Efficient Monte Carlo Simulation of Lattice QCD

Bálint Joó

Doctor of Philosophy

The University of Edinburgh

1999

I dedicate this thesis to my parents.

Abstract

This thesis is concerned with the efficient simulation of lattice QCD with dynamical fermions. We discuss two aspects of this theme, the tuning of existing algorithms and the investigation of novel algorithms. We begin with an introduction to lattice QCD and Monte Carlo Methods for its simulation. Particular emphasis is placed on the difficulties of the lattice formulation of fermion fields. We then continue with a description of the Hybrid Monte Carlo (HMC) algorithm, focusing on the conditions the algorithm must obey for correctness and on some of the numerical methods required for its implementation.

We then discuss issues of reversibility and instability for the Molecular Dynamics part of HMC algorithm. After considering the source of instabilities in the context of free field theory we adopt a working hypothesis by which we can relate this instability to the case of lattice QCD. Our tuning studies of HMC attempt to investigate the behaviour of reversibility violations and simulation cost in the molecular dynamics with varying solver target residue r. We also investigate the onset of instabilities in the molecular dynamics while varying the solver residue r and the stepsize $\delta\tau$.

Our second subject is the investigation of novel simulation algorithms. We consider the Parallel Tempering (PT) algorithm and its application to lattice QCD. We give an introduction to the algorithm and discuss the use of action matching technologies to tune the simulation parameters for maximal swap acceptance rates. We then discuss issues of cost for PT simulations by considering the CPU time needed by the algorithm for the estimation of the expectation value of an observable of interest and comparing this with the cost of reference HMC simulations. Finally we present some numerical results which indicate that we have a reasonable understanding of the algorithm but that we have not managed to maximise the acceptance rate through action matching. Due to large errors on our measured autocorrelation time we reserve judgement on the question of cost efficiency of the algorithm.

Declaration

This thesis has been wholly composed and written by me and contains details of research carried out by me as a member of the UKQCD collaboration.

The work on Parallel Tempering has been carried out in collaboration with Dr Alan Irving and Prof. James Sexton who were involved in the action matching part of the work and assisted in the study of autocorrelation times. The code to carry out the simulations in Parallel Tempering was written by me, using existing UKQCD GHMC Code written by Dr Stephen Pickles, Dr Zbigniew Sroczynski and Dr Stephen Booth. The Parallel Tempering simulations were carried out entirely by me. The code to carry out autocorrelation analysis and statistical analysis presented was also written by me.

The work on Hybrid Monte Carlo was carried out in collaboration with Prof Anthony Kennedy and Prof James Sexton. The simulations were carried out using the GHMC code and the data fitting analysis used components of the UKQCD Analysis code-base (the (in)famous HHHH code).

The results of chapters 3 and 4 of this thesis relating to instabilities, reversibility violation in the Molecular dynamics part of an HMC simulation were presented at the UK Theory institute held at Swansea, Wales, UK in August 1999. The results were also presented at VIELAT99, the 9th Workshop on Lattice Field Theory held at Vienna, Austria through 9-11 September 1999. A paper detailing these results is currently in preparation.

A summary of the results of the Parallel Tempering project were presented at Lattice '98, the XVI International Symposium on Lattice Field Theory, held in Boulder, Colorado, USA through 13-18 July 1998 and are published in the proceedings of this conference,

B. Joó, Nucl. Phys. B(Proc Suppl) 73 (1999) 877-879. (`hep-lat/9808052`)

A full report of our Parallel Tempering studies was also published:

B. Joó, A.C. Irving, J.C. Sexton, B. Pendleton, S.M. Pickles, Z. Sroczynski, Phys. Rev. D(59), 114501, 1999. (`hep-lat/9810032`)

Acknowledgements

I would like to acknowledge all the people who have helped and supported me throughout the Edinburgh Experience.

In particular I would like to thank my supervisors Dr Brian Pendleton and Prof Richard Kenway for giving me the support (both financial and academic) needed to carry out this PhD. I would also like to thank my collaborators Dr Alan Irving and Dr James Sexton for working with me. It has been a fruitful collaboration so far and I hope this is not the end. I would also like to thank Professor Anthony Kennedy for lots of direction and focus he has given to my final year of work since his arrival in Edinburgh at the Maxwell Institute. Particular thanks go to Dr Alan Irving for reading bits of this thesis and sending back encouraging remarks during my darkest hours of thesis writing. Thanks are also due to Prof Kenway for reading the draft of this thesis at the eleventh hour.

I would like to also acknowledge financial assistance from PPARC in the form of a part time job during my period of study here in Edinburgh.

I found it impossible to go through 3 years of PhD without meeting other students and colleagues who have had an impact on my life here so I'd like to acknowledge them for their friendship and assistance. I would very much like to thank Dr Mauro Talevi for brightening up life here in Edinburgh, for all his friendship and help. Also deserving of thanks are Dr David Richards, Dr Stephen Pickles, Dr Harry Newton, Dr Zbigniew Scroczynski, Dr David Chi-Jen Lin, Dr Chris Maynard, Dr Douglas Smith and Dr Peter Boyle, who were finishing as I was arriving and have assisted me with pints, hospitality, understanding, fixing my bike and even with Physics. I would like to thank Joyce Garden for her collaboration and the occasional deep conversation. I would also like to thank Giuseppe Lacagnina, not only for discussions on Physics but also for being a good flatmate. Particular thanks go to Chris Maynard for assisting with the proof reading of this thesis.

I would also like to thank some other people whom I know from outwith Physics who have been very good friends. I would like to express my gratitude to Jonathan McCalla, Martin Ramsay, Conrad Hughes, Alex Regier, and Alex Hunsley. A special mention is due to Jason and Serena Ayers. Others to thank are János and Éva Hajtó, Maryanne Matthews, Jocelyn Kennedy, Anikó Miski and all the members of the Pásztori-Kupán family.

Thanks go also to Gabriella Rácz and Pauline Wilcox for making my day to day life interesting and varied, looking after me when I was ill, encouraging me when I was down and for all their love and support.

I would also like to thank all those people whose names are too many to list individually, flatmates, colleagues and friends both here in Edinburgh and around the world

Finally, but most importantly I would like to thank my parents Dr Susan Bardócz, Dr Árpád Pusztai and Prof Ferenc Joó for all their help, encouragement, love and support (with the occasional bit of controversy thrown in), my brother Ferenc Joó for coming to visit me occasionally and all my family both here and in Hungary.

Contents

Chapter 1

Introduction

The primary interest of this thesis is the efficient simulation of *lattice Quantum Chromodynamics* (lattice QCD). This theory is a non–perturbative formulation of *continuum Quantum Chromodynamics* (QCD) which is the gauge theory believed to correctly describe the fundamental particles and interactions of the strong nuclear force. Continuum QCD is amenable to analysis by perturbation theory in the ultra–violet regime, where due to asymptotic freedom, its coupling constant α_s is small. However in the infra–red regime, this coupling constant is large – a phenomenon known as infra–red slavery – and perturbation theory fails. In this region, lattice methods are the only ones known that can be used to perform model–independent, *ab–initio* investigations of QCD.

Such calculations have always been enormously demanding in terms of computer time. Until recently most simulations have made use of the *quenched approximation* thus neglecting the effects of virtual quark–antiquark pairs (*dynamical fermions*) on the vacuum. In terms of particle physics phenomenology, this approximation is a rather good one. The spectrum of light hadrons and certain decay constants are reproduced to within 10% - 20% [1, 2] .

Nonetheless, making use of the quenched approximation can lead to large errors elsewhere, in particular the masses of the light quarks are found to be too high [2]. This and other reasons make it necessary from a practical viewpoint as well as in principle to carry out simulations with dynamical fermions.

Dynamical fermion simulations are, however, orders of magnitude more expensive in terms of computer time than ones in the quenched approximation due to having to incorporate the effects arising from the determinant of the *fermion matrix* in the path integral. Thus research into novel algorithms to carry out such calculations is important to making advances in this area.

In this thesis we consider two aspects of performing efficient dynamical fermion simulations. In the first instance we examine the Hybrid Monte Carlo (HMC) method [3], which at the current time is one of the most popular algorithms for performing dynamical fermion simulations. This algorithm is quite mature by now, having been invented in the late 1980s, however there are still unanswered questions concerning some of its features. In particular, we will examine issues of finite precision in Hybrid Monte Carlo on large lattices and small quark masses.

Since with currently available computing resources, HMC simulations of dynamical fermion systems can take months of computer time to complete, tuning the algorithm for maximum performance is certainly desirable. In our study of the HMC algorithm we shall present the results of such a tuning exercise.

The second aspect of this thesis is the investigation of novel algorithms for lattice QCD simulations. In particular we shall consider the Parallel Tempering (PT) algorithm of [4]. This technique was originally proposed to improve the efficiency of simulations of spin–glass systems. We examine whether or not it can be used in combination with HMC to improve the efficiency of simulations of lattice QCD. In the course of this discussion we shall require the use of action matching technologies [5, 6] which can clarify our understanding of the PT algorithm, and in an ideal world can be used to tune the parameters of PT simulations for maximum efficiency.

1.0.1 Thesis Organisation

In the remainder of this chapter we will outline the formulation of gauge theories on a lattice and describe the use of Markov processes in the generation of lattice gauge field configurations. We shall then discuss the Metropolis algorithm for the generation of gauge configurations as one of the most basic of its kind. We will close the chapter by discussing some observables of interest in lattice QCD.

We present an introduction to the Hybrid Monte Carlo algorithm in chapter 2. We shall cover the formulation of the algorithm for a general Hamiltonian system and illustrate its requirements for correctness. We then discuss its formulation for lattice QCD while also describing some of the numerical procedures needed in an actual implementation of the algorithm.

In chapters 3 and 4 we present our study of the HMC algorithm. Chapter 3 focuses on issues of finite precision, and in particular, on tuning the solver residual during the Molecular Dynamics part of an HMC simulation.[1] Chapter 4 on the other hand discusses instabilities in the Molecular Dynamics, which can cause an HMC simulation to give the wrong answer, and attempts to identify the source of these as well as the boundaries of the region of parameter space where one is safe from them.

At this point we leave the subject of HMC and turn to the PT algorithm. We define the algorithm in chapter 5, where we also discuss the relevance of action matching technologies. In chapter 6 we consider the cost–effectiveness of the algorithm. In chapter 6 we outline the criteria for cost effectiveness and discuss a worst case calculation to see whether the algorithm can satisfy these criteria. In chapter 7 we introduce a more flexible, but less generally applicable matrix model for the costs of PT simulations and discuss when the algorithm can achieve the maximum gain. Last but not least, in chapter 8 we present our numerical results in an attempt to answer some questions raised in the preceding chapters.

Finally, we shall collect our principal results and conclusions in chapter 9, where we shall also suggest some avenues for future research.

[1]We assure the reader that the jargon used here will be explained in detail at the relevant part of this thesis

1.1 Lattice Gauge Theories

We begin by outlining those features of lattice gauge theory which are relevant to this thesis. For a detailed introduction to lattice gauge theories, the reader is referred to the standard textbooks and some excellent review articles [7, 8, 9].

1.1.1 The Space Time Lattice

We will consider lattice gauge theories in d dimensions. Our main interest is lattice QCD in 4 dimensions, but the following treatment is quite general.

We discretise space time as a homogeneous lattice with lattice spacing a. Most of the time we shall use dimensionless units in which the lattice spacing is unity to simplify our formulae. The lattice spacing in physical units depends on the coupling constants of the theory. The connection between the two units can be made through the formalism of the *renormalisation group*. In a numerical simulation, the physical lattice spacing is an observable that has to be determined by comparing the values of some dimensionful lattice observable with its experimental counterpart.

In this thesis we shall consider lattice field theories formulated in Euclidean space–time, using the path integral formalism. The relevant Euclidean path integrals can be obtained from their Minkowski space counterparts by analytic continuation. Alternatively, the path integral formalism can be derived directly in Euclidean space–time as is done in [7].

We can label a site on the lattice by a *site index* x. The set of vectors

$$\{\hat{\mu} \mid \hat{\mu} = a\mathbf{e}_\mu,\ \mu = 1, 2, ..., d\} \tag{1.1}$$

with $\mathbf{e}_\mu$ being the Euclidean unit vector in direction μ, form a basis which can be used to reach any site y on the lattice from a given site x. In particular the site neighbouring x in the direction μ shall be denoted $x + \hat{\mu}$.

Continuum derivatives become finite difference operators on the lattice and integrals become sums over the lattice sites:

$$\frac{\partial}{\partial x^{\mu}} \rightarrow \Delta^{\mu}_{x,y} = \frac{1}{2a}\left[\delta_{x,y+\hat{\mu}} - \delta_{x,y-\hat{\mu}}\right] \ , \tag{1.2}$$

$$\nabla^2 \rightarrow \nabla^2_{x,y} = \frac{1}{a^2}\sum_{\mu}\left[\delta_{x,y+\hat{\mu}} + \delta_{x,y-\hat{\mu}} - 2\delta_{x,y}\right] \ , \tag{1.3}$$

$$\int d^n x \rightarrow a^4 \sum_{x} \ . \tag{1.4}$$

1.1.2 Gauge Fields

Gauge fields are represented in lattice gauge theories by *link variables*. These link variables are members of the gauge group $\mathcal{G}$ by which the *action* of the lattice gauge theory is defined. In the case of lattice QCD they are elements of the group $SU(3)$. A link variable associated with the directed link on the lattice beginning on site x pointing in direction μ will be denoted $U_{\mu}(x)$. In an $SU(N)$ gauge theory the link variables are *unitary matrices* and so

$$U^{\dagger}_{\mu}(x) = U^{-1}_{\mu}(x) = U_{-\mu}(x+\hat{\mu}) \ . \tag{1.5}$$

In a continuum gauge theory, gauge fields A_{μ} are associated with the *Lie algebra* of the gauge group. The link variable $U_{\mu}(x)$ corresponds to the *parallel transport operator* of the continuum field theory over the directed link between sites x and $x+\hat{\mu}$:

$$U_{\mu}(x) = \exp\left\{ig_0 \int_{x}^{x+\hat{\mu}} A_{\mu}(x')dx'_{\mu}\right\} \ , \tag{1.6}$$

which in the limit of small lattice spacing reduces to

$$U_{\mu}(x) = \exp\left\{ig_0 a A_{\mu}(x)\right\} \tag{1.7}$$

where g_0 is the bare gauge coupling constant.

Under a local gauge transformation $G(x) \in \mathcal{G}$ the link variables transform as:

$$U_{\mu}(x) \rightarrow G(x)U_{\mu}(x)G^{-1}(x+\hat{\mu}) \tag{1.8}$$

leading to the result that *the trace of path ordered products* of link matrices around closed loops on the lattice are *gauge invariant*. In particular the trace of the elementary *plaquette*

$$\begin{aligned} U_{\mu\nu}(x) &= U_\mu(x)U_\nu(x+\hat{\mu})U_{-\mu}(x+\hat{\mu}+\hat{\nu})U_{-\nu}(x+\hat{\nu}) \\ &= U_\mu(x)U_\nu(x+\hat{\mu})U^\dagger_\mu(x+\hat{\nu})U^\dagger_\nu(x) \text{ for } U \in SU(N) \end{aligned} \quad (1.9)$$

plays an important rôle in lattice gauge theories as it is the simplest gauge invariant operator that can be constructed from link variables.

All that is now required to formulate a pure–gauge theory on the lattice is a suitable lattice action and path integral measure. There exists a many–to–one correspondence between lattice gauge theory actions and the action of a continuum field theory. This is because once one has found a lattice gauge action with the desired continuum limit one is free to add *irrelevant* terms to this action which will disappear in the continuum.

The pure gauge action used in the simulations detailed in this thesis is the one due to Wilson. It is

$$S_{pg}(U) = -\frac{\beta}{2N}\sum_x \sum_{\mu\neq\nu} \text{Re Tr } U_{\mu\nu} \quad (1.10)$$

for an $SU(N)$ lattice gauge theory, where $\beta = 2N/g_0^2$ and g_0 is the bare coupling constant. Here we have also introduced the notation U to label the collection of all the link variables on the lattice. We shall refer to U as a lattice gauge field configuration or simply just as a gauge configuration. This action corresponds to the continuum $SU(N)$ Yang–Mills action with a discretisation error that is of second order in the lattice spacing a

$$S_{pg}(U) = \frac{1}{2}\text{Tr } F_{\mu\nu}F_{\mu\nu} + O(a^2)\ . \quad (1.11)$$

The path integral measure $\mathcal{D}U$ is defined on the lattice as the product of group integration measures on the links of the lattice

$$\mathcal{D}U = \prod_\mu \prod_x dU_\mu(x)\ . \quad (1.12)$$

where $dU_\mu(x)$ is just the usual group invariant (Haar) measure associated with the link between x and $x+\hat{\mu}$.

The expectation value of an operator $\mathcal{O}$ in a pure–gauge lattice gauge theory is thus defined through the Euclidean path integral as

$$\langle \mathcal{O} \rangle = \frac{1}{Z} \int \mathcal{D}U \mathcal{O}(U) \; e^{-S_{pg}(U)} \tag{1.13}$$

where

$$Z = \int \mathcal{D}U \;\; e^{-S_{pg}(U)} \tag{1.14}$$

1.1.3 Lattice Fermion Fields

Fermionic fields pose two major problems in lattice gauge theory. Firstly, the naive discretisation of the fermionic action leads to the infamous fermion doubling problem. What is more, there is no easy way out of this problem that leaves the theory with the correct continuum limit without sacrificing such desirable features of the fermionic theory as locality or chiral symmetry. This is the consequence of the celebrated *no-go theorem* of Nielsen and Ninomiya [10].

Secondly, the fermionic fields appearing in the action of a field theory are members of a *Grassmann algebra* as this is necessary for them to obey the correct anticommutation relations. This feature of fermions is mainly awkward for numerical simulations, as such mathematical objects are difficult to deal with on a computer. However, this particular limitation can be circumvented in several ways as will be demonstrated later on in this section.

We shall examine these two problems in turn. Consider first the case of continuum fermion fields $\bar{\psi}(x)$ and $\psi(x)$ with the corresponding continuum fermion action

$$S_f = \int dx \;\, dy \;\, \bar{\psi}(x) M(x,y) \psi(y) \; . \tag{1.15}$$

In (1.15), $M(x,y)$ is the Dirac operator,

$$M(x,y) = (\gamma_\mu D_\mu + m)\delta^4(x-y) \tag{1.16}$$

with fermion mass m. The fields $\bar{\psi}, \psi$ are members of a Grassmann algebra. The operator

$$D_\mu = \partial_\mu + i g_0 A_\mu(x) \tag{1.17}$$

is the usual continuum covariant derivative, A_μ is the continuum gauge field and g_0 is the gauge coupling constant. The γ_μ are matrices satisfying the anticommutation relation

$$\{\gamma_\mu, \gamma_\nu\} = 2\delta_{\mu\nu} \ . \tag{1.18}$$

Let us now consider the problem of fermion doubling which arises from the naive discretisation of the fermionic fields and $M(x,y)$. For the purposes of this discussion we shall consider the case of a single flavour of free fermion, when

$$M(x,y) = \gamma_\mu \partial^\mu + m \ . \tag{1.19}$$

The naive discretisation prescription for the fields and the action proceeds by replacing the fields $\bar{\psi}(x)$ and $\psi(x)$ with lattice vectors $\bar{\psi}_x$ and ψ_x. In this notation, the subscripts x are lattice site indices and we have suppressed spinor and colour indices.

The matrix $M(x,y)$ is discretised by replacing the continuum derivatives ∂^μ with the finite difference $\Delta^\mu_{x,y}$ of (1.2) to give the lattice fermion matrix

$$M_{x,y} = \gamma_\mu \Delta^\mu_{x,y} + m\delta_{x,y} \tag{1.20}$$

The free field *naive lattice fermion propagator* $\Delta^F = M^{-1}$ can be found by solving the Greens function equation

$$\Delta^F_{x,y} \ M_{y,z} = \delta_{x,z} \tag{1.21}$$

resulting in

$$\Delta^F_{x,y} = \int_{-\frac{\pi}{a}}^{\frac{\pi}{a}} \frac{d^d p}{(2\pi)^d} \frac{-i\gamma_\mu \hat{p}_\mu + m}{\sum_\mu \hat{p}_\mu^2 + m^2} \ e^{ip(x-y)} \tag{1.22}$$

with

$$\hat{p}_\mu = \frac{1}{a} \sin(p_\mu a). \tag{1.23}$$

It can be seen from (1.23) that $\hat{p}$ has two zeroes in a given direction μ, one being at $p_\mu = 0$ and the other being at the edge of the first Brillouin Zone ($p_\mu = \frac{\pi}{a}$). In the neighbourhood around both of these zeroes the propagator Δ^F corresponds to a single particle propagator with momentum p_μ. However these two particles have *opposite chirality*. In effect we now have 2^d particles instead of just one. This unfortunate circumstance persists at all lattice spacings including the continuum limit when $a \to 0$. This is the phenomenon known as *fermion doubling.*

There are two popular formulations of the lattice Dirac operator that alleviate the problem of fermion doubling. The first approach by Wilson sacrifices chiral symmetry in order to remove the doublers. The second approach sacrifices flavour symmetry. The latter approach known as the *staggered* formulation of fermions is not relevant to the work in this thesis and will not be discussed further. The Wilson formulation is sketched briefly below.

As mentioned in section 1.1.2 one is free to add terms to the lattice action which disappear in the continuum limit. In the Wilson formulation one adds to the naive fermion operator a term involving the lattice second derivative. The Wilson fermion operator, on a d dimensional lattice is then

$$\begin{aligned} M^W_{x,y} &= M_{x,y} - \frac{r}{2}\nabla^2_{x,y} \\ &= \{m + dr\}\,\delta_{x,y} - \frac{1}{2}\sum\{(r-\gamma_\mu)\delta_{x,y+\hat{\mu}} + (r+\gamma_\mu)\delta_{x,y-\hat{\mu}}\} \end{aligned} \tag{1.24}$$

where r is called the Wilson parameter.

