as the tadpoles are removed first from the gluon action, through use of an improved coupling, and then also from the quark action.

The Creutz ratio and the critical quark mass are both very similar to the couplings we need to compute for improved lagrangians. Tadpole improvement has been very successful in a wide range of applications.

Asymptotic freedom implies that short-distance QCD is simple (perturbative) while long-distance QCD is difficult (nonperturbative). The lattice separates short from long distances, allowing us to exploit this dichotomy to create highly efficient algorithms for solving the entire theory: $p > \pi/a$ QCD is included via corrections $\delta\mathcal{L}$ to the lattice lagrangian that are computed using perturbation theory; $p < \pi/a$ QCD is handled nonperturbatively using Monte Carlo integration. Thus, while we wish to make the lattice spacing a as large as possible, we are constrained by two requirements. First a must be sufficiently small that our finite-difference approximations for derivatives in the lagrangian and field equations are sufficiently accurate. Second a must be sufficiently small that π/a is a perturbative momentum. Numerical experiments indicate that both constraints can be satisfied when $a \approx 1/2$ fm or smaller, provided all lattice operators are tadpole improved.
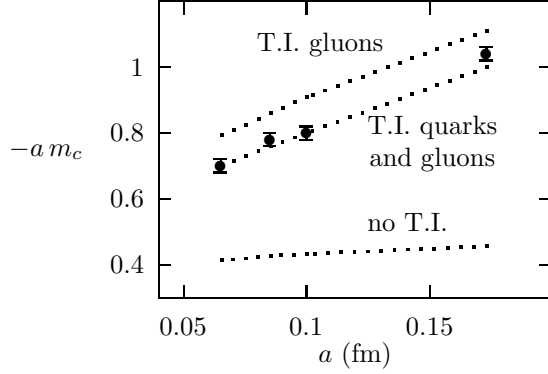
Figure 11: The critical bare quark mass for Wilson's lattice quark action versus lattice spacing. Monte Carlo data points are compared with perturbation theories in a theory with no tadpole improvement (T.I), tadpole-improved gluon dynamics, and tadpole-improved quark and gluon dynamics.

4   QCD on a Lattice

4.1   Classical Gluons[10]

The continuum action for QCD is

$$S = \int d^4x \, \tfrac{1}{2} \sum_{\mu,\nu} \mathrm{Tr}\, F_{\mu\nu}^2(x) \tag{81}$$

where

$$F_{\mu\nu} \equiv \partial_\mu A_\nu - \partial_\nu A_\mu + ig[A_\mu, A_\nu] \tag{82}$$

is the field tensor, a traceless $3{\times}3$ hermitian matrix. The defining characteristic of the theory is its invariance with respect to gauge transformations where

$$F_{\mu\nu} \to \Omega(x)\, F_{\mu\nu}\, \Omega(x)^\dagger \tag{83}$$

and $\Omega(x)$ is an arbitrary x-dependent SU_3 matrix.

The standard discretization of this theory seems perverse at first sight. Rather than specifying the gauge field by the values of $A_\mu(x)$ at the sites of the lattice, the field is specified by variables on the links joining the sites. In the classical theory, the "link variable" on the link joining a site at x to one at $x + a\hat{\mu}$ is determined by the line integral of A_μ along the link:

$$U_\mu(x) \equiv \mathcal{P} \exp\left(-i \int_x^{x+a\hat{\mu}} gA \cdot dy\right) \tag{84}$$

27

where the $\mathcal{P}$-operator path-orders the A_μ's along the integration path. We use U_μ's in place of A_μ's on the lattice, because it is impossible to formulate a lattice version of QCD directly in terms of A_μ's that has exact gauge invariance. The U_μ's, on the other hand, are SU_3 matrices that transform very simply under a gauge transformation:

$$U_\mu(x) \to \Omega(x)\, U_\mu(x)\, \Omega(x + a\hat{\mu})^\dagger. \tag{85}$$

This makes it easy to build a discrete theory with exact gauge invariance.

A link variable $U_\mu(x)$ is represented pictorially by a directed line from x to $x + \hat{\mu}$, where this line is the integration path for the line integral in the exponent of $U_\mu(x)$:
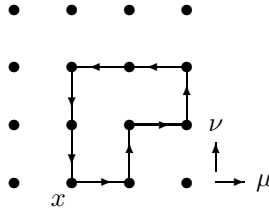

$U_\mu(x)$
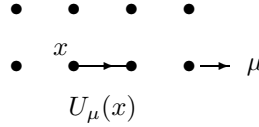
In the conjugate matrix $U_\mu^\dagger(x)$ the direction of the line integral is flipped and so we represent $U_\mu^\dagger(x)$ by a line going backwards from $x + \hat{\mu}$ to x:

$U_\mu^\dagger(x)$
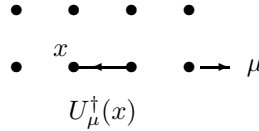
A Wilson loop function,

$$W(\mathcal{C}) \equiv \tfrac{1}{3}\,\mathrm{Tr}\,\mathcal{P}\mathrm{e}^{-i \oint_{\mathcal{C}} gA\cdot dx}, \tag{86}$$

for any closed path $\mathcal{C}$ built of links on the lattice can be computed from the path-ordered product of the U_μ's and $U_\mu^\dagger$'s associated with each link. For example, if $\mathcal{C}$ is the loop

then

$$W(\mathcal{C}) = \tfrac{1}{3}\,\mathrm{Tr}\left(U_\mu(x) U_\nu(x + a\hat{\mu}) \dots U_\nu^\dagger(x) \right). \tag{87}$$

28

Such quantities are obviously invariant under arbitrary gauge transformations Eq. (85).