One finds that the free Wilson fermion propagator $\Delta^W_{x,y}$ is

$$\Delta^W_{x,y} = \int_{-\frac{\pi}{a}}^{\frac{\pi}{a}} \frac{d^d p}{(2\pi)^2}\frac{-i\gamma_\mu\hat{p}_\mu + m(p)}{\sum_\mu \hat{p}^2_\mu + m(p)^2}\, e^{ip(x-y)} \tag{1.25}$$

where

$$m(p) = m + 2\frac{r}{a}\sum_\mu \sin^2\left(\frac{p_\mu a}{2}\right) \tag{1.26}$$

It can be seen that the mass term m has suffered an additive renormalisation. This is because the term $\nabla^2_{x,y}$ in (1.24) violates chiral symmetry explicitly. It is this symmetry that would otherwise protect m from such renormalisations. The renormalised mass term $m(p)$ diverges at the edges of the Brillouin zone as $a \to 0$, and thus the doubler modes are suppressed in Δ_F. Hence we have relaxed the requirement of chiral symmetry in the no-go theorem, and have thus avoided doublers.

Before leaving the Wilson formulation of fermions on the lattice it should be mentioned that the Wilson Fermion matrix is usually written in terms of the *hopping parameter* κ instead of the mass term m. Furthermore for the purposes of gauge theories the lattice derivatives must be made *covariant* through the use of the parallel transport operators (link matrices). Conventionally, the Wilson fermion matrix in (1.25) is rescaled so that the coefficient of the delta function becomes unity. Correspondingly, the fermion fields of the theory are also rescaled to take this into account. The full, gauge covariant Wilson fermion operator is thus

$$M^W_{x,y} = \delta_{x,y} - \kappa D^W_{x,y} \ , \tag{1.27}$$

the fields are rescaled as

$$\psi \to \frac{1}{\sqrt{2\kappa}}\psi \ , \tag{1.28}$$

and

$$D^W_{x,y} = \sum \left\{ U_\mu(x)(1-\gamma_\mu)\delta_{x,y+\hat{\mu}} + U^\dagger_\mu(x-\hat{\mu})(1+\gamma_\mu)\delta_{x,y-\hat{\mu}} \right\} \tag{1.29}$$

where r has been set to unity and κ is defined through the relation

$$am = \frac{1}{2}\left(\frac{1}{\kappa} - \frac{1}{\kappa_c}\right) \ . \tag{1.30}$$

Here κ_c is the so–called "critical" value of κ needed to ensure $m = 0$.

In the case where the fermions are coupled to a gauge field, κ_c is not known *a priori* and careful tuning of κ is required during a course of a simulation to reach a region of parameter space corresponding to light quarks.

In most of the work described in this thesis a formulation of the Wilson fermion operator known as the *Sheikholeslami–Wohlert improved* fermion operator was used. This operator is also known as the *clover improved* fermion operator. The difference between the Wilson and the clover fermion operators is that in the latter an extra term is added to the Wilson fermion operator called the *clover term*. The clover fermion operator is defined as:

$$\begin{aligned} M_{x,y} &= M^W_{x,y} + \frac{i}{2}c\sum_{\mu,\nu} F_{\mu\nu}(x)\sigma_{\mu\nu}\delta_{x,y} \\ &= A_{x,y} - \kappa D^W_{x,y} \end{aligned} \tag{1.31}$$

$$A_{x,y} = \left(1 + \frac{i}{2}c\sum_{\mu,\nu} F_{\mu\nu}(x)\sigma_{\mu\nu}\right)\delta_{x,y} \tag{1.32}$$

In the equations above $F_{\mu\nu}$ is the lattice field strength tensor defined as

$$F_{\mu\nu} = \frac{1}{8}\left(f_{\mu\nu}(x) - f^{\dagger}_{\mu\nu}(x)\right) \tag{1.33}$$

where

$$f_{\mu\nu}(x) = U_{\mu\nu}(x) + U_{\mu\nu}(x-\hat{\mu}) + U_{\mu\nu}(x-\hat{\mu}-\hat{\nu}) + U_{\mu\nu}(x-\hat{\nu}), \tag{1.34}$$

with $U_{\mu\nu}(x)$ being the plaquette operator defined in (1.9). The matrix $\sigma_{\mu\nu}$ is proportional to the commutator of the γ-matrices

$$\sigma_{\mu\nu} = \frac{i}{2}\left[\gamma_\mu, \gamma_\nu\right]. \tag{1.35}$$

The purpose of the clover term is to cancel from certain observables, lattice discretisation errors of $O(a)$, which arise in the discretisation of derivatives. This programme of removing discretisation errors is called an *improvement programme* [11]. In order to implement it fully, certain lattice observables also need to be modified (improved). The coefficient c is called the clover coefficient and has to be carefully tuned if all the $O(a)$ discretisation errors are to be removed from the final results. Details of the Sheikholeslami–Wohlert improvement programme and the tuning of the clover coefficient can be found in [11, 12].

The last point we wish to mention about the lattice fermion matrix is that it is often possible to make lattice computations more efficient through a reformulation of the fermion action known as preconditioning. Our simulations use the action of (1.31) in combination with "even–odd" preconditioning. We shall discuss more details of this preconditioning scheme in chapter 2.

Let us now consider the potential problem of the Grassmannian nature of the lattice fermion fields $\bar{\psi}_x$ and ψ_x. We shall assume that the fermion matrix $M(U)_{x,y}$ is now a suitable, gauge covariant, discretisation of the continuum fermion operator $M(x,y)$, say for example, the one of (1.31). The fermion action is then

$$S_f(U,\bar{\psi},\psi) = \bar{\psi}_x M(U)_{x,y} \psi_y \ . \tag{1.36}$$

Observables $\mathcal{O}(U)$ of the gauge fields U only, are given by the path integral,

$$\langle \mathcal{O} \rangle = \frac{1}{Z}\int \mathcal{D}U\ \mathcal{D}\bar{\psi}\ \mathcal{D}\psi\ \mathcal{O}(U)\ e^{-S_{pg}(U) - S_f(\bar{\psi},\psi,U)} \tag{1.37}$$

where Z is the generating functional for the theory in the absence of any fermionic sources,

$$Z = \int \mathcal{D}U \ \mathcal{D}\bar{\psi} \ \mathcal{D}\psi \ e^{-S_{pg}(U)-S_f(\bar{\psi},\psi,U)} \ . \tag{1.38}$$

To calculate observables depending on the fermion fields $\bar{\psi}$ and ψ, one needs to introduce Grassmann valued sources into the action. The generating functional in the presence of Grassmannian sources $\bar{\eta}_x$ and η_x is

$$Z[\bar{\eta},\eta] = \int \mathcal{D}U\mathcal{D}\bar{\psi}\mathcal{D}\psi \ e^{-S_{pg}(U)-\bar{\psi}_x M(U)_{x,y}\psi_y+\bar{\eta}_x\psi_x-\eta_x\bar{\psi}_x} \ . \tag{1.39}$$

One can carry out the Gaussian integration over the fermion fields $\bar{\psi}$ and ψ in (1.39) to obtain

$$Z[\bar{\eta},\eta] = \int \mathcal{D}U \ \det M(U) \ e^{-S_{pg}(U)-\bar{\eta}_x M^{-1}_{x,y}(U)\eta_y} \tag{1.40}$$

and setting the source terms to zero gives

$$Z = \int \mathcal{D}U \ \det M(U) \ e^{-S_{pg}(U)} \ . \tag{1.41}$$

It can thus be seen from (1.41) that for observables depending on the gauge fields only, the effects of the Grassmann valued fermion fields in the action, can be expressed by reweighting the generating functional of a pure gauge theory with the gauge configuration dependent factor $\det M(U)$. For the case of fermionic correlation functions, carrying out the appropriate functional differentiation in (1.40) and setting the source terms to zero will result in path integrals that are independent of both the fields $\bar{\psi}$ and ψ as well as the source terms $\bar{\eta}$ and η, as the effects of $\bar{\psi}$ and ψ in (1.40) are once again incorporated in $\det M(U)$, and terms coupling to $\bar{\eta}$ and η will vanish when the sources are set to zero. Hence in both cases, the evaluation of the expectation value of a general observable $\mathcal{O}(U)$, which can now in principle depend on the fermionic fields in the action, reduces to evaluating path integrals of the form

$$\langle \mathcal{O} \rangle = \frac{1}{Z} \int \mathcal{D}U \ \det M(U) \ \mathcal{O}(U) \ e^{-S_{pg}(U)} \ . \tag{1.42}$$

We have now removed the Grassmann fields from the path integral, at the expense of having to deal with the determinant of the fermion matrix. There are at the moment three common techniques for handling the determinant:

1. **Ignore it completely:** This is the so called *quenched approximation*. Here the determinant is set to a constant, and this constant cancels between the numerator and the partition function of (1.42). Physically this corresponds to neglecting virtual quark-antiquark loops in the vacuum. A nice exposition of why this is the case can be found in [13].

2. **Re–express it to obtain an effective gauge action:** In this case we write Z as,

$$\begin{aligned} Z &= \int \mathcal{D}U \det M(U)\ e^{-S_{pg}(U)} \\ &= \int \mathcal{D}U\ e^{-S_{pg}(U)+\ln\det M(U)} \\ &= \int \mathcal{D}U\ e^{-S_{pg}(U)+\mathrm{Tr}\ \ln M(U)} \end{aligned} \tag{1.43}$$

Thus the generating functional in the absence of sources is now that of a gauge theory with an *effective action*

$$S_{\text{eff}} = S_{pg}(U) - \mathrm{Tr}\ \ln M(U)\ . \tag{1.44}$$

3. **The method of pseudofermions:** In this case one re–expresses (1.41) as an integral over scalar lattice fields ϕ_x and $\phi^\dagger_x$:

$$\begin{aligned} Z &= \int \mathcal{D}U\ \det M(U)\ e^{-S_{pg}(U)} \\ &= \int \mathcal{D}U\ \mathcal{D}\phi^\dagger\ \mathcal{D}\phi\ e^{\left\{-S_{pg}(U)-\phi^\dagger_x M(U)^{-1}_{x,y}\phi_y\right\}} \end{aligned} \tag{1.45}$$

The fermionic theory in question has been transformed into a bosonic one over the fields $U, \phi^\dagger$ and ϕ, with the effective action

$$S_{\text{eff}} = S_{pg}(U) + \phi^\dagger_x M(U)^{-1}_{x,y}\phi_y \tag{1.46}$$

In methods employing this trick, the fields $\phi^\dagger$ and ϕ are called *pseudofermions*.

1.1.4 Lattice QCD with Dynamical Fermions

Up until now, our descriptions of how to formulate gauge and fermion fields on the lattice have been rather general. Let us now make the situation concrete.

Almost all the simulations detailed in this thesis are of a system with action

$$S(U,\phi^\dagger,\phi;\beta,\kappa,c) = S_{pg}(U;\beta) + \phi_x^\dagger Q(U;\kappa,c)_{x,y}\phi_y \tag{1.47}$$

where S_{pg} is the Wilson gauge action of (1.10) and

$$Q(U;\kappa,c) = \left\{M^\dagger(U;\kappa,c)M(U;\kappa,c)\right\}^{-1} \tag{1.48}$$

is the inverse of the *fermion matrix* for two flavours of degenerate quarks and $\phi^\dagger$ and ϕ are pseudofermion fields. The quantities β, κ and c are as defined in previous sections and are the physical parameters to a given simulation.

The expectation value of a general observable $\mathcal{O}$ of the gauge fields is given by the path integral

$$\langle\mathcal{O}\rangle = \frac{1}{Z}\int \mathcal{D}U\ \mathcal{D}\phi^\dagger\ \mathcal{D}\phi\ \mathcal{O}(U)\ e^{-S(U,\phi^\dagger,\phi;\beta,\kappa,c)} \tag{1.49}$$

$$Z = \int \mathcal{D}U\ \mathcal{D}\phi^\dagger\ \mathcal{D}\phi\ e^{-S(U,\phi^\dagger,\phi;\beta,\kappa,c)} \tag{1.50}$$

with $\mathcal{D}U$ as specified in (1.12) and

$$\mathcal{D}\phi = \prod_x d\phi_x\ . \tag{1.51}$$

Such a system in d space–time dimensions on a lattice with N sites is formally equivalent to a classical equilibrium statistical mechanics system with an $O(N)$ dimensional configuration space

$$\Gamma = \Gamma(U)\otimes\Gamma(\phi^\dagger)\otimes\Gamma(\phi)\ . \tag{1.52}$$

Simulations of lattice QCD, are calculations which attempt to estimate path integrals like (1.49). Due to the enormously large number of degrees of freedom possessed by a lattice system, evaluation of path integrals via traditional numerical methods such as the Trapezium Rule and Simpson's rule is not numerically feasible. Most computations thus rely on Monte Carlo methods which we now proceed to discuss.

1.2 Monte Carlo Integration

1.2.1 Sampling Methods

Sampling methods attempt to approximate path integrals such as (1.49) by generating a sequence of N samples of the configuration space of the system in question. In our case, the samples are just gauge field configurations U. For purposes of clarity we shall not consider pseudofermion fields in the ensuing discussion as they would cause unnecessary complication. Instead, to save us from writing factors of $\det M(U)$ we assume that we are using some effective action such as $S = S_{pg}(U) - \mathrm{Tr}\ \ln Q^{-1}(U)$ to account for the fermionic determinant.

Random Sampling methods produce a sequence of N gauge configurations $\{U_i, i = 0 \ldots N\}$ in which every configuration has equal probability of occurring. For an observable $\mathcal{O}$, we can then construct the sequence $\{\mathcal{O}(i), i = 0 \ldots N\}$ where $\mathcal{O}(i) = \mathcal{O}(U_i)$. The path integral for the expectation value of an observable $\mathcal{O}$ could then be approximated as the weighted average,

$$\langle \mathcal{O} \rangle \approx \frac{\sum_{i=0}^{N} \mathcal{O}(i) e^{-S(U_i)}}{\sum_{i=0}^{N} e^{-S(U_i)}} \ . \tag{1.53}$$

However, the probability density function $P(\mathcal{O}(i))$ of $\mathcal{O}(i)$ is expected to be sharply peaked around the expectation value $\langle \mathcal{O}(i) \rangle$, with a width of $\frac{1}{\sqrt{N}}$. Hence we would expect that most of the $\mathcal{O}(i)$ have negligible contribution to the sum (1.53).

Importance sampling methods improve upon this deficiency of random sampling by attempting to sample mostly those parts of phase space with a non–negligible contribution to the path integral. Consider then generating a sequence $\{U_i\}$ and the corresponding sequence $\{\mathcal{O}(i)\}$ in such a way that the occurrence of configuration U_i in the sequence has probability density function $P_o(U_i)$.

In order to account for the fact that the configurations U_i no longer have a constant probability distribution function we have to introduce additional reweighting factors of $\frac{1}{P_o(U_i)}$ into the numerator and denominator of (1.53). The expectation value is then approximated as

$$\langle \mathcal{O} \rangle \approx \frac{\sum_{i=0}^{N} \mathcal{O}(i) \frac{e^{-S(U_i)}}{P_o(U_i)}}{\sum_{i=0}^{N} \frac{e^{-S(U_i)}}{P_o(U_i)}} \; . \tag{1.54}$$

In particular, if it can be arranged that $P_o(U_i) = \frac{1}{Z} e^{-S(U_i)}$ then

$$\langle \mathcal{O} \rangle \approx \frac{1}{N} \sum_{i=0}^{N} \mathcal{O}(i) = \bar{\mathcal{O}}(i), \tag{1.55}$$

in other words, the expectation value would be approximated by the *sample mean*, $\bar{\mathcal{O}}$ of the sequence $\{\mathcal{O}(i)\}$.

1.2.2 Stochastic Processes

We now concern ourselves with a generation of a sequence of configurations such as is required for importance sampling. Consider an iterative process which at iteration l produces a gauge configuration U_l which is a random variable with probability density function $P^l(U)$. We shall refer to such an iterative process as a stochastic process and to iteration l as step l, or simulation timestep l respectively.

Ideally we would like to have that

$$P^l(U) = P^{\mathrm{eq}}(U) = \frac{1}{Z} e^{-S(U)} \tag{1.56}$$

for all l, as in this case it would necessarily follow that the probability density function for U_l occurring in the sequence would also be $P_o(U_l) = P^{\mathrm{eq}}(U_l)$, as is required for importance sampling. Stochastic processes satisfying condition (1.56) are called *strictly stationary*. In practice the desired condition of strict stationarity is difficult to realise. However, one can achieve the next best thing, which is to generate a process for which $P^l(U) \to P^{\mathrm{eq}}(U)$ in the limit when $l \to \infty$.

To clarify this idea consider again a process, which on step l produces a random variable U_l which has probability density function $P^l(U)$. We can define a distance norm between $P^l(U)$ and the desired probability density function $P^{\mathrm{eq}}(U)$ as

$$d(P^l(U), P^{\mathrm{eq}}(U)) = \int |P^l(U) - P^{\mathrm{eq}}(U)| \mathcal{D}U \; . \tag{1.57}$$

If $d(P^l(U), P^{\rm eq}(U)) \to 0$ as $l \to \infty$, then by the definition of limits, for any arbitrary positive, real ϵ, there must exist some integer L such that for $l > L$, $d(P^l(U), P^{\rm eq}(U)) < \epsilon$. In particular, ϵ can be made infinitesimally small, in which case we can say that the process has converged to its fixed point $P^{\rm eq}(U)$, or alternatively, that the process has reached equilibrium.

For a given, small ϵ we shall call the corresponding L the equilibration time of the system. In principle, we should discard U_l for $l < L$ from the sequence of configurations which we use to estimate expectation values. The remaining configurations will effectively all have the desired probability density function $P^{\rm eq}(U)$ within the small error ϵ. The fact that there is a difference of $O(\epsilon)$ between $P^l(U)$ and $P^{\rm eq}(U)$ does introduce some bias into our estimates of expectation values, however, we can certainly choose ϵ so small that this bias is negligible with respect to the statistical errors on these estimates.

1.2.3 Markov Processes

A particular means of realising processes which converge to the required equilibrium is through the use of Markov processes. We shall now give definitions of these processes, and discuss what conditions they must satisfy to ensure their convergence. For clarity of discussion, we shall consider a system consisting of a finite number of discrete states x. We shall generalise the discussion at the end to the case when the random variables come from a continuously infinite set, such as from the configuration space of lattice QCD.

The process in question constructs a sequence of states $\{x_0, x_1, \ldots, x_n\}$, with state x_l having probability distribution $P^l(x)$. The process is called *Markov* if the probability that state x_i takes a certain value depends only on state x_{i-1}

$$P(x_i = x | x_{i-1}, x_{i-2}, \ldots, x_0) = P(x_i = x | x_{i-1}) \tag{1.58}$$

The *probability of transition* from state x to state x' can be expressed in matrix notation as

$$\mathcal{M}_{x'x} = P(x_i = x' | x_{i-1} = x) \ . \tag{1.59}$$

where $\mathcal{M}$ is called the *Markov Matrix*. Since $\mathcal{M}_{x'x}$ is a probability it follows that

$$\mathcal{M}_{x'x} \geq 0 \qquad \text{for all } x', x \tag{1.60}$$

and that

$$\sum_{x'} \mathcal{M}_{x'x} = 1 \qquad \text{for all } x \ . \tag{1.61}$$

A Markov process is called *strongly ergodic* if

$$\mathcal{M}_{x'x} > 0 \qquad \text{for all } x', x \ . \tag{1.62}$$

In this notation, the probability distribution of states $P^l(x)$ on step l can be expressed as a vector p^l, with components p^l_x. The probability distribution of states at a time $l+1$ can be obtained from the ones at time l using the Markov matrix as

$$p^{l+1}_{x'} = \sum_x \mathcal{M}_{x'x} p^l_x \qquad \text{or equivalently} \qquad p^{l+1} = \mathcal{M} p^l \ , \tag{1.63}$$

where in the second of the two equations we lapsed completely into the language of matrices and vectors. In addition we point out that given an initial probability distribution of states p^0 the one at step l is formally given by $p^l = \mathcal{M}^l p^0$, where the superscript on $\mathcal{M}$ is not a label, but implies that $\mathcal{M}$ should be raised to the power l. We also note that at each step l, the space of all probability distributions p^l is a complete set.

If $p^{l+1} = p^l = p^{\text{eq}}$ then it is clear that that the Markov process is strictly stationary. In this case p^{eq} is called the *fixed point* of the Markov process. We can write the fixed point condition as

$$p^{\text{eq}} = \mathcal{M} p^{\text{eq}} \tag{1.64}$$

from which we see that p^{eq} is an eigenvector[2] of M with an eigenvalue of 1.

A Markov process is said to satisfy *detailed balance* with respect to a probability distribution p if

$$\mathcal{M}_{x'x} p_x = \mathcal{M}_{xx'} p_{x'} \tag{1.65}$$

[2] Absolutely rigorous treatments make a distinction between left and right eigenvectors of M [13]. Such distinctions are not truly relevant for us at this level of treatment

and this is sufficient (but not necessary) to imply that p is a fixed point of the process. To see this we simply sum over one of the indices, say x, in (1.65):

$$\begin{aligned}\sum_x \mathcal{M}_{x'x} p_x &= \sum_x \mathcal{M}_{xx'} p_{x'} \\ &= p_{x'} \sum_x \mathcal{M}_{xx'} \\ &= p_{x'} \end{aligned} \tag{1.66}$$

and $p = \mathcal{M}p$ which is the condition for p to be a fixed point.

In the case of discrete variables, the distance norm of the last section between any two probability distributions p^l and q^l at step l of the process becomes

$$d(p^l, q^l) = \sum_x |p^l_x - q^l_x| \; . \tag{1.67}$$

It can be shown [14] that a strongly ergodic Markov process with Markov matrix M is a contraction mapping with respect to this distance metric so that

$$d(p^{l+1}, q^{l+1}) = d(\mathcal{M}p^l, \mathcal{M}q^l) \leq \alpha \; d(p^0, q^0) \; , \;\; \alpha < 1 \tag{1.68}$$

implying that the sequence $\{d^l, l = 1 \ldots\}$ with

$$d^l = d(Mp^{l-1}, Mq^{l-1}) = d(M^l p^0, M^l q^0) \tag{1.69}$$

is Cauchy with

$$\lim_{l \to \infty} d^l = 0 \tag{1.70}$$

This together with the completeness of the space of probability distributions is sufficient to imply that a strongly ergodic Markov process has a unique fixed point distribution $p^{\mathrm{eq}} = P^{\mathrm{eq}}(x)$ to which it will converge regardless of the initial probability distribution of the states.

The condition of strong ergodicity is not always fulfilled. However if the weaker condition of *ergodicity* is satisfied that

$$\inf_{t,x,x'} \mathcal{M}^t_{x'x} > 0 \tag{1.71}$$

then there exists some n such that for $l \geq n$, $\mathcal{M}^l_{x'x} > 0$ for all states x' and x. One can then construct a Markov process out of the subprocesses $\mathcal{M}^n$ which is in turn strongly ergodic and the process will converge to the equilibrium limit.

If $\mathcal{M}$ has N_s linearly independent eigenvectors $\{e_i\}$, where N_s is the number of discrete states that x can take, then it can be diagonalised with some nonsingular matrix L such that

$$L^{-1}\mathcal{M}L = D \tag{1.72}$$

with D being a diagonal matrix of the eigenvalues $\{\lambda_i\}$ of $\mathcal{M}$ and L being a matrix whose columns are the corresponding eigenvectors $\{e_i\}$. If $\mathcal{M}$ is strongly ergodic, then by the foregoing it is necessary that

- $p^{\rm eq}$ is an eigenvector of M with eigenvalue $\lambda_{eq} = 1$
- $0 < |\lambda_i| < 1$ for all $i > 0$, where the eigenvalues have been labelled so that $\lambda_{eq} = \lambda_0$ and $|\lambda_{i+1}| \leq |\lambda_i|$
- there exists a finite gap $\delta = \lambda_0 - \lambda_1 = 1 - \lambda_1 > 0$.

otherwise the necessary limit 1.70 would not be reached. The implication of the above is that as $n \to \infty$,

$$\sup d^n = |\lambda_1|^n d_0 = e^{n \ln|\lambda_1|} d_0 \tag{1.73}$$

and the process approaches its equilibrium distribution exponentially.

Generalisation to the case where the random variables on step l come from a continuously infinite configuration space is straightforward. Instead of the probability distributions of states at step l, we consider probability density functions. The probability density functions at step l can be considered as state vectors $|p^l\rangle$ in a similar style to quantum mechanics. The Markov matrix $\mathcal{M}$ becomes a linear operator on the space of these state vectors. The element $\mathcal{M}_{U'U}$ is more conventionally written as $W(U' \leftarrow U)$ and is the probability density of transition from a small volume of configuration space $\mathcal{D}U$ centred around U to another one, $\mathcal{D}U'$, centred around U'. The definition of the distance norm has already been given in (1.57)

In this notation the fixed point condition becomes,

$$P^{\rm eq}(U')\mathcal{D}U' = \mathcal{D}U' \int \mathcal{D}U \; W(U' \leftarrow U) P^{\rm eq}(U) \tag{1.74}$$

and the detailed balance condition is written as

$$W(U' \leftarrow U)P^{\mathrm{eq}}(U)\ \mathcal{D}U\ \mathcal{D}U' = W(U \leftarrow U')P^{\mathrm{eq}}(U')\ \mathcal{D}U'\ \mathcal{D}U\ . \tag{1.75}$$

We note that the measures are the same on both sides of the detailed balance condition above, and in this case one is free to deal simply with the probability density functions $W(U' \leftarrow U)$ and $P^{\mathrm{eq}}(U)$.

1.2.4 Generation of Configurations

Next we require to construct a Markov process which is ergodic and satisfies detailed balance with respect to our desired equilibrium distribution. To this end, we consider the following generic algorithm:

Start with some initial configuration x_0. Generate configuration x_{i+1} from configuration $x_i = x$ as follows

i) Choose a trial configuration x' with probability $P_{\mathrm{c}}(x' \leftarrow x)$.

i) Accept the x' with probability $P_{\mathrm{acc}}(x' \leftarrow x)$. This can be done by choosing a uniformly random (or pseudo-random) number u, in the interval $[0, 1)$. If $0 \leq u < P_{\mathrm{acc}}$ the configuration is accepted otherwise the configuration is rejected.

iii) If the x' is accepted then $x_{i+1} = x'$ otherwise $x_{i+1} = x$.

The process is clearly Markov if $P_{\mathrm{c}}(x' \leftarrow x)$ and $P_{\mathrm{acc}}(x' \leftarrow x)$ depend on x' and x only. The element $M_{x'x}$ of the Markov Matrix is the probability

$$M_{x,x} = P_{\mathrm{acc}}(x \leftarrow x)P_{\mathrm{c}}(x \leftarrow x) - \left(1 - \sum_{x' \neq x} P_{\mathrm{acc}}(x' \leftarrow x)P_{\mathrm{c}}(x' \leftarrow x)\right) \tag{1.76}$$

when $x' = x$ and

$$M_{x'x} = P_{\mathrm{acc}}(x' \leftarrow x)P_{\mathrm{c}}(x' \leftarrow x) \tag{1.77}$$

otherwise. The condition for detailed balance with respect to $P^{\mathrm{eq}}(x)$ is

$$P_{\mathrm{acc}}(x' \leftarrow x)P_{\mathrm{c}}(x' \leftarrow x)P^{\mathrm{eq}}(x) = P_{\mathrm{acc}}(x \leftarrow x')P_{\mathrm{c}}(x \leftarrow x')P^{\mathrm{eq}}(x')\ . \tag{1.78}$$

In most algorithms the probability P_c is chosen *reversibly* so that

$$P_c(x' \leftarrow x) = P_c(x \leftarrow x') \tag{1.79}$$

and in this case the P_c terms cancel out of the detailed balance condition.

1.2.5 The Metropolis Algorithm

The Metropolis algorithm [15] is typical of the procedure outlined above. In this algorithm steps ii) and iii) of the template algorithm in 1.2.4 are restricted to changing only a single degree of freedom, say a single link variable in the case of a lattice gauge theory.

The trial configurations may be chosen with any reasonable P_c and P_{acc}, in the accept/reject step, is the famous *Metropolis acceptance probability*

$$P_{acc}(x' \leftarrow x) = \min\left(1, \frac{P_c(x \leftarrow x')P^{eq}(x')}{P_c(x' \leftarrow x)P^{eq}(x)}\right) \tag{1.80}$$

We note that this acceptance probability satisfies detailed balance with respect to P^{eq} by construction. Consider the case when,

$$\frac{P_c(x \leftarrow x')P^{eq}(x')}{P_c(x' \leftarrow x)P^{eq}(x)} < 1. \tag{1.81}$$

In this case

$$P_{acc}(x' \leftarrow x) = \frac{P_c(x \leftarrow x')P^{eq}(x')}{P_c(x' \leftarrow x)P^{eq}(x)} \tag{1.82}$$

and

$$P_{acc}(x \leftarrow x') = 1 \ . \tag{1.83}$$

The left hand side of the detailed balance condition in (1.78) is

$$\begin{aligned} P_{acc}(x' \leftarrow x)P_c(x' \leftarrow x)P^{eq}(x) &= \frac{P_c(x \leftarrow x')P^{eq}(x')}{P_c(x' \leftarrow x)P^{eq}(x)}P_c(x' \leftarrow x)P^{eq}(x) \\ &= P_c(x \leftarrow x')P^{eq}(x') \ , \end{aligned} \tag{1.84}$$

while the right hand side is

$$P_{acc}(x \leftarrow x')P_c(x \leftarrow x')P^{eq}(x') = P_c(x \leftarrow x')P^{eq}(x') \ . \tag{1.85}$$

Clearly the two sides are equal, and detailed balance is satisfied. The relation is trivially satisfied when $x' = x$.

A single update step itself in this algorithm is not ergodic as only one degree of freedom is updated, all the remaining ones being held fixed. However, the sequence of updates when all the degrees of freedom are updated, known as a *sweep*, is ergodic.

1.2.6 Metropolis–like Algorithms

The Metropolis accept/reject strategy can also be used if more than one degree of freedom is changed in step ii) of the template algorithm. In particular it is possible that a trial configuration is chosen in such a way that all the degrees of freedom are updated. Such procedures will be called *Global Metropolis–like procedures* in the future. In particular the HMC procedure described in the following chapter is a global Metropolis–like algorithm where the trial configurations are chosen from a starting configuration x by integrating the Hamiltonian equations of motion for the system in question.

In other situations it may be possible to choose the trial configurations directly from the equilibrium distribution $P^{\mathrm{eq}}(x)$. In this case the acceptance probability $P_{\mathrm{acc}} = 1$. This method is known as the *heat–bath method.*

1.2.7 Sample Mean, Variance and Autocorrelations

We now examine briefly the convergence of our estimates of path integrals. We assume that we are able to generate a suitable sequence of configurations $\{U_i, i = 0 \ldots N\}$ via importance sampling, and that we have generated from this the corresponding sequence $\{\mathcal{O}(i), i = 0 \ldots N\}$.

As indicated before, the sample mean $\bar{\mathcal{O}}$ is a suitable estimator for $\langle \mathcal{O} \rangle$. If successive samples $\mathcal{O}(i)$ were independent, the naive variance would be defined as

$$\sigma_I^2(\mathcal{O}) = \frac{1}{N} \left\langle (\mathcal{O}(i) - \langle \mathcal{O}(i) \rangle)^2 \right\rangle \tag{1.86}$$

and

$$\hat{\sigma}_I^2(\mathcal{O}) = \frac{1}{N(N-1)} \sum_{i=0}^{N} (\mathcal{O}(i) - \bar{\mathcal{O}})^2 \tag{1.87}$$

would be a suitable estimator for $\sigma_I^2(\mathcal{O})$.

In the case of Markov processes, successive configurations are usually correlated. In this case the variance is defined as

$$\sigma^2(\mathcal{O}) = \sigma_I^2(\mathcal{O}) \sum_{t=-(N-1)}^{t=+(N-1)} \left(1 - \frac{|t|}{N}\right) C_{\mathcal{O}}(t) \; , \tag{1.88}$$

where

$$C_{\mathcal{O}}(t) = \frac{1}{\sigma_I^2(\mathcal{O})} \langle (\mathcal{O}(|t| + t_0) - \langle \mathcal{O} \rangle)(\mathcal{O}(t_0) - \langle \mathcal{O} \rangle) \rangle_{t_0} \tag{1.89}$$

is the autocorrelation function for observable $\mathcal{O}$. By the subscript on the expectation value we mean that the averaging is to be carried out over all pairs of configurations $\mathcal{O}(t_0)$ and $\mathcal{O}(|t| + t_0)$ separated by t steps. We shall drop the subscript $\mathcal{O}$ from now on. We point out that $C(t)$ is even, and is normalised to unity.

By making the assumption that $C(t)$ decays exponentially, we can evaluate the sum in (1.88) in the limit that $N \to \infty$:

$$\begin{aligned} \sigma^2(\mathcal{O}) &= \sigma_I^2(\mathcal{O}) \left(\sum_{t=-\infty}^{\infty} C(t) - \sum_{t=-\infty}^{\infty} \frac{|t|}{N} C(t) \right) \\ &= \sigma_I^2(\mathcal{O})(1 + 2\mathcal{A}_{\mathcal{O}}) \; , \end{aligned} \tag{1.90}$$

where we have introduced the integrated autocorrelation time $\mathcal{A}_{\mathcal{O}}$, for observable $\mathcal{O}$ defined as

$$\mathcal{A}_{\mathcal{O}} = \sum_{t=1}^{\infty} C(t) \; . \tag{1.91}$$

In the case of independent configurations $\mathcal{A} = 0$.

Let now consider the standard error on the mean

$$\epsilon(N, \sigma^2(\mathcal{O})) = \sqrt{\frac{\sigma^2}{N}} = \sqrt{\frac{(2\mathcal{A} + 1)}{N} \sigma_I^2(\mathcal{O})} \; . \tag{1.92}$$

We can clearly see that ϵ decreases as $\frac{1}{\sqrt{N}}$, and that in the case when successive configurations are correlated, a factor of $(2\mathcal{A} + 1)$ more configurations are needed to reach the same value of ϵ than one would need in the case of independent configurations.

Since in practice it requires a finite amount of time for a computer program implementation of a given Markov process to generate each configuration, it is clear that the integrated autocorrelation time $\mathcal{A}$ is an important metric of the performance of such an implementation. However, $\mathcal{A}$ depends on the process, and the observable rather than how the implementation is coded. Hence $\mathcal{A}$ is a good characterisation of the speed of convergence of the observable $\mathcal{O}$ for a given process. In particular we can use $\mathcal{A}$ to decide which of two processes is more efficient. However, the numerical estimation of $\mathcal{A}$ for a given Markov process is not entirely straightforward. We discuss the details in appendix A.

1.3 Observables

Now that we have outlined the basics of generating gauge configurations we will outline a few observables that are of interest in lattice QCD and are relevant to this thesis.

1.3.1 Plaquette

One of the simplest observables to calculate is the plaquette operator of (1.9) averaged over all lattice sites and planes (μ, ν) of the lattice. In terms of computer time used this operator is very cheap to calculate and is frequently calculated for every update step in a Metropolis–like algorithm. It is useful for example to monitor the convergence of the algorithm to equilibrium, and to validate gauge configurations that have been stored on some permanent storage medium, when they are loaded to carry on an existing simulation or to compute some other observable. As the operator is typically calculated after every update step of an algorithm reasonable estimates of its integrated autocorrelation time may be calculated, in order to estimate the efficiency of the algorithm used.

1.3.2 Quark propagators and Hadronic Correlation Functions

In hadron spectroscopy one is interested in two point correlation functions of the form

$$C(x) = \left\langle \Omega | \mathcal{C}(x) \mathcal{C}^{\dagger}(0) | \Omega \right\rangle \tag{1.93}$$

where $|\Omega\rangle$ is the vacuum state, $\mathcal{C}^{\dagger}(0)$ creates some current at the origin and $\mathcal{C}(x)$ annihilates it at some point x.

In particular for the study of mesons $\mathcal{C}(x)$ is a bilinear of lattice fermion fields[3]

$$\mathcal{C}(x) = \bar{\psi}_x \Gamma \psi_x \tag{1.94}$$

where Γ is an operator that has the same spin–parity structure as the meson state under investigation, for example $\Gamma = \gamma^5$ for the pseudo-scalar particle. Written out in full

$$C(x) = \left\langle \Omega | \bar{\psi}(x) \Gamma \psi(x) \bar{\psi}(0) \Gamma^{\dagger} \psi(0) | \Omega \right\rangle \ . \tag{1.95}$$

Wick contracting $\psi(0)$ with $\bar{\psi}(x)$ and $\psi(x)$ with $\psi(0)$ one obtains

$$C(x) = \left\langle \Omega | \mathrm{Tr}\ \Delta_{0,x} \Gamma \Delta_{x,0} \Gamma^{\dagger} | \Omega \right\rangle \tag{1.96}$$

where the trace is carried out over spin indices (suppressed in our notation) and $\Delta_{x,y} = M^{-1}_{x,y}$ is the lattice fermion propagator.

Using the fact that the fermion matrix is γ_5-Hermitian it is straightforward to establish that $\Delta_{0,x} = \gamma_5 \Delta^{\dagger}_{x,0} \gamma_5$. Inserting this relation into (1.96) and using the cyclic properties of traces it can be established that

$$C(x) = \left\langle \Omega | \mathrm{Tr}\ \Delta_{x,0} \Gamma^{\dagger} \gamma_5 \Delta^{\dagger}_{x,0} \gamma_5 \Gamma | \Omega \right\rangle \ . \tag{1.97}$$

The correlation functions $C(x)$ can hence be built out of the γ matrices and the fermion propagator $\Delta_{x,0}$ from the origin to the point x. For other uses of quark propagators, such as the construction of baryonic two point functions for mass spectroscopy, the construction of three point functions and four quark operators for the study of semileptonic decays and flavour changing neutral current interactions, we refer the reader to [16, 17, 18].

[3] In the case of baryons, the situation is similar, but a little more complicated. The reader is referred to [8] for details.

1.3.3 Time Decay of Correlation Functions

We now briefly outline how correlation functions of operators decay with Euclidean time. Let us consider the decay of the time dependent, position independent operator $\mathcal{O}(t)$ on a lattice of infinite temporal extent. The case of finite temporal extent causes minor complications associated with the boundary conditions of the system, which we will not discuss here.

The time evolution of the operator is given by the transfer matrix formalism as

$$\mathcal{O}(t) = e^{\hat{H}t}\mathcal{O}(0)e^{-\hat{H}t} \ , \tag{1.98}$$

where $\hat{H}$ is the Hamiltonian operator of the system. The two point function becomes

$$\left\langle \Omega|\mathcal{O}(t)\mathcal{O}^{\dagger}(0)|\Omega\right\rangle = \left\langle \Omega|e^{\hat{H}t}\mathcal{O}(0)e^{-\hat{H}t}\mathcal{O}^{\dagger}(0)|\Omega\right\rangle \tag{1.99}$$

Inserting complete sets of eigenstates $|n\rangle$ of the Hamiltonian gives

$$\begin{aligned} \left\langle \Omega|\mathcal{O}(t)\mathcal{O}^{\dagger}(0)|\Omega\right\rangle &= \sum_n \left\langle \Omega|e^{\hat{H}t}\mathcal{O}(0)e^{-\hat{H}t}|n\right\rangle \left\langle n|\mathcal{O}^{\dagger}(0)|\Omega\right\rangle & (1.100) \\ &= \sum_n e^{-(E_n-E_\Omega)t}|\left\langle \Omega|\mathcal{O}(0)|n\right\rangle|^2 & (1.101) \end{aligned}$$

where E_n is the eigenvalue of the n-th eigenstate of the Hamiltonian, and E_Ω is the vacuum state energy.

If the *ground state* $|0\rangle$ of the Hamiltonian has energy $E_0 > E_\Omega$ then in the limit of large Euclidean time, the contributions from the *excited states* ($n > 0$) will die away faster than the contribution from the ground state, and

$$\left\langle \Omega|\mathcal{O}(t)\mathcal{O}^{\dagger}(0)|\Omega\right\rangle \stackrel{t\to\infty}{\longrightarrow} Ae^{-mt} \tag{1.102}$$

where the constant $A = |\left\langle \Omega|\mathcal{O}(0)|0\right\rangle|^2$, and $m = E_0 - E_\Omega$ is the *mass gap* of the system.

In particular for the case of a hadronic correlation function

$$C(\mathbf{p},t) = \frac{1}{V}\sum_{\mathbf{x}} C(\mathbf{x},t)e^{i\mathbf{p}.\mathbf{x}} \tag{1.103}$$

with V being the spatial lattice volume and $C(\mathbf{x},t) = C(x)$ being a correlation function as defined in (1.97), $C(\mathbf{p},t)$ decays exponentially as above and for three momentum $\mathbf{p} = \mathbf{0}$, the mass gap m corresponds to the rest mass of the meson [13, 9, 8, 7].

1.3.4 Wilson loops, the static potential, r_0 and the string tension

One is often interested in the expectation values of the trace of path ordered products of link matrices around a contour C on the lattice. An operator of this sort is called a Wilson loop W is defined as

$$W(C) = \mathrm{Tr}\ P\left(\prod_{l\in C} U_l\right) \tag{1.104}$$

where P denotes the path ordering operator and l labels a link belonging to the loop. Such observables are useful in many situations, for example the tuning of the coefficients c_i in an effective action

$$S_{\mathrm{eff}} = \sum_i c_i W(C_i) \tag{1.105}$$

when one wishes to match this to some other action S [5].

A rectangular loop in a space–time plane p of the lattice with width R in the spatial direction and length T in the temporal dimension will be denoted $W_p(R,T)$. In particular the expectation value $\langle W_p(R,T)\rangle$ corresponds to the time evolution of the potential between a static quark–antiquark pair with a separation R. The expectation value of $W_p(R,T)$ can be shown, using similar arguments as in section 1.3.3, to be of the form

$$\langle W_p(R,T)\rangle = C(R)e^{-V(R)T} + \text{excited states} \tag{1.106}$$

where $V(R)$ is the mass gap of the observable $\langle W_p(R,T)\rangle$ known as the *static potential.* The function $C(R)$ describes the overlap of the operator with the ground state contribution – c.f the constant A in (1.102).

The practicalities of extracting the $V(R)$ from $\langle W_p(R,T)\rangle$ are outlined in [19], where the focus is on maximising the overlap $C(R)$ with the ground state and the elimination of the excited states.

The operator $V(R)$ is interesting for several reasons. In particular it has been used in the past to study the QCD Λ-parameter and the running of the QCD coupling α_s in [19]. It has also been used as probe for string breaking and for the study of sea quark effects in dynamical fermion simulations [2, 1, 20].