One might wonder why we go to so much trouble to preserve gauge invariance when we quite willing give up Lorentz invariance, rotation invariance, etc. The reason is quite practical. With gauge invariance, the quark-gluon, three-gluon, and four-gluon couplings in QCD are all equal, and the bare gluon mass is zero. Without gauge invariance, each of these couplings must be tuned independently and a gluon mass introduced if one is to recover QCD. Tuning this many parameters in a numerical simulation is very expensive. This is not much of a problem in the classical theory, where approximate gauge invariance keeps the couplings approximately equal; but it is serious in the quantum theory because quantum fluctuations (loop-effects) renormalize the various couplings differently in the absence of exact gauge invariance. So while it is quite possible to formulate lattice QCD directly in terms of A_μ's, the resulting theory would have only approximate gauge invariance, and thus would be prohibitively expensive to simulate. Symmetries like Lorentz invariance can be given up with little cost because the symmetries of the lattice, though far less restrictive, are still sufficient to prevent the introduction of new interactions with new couplings (at least to lowest order in a).

We must now build a lattice lagrangian from the link operators. We require that the lagrangian be gauge invariant, local, and symmetric with respect to axis interchanges (which is all that is left of Lorentz invariance). The most local nontrivial gauge invariant object one can build from the link operators is the "plaquette operator," which involves the product of link variables around the smallest square at site x in the $\mu\nu$ plane:

$$P_{\mu\nu}(x) \equiv \tfrac{1}{3} \operatorname{Re} \operatorname{Tr} \left(U_\mu(x) U_\nu(x + a\hat{\mu}) U_\mu^\dagger(x + a\hat{\mu} + a\hat{\nu}) U_\nu^\dagger(x) \right). \qquad (88)$$

To see what this object is, consider evaluating the plaquette centered about a point x_0 for a very smooth weak classical A_μ field. In this limit,

$$P_{\mu\nu} \approx 1 \qquad (89)$$

since

$$U_\mu \approx \mathrm{e}^{-igaA_\mu} \approx 1. \qquad (90)$$

Given that A_μ is slowly varying, its value anywhere on the plaquette should be accurately specified by its value and derivatives at x_0. Thus the corrections to Eq. (89) should be a polynomial in a with coefficients formed from gauge-invariant combinations of $A_\mu(x_0)$ and its derivatives: that is,

$$P_{\mu\nu} = 1 \quad - \quad c_1 a^4 \operatorname{Tr} \left(g F_{\mu\nu}(x_0) \right)^2$$

29

$$
\begin{aligned}
&- \quad c_2\, a^6\, \mathrm{Tr}\left(gF_{\mu\nu}(x_0)(\mathrm{D}_\mu^2 + \mathrm{D}_\nu^2)gF_{\mu\nu}(x_0)\right) \\
&+ \quad \mathcal{O}(a^8) \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad (91)
\end{aligned}
$$

where c_1 and c_2 are constants, and D_μ is the gauge-covariant derivative. The leading correction is order a^4 because $F_{\mu\nu}^2$ is the lowest-dimension gauge-invariant combination of derivatives of A_μ, and it has dimension 4. (There are no F^3 terms because $P_{\mu\nu}$ is invariant under $U_\mu \to U_\mu^\dagger$ or, equivalently, $F \to -F$.)

It is straightforward to find the coefficients c_1 and c_2. We need only examine terms in the expansion of $P_{\mu\nu}$ that are quadratic in A_μ; the cubic and quartic parts of $F_{\mu\mu}^2$ then follow automatically, by gauge invariance. Because of the trace, the path ordering is irrelevant to this order. Thus

$$
\begin{aligned}
P_{\mu\nu} &= \tfrac{1}{3}\,\mathrm{Re}\,\mathrm{Tr}\,\mathcal{P}\mathrm{e}^{-i\oint_\square gA\cdot dx} \\
&= \tfrac{1}{3}\,\mathrm{Re}\,\mathrm{Tr}\left[1 - i\oint_\square gA\cdot dx - \tfrac{1}{2}\left(\oint_\square gA\cdot dx\right)^2 + \mathcal{O}(A^3)\right] \quad (92)
\end{aligned}
$$

where, by Stoke's Theorem,

$$
\begin{aligned}
\oint_\square A\cdot dx &= \int_{-a/2}^{a/2} dx_\mu dx_\nu\,[\partial_\mu A_\nu(x_0 + x) - \partial_\nu A_\mu(x_0 + x)] \\
&= \int_{-a/2}^{a/2} dx_\mu dx_\nu\,[F_{\mu\nu}(x_0) + (x_\mu\mathrm{D}_\mu + x_\nu\mathrm{D}_\nu)F_{\mu\nu}(x_0) + \cdots] \\
&= a^2\,F_{\mu\nu}(x_0) + \frac{a^4}{24}(\mathrm{D}_\mu^2 + \mathrm{D}_\nu^2)F_{\mu\nu}(x_0) + \mathcal{O}(a^6, A^2). \qquad (93)
\end{aligned}
$$