The lattice observable $\hat{r}_0$ [21] is defined through the static potential as

$$F(\hat{r}_0)\hat{r}_0^2 = 1.65 \tag{1.107}$$

where $F(R)$ is the force between a static quark–antiquark pair separated by distance R, in other words

$$F(\hat{r}_0) = -\frac{dV(R)}{dR}|_{R=\hat{r}_0} \ . \tag{1.108}$$

The continuum experimental value of r_0 is approximately $r_0 \approx 0.49\text{fm}$. Hence $\hat{r_0}$ can be used to set the lattice spacing non–perturbatively as $\hat{r}_0 = \frac{r_0}{a}$ implying that $a = \frac{r_0}{\hat{r}_0}$.

Another observable that can be used to set the lattice spacing is the *string tension*

$$\sigma = \lim_{R\to\infty} F(R) \tag{1.109}$$

which is the limit of energy stored in a flux–tube between a quark–antiquark pair held at an infinite separation. This quantity was used extensively in quenched QCD simulations to set the lattice scale. However in dynamical fermion simulations the flux tube is expected to break when it carries enough energy to create a quark–antiquark pair, and hence the preferred method of fixing the lattice spacing is through measuring $\hat{r}_0$.

1.3.5 Data Analysis

Once a simulation has been performed one generally has to subject the resulting data to some sort of analysis. This analysis can sometimes be quite simple, for example to estimate confidence limits on some ensemble average, or it can occasionally be quite complex, such as fitting a model to the data. All the techniques of analysis used in this thesis are fairly standard, and have little bearing on our main theme which is the subject of efficient Monte Carlo simulations of lattice QCD. Hence we have relegated details of these methods to appendix B

1.4 Summary of Chapter

In this introductory chapter we have outlined the main goals of this thesis, which is the efficient simulation of dynamical fermion lattice QCD systems.

We have shown how to formulate gauge and fermionic theories on a Euclidean space time lattice, stressing particularly the problems encountered when one deals with fermions.

We have outlined the basic ideas behind stochastic sampling methods and presented some of the theory of Markov processes paying particular regard to how they can be used to generate a sequence of gauge configurations distributed with the desired lattice QCD probability distribution.

We have outlined some observables of interest in lattice QCD, simulations in particular the mesonic correlation functions and the static potential and related observables.

In the next chapter we shall discuss the Hybrid Monte Carlo algorithm.

Chapter 2

Introduction To Hybrid Monte Carlo

In the previous chapter we have demonstrated how Monte–Carlo methods can be used to calculate path integrals. Unfortunately for dynamical fermion lattice QCD simulations the Metropolis method is not very efficient. In this chapter we present an overview of the Hybrid Monte Carlo (HMC) algorithm of [3] which has been the method of choice for performing dynamical fermion simulations for the last decade.

The HMC algorithm is fundamental to the remainder of this thesis. In the next few chapters we discuss the tuning of existing HMC simulations, whereas in the chapters thereafter we will use HMC as a component of other simulation algorithms.

Our overall aim is to give some idea of the computational cost of HMC simulations, so that in the chapters following we can discuss how tuning and novel algorithms can reduce (or increase) this cost. Hence our presentation will focus mostly on computational issues in the simulation. However there are some aspects of theoretical background that we also wish to clarify, in particular what conditions are required by the algorithm for correctness. These requirements of course also set some conditions on the software components used by the algorithm.

Our discussion of the algorithm will be split into four main sections. In the first we discuss the HMC algorithm for a single one dimensional Hamiltonian system, and show the conditions required for it to produce the correct equilibrium. Secondly, we will describe the relevant algorithmic steps when applied to lattice QCD. In the third section, we shall take a foray into the realm of Krylov subspace solvers. Finally we will discuss some issues of computational cost.

2.1 The HMC Algorithm

The HMC algorithm constructs a Markov process which satisfies detailed balance with respect to our desired probability distribution and is ergodic, as is required for it to reach the correct equilibrium distribution. In what follows we consider a system with a Hamiltonian function H of the form

$$H = \frac{1}{2}p^2 + S(q) \ , \tag{2.1}$$

where the S is the desired action, q is a canonical position and p is the momentum canonically conjugate to q.

Given an initial configuration (q, p_{last}) the algorithm proceeds according to the following scheme

1. **Momentum Refreshment:** A new set of canonical momenta p are chosen from a Gaussian heat-bath with zero mean and unit variance, to get configuration (q, p).

2. **Molecular Dynamics (MD):** Hamilton's equations of motion are integrated for some fictitious time τ with $(q(0), p(0)) = (q, p)$ to obtain the trial configuration $(q', p') = (q(\tau), p(\tau))$.

3. **Accept Reject Step:** The trial configuration (q', p') is accepted with a global Metropolis acceptance probability:

$$P_{\mathrm{acc}}(q', p' \leftarrow q, p) = \min\left(1, e^{-\Delta H}\right) , \tag{2.2}$$

where

$$\Delta H = H(q', p') - H(q, p) \ . \tag{2.3}$$

If the trial configuration is rejected the new state is (q, p),

2.1.1 Equilibrium Distribution

The scheme outlined above then generates a Markov process which, providing it is ergodic and satisfies detailed balance, converges asymptotically to the equilibrium distribution

$$P^{\mathrm{eq}}(q,p)\ \mathcal{D}q\ \mathcal{D}p = \frac{1}{Z} e^{-H(q,p)}\ \mathcal{D}q\ \mathcal{D}p \tag{2.4}$$

with

$$Z = \int \mathcal{D}q\ \mathcal{D}p\ e^{-H(q,p)}. \tag{2.5}$$

We note that for a Hamiltonian of form (2.1), the partition function factorises as

$$Z = \int \mathcal{D}p\ e^{-\frac{p^2}{2}} \int \mathcal{D}q\ e^{-S(q)} = Z_p Z_q \tag{2.6}$$

and the distribution of the coordinates q is given by

$$P^{\mathrm{eq}}(q)\ \mathcal{D}q = \frac{\mathcal{D}q}{Z} \int \mathcal{D}p\ e^{-H(q,p)} = \frac{\mathcal{D}q}{Z_p Z_q} e^{-S(q)} Z_p = \frac{1}{Z_q} e^{-S(q)}\ \mathcal{D}q\ . \tag{2.7}$$

Thus it can be seen that given an action $S(q)$, the transition to a Hamiltonian system does not affect the equilibrium, as the coordinates are still distributed with $P^{\mathrm{eq}}(q)\ \alpha\ e^{-S(q)}$. We also note that the "integration with respect to the momenta" is carried out by refreshing the momenta before every MD step.

However for convergence to equilibrium, the Markov process must be ergodic and obey detailed balance with respect to $P^{\mathrm{eq}}(q,p)$.

2.1.2 Ergodicity

If Hamilton's equations were integrated exactly in the MD step of the algorithm, the energy change ΔH along an MD[1] trajectory would be zero, as the Hamiltonian

[1] Whether the correct way of writing this should be "a MD trajectory" or "an MD trajectory" is a problem that is seemingly more taxing than the solution of QCD. We make the arbitrary choice of "an MD trajectory"

contains no explicit time dependence. Hence the MD would be restricted to a hyper-surface of constant H and the algorithm would not be ergodic in phase space. The momentum refreshment step however boosts the system to a different hyper-surface before each MD trajectory, thus ensuring ergodicity.

2.1.3 Detailed Balance, Reversibility and Area Preservation

As discussed in the previous chapter we would like the algorithm to have our desired equilibrium distribution as its fixed point, for which it is sufficient that the algorithm satisfies detailed balance. The HMC algorithm effectively makes two Markov steps, one is refreshing the momenta and the other is executing MD followed by an accept reject step. Both these steps have to have the same fixed point distribution. Since the momenta can be sampled directly from the correct equilibrium, this step trivially has the correct fixed point.

The detailed balance equation for the MD is

$$\begin{aligned} &P_{\text{acc}}(q',p' \leftarrow q,p)\ P_{\text{c}}(q',p' \leftarrow q,p)\ P^{\text{eq}}(q,p)\ \mathcal{D}q'\mathcal{D}p'\mathcal{D}q\mathcal{D}p \\ &\quad = P_{\text{acc}}(q,p \leftarrow q',p')\ P_{\text{c}}(q,p \leftarrow q',p')\ P^{\text{eq}}(q',p')\ \mathcal{D}q\mathcal{D}p\mathcal{D}q'\mathcal{D}p'\ . \end{aligned} \tag{2.8}$$

The definition for the global acceptance probability

$$P_{\text{acc}}(q',p' \leftarrow q,p) = \min\left(1, e^{-\Delta H}\right) = \min\left(1, \frac{P^{\text{eq}}(q',p')}{P^{\text{eq}}(q,p)}\right) \tag{2.9}$$

ensures that the relation:

$$P_{\text{acc}}(q',p' \leftarrow q,p)P^{\text{eq}}(q,p) = P_{\text{acc}}(q,p \leftarrow q',p')P^{\text{eq}}(q',p') \tag{2.10}$$

is identically satisfied. Hence for detailed balance we need

$$P_{\text{c}}(q',p' \leftarrow q,p)\ \mathcal{D}q'\mathcal{D}p'\mathcal{D}q\mathcal{D}p = P_{\text{c}}(q,p \leftarrow q',p')\ \mathcal{D}q\mathcal{D}p\mathcal{D}q'\mathcal{D}p'\ . \tag{2.11}$$

In HMC, the role of P_c is played by the MD step. Since MD is deterministic, P_c is of the form

$$P_{\text{c}}(q',p' \leftarrow q,p) = \delta(q',p' - M(q,p,\tau)), \tag{2.12}$$

where $M(q,p,\tau)$ is the mapping that represents MD evolution for some time τ starting with configuration (q,p). The delta function just states, that the outcome of the mapping must be (q',p').

Hence the detailed balance condition reduces to

$$\delta(q',p'-M(q,p,\tau))\ \mathcal{D}q'\mathcal{D}p'\mathcal{D}q\mathcal{D}p = \delta(q,p-M(q',p',-\tau))\mathcal{D}q'\mathcal{D}p'\mathcal{D}q\mathcal{D}p\ . \quad (2.13)$$

and it can be seen that we require M to be *reversible* in the sense that

$$M(q',p',-\tau) = M^{-1}(q,p,\tau)\ . \quad (2.14)$$

If this reversibility condition is satisfied, we can write

$$\begin{aligned}\delta(q,p-M(q',p',-\tau)) &= \delta(q,p-M^{-1}(q',p',\tau)) \\ &= \delta(M(q,p-M^{-1}(q',p',\tau),\tau))|\det(J(M))| \quad (2.15) \\ &= \delta(M(q,p,\tau)-q',p')|\det(J(M))|\ , \quad (2.16)\end{aligned}$$

where $J(M)$ is the Jacobian of the transformation M.

Clearly then (2.13) can only hold if

$$\det(J(M)) = 1 \quad (2.17)$$

which is the property of area preservation. Hence we have shown that reversibility and area preservation of the MD are sufficient for HMC to satisfy detailed balance.

We note that lack of reversibility and area preservation can be accounted for by changing the acceptance probability. However, it is generally non-trivial to compute how the acceptance probability needs to be modified. In practice therefore, both these conditions need to be satisfied.

2.2 Molecular Dynamics

We now go on to discuss some algorithms to integrate Hamilton's equations of motion. To set the scene we begin by recapping some ideas from classical mechanics

2.2.1 Classical Mechanics

Hamilton's equations of motion are (in terms of fictitious time τ):

$$\dot{q} = \frac{dq}{d\tau} = \frac{\partial H}{\partial p} = p \tag{2.18}$$

$$\dot{p} = \frac{dp}{d\tau} = -\frac{\partial H}{\partial q} = -\frac{\partial S}{\partial q} = F \tag{2.19}$$

where $F = -\frac{dS}{dq}|_\tau$ is the force at time τ.

Using these equations, the time evolution of a function of the co–ordinate q and momentum p, with no explicit time dependence, can be written as

$$\begin{aligned}\frac{df(q,p)}{d\tau} &= \frac{dq}{d\tau}\frac{\partial f}{\partial q} + \frac{dp}{d\tau}\frac{\partial f}{\partial p} \\ &= \frac{\partial H}{\partial p}\frac{\partial f}{\partial q} - \frac{\partial H}{\partial q}\frac{\partial f}{\partial p} \\ &= \{H, f(q,p)\}_{\mathrm{P}} \end{aligned} \tag{2.20}$$

where we have introduced $\{\}_{\mathrm{P}}$, the classical Poisson bracket of two functions of p and q. In a general $2N$ dimensional phase space, for functions f and g of the coordinates and momenta, the Poisson bracket is defined as

$$\{f, g\}_{\mathrm{P}} = \sum_i^N \left(\frac{\partial f}{\partial p_i} \partial_i g - \partial_i f \frac{\partial g}{\partial p_i} \right) \ , \tag{2.21}$$

where the ∂_i is a suitable derivative with respect to the coordinates q_i. In the case of our classical $1D$ system where p and q are just numbers we can write

$$\partial_i = \frac{\partial}{\partial q_i} \tag{2.22}$$

whereas for QCD the situation is more complicated due to the coordinates being members of a group.

From this formalism, it is of course immediately clear that if H contains no explicit time dependence, then it is a constant of the motion, since $\{H, H\}_{\mathrm{P}} = 0$.

If we define the operator $\hat{\mathcal{L}}_{\mathrm{H}}$ as

$$\hat{\mathcal{L}}_{\mathrm{H}} f(q,p) = \left(\frac{\partial H}{\partial p_i} \partial_i - \partial_i H \frac{\partial}{\partial p_i} \right) f(q,p) = \{H, f(q,p)\}_{\mathrm{P}} \ , \tag{2.23}$$

then the time evolution equation (2.20) can be formally solved as

$$f(q,p,\tau) = e^{\tau\hat{\mathcal{L}}_{\mathrm{H}}} f(q,p,0) = \hat{U}(\tau) f(q,p,0) \ , \tag{2.24}$$

where we have defined the time evolution operator $\hat{U}(\tau)$ as

$$\hat{U}(\tau) = e^{\tau\hat{\mathcal{L}}_{\mathrm{H}}} = e^{\tau\left(\frac{\partial H}{\partial p_i}\partial_i - \partial_i H \frac{\partial}{\partial p_i}\right)} \ . \tag{2.25}$$

In our simple system, we assume that the phase space is flat. In this case we can write

$$\hat{\mathcal{L}}_{\mathrm{H}} = p\frac{\partial}{\partial q} + F\frac{\partial}{\partial p} \ . \tag{2.26}$$

In particular, Liouville's theorem guarantees that $\hat{U}(\tau)$ preserves the phase space volume. Reversibility can be shown using the Baker-Campbell-Hausdorff formula:

$$e^A e^B = e^{A+B+\frac{1}{2}[A,B]+\mathrm{h.o.c}} \tag{2.27}$$

where h.o.c (higher order commutators) refers to commutators involving three or more operators. To do this, we write

$$\hat{U}(-\tau)\hat{U}(\tau) = e^{-\tau\hat{\mathcal{L}}_{\mathrm{H}}} e^{\tau\hat{\mathcal{L}}_{\mathrm{H}}} = e^{0-\frac{\tau^2}{2}[\hat{\mathcal{L}}_{\mathrm{H}},\hat{\mathcal{L}}_{\mathrm{H}}]+\mathrm{h.o.c}} \ . \tag{2.28}$$

However as $\hat{\mathcal{L}}_{\mathrm{H}}$ commutes with itself, all the commutators vanish leaving

$$\hat{U}(-\tau)\hat{U}(\tau) = 1 \ \Rightarrow \ \hat{U}(-\tau) = \hat{U}(\tau)^{-1} \tag{2.29}$$

which is the desired reversibility condition.

2.2.2 Leapfrog Integration

Let us now consider splitting the operator $\hat{\mathcal{L}}_{\mathrm{H}}$ into two parts $\hat{\mathcal{L}}_{\mathrm{p}}$ and $\hat{\mathcal{L}}_{\mathrm{q}}$ with

$$\hat{\mathcal{L}}_{\mathrm{q}} = \frac{\partial H}{\partial p}\frac{\partial}{\partial q} = p\frac{\partial}{\partial q} \tag{2.30}$$

$$\hat{\mathcal{L}}_{\mathrm{p}} = \frac{-\partial H}{\partial q}\frac{\partial}{\partial p} = F\frac{\partial}{\partial p} \tag{2.31}$$

so that

$$\hat{\mathcal{L}}_{\mathrm{H}} = \hat{\mathcal{L}}_{\mathrm{q}} + \hat{\mathcal{L}}_{\mathrm{p}} \ . \tag{2.32}$$

While it is not in general possible to write down an operator $\hat{U}(\tau) = e^{\tau\hat{\mathcal{L}}_{\mathrm{H}}}$ in closed form it is possible to write the operators as

$$\begin{aligned} \hat{U}_q(\tau) &= e^{\tau\hat{\mathcal{L}}_{\mathrm{q}}} = 1 + p\tau\frac{\partial}{\partial q} + \text{h.o.t} && (2.33) \\ \hat{U}_p(\tau) &= e^{\tau\hat{\mathcal{L}}_{\mathrm{p}}} = 1 + F\tau\frac{\partial}{\partial p} + \text{h.o.t} && (2.34) \end{aligned}$$

where h.o.t refers to higher order terms in the series expansion of the exponential. When applying $\hat{U}_q$ to a coordinate $q(0)$, terms with more than one derivative with respect to q vanish. Likewise derivatives of second order or greater with respect to p vanish when applying $\hat{U}_p$ to a momentum $p(0)$, hence

$$\begin{aligned} \hat{U}_q(\tau)(q(0),p(0)) &= ((1 + p(0)\tau\frac{\partial}{\partial q}|_{\tau=0} + \text{h.o.t})\ q(0), p(0)) \\ &= (q(0) + p(0)\tau, p(0)) && (2.35) \end{aligned}$$

and

$$\begin{aligned} \hat{U}_p(\tau)(q(0),p(0)) &= (q(0), (1 + F\tau\frac{\partial}{\partial p}|_{\tau=0} + \text{h.o.t})\ p(0)) \\ &= (q(0), p(0) + F\tau)\ . && (2.36) \end{aligned}$$

Unfortunately, as $\hat{\mathcal{L}}_{\mathrm{q}}$ and $\hat{\mathcal{L}}_{\mathrm{p}}$ do not commute,

$$\hat{U}_q(\tau)\hat{U}_p(\tau) = e^{\tau\hat{\mathcal{L}}_{\mathrm{q}}}e^{\tau\hat{\mathcal{L}}_{\mathrm{p}}} = e^{\tau(\hat{\mathcal{L}}_{\mathrm{q}}+\hat{\mathcal{L}}_{\mathrm{p}})+\frac{\tau^2}{2}[\hat{\mathcal{L}}_{\mathrm{q}},\hat{\mathcal{L}}_{\mathrm{p}}]} \neq e^{\hat{\mathcal{L}}_{\mathrm{q}}+\hat{\mathcal{L}}_{\mathrm{p}}} = e^{\hat{\mathcal{L}}_{\mathrm{H}}}\ . \quad (2.37)$$

However, the update $\hat{U}_p(\tau)\hat{U}_q(\tau) = e^{\tau\hat{\mathcal{L}}_{\mathrm{H}}+O(\tau^2)}$. This suggests that for small timestep $\delta\tau$ the true evolution operator will be well approximated by $\tilde{U}(\delta\tau) = \hat{U}_p(\delta\tau)\hat{U}_q(\delta\tau)$. This leads us to the class of symplectic integrator algorithms for integrating Hamilton's equations numerically. The operators $\hat{U}_p(\delta\tau)$ and $\hat{U}_q(\delta\tau)$ are known as symplectic building blocks. One can then consider building various approximations to $\hat{U}(\delta\tau)$ from them.

It is easy to show that $\hat{U}_p$ and $\hat{U}_q$ are individually area preserving by examining the determinant of the Jacobian matrix for each one:

$$\det(J(\hat{U}_q(\tau))) = \det\begin{pmatrix} \frac{\partial(q(0)+p(0)\tau)}{\partial q(0)} & \frac{\partial p(0)}{\partial q(0)} \\ \frac{\partial(q(0)+p(0)\tau)}{\partial p(0)} & \frac{\partial p(0)}{\partial p(0)} \end{pmatrix} = \det\begin{pmatrix} 1 & 0 \\ \frac{\partial(q(0)+p(0)\tau)}{\partial p(0)} & 1 \end{pmatrix} = 1 \quad (2.38)$$

$$\det(J(\hat{U}_p(\tau))) = \det \begin{pmatrix} \frac{\partial q(0)}{\partial q(0)} & \frac{\partial (p(0)+F(q(0))\tau}{\partial q(0)} \\ \frac{\partial q(0)}{\partial p(0)} & \frac{\partial (p(0)+F(q(0))\tau)}{\partial p(0)} \end{pmatrix} = \det \begin{pmatrix} 1 & \frac{\partial (p(0)+F(q(0))\tau}{\partial q(0)} \\ 0 & 1 \end{pmatrix} = 1 \ . \tag{2.39}$$

Hence an approximation for $\hat{U}$ built from any product of the $\hat{U}_q$ and the $\hat{U}_p$ will also be area preserving.

The symplectic building block operators are themselves also reversible:

$$e^{-\delta\tau\hat{\mathcal{L}}_{\mathrm{q}}} e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{q}}} = e^{0-\frac{\delta\tau^2}{2}[\hat{\mathcal{L}}_{\mathrm{q}},\hat{\mathcal{L}}_{\mathrm{q}}]+\mathrm{h.o.c}} = 1 \tag{2.40}$$

$$e^{-\delta\tau\hat{\mathcal{L}}_{\mathrm{p}}} e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{p}}} = e^{0-\frac{\delta\tau^2}{2}[\hat{\mathcal{L}}_{\mathrm{p}},\hat{\mathcal{L}}_{\mathrm{p}}]+\mathrm{h.o.c}} = 1 \tag{2.41}$$

as the commutators of operators with themselves vanish. Unfortunately the suggested $\tilde{U}(\delta\tau) = \hat{U}_q(\delta\tau)\hat{U}_p(\delta\tau)$ is not reversible as

$$\begin{aligned} \tilde{U}(\delta\tau)^{-1} &= \left(\hat{U}_q(\delta\tau)\hat{U}_p(\delta\tau)\right)^{-1} \\ &= \hat{U}_p(\delta\tau)^{-1}\hat{U}_q(\delta\tau)^{-1} = \hat{U}_p(-\delta\tau)\hat{U}_q(-\delta\tau) \\ &\neq \hat{U}_q(-\delta\tau)\hat{U}_p(-\delta\tau) = \tilde{U}(\delta\tau) \ . \end{aligned} \tag{2.42}$$

The time symmetric combination

$$\tilde{U}_3(\delta\tau) = \hat{U}_p\left(\frac{\delta\tau}{2}\right)\hat{U}_q(\delta\tau)\,\hat{U}_p\left(\frac{\delta\tau}{2}\right) \tag{2.43}$$

is reversible however,

$$\begin{aligned} \tilde{U}_3(\delta\tau)^{-1} &= \left[\hat{U}_p\left(\frac{\delta\tau}{2}\right)\hat{U}_q(\delta\tau)\,\hat{U}_p\left(\frac{\delta\tau}{2}\right)\right]^{-1} \\ &= \hat{U}_p\left(\frac{\delta\tau}{2}\right)^{-1}\hat{U}_q(\delta\tau)^{-1}\,\hat{U}_p\left(\frac{\delta\tau}{2}\right)^{-1} \\ &= \hat{U}_p\left(\frac{-\delta\tau}{2}\right)\hat{U}_q(-\delta\tau)\,\hat{U}_p\left(\frac{-\delta\tau}{2}\right) = \tilde{U}_3(-\delta\tau) \ . \end{aligned} \tag{2.44}$$

It is clear that the reversibility is a result of the symmetric nature of $\tilde{U}_3$ with respect to the reversal of the order of the building blocks, and the reversibility of each one. Such approaches to constructing exactly reversible approximations to $\hat{U}$ are known as *symmetric symplectic algorithms*. In particular, the method for constructing $\tilde{U}_3$ is known as the *leap–frog* algorithm, due to the nature of updating initially the momenta for half a timestep, followed by updating the coordinates for a whole timestep ("leapfrogging over the momenta") and then finally updating the momenta for half a timestep to complete the full update.

The reversible nature of $\tilde{U}_3(\delta\tau)$ also ensures that errors in the coordinates, momenta and energy arise at $O(\delta\tau^3)$ for a single step of the leapfrog algorithm. Using the Baker-Campbell-Hausdorff formula one finds that

$$\begin{aligned}
\tilde{U}_3(\delta\tau) &= e^{\frac{\delta\tau}{2}\hat{\mathcal{L}}_{\rm p}} e^{\delta\tau\hat{\mathcal{L}}_{\rm q}} e^{\frac{\delta\tau}{2}\hat{\mathcal{L}}_{\rm p}} \\
&= e^{\delta\tau(\hat{\mathcal{L}}_{\rm p}+\hat{\mathcal{L}}_{\rm q})+\frac{\delta\tau^2}{4}[\hat{\mathcal{L}}_{\rm p},\hat{\mathcal{L}}_{\rm q}]+\frac{\delta\tau^2}{4}[\hat{\mathcal{L}}_{\rm q},\hat{\mathcal{L}}_{\rm p}]+O(\delta\tau^3)} \\
&= e^{\delta\tau(\hat{\mathcal{L}}_{\rm p}+\hat{\mathcal{L}}_{\rm q})+\frac{\delta\tau^2}{4}[\hat{\mathcal{L}}_{\rm p},\hat{\mathcal{L}}_{\rm q}]-\frac{\delta\tau^2}{4}[\hat{\mathcal{L}}_{\rm p},\hat{\mathcal{L}}_{\rm q}]+O(\delta\tau^3)} = e^{\delta\tau\hat{\mathcal{L}}_{\rm H}+O(\delta\tau^3)} \qquad (2.45)
\end{aligned}$$

This result can also be shown by considering the reversibility condition

$$\tilde{U}_3(-\delta\tau) = \tilde{U}_3(\delta\tau)^{-1}\ . \qquad (2.46)$$

Since $\tilde{U}_3(\delta\tau)$ is an operator of the form $e^{f(\delta\tau)}$, equation (2.46) basically states that $f(-\delta\tau) = -f(\delta\tau)$ so that the exponent is an odd function of $\delta\tau$. This guarantees the disappearance of all even powers of $\delta\tau$ from the exponent in (2.45).

2.2.3 Higher Order Integration Schemes

It has been noticed in [22] that the combination

$$\tilde{U}_5(\delta\tau) = \tilde{U}_3\left(\frac{\delta\tau}{2-2^{\frac{1}{3}}}\right)\tilde{U}_3\left(\frac{-\delta\tau}{2^{\frac{2}{3}}-1}\right)\tilde{U}_3\left(\frac{\delta\tau}{2-2^{\frac{1}{3}}}\right) \qquad (2.47)$$

is accurate to $O(\delta\tau^5)$. This result was later generalised in [23] to allow one to construct schemes accurate to arbitrary odd orders of $\delta\tau$.

The construction is recursive, beginning with the leap-frog method as the base case. We have already shown that a single step of this method is accurate to $O(\delta\tau^3)$. Let us now consider the time following symmetric combination of the $\tilde{U}_3$:

$$\tilde{U}_5(\delta\tau) = \tilde{U}_3(\delta\tau_1)\tilde{U}_3(\delta\tau_2)\tilde{U}_3(\delta\tau_1))\ . \qquad (2.48)$$

Clearly if the first and last updates use the same timestep $\delta\tau_1$ the scheme is still reversible. This implies that there are no errors of $O(\delta\tau^4)$ (or any other even order of $\delta\tau$). The individual $\tilde{U}_3$ updates do have leading order discretisation errors of $O(\delta\tau^3)$, however, the timestep $\delta\tau_2$ in the central leapfrog update can be chosen to cancel exactly the $O(\delta\tau^3)$ errors from the first and last update, leaving a leading discretisation error of $O(\delta\tau^5)$. One can then construct $O(\delta\tau^7)$ algorithms from the $O(\delta\tau^5)$ schemes and so on.

More generally it has been shown in [23] that if one has a scheme $\tilde{U}_n$ which is accurate in terms of discretisation errors to some even power n of $\delta\tau$ and is reversible and area preserving, the next higher order integrator can be constructed as

$$\tilde{U}_{n+2}(\delta\tau) = \tilde{U}_n(\delta\tau_1)^i \tilde{U}_n(\delta\tau_2) \tilde{U}_n(\delta\tau_1)^i \ , \tag{2.49}$$

where suitable choices for $\delta\tau_1$ and $\delta\tau_2$ to cancel errors of $O(\delta\tau^{n+1})$ are[2]

$$\begin{aligned} \delta\tau_1 &= \frac{\delta\tau}{2i - s} && (2.50) \\ \delta\tau_2 &= \frac{-\delta\tau}{\frac{2i}{s} - 1} \ . && (2.51) \end{aligned}$$

In the equations above, i is an arbitrary positive integer and

$$s = (2i)^{\frac{1}{n+1}} \ . \tag{2.52}$$

Reversibility ensures that there are no discretisation errors of $O(\delta\tau^{n+2})$ and hence in such a scheme the leading order discretisation error is $O(\delta\tau^{n+3})$. In particular the scheme of [22], with which we began this discussion is a version of this general scheme with $i = 1$ and $n = 2$.

2.2.4 The Sexton–Weingarten Trick

In [24] an update algorithm was proposed which is possibly more efficient than leapfrog, in the situation when the Hamiltonian can be split as

$$H(q,p) = \frac{1}{2}p^2 + S_1(q) + S_2(q) \ . \tag{2.53}$$

Lattice QCD with dynamical fermions is just such a case as the action can be split into gauge and fermionic parts

$$S(U,\phi^\dagger,\phi) = S_{pg}(U) + S_f(U,\phi^\dagger,\phi) \ . \tag{2.54}$$

[2] We have rescaled the timestep here so that the overall "time–of–flight" for the update is $\delta\tau$

Proceeding as before in the case of the leapfrog updates one can now define

$$\hat{\mathcal{L}}_{\mathrm{p}_1} = F_1 \frac{\partial}{\partial p} \quad \text{and} \tag{2.55}$$

$$\hat{\mathcal{L}}_{\mathrm{p}_2} = F_2 \frac{\partial}{\partial p} , \tag{2.56}$$

with $F_1 = -\frac{\partial S_1(q)}{\partial q}$ and $F_2 = -\frac{\partial S_2(q)}{\partial q}$. We then define the corresponding partial update operators

$$\tilde{U}_1(\delta\tau) = e^{\frac{1}{2}\delta\tau \hat{\mathcal{L}}_{\mathrm{p}1}} e^{\delta\tau \hat{\mathcal{L}}_{\mathrm{q}}} e^{\frac{1}{2}\delta\tau \hat{\mathcal{L}}_{\mathrm{p}1}} , \tag{2.57}$$

$$\tilde{U}_2(\delta\tau) = e^{\delta\tau \hat{\mathcal{L}}_{\mathrm{p}2}} \tag{2.58}$$

and the full update operator can be written as

$$\tilde{U}_S(\delta\tau) = \tilde{U}_2\left(\frac{\delta\tau}{2}\right)\left[\tilde{U}_1\left(\frac{\delta\tau}{n}\right)\right]^n \tilde{U}_2\left(\frac{\delta\tau}{2}\right) \tag{2.59}$$

for a positive integer n.

The splitting is accurate to $O(\delta\tau^3)$ as before, but now the coefficient multiplying the leading error term decreases as $\frac{1}{n^2}$. Similar error reduction techniques can also be employed in the higher order schemes outlined earlier.

The main advantage of the Sexton–Weingarten trick therefore is a smaller $O(\delta\tau^3)$ error term when the operator $e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{p}1}}$ is applied more often than $e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{p}2}}$. For this to be competitive $e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{p}1}}$ has to be much cheaper to compute than $e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{p}2}}$. This is the situation in the case of lattice QCD where $e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{p}1}}$ corresponds to the update of momenta due to the force from gauge fields, which is a cheap operation to compute. However, the $e^{\delta\tau\hat{\mathcal{L}}_{\mathrm{p}2}}$ corresponds to the momentum update due the force from the fermion fields which is expensive as it requires a solution of a large system of linear equations.

2.3 Acceptance Rate

It can be seen from equation (2.2) that the acceptance probability, P_{acc}, following an MD trajectory, depends on the energy change, ΔH, along the trajectory. In our discussion of instabilities we will show that one of the problems these cause is to increase ΔH on average and consequently cause the rate of acceptance to drop. To support this claim, we will now derive the formula for the acceptance rate of HMC, and thus make clearer the link between the average ΔH and the average acceptance. We note that this derivation is not new and can also be found elsewhere [25, 8]. We note also that this calculation basically relies on detailed balance, and area preservation and we shall use similar arguments when discussing the acceptance rate of the PT algorithm in chapter 5.

Consider two field configurations ($q' = q(\tau), p' = p(\tau)$) and ($q = q(0), p = p(0)$) at the two ends of a molecular dynamics trajectory. The corresponding values of the Hamiltonian are $H(q', p')$ and $H(q, p)$ respectively. The change in the Hamiltonian along the trajectory is ΔH.

As the MD is measure preserving we can write for the partition function that

$$Z = \int \mathcal{D}q' \ \mathcal{D}p' \ e^{-H(q',p')} = \int \mathcal{D}q \ \mathcal{D}p \ e^{-H(q',p')} = \int \mathcal{D}p \ \mathcal{D}q \ e^{-H(q,p)-\Delta H}. \quad (2.60)$$

We also have that

$$Z = \int \mathcal{D}q \ \mathcal{D}p \ e^{-H(q,p)} \ . \quad (2.61)$$

Dividing (2.60) by (2.61) we obtain the result of [26]:

$$1 = \frac{1}{Z} \int \mathcal{D}q \ \mathcal{D}p \ e^{-\Delta H} \ e^{-H(q,p)} = \left\langle e^{-\Delta H} \right\rangle \ . \quad (2.62)$$

If we now consider ΔH to be a random variable, we can appeal to the central limit theorem, and in the limit of a large number of samples, write down a Gaussian probability density

$$P(\Delta H) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\Delta H - \langle \Delta H \rangle)^2}{2\sigma^2}} \quad (2.63)$$

for ΔH, with mean $\langle \Delta H \rangle$ and variance σ^2. The constant coefficient $\frac{1}{\sqrt{2\pi\sigma^2}}$ in the above expression serves merely to normalise the integral over ΔH to unity.

Combining (2.63) and (2.62) it is immediately clear that

$$\left\langle e^{-\Delta H} \right\rangle = \frac{1}{\sqrt{2\pi\sigma^2}} \int d(\Delta H) \ e^{-\Delta H} \ e^{-\frac{(\Delta H - \langle \Delta H \rangle)^2}{2\sigma^2}} = 1 \ . \quad (2.64)$$

Carrying out the integral leaves

$$e^{\frac{\sigma^2}{2} - \langle \Delta H \rangle} = 1 \ , \quad (2.65)$$

from which it follows that

$$\sigma^2 = 2\langle \Delta H \rangle . \tag{2.66}$$

Inserting this value of σ^2 into (2.63) we obtain for the average acceptance rate,

$$\langle P_{\text{acc}} \rangle = \frac{1}{\sqrt{4\pi \langle \Delta H \rangle}} \int d(\Delta H) \, \min\left\{1, e^{-\Delta H}\right\} \, e^{-\frac{(\Delta H - \langle \Delta H \rangle)^2}{4\langle \Delta H \rangle}} . \tag{2.67}$$

which, on carrying out the integral over ΔH yields

$$\langle P_{\text{acc}} \rangle = \text{erfc}\left(\frac{1}{2}\langle \Delta H \rangle^{\frac{1}{2}}\right) = \text{erfc}\left(\left[\frac{1}{8}\sigma^2\right]^{\frac{1}{2}}\right) . \tag{2.68}$$

2.4 Formulation for Lattice QCD

The theoretical problem with formulating the HMC method for lattice QCD effectively boils down to the problem of defining molecular dynamics in a group manifold. This problem has been solved and it is not our intention to discuss it here. We refer the reader to [27] for more details. In this section, we content ourselves with presenting the definitions of the conjugate momenta and the gauge and momentum updates for the leap–frog algorithm.

2.4.1 Conjugate Momentum Fields and Hamiltonian

The conjugate momenta for the SU(3) group valued link matrices $U_\mu(x)$ are $\pi_\mu(x)$, which belong to the Lie algebra $su(3)$. In particular the π fields can be represented as a linear combination of the Gell–Mann (GM) matrices λ_i (the $SU(3)$ generators are given by $\frac{\lambda_i}{2}$). Thus

$$\pi_\mu(x) = \sum_{i=1}^{8} \pi_\mu^i(x)\lambda_i \tag{2.69}$$

and the storage of the momentum fields reduces to storing an 8-vector of the components $\pi_\mu^i(x)$ for each link of the lattice. The GM matrices are themselves normalised through the relation Tr $(\lambda^a \lambda^b) = 2\delta_{ab}$.

The corresponding Hamiltonian function in the presence of pseudofermion fields ϕ is

$$H(\pi, U, \phi) = \frac{1}{2} \sum_{x,\mu,a} (\pi^a_\mu(x))^2 + S(U, \phi) \ . \tag{2.70}$$

where the action S is the usual lattice QCD action of (1.47).

Hence evaluating the Hamiltonian requires the vector

$$X = Q(U; \kappa, c)\ \phi = \left(M^\dagger(U; \kappa, c) M(U; \kappa, c)\right)^{-1}\ \phi \tag{2.71}$$

where Q is as defined in (1.48), or alternatively

$$Y = M^{-\dagger}(U, \kappa, c)\phi \tag{2.72}$$

where the notation $M^{-\dagger}$ has been introduced as a convenient way of denoting $(M^\dagger)^{-1} = (M^{-1})^\dagger$. In the first case, $H = \phi X$ whereas in the second, $H = Y^\dagger Y$. Obtaining X and/or Y requires solution of the linear system(s)

$$M^\dagger M\ X = \phi \qquad \text{and/or} \qquad M^\dagger\ Y = \phi \tag{2.73}$$

respectively. Details of how this is done, will be given in section 2.5 on Krylov Subspace Iterative Solvers.

2.4.2 Leapfrog Momentum Update and the Pseudofermion Force

The leapfrog momentum update step for lattice QCD is the transformation

$$\pi_\mu(x, \tau) \rightarrow \pi_\mu(x, \tau + \delta\tau) = \pi_\mu(x, \tau) + F_\mu(x)\delta\tau \tag{2.74}$$

where F is the force, which has the form

$$F_\mu(x) = \dot{\pi} = -\frac{\partial S_{pg}}{\partial U_\mu(x)} + \left[\left(M^\dagger M\right)^{-1} \phi\right]^\dagger \frac{\partial}{\partial U_\mu(x)} \left(M^\dagger M\right) \left[\left(M^\dagger M\right)^{-1} \phi\right] \ . \tag{2.75}$$

We can see that we can write the force as $F = F_{\mathrm{gauge}} + F_{\mathrm{fermion}}$ allowing us to make use of the Sexton–Weingarten trick. The explicit form of F for the clover action used in our simulation can be found in Appendix B of [13].

It can also be seen from (2.75) that computation of the vector X defined in (2.73) is required to evaluate F. In the case of the clover action, the computation of the vector Y is also needed [13].

2.4.3 Leapfrog Gauge Update

The gauge field update in the leapfrog algorithm for lattice QCD is

$$U_\mu(x,\tau) \to U_\mu(x,\tau+\delta\tau) = \exp(i\ \pi_\mu(x)\ \delta\tau)U_\mu(x,\tau)\ . \tag{2.76}$$

For the gauge update step to be reversible it is necessary that the matrix $\exp(i\ \pi_\mu(x)\ \delta\tau)$ be unitary[3] so that

$$\exp(i\ \pi_\mu(x)\ \delta\tau)^\dagger = \exp(i\ \pi_\mu(x)\ \delta\tau)^{-1}\ . \tag{2.77}$$

The condition of reversibility can then be stated as

$$\exp(i\ \pi_\mu(x)\ \delta\tau) = \exp(-i\ \pi_\mu(x)\ \delta\tau)^{-1} = \exp(i\ \pi_\mu(x)\ \delta\tau)^\dagger \tag{2.78}$$

The challenging task is to find a method of computing the exponential of the momenta in (2.76) so that both the conditions (2.77) and (2.78) hold to a sufficiently high accuracy.

There exists a method presented originally in [28] and discussed also in [29, 30] for carrying out this matrix exponentiation, which uses the Cayley–Hamilton theorem. This theorem states that any function of a matrix $\mathcal{M} \in su(3)$ can be expressed as a second degree polynomial in $\mathcal{M}$. We now give a very brief introduction to the ideas underlying the algorithm. We refer the reader to [30] for details.

The determinant of the matrix $\mathcal{M}$ can always be expressed as a polynomial in powers of Tr $\mathcal{M}$. This fact also means that the characteristic equation and the eigenvalues of $\mathcal{M}$ can also be expressed in terms of this trace.

The exponential $e^{\mathcal{M}}$ can be written down in terms of these eigenvalues. However due to the Cayley–Hamilton theorem, $e^{\mathcal{M}}$ can also be written as

$$e^{\mathcal{M}} = a\mathcal{M}^2 + b\mathcal{M} + c\ , \tag{2.79}$$

[3] Even tho' we stated that the momenta are stored as 8-vectors, it must be borne in mind that in reality they are matrices

which allows one to set up a system of 3 linear equations relating the exponential in terms of the eigenvalues, to the coefficients a, b and c. This simple set of equations can also be solved analytically to give exact formulae for the coefficients in terms of Tr $\mathcal{M}$, from which the polynomial can be reconstructed. Thus for exponentiating a given matrix one merely needs to evaluate a few traces, the formulae for the coefficients a,b and c and a second degree matrix polynomial.

In particular a power series expansion of the desired coefficients a, b and c has been determined to a high order thus making the process of computing them very straightforward using Horner's rule at the expense of the programmer having to deal with a set of "magic–numbers". Should the coefficients not be accurate enough in terms of this power series definition one can always fall back to computing them from their analytic definitions at a slightly higher (but still low) cost.

This method is exact in the sense that it does not use truncated power series approximation for the exponential, neither does it need to numerically diagonalise the matrix $\mathcal{M}$, to perform the exponentiation of the eigenvalues.

2.5 On Krylov Solvers

We have shown that in order to compute the Hamiltonian function, or the force in the momentum update one has to compute the vector X of (2.73). This computation involves the solution of a very large sparse linear system.

There is a branch of numerical mathematics devoted exclusively to the study of algorithms for performing this task. A whole wealth of literature exists on the subject and the interested reader is referred to [31, 32] for a good introduction. Currently the most computationally competitive solver algorithms belong to the class of Iterative Krylov–Subspace solvers. We give here a brief introduction to how these methods work, and cite two algorithms in common use in lattice QCD simulations.

2.5.1 Krylov Solvers in general

The problem in hand is to solve the general linear system

$$Ax = b \tag{2.80}$$

where A is some matrix and x and b are vectors. We assume here that A is non-singular and there exists a unique solution x.

The residual r_i for some guess/approximation x_i to the true solution x is defined as

$$r_i = b - Ax_i \tag{2.81}$$

It can be shown by an inductive process, that if one begins with an initial guess vector x_0 and a corresponding residual r_0, then successive approximate solutions x_i will be members of the i-th dimensional Krylov subspace

$$K^i(A, x_0) = \text{span}\left\{A^i x_0, A^{i-1}x_0, \ldots, Ax_0, x_0\right\} \tag{2.82}$$

and the corresponding residuals r_i are members of $K^i(A, r_0)$.