Thus $c_1 = 1/6$ and $c_2 = 1/72$ in Eq. (91).

The expansion in Eq. (91) is the classical analogue of an operator product expansion. Using this expansion, we find that the traditional "Wilson action" for gluons on a lattice,

$$
S_{\mathrm{Wil}} = \beta\sum_{x,\mu>\nu}(1 - P_{\mu\nu}(x)) \qquad\qquad\qquad (94)
$$

where $\beta = 6/g^2$, has the correct limit for small lattice spacing up to corrections of order a^2:

$$
S_{\mathrm{Wil}} = \int d^4x \sum_{\mu,\nu}\left\{\tfrac{1}{2}\,\mathrm{Tr}\,F_{\mu\nu}^2 + \frac{a^2}{24}\,\mathrm{Tr}\,F_{\mu\nu}(\mathrm{D}_\mu^2 + \mathrm{D}_\nu^2)F_{\mu\nu} + \cdots\right\}. \qquad (95)
$$

We can cancel the a^2 error in the Wilson action by adding other Wilson loops. For example, the $2a \times a$ "rectangle operator"

$$R_{\mu\nu} = \tfrac{1}{3} \operatorname{Re} \operatorname{Tr} \quad \begin{array}{c} \rule{0pt}{0pt} \end{array} \quad \begin{array}{c} \nu \\ \mu \end{array} \tag{96}$$

has expansion

$$R_{\mu\nu} = 1 - \frac{4}{6}\, a^4 \operatorname{Tr} (gF_{\mu\nu})^2 - \frac{4}{72}\, a^6 \operatorname{Tr} \left(gF_{\mu\nu}(4\,\mathrm{D}_\mu^2 + \mathrm{D}_\nu^2)gF_{\mu\nu}\right) - \cdots. \tag{97}$$

The mix of a^4 terms and a^6 terms in the rectangle is different from that in the plaquette. Therefore we can combine the two operators to obtain an improved classical lattice action that is accurate up to $\mathcal{O}(a^4)$[12,13]:

$$S_{\text{classical}} \equiv -\beta \sum_{x,\mu>\nu} \left\{ \frac{5P_{\mu\nu}}{3} - \frac{R_{\mu\nu} + R_{\nu\mu}}{12} \right\} + \text{const} \tag{98}$$

$$= \int d^4x \sum_{\mu,\nu} \tfrac{1}{2} \operatorname{Tr} F_{\mu\nu}^2 + \mathcal{O}(a^4). \tag{99}$$

This process is the analogue of improving the derivatives in discretizations of non-gauge theories.[b]

Exercise: Defining the "twisted-rectangle operator"

$$T_{\mu\nu} = \tfrac{1}{3} \operatorname{Re} \operatorname{Tr} \quad \begin{array}{c} \rule{0pt}{0pt} \end{array} \quad . \tag{100}$$

show that

$$S_{\text{classical}} \equiv -\beta \sum_{x,\mu>\nu} \left\{ P_{\mu\nu} + \frac{T_{\mu\nu} + T_{\nu\mu}}{12} \right\} + \text{const} \tag{101}$$

$$= \int d^4x \sum_{\mu,\nu} \tfrac{1}{2} \operatorname{Tr} F_{\mu\nu}^2 + \mathcal{O}(a^4). \tag{102}$$

This is an alternative to the improved gluon action derived in the previous exercise.

---

[b] An important step that I have not discussed is to show that the gluon action is positive for any configuration of link variables[13]. This guarantees that the classical ground state of the lattice action corresponds to $F_{\mu\nu} = 0$.

4.2 Quantum Gluons[c]

In the previous section we derived improved classical actions for gluons that are accurate through order a^4. We now turn these into quantum actions. The most important step is to tadpole improve the action by dividing each link operator U_μ by the mean link u_0: for example, the action built of plaquette and rectangle operators becomes

$$S = -\beta \sum_{x,\mu>\nu} \left\{ \frac{5}{3} \frac{P_{\mu\nu}}{u_0^4} - \frac{R_{\mu\nu} + R_{\nu\mu}}{12\, u_0^6} \right\}. \tag{103}$$

The u_0's cancel lattice tadpole contributions that otherwise would spoil weak-coupling perturbation theory in the lattice theory and undermine our procedure for improving the lattice discretization. Note that $u_0 \approx 3/4$ when $a = .4\,\mathrm{fm}$, and therefore the relative importance of the $R_{\mu\nu}$'s is larger by a factor of $1/u_0^2 \approx 2$ than without tadpole improvement. Without tadpole improvement, we cancel only half of the a^2 error.