Given an approximate solution x_i and residual r_i in the Krylov subspace $K^i(A, r_0)$, the task of an iterative solution procedure is to find a better approximation x_{i+1}, for which a basis for $K^{i+1}(A, r_0)$ is needed. A single step of a Krylov subspace solver essentially involves enlarging an i dimensional Krylov subspace to an $i+1$ dimensional one. The manner in which this is done is the defining feature of the iterative method under consideration. In exact arithmetic all algorithms terminate in $O(N)$ steps, where $N = \dim(A)$, as then the Krylov Subspace, cannot be enlarged any further.

In practice for very large matrices, the presence of rounding errors implies that one is searching for a solution in a space that is effectively infinite dimensional. In this case the algorithm terminates according to some other stopping criterion such as the norm of the relative residual $\frac{||r_i||}{||b||}$ being smaller than some pre-defined "target relative residual" $\frac{||r||}{||b||}$.

If A is hermitean positive definite then the form

$$(r_i, r_j)_A = (r_i, Ar_j) = r_i^\dagger A_{ij} r_i \tag{2.83}$$

forms an inner product. In particular one can define a norm

$$||r||_A^2 = (r, r)_A \tag{2.84}$$

and one can define an orthogonality relation between the vectors r_i and r_j as

$$(r_i, r_j)_A = 0 \ . \tag{2.85}$$

If these conditions are satisfied, there exists an optimal way to enlarge an existing Krylov-subspace by choosing the new basis vector r_{i+1} orthogonal to all the existing basis vectors.

A particularly efficient way of generating a sequence of Krylov subspaces of increasing dimensionality with an orthogonal set of basis vectors at each step, is to use a symmetric Lánczos process. For this algorithm the residuals satisfy the three term recurrence relation

$$\alpha_{i+1} r_{i+1} = A r_i - \beta_i r_i - \gamma_i r_{i-1} \tag{2.86}$$

where α_{i+1}, β_i and γ_i are constants to be determined by the algorithm. Given that the process needs the vector Ar_i in the computation of β_i, one can compute r_{i+1} using only vector operations instead of using (2.81).

We note that computing the residuals using such a recurrence relation seems more stable with respect to roundoff errors than using (2.81). Hence in finite precision one has to be careful to make sure there is no significant deviation between r_i computed through the recurrence relation – *the accumulated residue* – and r_i calculated from x_i, b and A using (2.81) – *the real residue*. The discrepancy between these two variants of r can be made small by restarting the process once a solution has been found. If the solution is indeed the correct one, no more work will have to be done in the restarted solve, and the two definitions of the residue will agree as they are the same by definition on the first iteration.

The Conjugate Gradient (CG) algorithm [33] is equivalent to performing a symmetric Lánczos process, but is restricted to solving linear systems whereas the Lánczos process is more general and is usually used to calculate eigenspectra. In the CG algorithm, the three term recurrence (2.86) reduces to two coupled two term recurrence relations.

As r_i is a member of the Krylov subspace $K^i(A, r_0)$, it can be expressed as $P^i(A)r_0$ where $P^i(A)$ is a matrix polynomial of order i in A. Hence the CG process can also be thought of as generating a sequence of polynomials $P^i(A)$ of increasing order, which are mutually orthogonal with respect to some suitable inner product. The algorithm is optimal in the sense that $x_i = x_0 + P^i(A)r_0$ minimises the error $||x - x_i||_A^2$ with respect to all other polynomials $Q^i(A)$. The convergence of the algorithm can be analysed using the Chebyshev polynomials and it can be shown that the norm of the error decreases monotonically with the number of iterations.

If the matrix A is not hermitean or symmetric, one does not necessarily have an orthogonality relation to define the Lánczos process. However one can use a biorthogonality relation

$$(r, A^\dagger s) = (Ar, s) = 0 \tag{2.87}$$

to construct bases for two Krylov subspaces $K^i(A, r_0)$ and $K^i(A^\dagger, s_0)$. This is the idea behind the unsymmetric Lánczos process, and the Stabilised Bi-Conjugate Gradients (BiCGStab) algorithm [34]. These methods still use recurrence relations to compute the residual at step i but now a monotonic decrease in the error is no longer guaranteed.

2.5.2 Krylov Solvers in QCD

The most popular solvers in use in lattice QCD are the CG and BiCGStab methods. With the CG method, one can obtain X directly and the solution is guaranteed to converge monotonically as $M^\dagger M$ is a hermitean positive definite matrix. Should one need Y it can be obtained from X by a single matrix multiplication. Generally a few restarts can ensure that accumulated and computed residuals are equal.

With the BiCGStab method one usually proceeds by computing Y and then using Y to compute X. In this case the solution process is two stage and the error in X is compounded by the fact that there is an error already in the computation of Y. Hence it is prudent to restart the solution process one or more times using CG, with the solution X of the last solve as the initial guess. If X is accurate enough, the CG solve should take a couple of iterations only. An accurate value of Y can then be recovered from X through a single matrix multiplication.

It may seem, that solving via BiCGStab is a somewhat more complex procedure than solving with CG. However in the case of CG the matrix A is $M^\dagger M$ whereas for BiCGStab it is M or $M^\dagger$. Correspondingly, the condition number of the system solved by CG is the square of the one solved by BiCGStab. One therefore expects that BiCGStab should converges substantially faster than CG.

In practice our simulations use the BiCGStab method with a few CG restarts to make sure the results X and Y are accurate enough. We have noticed that sometimes in the case of light sea quarks, the first CG restart can take many iterations to converge to the desired accuracy. This can be a sign of reaching the limits of numerical precision, when the true residue can be several orders of magnitude larger than the accumulated residue. It is also probably a sign of making the algorithm termination criterion smaller than the maximum available machine precision.

2.5.3 Stopping Criteria

We have mentioned before that although in exact arithmetic Krylov subspace algorithms terminate in $O(N)$ steps, it should be mentioned that $N = O(V)$ where V is the number of sites on the lattice. If we add in rounding error effects we find that we are effectively looking for the solution in an infinite dimensional space, and that the solution algorithm can proceed for a very large number of iterations.

Often the solution is as accurate as finite precision allows much before one has exhausted the available Krylov subspaces and it is desirable for the algorithm to stop at this stage.

The stopping criterion used in our simulations is to check whether the relative error $\frac{||r_i||_2}{||b||_2}$ is smaller than some pre-defined tolerance r. We loosely refer to r as the *target residue* although in reality it should be called the target relative error. However, quite a wealth of possible definitions are available for r_i. It can be the accumulated or the real residue. Secondly it can be the residue for CG algorithm:

$$r_i^{CG} = \phi - M^\dagger(\kappa, c)M(\kappa, c))X_i \tag{2.88}$$

or for the BiCGStab algorithm

$$\begin{aligned} r_i^{BiCGStab}(Y) &= \phi - M^\dagger Y_i && (2.89) \\ r_i^{BiCGStab}(X) &= (Y + \epsilon) - MX_i && (2.90) \end{aligned}$$

where ϵ is the error in Y from the first solve.

The latter problem is relatively easy to answer. Since even when we use BiCGStab, the solution is finished using CG it is best if the target residual r is defined as the relative error using r_i^{CG}. As for the first quandary, we make the choice to use the variant of the relative error which uses the accumulated residue in the stopping criterion. In this case however we must remain aware of the fact that in a single precision (32-bit) solution we cannot really expect to get the relative error of the true residual numerically much less than $O(10^{-6})$, and therefore it makes no sense to set the target residual r smaller than $O(10^{-7})$. Furthermore, we must be aware that if we do set the target residual to $O(10^{-7})$, and the accumulated residue converges to this level, we may still have that the true relative error is only $O(10^{-6})$.

2.5.4 Preconditioning

Before we can carry on to our numerical investigations, we must stop one final time to mention the issue of preconditioning. In strict linear algebra terms, preconditioning usually refers to a splitting of the matrix A in (2.80) so that after the splitting the remaining linear system(s) have a reduced condition number with respect to the initial system and therefore are easier to solve. Although the purpose of preconditioning is merely to make the computation more efficient, in the case of the clover improved fermion matrix it involves a non–trivial reformulation of the matrices and pseudofermion fields of the theory. Consequently the equations of motion are changed somewhat. The impact of these changes does not have much bearing on our numerical investigations and details of the exact formulation of the force term can be found in [13] and [35]. The following presentation is merely for the sake of completeness.

In lattice QCD using Wilson fermions, which involve only nearest neighbour interactions one can implement a scheme which is known as even odd preconditioning. Each lattice site (x, y, z, t) is assigned a parity P where

$$P = \text{sign}\left[(-1)^{x+y+z+t}\right] \; . \tag{2.91}$$

In a basis where sites are ordered according to their parities the fermion matrix can be rewritten according to the following block structure

$$M = \begin{pmatrix} A_{ee} & -\kappa D_{eo} \\ -\kappa D_{oe} & A_{oo} \end{pmatrix} \; . \tag{2.92}$$

where D is the Wilson Hopping term of (1.29), A is the clover term of (1.32) and the indices indicate the parities of the sites connected by the block (ee stands for even sites coupled to even sites, eo stands for even sites connected with odd sites and so forth). As demonstrated in [13] one can then introduce the preconditioned fermion matrix

$$\tilde{M} = \begin{pmatrix} A_{ee} & 0 \\ 0 & \hat{M}_{oo} \end{pmatrix} \tag{2.93}$$

with

$$\hat{M}_{oo} = A_{oo} - \kappa^2 D_{oe} A_{ee}^{-1} D_{eo} \tag{2.94}$$

and it can be shown that

$$\det(M^\dagger M) = \det(A^\dagger A)_{ee} \det(\hat{M}^\dagger \hat{M})_{oo} \; . \tag{2.95}$$

If we make once again the identification that $Q_{oo} = (\hat{M}^\dagger \hat{M})^{-1}_{oo}$, the lattice action can be rewritten as

$$S(U, \phi_o) = S_G - 2 \sum_e \ln \det A_{ee} + \phi_o^\dagger Q_{oo} \phi_o \; . \tag{2.96}$$

This reformulation guarantees that configurations are still distributed with the correct weight of $\det(M^\dagger M)$ however now pseudofermion fields are needed only on one parity, and the dimensionality of the matrix to be inverted in calculating H and F has been reduced by a factor of two. This is done at the expense of having to compute $\ln \det A_{ee}$ explicitly. To facilitate this there exists the "clover-trick" of [36] in which a similarity transform affects a further block diagonalisation on A_{ee}.

We reiterate that this preconditioning trick does not affect our numerical tests of reversibility or instability. However it can potentially change the definition of the target residual r which should now be considered to be "the desired relative error, defined through the accumulated residual of the CG algorithm when applied to solutions of linear systems involving the even–odd preconditioned fermion matrix."

2.6 Simulation Cost

So far in our discussion of the HMC algorithm we have often referred to sub–algorithms as efficient or inefficient. We now propose to discuss simulation cost. We can think of three interrelated factors that affect the cost of Hybrid Monte Carlo calculations:

1. **Cost of a Markov Step:** This is basically the computational cost of performing all the steps in the HMC algorithm once.
2. **The acceptance rate:** Clearly, if the acceptance rate is low the progress through the phase space will be slow.
3. **The autocorrelation times:** These affect cost in two ways, the exponential autocorrelation time governs the time it takes the algorithm to equilibrate while the integrated autocorrelation time for an observable enters into the definition of the error on the sample mean (see (1.92)). Heuristically, a long autocorrelation time implies that one has to do more work to reach a given level of statistical errors than one would have to if the autocorrelation time was shorter.

One can then define various metrics of cost. In particular the cost function

$$\varrho = \frac{\varrho_{\mathrm{traj}}}{\langle P_{\mathrm{acc}} \rangle} \tag{2.97}$$

where ϱ_{traj} is some metric of the computational work associated with performing a trajectory and $\langle P_{\text{acc}} \rangle$ is the average acceptance rate can characterise the cost of the work involved in generating an accepted trajectory.

The number of trajectories N_{traj} one needs to perform to reach a fixed statistical error ϵ on an observable of interest grows as (using equation (1.92))

$$N_{\text{traj}} = \frac{(2\mathcal{A} + 1)\sigma^2}{\epsilon^2} \tag{2.98}$$

where $\mathcal{A}$ is the autocorrelation time of the observable in units of completed trajectories. Hence, for a given observable of interest one can define a cost for the whole simulation as

$$\varrho_{\text{sim}} = \varrho_{\text{traj}} \frac{(2\mathcal{A} + 1)\sigma^2}{\epsilon^2} \tag{2.99}$$

The cost of a trajectory itself is a sum of the costs of all its components. However, the solves for X and Y are by far the most time consuming steps in a dynamical fermion computation with reasonably light quarks. Hence a good guide for a cost of a trajectory is the cost function

$$\varrho_{\text{traj}} = N_{\text{iters}} \tag{2.100}$$

The various cost functions discussed here are in principle complicated functions of the physical and algorithmic parameters of a simulation. Generally tuning algorithmic parameters has the aim of minimising one or more cost functions. In this case a cost measure is often normalised to either the most costly, least costly or some typical value for itself. In our tuning exercise of HMC, which we will outline in the next two chapters we considered the cost function

$$\text{Cost} = \frac{\langle N_{\text{iters}} \rangle}{\langle P_{\text{acc}} \rangle} \tag{2.101}$$

as an indication of performance. It expresses the fact that making the solver target residue r more accurate increases the number of solver iterations per trajectory, and that a low acceptance rate means that a simulation is proceeding slowly. In this case we assumed that the autocorrelation time does not change with algorithmic parameters which is not necessarily a true statement, however measuring the change in autocorrelation time would have been far too costly in our studies.

However, in our studies of Parallel Tempering (PT), which used HMC as a building block, we took as a metric of efficiency the factor by which PT improved autocorrelation times over HMC. In this case our efficiency measure was the ratio $\frac{\mathcal{A}^{\mathrm{PT}}}{\mathcal{A}^{\mathrm{H}}}$, where $\mathcal{A}^{\mathrm{PT}}$ and $\mathcal{A}^{\mathrm{H}}$ are the autocorrelation functions of the same observable measured on an ensemble of configurations produced by PT and HMC respectively. The cost of PT simulations is discussed extensively in chapter 6.

The general message of this section is as follows. There can be many metrics of efficiency of a simulation depending on which aspect we want to measure the efficiency of. We were content here merely to present the metrics we used in this thesis.

2.7 Summary of Chapter

We finish here our discussion of the application of HMC to lattice QCD. As this chapter has been a rather long one we would like to collect some of the main points we have made for future use.

- We have shown that for HMC to converge to the correct equilibrium, the molecular dynamics has to be reversible and area preserving. We have also shown, through the formalism of classical mechanics, how this can be done in a 1-Dimensional general Hamiltonian system. We ended this discussion by describing the construction of higher order integration schemes, and the Sexton–Weingarten trick. The generalisation of the algorithms, but not necessarily the classical mechanics, to lattice QCD is straightforward. Generalising the classical mechanics to group manifolds is not trivial and we regard it as beyond the scope of this thesis.

- We have described, if not in great detail, the necessary numerical computations for simulating lattice QCD. We outlined briefly the famous "Piece of Paper" method for exponentiation of the momenta in the gauge update step, and we have discussed the computation of the fermionic force through the computation of the vectors X and Y using Krylov Subspace solvers.

During this latter discussion we touched upon symmetric (hermitian) and non–symmetric methods, the methods generally in use in lattice QCD and we have discussed the particular stopping criterion used in our simulations.

- Finally we briefly discussed the issues of simulation cost, noting that cost metrics depended chiefly on what one wants to measure. In this process we introduced a cost function for our HMC tuning studies and a metric of efficiency for the PT algorithm.

With this background, we should now be amply prepared for the results of our reversibility and tuning studies which are detailed in the forthcoming two chapters and for understanding our investigations of the parallel tempering algorithm.

Chapter 3

Reversibility of HMC and Tuning the Solver

3.1 Motivation

In this chapter and the next, we describe two main themes both to do with the problem of finite precision in performing lattice QCD simulations using HMC. These two themes are connected by the main concern of this thesis, the desire to perform efficient simulations. The two themes themselves are reversibility and instabilities in the MD integration algorithm.

3.1.1 Reversibility and finite precision

Reversibility along with area preservation is required for detailed balance to hold, as was demonstrated in the previous chapter. Unfortunately, lattice QCD simulations involve arithmetic with real numbers, which do not have an exact binary representation, and thus in digital computers they are approximated by floating point numbers of fixed precision. This unfortunate fact forbids exact reversibility in our integration of the MD equations of motion chiefly because arithmetic operations are not associative in finite precision, In particular two levels of precision are in common use, called single precision, which usually entails using 32 bits for storing a floating point number, and double precision where 64 bits are used. Currently there exists an international standard [37], which defines arithmetic in finite precision, to which most modern computer manufacturers make claims to adhere.

In our quest for efficient simulations of lattice QCD, it is desirable to use single precision, or in fact as few bits of precision as possible, as the size of the data structures used to represent numbers in a computer has direct relevance to computational cost. In particular using single precision in our current simulations can potentially save a factor of 1.7 over the double precision representation merely through the difference of memory bandwidth required for each implementation [38].

The important question relating to reversibility then, is whether or not single precision is sufficient for the reversibility violations incurred during an actual HMC simulation to remain small enough for the algorithm to remain correct within the attainable statistical errors. Unfortunately we cannot give a mathematically rigorous answer to this question and have to resort to numerical experiments.

3.1.2 Chaos in the MD evolution and finite precision

The presence of chaos in the molecular dynamics evolution [39], [40] has bearing on this issue as it implies that aside from the usual volume scaling of rounding errors (and also reversibility violations) there is a potential for rounding errors to be magnified exponentially along an MD trajectory. This growth is characterised by the leading Liapunov exponent ν of the system, which will be defined in more detail later. The only way to guard against such an accumulation of errors is to increase the number of bits in the binary representation of our real numbers, which again raises issues of cost.

3.1.3 Solver Strategies and Reversibility

In the case of dynamical fermion simulations there is another aspect which has bearing on reversibility; which is the procedure for calculating the fermion force or to be more specific, the vectors X and Y of (2.73). If one chooses the initial guess for the Krylov solver in a time symmetric manner, reversibility of the solution procedure should be maintained irrespective of the desired residual r even in finite precision [41]. However, the errors in the force will be increased and this can have the effect that the molecular dynamics becomes even more chaotic. Hence we must distinguish between the Liapunov exponent of the numerical MD procedure and that of the underlying equations of motion.

3.1.4 Tuning the solver residual

From the previous statements, the immediate suggestion is to try and optimise the target solver residual for computing X during the molecular dynamics. In the case of our HMC code an initial tuning study has been carried out on small lattices, details of which can be found in [13]. The overall results of this study suggest that one can make some gain in the efficiency of the simulation by optimising r. However, to the best of our knowledge, there has not so far been a study of this kind on lattices of the size that one would use in a modern simulation[1].

The aims of this chapter are twofold:

- to outline our initial investigations into the problems of reversibility violations.
- to outline our preliminary study of tuning the target solver residual r.

Details of the first study are shown chiefly to illustrate that our code is doing everything as well as is possible in single precision. We also take this chance to discuss the original source of reversibility violations, and to define some metrics of reversibility.

[1]We currently use lattices of size $V = 16^3 \times 32$

This then sets the scene for the second topic, which is tuning the residual r. We found that the behaviour of ΔH along a trajectory changed qualitatively as r was made more and more inaccurate. This result links the topics discussed in this chapter with the other one discussed in [39], the subject of instability in the molecular dynamics. The results of our study of instabilities are presented in the next chapter.

3.2 Microscopic sources of reversibility

At a fundamental level, reversibility violations in finite precision stem from the gauge and momentum updates in the molecular dynamics not being reversible. Before examining global aspects such as the scaling of reversibility violations with system volume or MD trajectory length, it is prudent to say a few words about these "microscopic" sources of reversibility violations.

3.2.1 Gauge Update

We mentioned in section 2.4.3, that for the gauge update to be reversible, the unitarity and hermiticity conditions of (2.77) and (2.78) must hold.

To obtain an upper bound on the unitarity bound violations, we interrupted the gauge update step to compute

$$\delta\mathrm{Unit} = \max_{x,\mu,a,b} \left| \left(\exp(i\ \pi_\mu(x)\ \delta\tau)^\dagger\ \exp(i\ \pi_\mu(x)\ \delta\tau) - 1 \right)_{ab} \right| \tag{3.1}$$

whereas to check the hermiticity/reversibility condition we computed

$$\delta\mathrm{Herm} = \max_{x,\mu,a,b} \left| \left(\exp(i\ \pi_\mu(x)\ \delta\tau) - \exp(-i\ \pi_\mu(x)\ \delta\tau)^\dagger \right)_{ab} \right| \tag{3.2}$$

We carried out these tests on a small lattice with volume $V = 4^4$ sites, in the quenched approximation. The parameters used were $\beta = 5.4$, $\tau = 1$, and $\delta\tau = \frac{1}{10}$. We found that the maximum values of both δUnit and δHerm along a trajectory were

$$\max_{\mathrm{traj}} \delta\ \mathrm{Unit} = \max_{\mathrm{traj}}\ \delta\mathrm{Herm} = 0.59604645 \times 10^{-7} \approx \frac{1}{2}\epsilon_{SP} \tag{3.3}$$

where ϵ_{SP} is the 32-bit (single precision) machine unit of least precision,

$$\epsilon_{SP} = 1.1290929 \times 10^{-7}. \tag{3.4}$$

We stress these are maximal values. The hermiticity condition is often fulfilled exactly to within machine precision in the sense that $\delta\text{Herm} = 0$ whereas the value for δUnit is more often the same as its maximum value. These results satisfied us that our implementation of the matrix exponentiation routine was as accurate as possible.