The mean link u_0 is computed numerically by guessing a value for use in the action, measuring the mean link in a simulation, and then readjusting the value used in the action accordingly. This tuning cycle converges rapidly to selfconsistent values, and can be done very quickly using small lattice volumes. The u_0's depend only on lattice spacing, and become equal to one as the lattice spacing vanishes.

The expectation value of the link operator is gauge dependent. Thus to minimize gauge artifacts, u_0 is commonly defined as the Landau-gauge expectation value, $\langle 0|\frac{1}{3}\,\mathrm{Tr}\,U_\mu|0\rangle_{\mathrm{LG}}$. Landau gauge is the axis-symmetric gauge that maximizes u_0, thereby minimizing the tadpole contribution; any tadpole contribution that is left in Landau gauge cannot be a gauge artifact. An alternative procedure is to define u_0 as the fourth root of the plaquette expectation value,

$$u_0 = \langle 0|P_{\mu\nu}|0\rangle^{1/4}. \tag{104}$$

This definition gives almost identical results and is more convenient for numerical work since gauge fixing is unnecessary.

Tadpole improvement is the first step in a systematic procedure for improving the action. The next step is to add in renormalizations due to contributions from $k > \pi/a$ physics not already included in the tadpole improvement. These renormalizations induce $a^2 \alpha_s(\pi/a)$ corrections,

$$\delta\mathcal{L} \;=\; \alpha_s\, r_1\, a^2 \sum_{\mu,\nu} \mathrm{Tr}(F_{\mu\nu} \mathrm{D}_\mu^2 F_{\mu\nu})$$

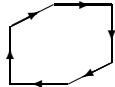[c]This section is based on work with M. Alford, W. Dimm, G. Hockney and P. Mackenzie that is described in [14].

$$+ \quad \alpha_s\, r_2\, a^2 \sum_{\mu,\nu} \text{Tr}(\text{D}_\mu F_{\nu\sigma}\text{D}_\mu F_{\nu\sigma})$$

$$+ \quad \alpha_s\, r_3\, a^2 \sum_{\mu,\nu} \text{Tr}(\text{D}_\mu F_{\mu\sigma}\text{D}_\nu F_{\nu\sigma})$$

$$+ \quad \cdots, \tag{105}$$

that must be removed. The last term is harmless; its coefficient can be set to zero by a change of field variable (in the path integral) of the form

$$A_\mu \rightarrow A_\mu + a^2\, \alpha_s\, f(\alpha_s) \sum_\nu \text{D}_\nu F_{\nu\mu}. \tag{106}$$

Since changing integration variables does not change the value of an integral, such field transformations must leave the physics unchanged.[d] Operators that can be removed by a field transformation are called "redundant." The other corrections are removed by renormalizing the coefficient of the rectangle operator $R_{\mu\nu}$ in the action, and by adding an additional operator. One choice for the extra operator is

$$C_{\mu\nu\sigma} \equiv \tfrac{1}{3}\,\text{Re Tr}\ \ \ \ . \tag{107}$$

Then the action, correct up to $\mathcal{O}(a^2\alpha_s^2, a^4)$, is[15]

$$S = -\beta \sum_{x,\mu>\nu} \left\{ \frac{5}{3}\frac{P_{\mu\nu}}{u_0^4} - r_\text{g}\,\frac{R_{\mu\nu}+R_{\nu\mu}}{12\,u_0^6} \right\} + c_\text{g}\,\beta \sum_{x,\mu>\nu>\sigma} \frac{C_{\mu\nu\sigma}}{u_0^6}, \tag{108}$$

where

$$r_\text{g} \quad = \quad 1 + .48\,\alpha_s(\pi/a) \tag{109}$$

$$c_\text{g} \quad = \quad .055\,\alpha_s(\pi/a). \tag{110}$$

The coefficients r_g and c_g are computed by "matching" physical quantities, like low-energy scattering amplitudes, computed using perturbation theory in the lattice theory with the analogous quantity in the continuum theory. The lattice result depends upon r_g and c_g; these parameters are tuned until the lattice amplitude agrees with the continuum amplitude to the order in a and α_s required:

$$T_\text{lat}(r_\text{g},c_\text{g}) = T_\text{contin}. \tag{111}$$

---

[d]One must, of course, include the jacobian for the transformation in the transformed path integral. This contributes only in one-loop order and higher; it has no effect on tree-level calculations.

Note that tadpole improvement has a big effect on these coefficients. Without tadpole improvement, $r_{\mathrm{g}} = 1 + 2\alpha_s$; that is, the coefficient of the radiative correction is four times larger. Tadpole improvement automatically supplies 75% of the one-loop contribution needed without improvement. Since $\alpha_s \approx 0.3$, the unimproved expansion for r_{g} is not particularly convergent. However, with tadpole improvement, the one-loop correction is only about 10–20% of r_{g}. Indeed, for most current applications, one-loop corrections to tadpole-improved actions are negligible.