3.2.2 Momentum Update

In the momentum update there are two possible sources of reversibility violation. Firstly violations can arise from roundoff errors in the update computation itself and secondly it is possible that the force is computed non-reversibly. However, the gauge fields are unchanged by a momentum update, and so when carrying out a momentum step forward followed immediately by carrying out a step backwards, the values of the force should be equal on both steps. Hence when one carries out a forward momentum step, followed by a backward one, using the same values of the force on both steps, one can ascertain the level of reversibility violation incurred by the momentum update from the non associativity in the addition $p + F\delta\tau$.

Consider a starting configuration (U, π, ϕ), where the π fields have just been refreshed from a heat-bath. We first compute the force F. After this we make a momentum update forward for time $\delta\tau$ resulting in (U, π', ϕ) and then one backward in time to obtain configuration (U, π'', ϕ), using the same force on both the forward and backward steps. We can then define the quantity

$$\delta\pi^i_\mu(x) = \pi^i_\mu(x)'' - \pi^i_\mu(x) \; , \tag{3.5}$$

on all links of the lattice and all the momentum components i, as a measure of the reversibility violation for such a test.

In our computations we carried out this test, and averaged the results over a hundred such forward–reverse steps with corresponding momentum refreshes. We thus obtained an estimate of $\left\langle \delta\pi^i_\mu(x) \right\rangle$, the *average* reversibility violation due to lack of associativity in the addition. Note that links and momentum components are not averaged over. We then created 8 datasets $i = 1 \ldots 8$ from the collection of measurements for $\left\langle \delta\pi^i_\mu(x) \right\rangle$, into each of which we binned the values of $\left\langle \delta\pi^i_\mu(x) \right\rangle$ from all the links of the lattice for a single momentum component i. We then histogrammed the values in each dataset, to obtain the distribution of the average reversibility violation for each component of the momenta.

The results are shown in figure 3.1. We show the histograms of all the components. We have obtained error estimates on the data points, but they are rather small, and clutter the picture, hence for the sake of clarity we do not show them. The lattice size used for this test was $V = 4^3 \times 8$, trajectories consisted of 10 steps of length $\delta\tau = 0.1$ giving a trajectory length of $\tau = 1$. The physical parameters used were $\beta = 5.2$, $c = 0$, and $\kappa = 0.1360$. For the data shown we used 500 bins in the histogramming process.

We can see from figure 3.1, that the distributions are very narrow and appear centred on zero. The width seems to be of order $O(10^{-1}\epsilon_{SP})$. It may seem strange that the distribution appears narrower than ϵ_{SP}. This is a result of carrying out the averaging and histogramming procedure in double precision.

With these investigations we were satisfied, that our underlying gauge and momentum update steps were as reversible as possible.

3.2.3 Solver Strategies

In the previous section we mentioned that it is possible to destroy reversibility by computing the force in a non-reversible way. Naturally one does not wish to do this, but there are certain computing optimisations that are tempting to make and yet spoil reversibility. In our previous tests, we kept the force constant over our forward and reverse updates of the momenta. We now feel it necessary to say a few words about solver strategies for the computation of X and how they can affect reversibility.

Let us consider the reversibility of an individual momentum update, assuming

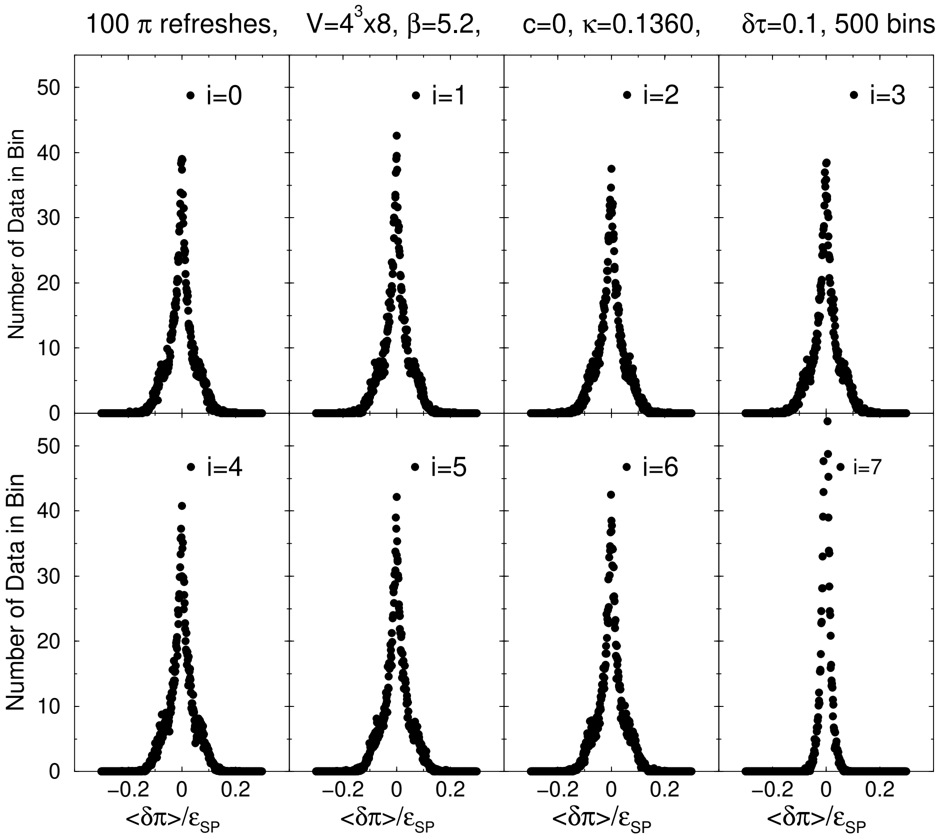

Figure 3.1: Distributions of microscopic reversibility violation obtained from histogramming the components of $\langle \delta\pi \rangle$. Each graph corresponds to a different component of the momentum. ϵ_{SP} is the single precision machine accuracy.

for now that we can do the vector addition in an associative manner. This is not a bad assumption seeing that we have shown that the average error is less than ϵ_{SP}.

We then carry out a momentum step forward followed by one backward as before, but now at the start of each step we recalculate X. We denote by X_f the value for X on the forward step and by X_b the value on the backwards one.

One can then have the following two viewpoints (both true)

1. The object X is just a function of U and ϕ. As U is not updated by the momentum step and ϕ is constant along an MD trajectory, we should have

that $X_f = X_b$. Consequently we can use X_f as the initial guess for the solution of X_b and then the latter solve will not cost any work.

2. The object X is obtained by a solution procedure which is a mapping $X = \mathcal{M}(U, \phi, X^0, r)$, where X^0 is some initial guess at the solution. Hence $X_f = \mathcal{M}(U, \phi, X^0_f, r)$ and $X_b = \mathcal{M}(U, \phi, X^b_0, r)$.

   For reversibility we need that $X_f = X_b$, in other words that $\mathcal{M}(U, \phi, X^0_f, r) = \mathcal{M}(U, \phi, X^0_b, r)$, which implies that we need $X^0_f = X^0_b$, so that the initial guess of the solver should be time symmetric. Clearly using $X^0_b = X_f$ violates this time symmetry, unless $X^0_f = X_f$ also.

These two viewpoints give rise to two kinds of solver strategies, one in which we use some time symmetric (usually time independent) initial guess in our solver or one in which we use a time asymmetric one.

The viewpoint for the case of asymmetric starts is only true when both X_b and X_f are exact solutions. For this viewpoint to be valid using an iterative solver, X_b and X_f would need to be known very accurately, making the solution process expensive. Varied strategies have been used in the past such as using the previous solution or making extrapolations from the last few solutions to obtain a good initial guess, so that time spent in the solution process could be reduced. However as is made clear in [39], unless the residue is very small reversibility metrics show signs of deterioration.

When using a time symmetric start, however, the solver target residue r becomes a tunable parameter, the variation of which can increase or decrease the time spent in the solution for X and Y. However making the solution less and less accurate brings with it corresponding increases in ΔH and decreases in the acceptance rate. If the resulting F is sufficiently inaccurate the solver can become unstable, resulting in large ΔH, and the cost of simulation will become prohibitively expensive.

3.3 Global Reversibility Violations

Having satisfied ourselves that on a microscopic level the reversibility of our algorithm is as good as we can make it, we now discuss some global aspects of reversibility. In particular we are interested in how the accumulated rounding errors scale with volume and trajectory length. With this purpose, motivated by [39] we define some global metrics of reversibility.

Consider performing an HMC trajectory of length τ, with stepsize $\delta\tau$ and target residual r (for the calculation of X) starting from initial configuration (U, π) in the presence of a fixed ϕ field. The result of the trajectory is (U', π') and the corresponding energy change is ΔH.

Now consider performing an MD trajectory backwards in time with same length τ, stepsize $\delta\tau$ and target residual r. We will denote the result of this trajectory (U'', π''). The ϕ fields remain unchanged for both trajectories in such a test.

We can then define a measure of difference between U'' and U as

$$||\delta\Delta U|| = \sqrt{\sum_{x,\mu,a,b}\left[U''^{ab}_{\mu}(x) - U^{ab}_{\mu}(x)\right]^2} \tag{3.6}$$

and between the π'' and π as

$$||\delta\Delta\pi|| = \sqrt{\sum_{x,\mu,i}\left[\pi''^{i}_{\mu}(x) - \pi^{i}_{\mu}(x)\right]^2} \tag{3.7}$$

and a measure of the reversibility violation in terms of the energy

$$\delta\Delta H = H(U'', \pi'', \phi) - H(U, \pi, \phi) \tag{3.8}$$

We can also define a measure of the relative error in our computation of ΔH as $\frac{|\delta\Delta H|}{|\Delta H|}$. This latter metric provides a rough idea of our error in computing the acceptance probability P_{acc}.

3.3.1 Volume Scaling of metrics

Examining the definitions of our metrics, it should be clear that $||\delta\Delta U||$ and $||\delta\Delta\pi||$ should grow like $O(\sqrt{V})$. The reason for this is that the squares of the norms are sums of positive definite quantities of equal size on average, over the whole lattice so $||\delta\Delta U||^2$ and $||\delta\Delta\pi||^2$ should grow as the volume V and the norms themselves are the square roots of these quantities.

The metric $\delta\Delta H$ however is expected to grow more slowly, as the terms involved in its evaluation over the lattice degrees of freedom are not positive definite and hence one would expect local cancellations in (3.8).

To test that errors over the individual degrees of freedom do not grow too dangerously with the volume we were motivated to define variants of $||\delta\Delta U||$, $||\delta\Delta\pi||$ and $||\delta\Delta H||$ averaged over the relevant degrees of freedom:

$$\frac{||\delta\Delta U||}{\text{d.o.f}} = \frac{1}{\sqrt{N^{U}_{\text{d.o.f}}}}||\delta\Delta U|| \tag{3.9}$$

$$\frac{||\delta\Delta\pi||}{\text{d.o.f}} = \frac{1}{\sqrt{N^{\pi}_{\text{d.o.f}}}}||\delta\Delta\pi|| \tag{3.10}$$

$$\frac{|\delta\Delta H|}{\text{d.o.f}} = \frac{1}{N^{H}_{\text{d.o.f}}}|\delta\Delta H| \tag{3.11}$$

where $N^{U}_{\text{d.o.f}} = N^{\pi}_{\text{d.o.f}} = 8\times 4\times V$ are the number of gauge and momentum degrees of freedom and $N^{H}_{\text{d.o.f}} = N^{U}_{\text{d.o.f}} + N^{\pi}_{\text{d.o.f}}$ are the number of degrees of freedom in the quenched approximation. For dynamical fermion simulations one must also consider the number of fermionic degrees of freedom. Naively the number of fermionic degrees of freedom are $N^{f}_{\text{d.o.f}} = 4\times 3\times V$ complex numbers, i.e. a pseudofermion field variable at every lattice site. The presence of preconditioning, implies that only half of these degrees are accounted for by pseudofermion fields and the others are swept into the definition of the clover determinant (on the opposite parity sites). However our volume scaling study was carried out in the quenched approximation.

We have carried out reversed HMC trajectories, (as described in the definition of the metrics earlier) in the quenched configuration with $\tau = 1$, $\delta\tau = \frac{1}{180}$ and

$\beta = 5.4$, on several lattice sizes. The volumes of lattice used were

$$V \in \{4^4, 8^4, 10^4, 10^3 \times 16, 16^3 \times 32\} \ . \tag{3.12}$$

We plot the results for our metrics in figure 3.2

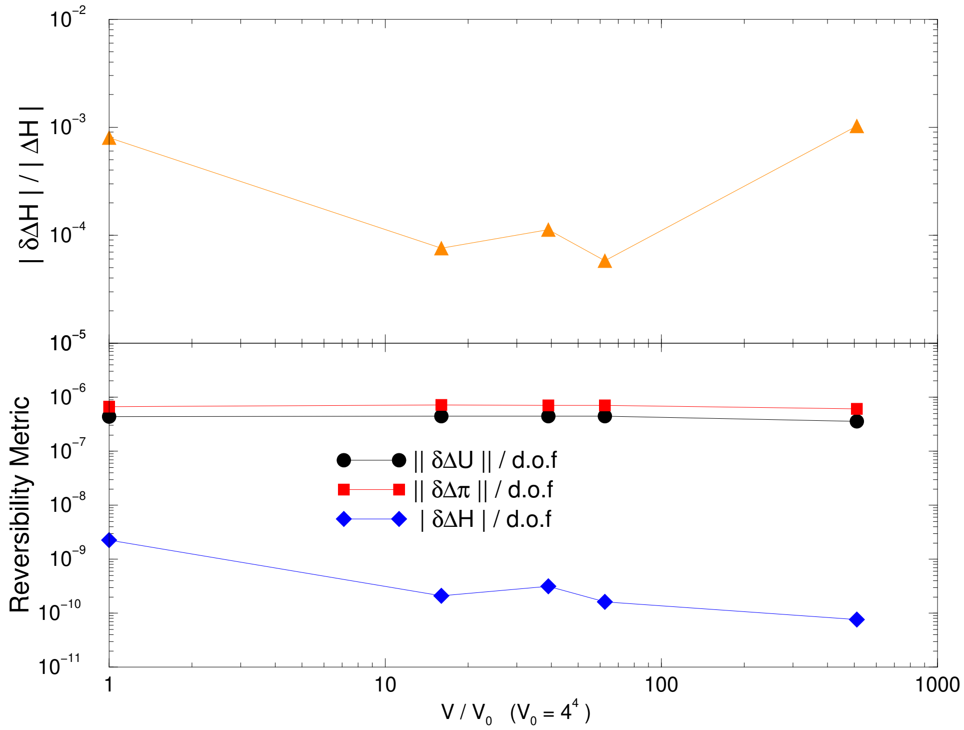

Figure 3.2: Volume scaling of reversibility metrics, $\frac{||\delta\Delta U||}{\mathrm{d.o.f}}$, $\frac{||\delta\Delta\pi||}{\mathrm{d.o.f}}$, $\frac{|\delta\Delta H|}{\mathrm{d.o.f}}$ and $\frac{|\delta\Delta H|}{|\Delta H|}$. The volumes are normalised by the smallest one for which $V = 4^4$.

Figure 3.2 shows that the metrics $\frac{||\delta\Delta U||}{\mathrm{d.o.f}}$ and $\frac{||\delta\Delta\pi||}{\mathrm{d.o.f}}$ scale with the volume. The metric $\frac{|\delta\Delta H|}{\mathrm{d.o.f}}$ is always smaller than $\frac{||\delta\Delta U||}{\mathrm{d.o.f}}$ and $\frac{||\delta\Delta\pi||}{\mathrm{d.o.f}}$. These two results are exactly as we would expect from the foregoing scaling arguments.

Finally we show in figure 3.2 the metric $\frac{|\delta\Delta H|}{|\delta H|}$. This metric seems always to be less than 10^{-3}. This is reassuring as it seems to indicate that the relative error in our computations of ΔH is less than one percent.

3.3.2 Scaling of metrics with time

We mentioned earlier that the authors of [13, 39, 40] found that the MD evolution in lattice QCD is chaotic, and the accumulation of rounding errors along a molecular dynamics trajectory could be characterised by the leading Liapunov exponent ν. We would now like to give an operational definition of this exponent.

Consider two HMC configurations $(U(\tau_0), \pi(\tau_0))$ and $(U'(\tau_0), \pi'(\tau_0))$. We can define distances $d_U(\tau_0) = ||U(\tau_0) - U'(\tau_0)||$ and $d_P(\tau_0) = ||\pi(\tau_0) - \pi'(\tau_0)||$ between them using the norms of (3.6) and (3.7). Now consider performing MD for some time τ using both $(U(\tau_0), \pi(\tau_0))$ and $(U'(\tau_0), \pi'(\tau_0))$ as initial conditions. One can then measure the distance $d_U(\tau_0+\tau)$ and $d_\pi(\tau_0+\tau)$ on the resulting configurations.

If the MD is chaotic it is possible to characterise, to leading order in τ, d_U and d_π as

$$d_U(\tau_0 + \tau) \propto e^{\nu_U \tau} \tag{3.13}$$

$$d_\pi(\tau_0 + \tau) \propto e^{\nu_\pi \tau} \ . \tag{3.14}$$

The coefficients ν_U and ν_π are called the leading Liapunov exponents for the system. In principle, one can associate a Liapunov exponent with every degree of freedom for the system, but the leading ones are the ones that dominate the magnification of the errors.

During a reversed MD trajectory, the accumulation of rounding errors leads to the MD deviating from the classical path along the forward trajectory. Hence on the reverse trajectory, there is a difference between the initial configuration actually used by the MD and what the correct classical starting position should be. Thus we can also define a Liapunov exponent for $||\delta\Delta U||$ and $||\delta\Delta\pi||$. Following the notation convention of [39] we define $\nu = \nu_{||\delta\Delta U||}$ as *the* Liapunov exponent and will drop the subscript on ν from now on.

Usually in such a study one is mostly interested in the sign of a given Liapunov exponent. A positive one indicates that the system is chaotic and one has exponential amplification of rounding errors whereas a negative one indicates that trajectories converge in phase space.

3.3.3 Liapunov Exponents and Tuning

The authors of [13, 39, 40] all found that the Liapunov exponents they measured were positive. In particular [39] proceeded to show that the chaos was a feature of the underlying system of equations by demonstrating that when algorithmic parameters were made accurate enough, both r and $\delta\tau$, the graphs of the Liapunov exponents plotted against the control parameters showed a plateaux, indicating that the Liapunov exponents were independent of the algorithmic parameters (their figures 14 and 18).

In terms of the behaviour of ν with step size $\delta\tau$, they found that this plateau ended and the exponents started growing at values of $\delta\tau > 0.6$ in the quenched approximation and in the case of heavy quarks for dynamical fermion simulations. However when using light fermions they found that the growth of the Liapunov exponent was much more rapid and the plateau ended at $\delta\tau = 0.08$. The actual values are not important, but the qualitative change is. However this discussion is reserved for the next chapter.

The authors of [39] also made a cursory study of the change in Liapunov exponents with increasing r. This aspect of their study is not as detailed as the one of the stepsize dependence, but it suggests that as they made their solver residue less and less accurate there was an increase in the Liapunov exponent (although the errors quoted are large) accompanied by a rapid drop in acceptance rate.

This latter result was an additional motivation for our study of tuning r. We now present the results.

3.4 Results from tuning r

3.4.1 Simulation Details

In this study, we used 10 configurations taken from an equilibrated ensemble with physical parameters $\beta = 5.2$, $c = 2.0171$ and $\kappa = 0.1355$. The lattice volume was

$V = 16^3 \times 32$. These parameters correspond to $\frac{m_\pi}{m_\rho} \approx 0.6$ [16]. In terms of current simulations this implies that the dynamical fermions are quite light.

Using our 10 starting configurations we carried out reversed MD trajectories of varying length τ with a constant stepsize of $\delta\tau = \frac{1}{180}$. We repeated this procedure for values of r ranging from $r = 10^{-7}$ to $r = 10^{-4}$ on each of our sample configurations. All these computations were carried out in single precision. We reiterate that our definition of r is the desired relative error on the restart CG solve, using the the accumulated residue.

On each MD trajectory we measured the metric $||\delta\Delta U||$, ΔH and N_{iters}, where N_{iters} is the total number of solver iterations (in both the BiCGStab solves and the restart CG solve) averaged over the forward and backward trajectories.

3.4.2 Results

We fitted $\ln ||\delta\Delta U||$ as a linear function of $\delta\tau$ to extract our Liapunov exponents. The quality of the data is shown in figure 3.3. The fits were carried out using the standard technique of maximum likelihood estimation. Error estimates were made by bootstrapping the 10 configurations. Both these methods are outlined in Appendix B.

Our measurements of $\langle \Delta H \rangle$ as a function of τ are shown in figure 3.4. We distinguish three types of behaviour for $\langle \Delta H \rangle$ – oscillatory (bottom graph), transitional (middle graph) and divergent (top graph).

Finally our acceptance rates, Liapunov exponents and cost function values (c.f (2.101)) are shown in figure 3.5. Due to the sharp drop in acceptance rate as the ΔH changes behaviour, we show an enlarged version of the cost function graph as well to illustrate the minimum. In this instance we normalised the cost function with its value at $r = 10^{-7}$, the case with the most solver iterations per trajectory.