Exercise: Show that the gauge that maximizes $\langle 0| \frac{1}{3}\,\mathrm{Tr}\,U_\mu |0\rangle$ becomes Landau gauge ($\partial \cdot A = 0$) in the $a \to 0$ limit.

4.3   Monte Carlo Evaluation of Gluonic Path Integrals

A computer code for the Monte Carlo evaluation of gluonic path integrals can be designed in close analogy with our code for one-dimensional quantum mechanics. The Metropolis algorithm for generating random configurations must be adapted to work with SU_3 matrices. In our quantum mechanics example, the coordinate was updated by adding a random number. In QCD the gluon field is specified by link variables $U_\mu(x)$, which are exponentials of the fundamental field. Thus to update a link variable we must multiply by the exponential of a random field; that is we must multiply by a random SU_3 matrix M:

$$U_\mu \to M\,U_\mu \tag{112}$$

Typically the matrix M is chosen randomly from a set of 50 or 100 random SU_3 matrices that is generated once, at the start of the simulation; the only restrictions on this set are that it be large enough so that products of the various M's cover the entire space of SU_3 matrices, and that the inverse, $M^\dagger$, for each matrix M in the set also be included in the set. The M's can be generated by first creating a set of hermitian matrices H whose matrix elements are random numbers between -1 and 1. These are converted to SU_3 matrices by forming $1 + i\epsilon H$ and unitarizing it.[e] Parameter ϵ determines the size of the update; as before, it is adjusted so that roughly half of all trial updates are accepted.

A second modification of the Metropolis algorithm that is useful for QCD is to update each link variable several times (rather than just once) before moving on to the next variable in a single sweep through the lattice. This allows the

---

[e] To convert an arbitrary matrix $M = (m_1\, m_2\, m_3)$ into an SU_3 matrix: first normalize the first column m_1 to unity, then make the second column m_2 orthogonal to the (new) first column and normalize it, and replace the third column by the cross product of the (new) first two columns.

link variable to come into statistical equilibrium with its immediate neighbors on the lattice. The additional cost for these extra updates is relatively small because standard gluon actions are linear in the link variables. Thus the part of the action that must be computed when updating a particular $U_\mu(x)$ can be written

$$\Delta S(x,\mu) = \text{ Re Tr} \left(U_\mu(x) \Gamma_\mu(x) \right) \tag{113}$$

where $\Gamma_\mu(x)$, which is a sum of products of the link variables, is independent of $U_\mu(x)$. Computing $\Gamma_\mu(x)$ is the most expensive part of the Metropolis update, but it need be computed only once for each set of successive updates of $U_\mu(x)$. Typically one does about 10 "hits" of the Metropolis algorithm before moving on the the next link variable.

Exercise: Design a computer code for evaluating gluonic path integrals using the Metropolis algorithm. Do this first for the simplest lattice action, the Wilson action (Eq. (94)):

$$S_{\text{Wil}} = -\tilde{\beta} \sum_{x,\mu>\nu} \frac{P_{\mu\nu}(x)}{u_0^4} \tag{114}$$

Run simulations at $\beta \equiv \tilde{\beta}/u_0^4 = 5.5$, which corresponds to a lattice spacing of around 0.25 fm. The lattice volume should be of order 2 fm on a side for typical QCD simulations; use $L/a = 8$ points on a side in your simulation. Set the Metropolis step size $\epsilon = 0.24$ and omit $N_{\text{cor}} = 50$ sweeps between Monte Carlo measurements. Compute Monte Carlo averages of $a \times a$ and $a \times 2a$ Wilson loops; you should obtain about 0.50 and 0.26, respectively.

Also try the improved action, Eq. (103). Use $\beta = 1.719$ and $u_0 = 0.797$ to again obtain $a \approx 0.25\,\text{fm}$[16]. The $a \times a$ and $a \times 2a$ Wilson loops have values of 0.54 and 0.28, respectively. (Wilson loops are unrenormalized and so these values need not agree with those from the Wilson action.)

4.4 A First Simulation

Perhaps the simplest physics calculation that involves just gluons is to compute the potential energy between a static quark and a static antiquark separated by a distance r. This "static potential" should be approximately Coulombic at short distances, but grow linearly at large distances, demonstrating quark confinement. It can be used in a Schrödinger equation to predict energy levels for the ψ/J and Υ families of mesons.

The euclidean Green's function or propagator G for a heavy nonrelativistic quark in a background gauge field A_μ satisfies the equation

$$\left(\text{D}_t - \frac{\mathbf{D}^2}{2M} \right) G(x) = \delta^4(x) \tag{115}$$

35

where $D_\mu = \partial_\mu - igA_\mu(x)$ is the gauge-covariant derivative. This equation is easily solved in the static-quark limit, where $M \to \infty$, to obtain

$$G_\infty(\mathbf{x}, t) = \left[\mathcal{P} e^{-i \int_0^t gA_0(\mathbf{x},t)dt} \right]^\dagger \delta^3(\mathbf{x}), \qquad (116)$$

which on the lattice becomes

$$G_\infty(\mathbf{x}, t) = U_t^\dagger(\mathbf{x}, t-a)\, U_t^\dagger(\mathbf{x}, t-2a) \,\ldots\, U_t^\dagger(\mathbf{x}, 0). \qquad (117)$$

Propagation of a static antiquark is described by $G_\infty^\dagger$. Therefore we obtain the static potential $V(r)$, which is the energy of a static quark and antiquark a distance r apart, from expectation values of $r \times t$ Wilson loops:

$$W(r,t) \equiv \langle 0| \tfrac{1}{3}\,\mathrm{Tr}\;\;\;\;|0\rangle \qquad (118)$$

where for large t

$$W(r,t) \to \mathrm{const}\; e^{-V(r)\,t}. \qquad (119)$$

Thus, to calculate the static potential $V(r)$, we compute $W(r,t)$ for a variety of t's and then take the large-t limit

$$W(r,t)/W(r,t+a) \to a\,V(r). \qquad (120)$$

There is one modification of this procedure that greatly improves the results. This is to replace the spatial link matrices, in the r direction, by "smeared" link variables $\tilde{U}_\mu(x)$:

$$\tilde{U}_\mu(x) \equiv \left(1 + \epsilon a^2 \Delta^{(2)} \right)^n U_\mu(x) \qquad (121)$$

where

$$\Delta^{(2)} U_\mu(x) \equiv \sum_\rho \Delta_\rho^{(2)} U_\mu(x) \qquad (122)$$

and $\Delta_\rho^{(2)}$ is a discretized, gauge-covariant derivative:

$$\begin{aligned}
\Delta_\rho^{(2)} U_\mu(x) \;\equiv\; & \frac{1}{u_0^2\, a^2} \left(U_\rho(x)\, U_\mu(x+a\hat{\rho})\, U_\rho^\dagger(x+a\hat{\mu}) - 2\, u_0^2\, U_\mu(x) \right. \\
& \left. + U_\rho^\dagger(x-a\hat{\rho})\, U_\mu(x-a\hat{\rho})\, U_\rho(x-a\hat{\rho}+a\hat{\mu}) \right). \quad (123)
\end{aligned}$$

36

The low-momentum components of the smeared link variable are unchanged by the smearing, up to corrections of order a^2p^2; but at high momentum the smearing acts as an ultraviolet cutoff:

$$\left(1 + \epsilon a^2 \Delta^{(2)}\right)^n \rightarrow \left(1 - \epsilon a^2 p^2\right)^n \approx \mathrm{e}^{-\epsilon a^2 p^2 n}. \tag{124}$$

By suppressing high-momentum gluons in the initial and final states of our matrix element, we suppress contributions from gluonic excitations and thereby hasten the convergence of $W(r,t)/W(r,t+a)$ to its asymptotic value. This significantly reduces the Monte Carlo statistical errors since these are generally much smaller at smaller t's.

Exercise: Add measurement code for $W(r,t)$ to your gluon Monte Carlo program. Run this code for the parameter sets given above and extract the static potential. Try this with unsmeared spatial U_μ's on the ends of the Wilson loop, and again with smeared links (try $n = 4$ smearings with $\epsilon = 1/12$). In the smeared case asymptotic results appear at the first or second time step

My results for the Wilson and improved actions are shown in Fig. 12. Each run took about 3 hours on a 300 MHz personal computer (with 64 MB of RAM). The potential was computed for quarks separated along the lattice axes, as well as for separations along diagonal directions on the lattice (using nonplanar Wilson loops). The plots clearly show the linear rise of the potential at large r, which results in quark confinement; and careful examination shows the onset of Coulombic behavior at small r, which is due to asymptotic freedom. The Monte Carlo results are compared in the figure with global fits to the form

$$V(r) = \sigma r - b/r + c \tag{125}$$

where σ is the string tension. This parameterization works well for the range of r's we are considering (0.25 fm–1 fm). Note that the Monte Carlo data is not as smooth in the Wilson case as it is for the improved action. This is caused the $\mathcal{O}(a^2)$ errors in the Wilson action, coming from the second term in Eq. (95). This term breaks rotational invariance; it is the leading manifestation in the gluon dynamics of the cubic structure of our lattice. While the true potential is a function only of the magnitude of the quark separation, we expect differences on the lattice between potentials for separations that are along the grid axes and those for separations along diagonals. This is particularly apparent at $r = 3a$ where there are two Monte Carlo points, one for separation $(3a, 0, 0)$ and the other for $(2a, 2a, a)$. The two points are clearly separated on the plot for the Wilson action. The a^2 errors are removed from the improved action; the two $r = 3a$ points merge on the plot for that action. The difference $a(V(2a, 2a, a) - V(3a, 0, 0))$ is reduced from 0.065 ± 0.007 with the Wilson action to 0.003 ± 0.006 with the improved action in these particular simulations. The a^2 errors are larger at smaller r's, but more difficult to quantify there.
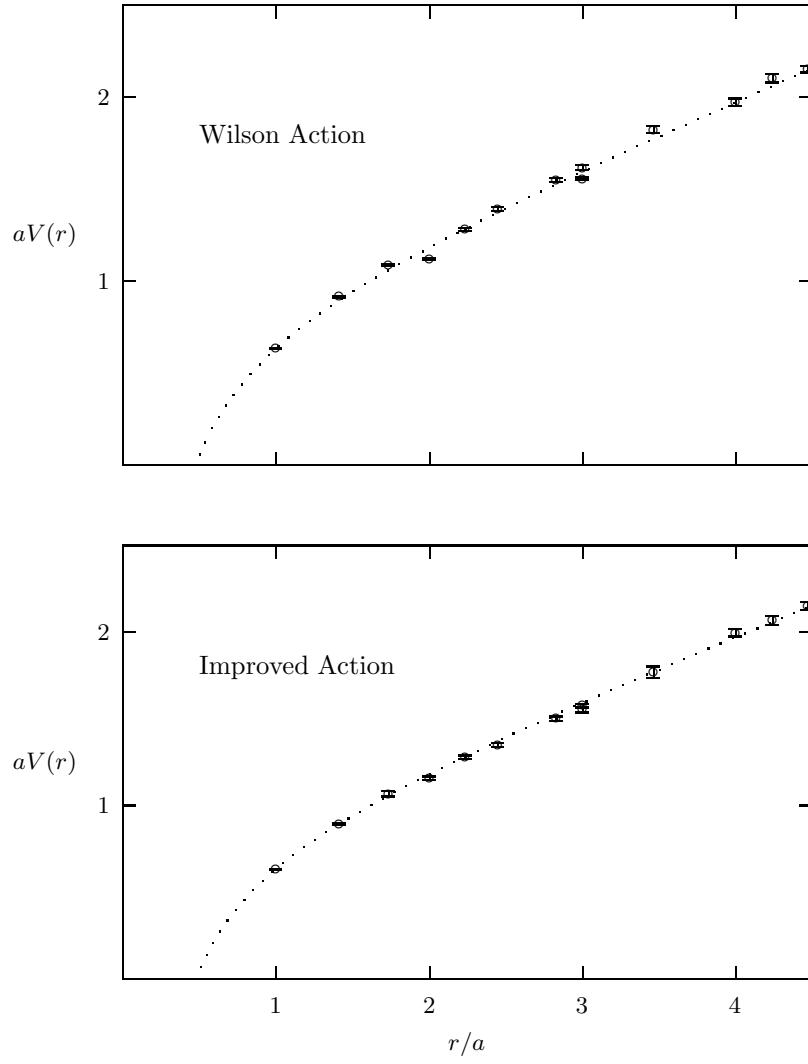
Figure 12: Static-quark potential computed using the Wilson gluon action and the $\mathcal{O}(a^2)$ improved action. The dotted line in each case is a fit of the Monte Carlo results to $\sigma r - b/r + c$. The lattice spacing $a \approx 0.25\,\mathrm{fm}$ in each case. Each plot required about 3 hours of computer time using a 300 MHz personal computer. The improved action gives a smoother curve.

38

5 Conclusions

The QCD simulation discussed in the previous section is, of course, just a beginning. A reasonable next step would be to add heavy quarks to the simulation, using nonrelativistic QCD, and to compute spectra and wavefunctions for ψ/J and Υ mesons[10,17,18]. Far more costly to simulate, although no more subtle theoretically, are light quarks[10,16]. These are needed to analyze protons, neutrons and other standard hadrons. Finally, and most costly, one might include light-quark vacuum polarization, which is an essential step for high-precision calculations (better than 10–15%).

Acknowledgements

Appendix

The sample simulation code in Section 2 is written in Python, a simple but powerful computer language that is freely available for just about any type of computer from the web site www.python.org. These simulations are most efficient when the Numeric package is used with Python; this is included with some Python distributions but must downloaded separately (from www.python.org) for others. To use the code given in the text, collect it together in a single file as follows:

```
import Numeric
from whrandom import uniform
from math import *

# ... code from text goes here

# set parameters:
N = 20
N_cor = 20
N_cf = 100
a = 0.5
eps = 1.4

# create arrays:
x = Numeric.zeros((N,),Numeric.Float)
G = Numeric.zeros((N_cf,N),Numeric.Float)
```

```
# do the simulation:
MCaverage(x,G)
```

If the file is called `simulation.py`, it is run with the command `python simulation.py`.
To test the binning and bootstrap codes add the following to the the the file:

```
def avg(G):                        # MC avg of G
    return Numeric.sum(G)/len(G)

def sdev(G):                       # std dev of G
    g = Numeric.asarray(G)
    return Numeric.absolute(avg(g**2)-avg(g)**2)**0.5

print 'avg G\n',avg(G)
print 'avg G (binned)\n',avg(bin(G,4))
print 'avg G (bootstrap)\n',avg(bootstrap(G))
```

The average of the binned copy of `G` should be the same as the average of `G`
itself; the average of the bootstrap copy should be different by an amount of
order the Monte Carlo error. Compute averages for several bootstrap copies
to get a good feel for the errors.

Finally one wants to extract energies. This is done by adding code to
compute $\Delta E(t)$:

```
def deltaE(G):                    # Delta E(t)
    avgG = avg(G)
    adE = Numeric.log(Numeric.absolute(avgG[:-1]/avgG[1:]))
    return adE/a

print 'Delta E\n',deltaE(G)
print 'Delta E (bootstrap)\n',deltaE(bootstrap(G))
```

Again repeating the evaluation for 50 or 100 bootstrap copies of `G` gives an
estimate of the statistical errors in the energies. Additional code can be added
to evaluate standard deviations from these copies:

```
def bootstrap_deltaE(G,nbstrap=100): # Delta E + errors
    avgE = deltaE(G)              # avg deltaE
    bsE = []
```

```
    for i in range(nbstrap):    # bs copies of deltaE
        g = bootstrap(G)
        bsE.append(deltaE(g))
    bsE = Numeric.array(bsE)
    sdevE = sdev(bsE)           # spread of deltaE's
    print "\n%2s  %10s  %10s" % ("t","Delta E(t)","error")
    print 26*"-"
    for i in range(len(avgE)/2):
        print "%2d  %10g  %10g" % (i,avgE[i],sdevE[i])

bootstrap_deltaE(G)
```

The entire program should take only a few seconds to run on a modern personal computer.

This example almost completely ignores the powerful object-oriented features that distinguish Python from most other scripting languages.

References

1. See for example Chapter 3 in G. Baym's *Lectures on Quantum Mechanics* (Benjamin/Cummings, Menlo Park, 1973).
2. G.P. Lepage, J. Comp. Phys. **27** (1978) 192.
3. An extensive discussion of the systematics of Monte Carlo errors can be found in: G.P. Lepage, *The Analysis of Algorithms for Lattice Field Theory*, in *From Actions to Answers*, edited by T. DeGrand and D. Toussaint (World Scientific, Singapore, 1989).
4. See for example M. Creutz, *Quarks, gluons, and lattices* (Cambridge University Press, Cambridge, 1985).
5. Download python from www.python.org. See also M. Lutz and D. Ascher, *Learning Python* (O'Reilly and Associates, 1999).
6. G.P. Lepage and P.B. Mackenzie, *Phys. Rev.* **D48** (1993) 2250.
7. For an overview see K.G. Wilson, *Rev. Mod. Phys.* **55** (1983) 583.
8. G.P. Lepage, *Lattice QCD for Small Computers*, in *The Building Blocks of Creation,* edited by S. Raby and T. Walker (World Scientific Press, Singapore, 1994).
9. G.P. Lepage, Nucl. Phys. Proc. Suppl. **60A** (1998) 267
10. Parts of Section 3 and 4 are based on the following article; it is more detailed and also discusses simulations of light and heavy quarks: G.P. Lepage, *Redesigning Lattice QCD*, in *Perturbative and Nonperturbative Aspects of Quantum Field Theory* (Lecture Notes in Physics, 479), edited

by H. Latal and W. Schweiger (Springer-Verlag, Berlin, 1997).

11. See, for example, H. Kawai, R. Nakayama and K. Seo, *Nucl. Phys.* **B189** (1981) 40.
12. G. Curci, P. Menotti, and G. Paffuti, *Phys. Lett.* **130B** (1983) 205; Erratum: *ibid.* **135B** (1984) 516.
13. M. Lüscher and P. Weisz, *Comm. Math. Phys.* **97** (1985) 59.
14. M. Alford, W. Dimm, G.P. Lepage, G. Hockney and P. Mackenzie, *Phys. Lett.* **B361** (1995) 87.
15. M. Lüscher and P. Weisz, *Phys. Lett.* **158B**, (1985) 250, and references therein.
16. See for example: M. Alford, T.R. Klassen and G.P. Lepage, Phys. Rev. **D58** (1998) 034503.
17. G.P. Lepage and B. Thacker, *Nucl. Phys.* **B**(Proc. Suppl.)**4** (1988) 199; G.P. Lepage, K. Hornbostel, L. Magnea, U. Magnea and C. Nakhleh, *Phys. Rev.* **D46** (1992) 4052.
18. C. Davies et al., *Phys. Rev.* **D50** (1994) 6963; P. McCallum and J. Shigemitsu, *Nucl. Phys.* **B47** (Proc. Suppl.) (1996) 409; C. Davies et al., *Phys. Rev. Lett.* **73** (1994) 2654; C. Davies et al., *Phys. Lett.* **B345** (1995) 42; J. Shigemitsu's talk at *Lattice '96* (June 1996, St. Louis).