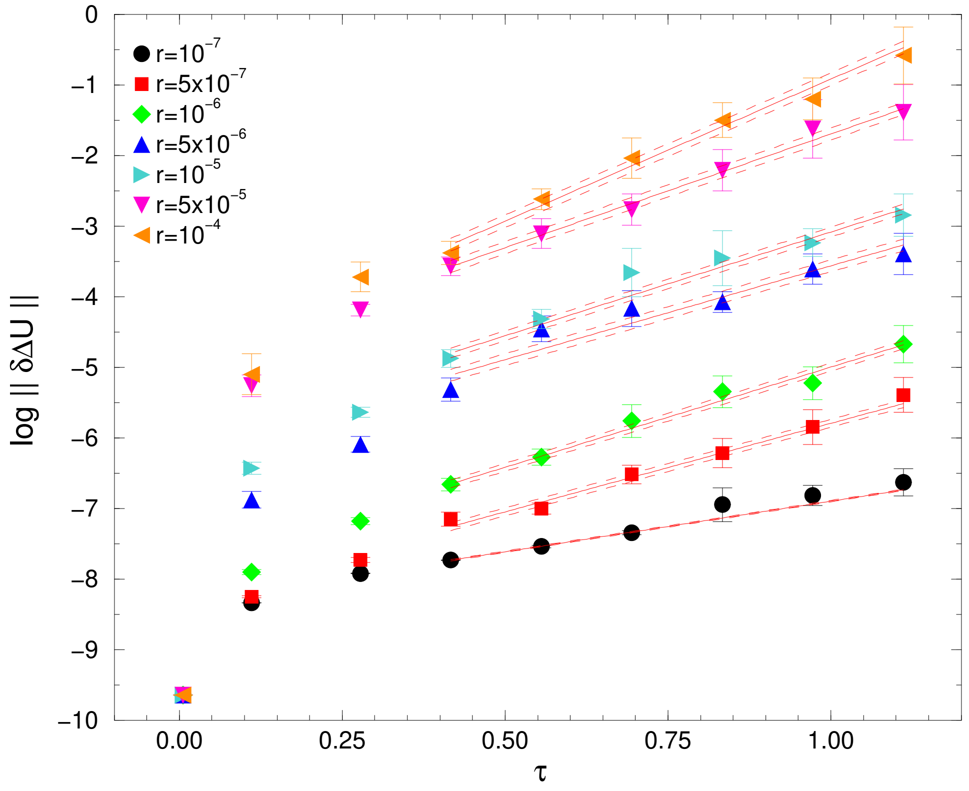

Figure 3.3: Fits for the exponent ν

3.4.3 Discussion

Figure 3.3 shows that our system is clearly chaotic, as the plots of $\ln ||\delta\Delta U||$ all have a positive slope. In this respect we have merely confirmed the results of other authors on a larger volume. We note that apart from the cases when $r = 5 \times 10^{-6}$ and $r = 10^{-5}$ the overall tendency seems to be for the Liapunov exponents to increase with r. We also note that apart from these two parameter values the signal for the fit is quite good.

In the situation when $r = 5 \times 10^{-6}$ and $r = 10^{-5}$ the graphs of figure 3.3 seem to show a marked break at around $\tau \approx 0.5$. In fact it was not possible to fit consistent values of ν for these two parameter values. Good fits could be obtained either side of the "break".

This situation is made more clear in figure 3.5. The bottom graph shows fits

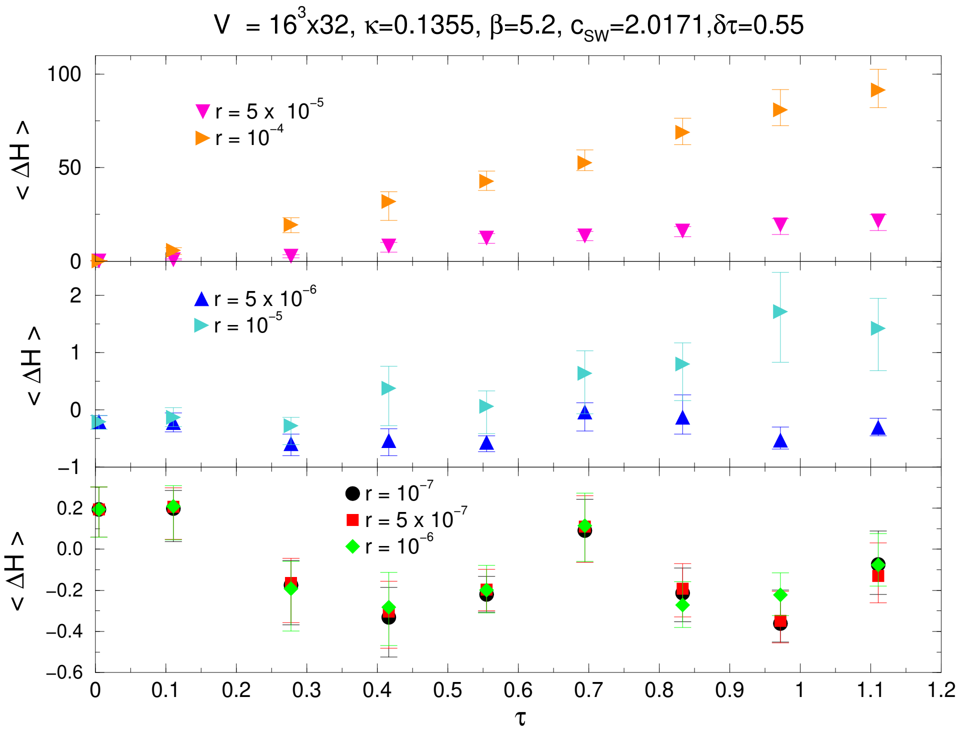

Figure 3.4: ΔH as a function of r

for the Liapunov exponent over varying fit ranges[2]. For the two parameter values highlighted it can be seen that a consistent fit cannot be made. For values of r less than $r = 5 \times 10^{-6}$, consistent values can be obtained for the Liapunov exponents, which seem to show a slowly increasing tendency with increasing r. For values of $r > 10^{-5}$ the values from the different fits tend to converge and are consistent with forming a plateau.

The authors of [39] showed that for sufficiently small r the ν should plateau. We do not see this on our plots and we presume that we need r to be smaller than we can reach using single precision solves. The region where consistent determination of ν becomes difficult corresponds to a rapid drop in acceptance rate similar to the one indicated in [39]. Correspondingly the cost function rises rapidly and the simulation becomes prohibitively costly.

[2]Consider arranging the data points of figure 3.3 in increasing order of τ, and labelling each data point by its rank in this sequence. The numbers quoted for the fit range are then the labels of the first and last elements of the subsequence over which the fit was carried out.

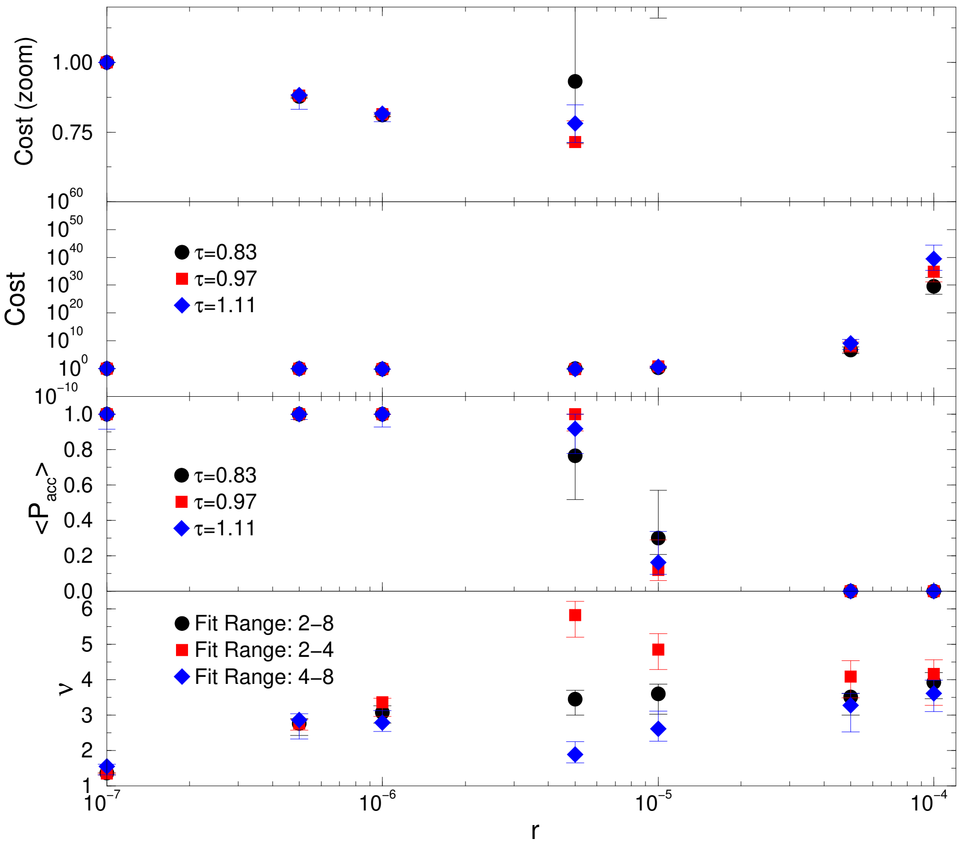

Figure 3.5: Liapunov exponents, Acceptance Rate, and Cost as a function of r

We note that figure 3.5 shows that there is a value of r for which the cost is minimised. However the saving seems no greater than 25% the cost of running the solver to exhaustion ($r = 10^{-7}$) in single precision. The saving may be greater if solves are done in double precision. Furthermore the minimum of the cost function seems to be closest to where the acceptance rate starts to drop, which we suspect is due to the MD becoming unstable.

3.5 Conclusions for this Chapter

From the data presented here we conclude that our implementation of the HMC algorithm is as safe from the point of view of reversibility violations as possible. The microscopic violations from the gauge and momentum field updates of our program have been shown to be of the order of ϵ_{SP}, and the scaling of metrics with volume behaves as we expect.

When tuning the solver target residue we found that our Liapunov exponents are sensitive to r, and hence they are not the values from the underlying classical equations of motion. We also found that they have a tendency to grow with increasing r until at a value somewhere between $r = 5 \times 10^{-6}$ and $r = 10^{-5}$ our acceptance rate drops rapidly and the Liapunov exponents become difficult to determine consistently. However as r is further increased, consistent estimates seem possible again.

Finally we note that the behaviour of the energy is consistent with the molecular dynamics integrator becoming unstable, a situation we shall investigate in greater detail in the next chapter.

Chapter 4

Instability in the MD evolution

In the previous chapter we described the results of our initial exercise of tuning the solver target residual r. We showed that for a value of r greater than $r = 10^{-5}$ the behaviour of ΔH changed from an oscillatory one, to one which seemed to grow rapidly with MD time τ. We noted that this seemed to signal to us the presence of an instability in our MD evolution scheme.

In this chapter we make a fuller study of instabilities, motivated by these results. We reiterate that the instabilities are an issue of cost, for if the average ΔH is large, the acceptance rate will be very small and hence our simulation cost will be large.

We believe the instability in the MD occurs when the combination $F\delta\tau$ in the momentum update reaches a critical value, a result which we will show in free field theory. We will also show that the effect of this instability on the behaviour of $\langle \Delta H \rangle$ is very similar to the one demonstrated in figure 3.4.

To connect the free field result to the case of an interactive field theory we will use the hypothesis of [39] that the fermionic term in the MD force becomes large in norm as the fermion mass tends to zero. Hence the instability can arise either as a result of not computing the fermion force accurately enough, as shown by our tuning exercise of r in the last chapter, or as a result of the stepsize becoming too large or possibly as a combination of these two factors.

We make the point, that reversibility is not necessarily an important factor in this study nor is finite precision. The instability in free field theory can be shown analytically, when the leapfrog procedure is supposed to be exactly reversible. However there is an impact for single precision simulations. If the force computed is not precise enough due to single precision limitations, it may not be possible to control ΔH through decreasing the stepsize. Note that in this case the solution is merely to perform the force computation in double precision and the fields themselves may not need to be stored this precisely. Hence the instabilities merely set upper limits on the parameters of the MD in single precision.

There are several consequences of the instability being due merely to light fermions. The first one is that the instability is likely to be encountered in double precision also. Another question that may arise is whether or not the instabilities can be avoided through the use of higher order integration schemes. Finally one can attempt to make educated guesses at how the upper limits on the parameters will change as one tries to simulate with even lighter fermion masses.

We note that in the quenched/heavy fermion case, the instabilities were not as much of a problem as they are in the case of light quarks. As already alluded to in the last chapter it was shown in [39] that Liapunov exponents started growing at much higher values of the stepsize for quenched simulations and dynamical simulations with heavy fermions than they did for the light fermion case. It was also shown in that contribution, that for the case of quenched/heavy fermions, by the time the reversibility metrics started to grow large due to instability, the value of ΔH was already prohibitively large. However in the case of light fermions, the instability seems to occur at times when ΔH would otherwise be small.

We also make the point that if rounding errors were not present, the instability would not be a problem at all, as then ΔH could always be controlled by changing the step-size. As we noted before, a time symmetric solution process is reversible and the only effect of increasing the residual[1] seems to be an increase in the Liapunov exponent, which implies an increase in the magnification of roundoff errors. Hence if no rounding errors are present, problems due to tuning r could also be offset by reducing the stepsize.

[1] Making the solve less accurate

Motivated by our results from the previous chapter, and by the desire to tune our simulation on a large volume, we carried out a tuning process, where we identified the onset of instability for MD trajectories of length $\tau \approx 1$ in an attempt to find upper boundaries on r and $\delta\tau$ at physical parameters, $\beta = 5.2$, $c = 2.0171$ and $\kappa = 0.1355$. These parameters correspond to those of a large scale simulation carried out by the UKQCD collaboration, with quite light quarks.

In this chapter we will first describe the source of instability in free field theory, present some supporting evidence for the hypothesis linking the instability to the fermion mass and present the results of the tuning study.

4.1 Instability in Free field Theory

The onset of instability of free field theory was shown for the case of a lattice of field variables in [39]. In this section we summarise the case for a single oscillator mode, ϕ, with conjugate momentum π, and angular frequency ω.

4.1.1 The Leapfrog Algorithms for a Harmonic Oscillator

The Hamiltonian for such a system is

$$H = \frac{1}{2}\left(\pi^2 + \omega^2\phi^2\right). \tag{4.1}$$

and the force is

$$F = -\omega^2\phi\ . \tag{4.2}$$

If we write ϕ and π as the phase space vector (ϕ, π), the update operator for a single timestep $\delta\tau$ of the leapfrog algorithm can be written as the matrix

$$\hat{U}(\delta\tau) = \begin{pmatrix} 1 - \frac{1}{2}\omega^2\delta\tau^2 & \delta\tau \\ -\frac{1}{2}\omega^2\delta\tau + \frac{1}{4}\omega^4\delta\tau^3 & 1 - \frac{1}{2}\omega^2\delta\tau^2 \end{pmatrix}\ . \tag{4.3}$$

This can be parameterised [39] as

$$\hat{U}(\delta\tau) = \begin{pmatrix} \cos[\kappa(\omega\delta\tau)\delta\tau] & \dfrac{\sin[\kappa(\omega\delta\tau)\delta\tau]}{\rho(\omega\delta\tau)} \\ -\rho(\omega\delta\tau)\sin[\kappa(\omega\delta\tau)\delta\tau] & \cos[\kappa(\omega\delta\tau)\delta\tau] \end{pmatrix}\ , \tag{4.4}$$

with

$$\kappa(\omega\delta\tau) = \frac{\cos^{-1}(1 - \frac{1}{2}\omega^2\delta\tau^2)}{\delta\tau} \tag{4.5}$$

and

$$\rho(\omega\delta\tau) = \sqrt{1 - \frac{\omega^2\delta\tau^2}{4}} \ . \tag{4.6}$$

Using this parameterisation one can write down the update matrix due to a whole trajectory of length τ as

$$\hat{U}(\tau) = \begin{pmatrix} \cos[\kappa(\omega\delta\tau)\tau] & \frac{\sin[\kappa(\omega\delta\tau)\tau]}{\rho(\omega\delta\tau)} \\ -\rho(\omega\delta\tau)\sin[\kappa(\omega\delta\tau)\tau] & \cos[\kappa(\omega\delta\tau)\tau] \end{pmatrix} \ . \tag{4.7}$$

When $\omega\delta\tau = 2$ two things happen simultaneously. Firstly, $\rho = 0$ causing one of the elements of $\hat{U}(\tau)$ to become undefined, secondly a branch point is encountered in κ as the $\cos^{-1}$ term becomes imaginary.

An alternative way of considering the instability is to apply $\hat{U}$ to the initial condition when the harmonic oscillator is at amplitude, in other words when $(\phi, \pi) = (A, 0)$. The resulting phase space trajectories obey the equation

$$\frac{\phi^2(\tau)}{A^2} + \frac{\pi^2(\tau)}{A^2\rho^2(\omega\delta\tau)} = 1 \ . \tag{4.8}$$

Hence the trajectories are elliptical when $\omega\delta\tau < 2$, and hyperbolic when $\omega\delta\tau > 2$. We note that the elliptic or hyperbolic nature of these orbits is independent of the initial phase space co-ordinates. It is just for simplicity of exposition that we have chosen the particular initial vector used above.

The energy change along a trajectory is

$$\Delta H = H(\phi(\tau), \pi(\tau)) - H(\phi(0), \pi(0)) \ . \tag{4.9}$$

Using the same initial conditions as before

$$\Delta H = \frac{1}{2}\left(\rho^2(\omega\delta\tau) - 1\right)\cos^2(\kappa(\omega\delta\tau)) \tag{4.10}$$

which is an oscillating function of τ for $\omega\delta\tau < 2$, however when $\omega\delta\tau > 2$, $\kappa(\omega\delta\tau)$ becomes imaginary changing the $\cos^2$ term to one involving a hyperbolic cosine and ΔH grows with τ. We note that this is similar behaviour to the ΔH in figure 3.4 of the last chapter, when r is increased from 10^{-6} to about 10^{-4}.

4.2 A Hypothesis for QCD

A link can be drawn between free field theory and a weakly interacting field theory. In [39] the claim was made that a weakly interacting theory such as an asymptotically free field theory at small distance, can be considered as a collection of harmonic oscillator modes. It was further argued therein, that the onset of instability will still be caused when the highest frequency mode becomes unstable with $\omega_{\max}\delta\tau = 2$. However, due to interactions, the frequency of this mode will vary in a complicated fashion and thus the transition from stability to instability will not be so abrupt as in the case of the single free oscillator mode described above.

This mechanism is illustrated with numerical results in [39] from simulations of a harmonic oscillator whose frequency is chosen randomly from a Gaussian distribution with mean unity and variance σ^2 before each MD step. The random choice for the frequency is supposed to simulate the effects of the complex interactions. As the σ^2 in the model is increased, corresponding to stronger and stronger interactions, the Liapunov exponents of the model show smoother and smoother growth around the value of $\delta\tau = 2$, whereas the transition for the free case, when $\sigma^2 = 0$ is discontinuous as expected.

The next line in the argument of [39] is that the $\omega\delta\tau$ term of the free field theory corresponds to the $F\delta\tau$ term in a lattice QCD leapfrog momentum update. The final link in this chain of reasoning is that the fermionic contribution to the force is expected to grow in some inverse proportion with the fermion mass m_f. Hence as the fermion mass becomes lighter and lighter in a simulation, the force is expected to grow causing the onset of the instability to appear at smaller and smaller values of $\delta\tau$.

4.3 Supporting the Hypothesis

We decided to check if this hypothesis could be supported in lattice QCD. In our approach, we measured the magnitude of the average fermionic and gauge forces, in both the 2-norm and the ∞-norm. As the force is defined in the Lie algebra $su(3)$ like the momenta, we could use the same definition of the 2-norm, and its degree of freedom averaged variant as we used in (3.10) for the definition of $||\delta\Delta\pi||$. The corresponding ∞–norm is defined as

$$||F||_\infty = \max_{x,\mu,i} |F^i_\mu(x)| \ . \tag{4.11}$$

The ∞–norm then is the force component with the maximum magnitude over the lattice, and this can be likened, although tenuously, to ω_{max}. The ratio of infinity norm of the force to its degree of freedom averaged 2–norm can give a rough indication of the relative number of high frequency modes compared to the number of "average" frequency ones.

4.3.1 Computational Procedure

Our procedure for estimating the force was to perform an HMC trajectory starting from one of the 10 configurations we used before, and to measure the force norms for each leapfrog update along a trajectory. We repeated these tests for several values of κ to simulate fermions of various mass. One may argue, that a configuration which has a reasonable statistical weight in a configuration ensemble equilibrated at $\kappa = 0.1355$, may have a vastly different statistical weight in ensembles equilibrated at different values of κ, implying that the results from these tests are not physical. However, we make the argument that here we are not looking at physical ensemble properties but simply tuning a parameter in our "gauge generation machine", to see how it reacts.

The lattice volume used in these tests was made up of $V = 16^3 \times 32$ sites and the molecular dynamics consisted of $N_{MD} = 175$ steps of step-size $\delta\tau = \frac{1}{180}$, corresponding to trajectories of length $\tau \approx 0.97$. During these tests the target residue was set to be $r = 10^{-6}$ which was still within the stable region of the MD. Error bars for the forces shown were computed by bootstrapping the 175 values measured along the trajectory.

The value of κ_c for the full ensemble of configurations from which our small sample of starting configurations was taken has been determined in hadron spectroscopy studies [16] to be

$$\kappa_c = 0.1363 \ . \tag{4.12}$$

Thus we were able to associate a value of am_f for every κ we used through the relationship

$$am_f = \frac{1}{2}\left(\frac{1}{\kappa} - \frac{1}{\kappa_c}\right) \tag{4.13}$$

Anticipating an inverse relation between the fermion mass and the norm of the force, we fitted the values of the fermionic force with the formula

$$F = A(am_f)^{\alpha} = A\left(\frac{1}{2\kappa} - \frac{1}{2\kappa_c}\right)^{\alpha} \tag{4.14}$$

where A, κ_c and α were the parameters to be determined by the fitting process.

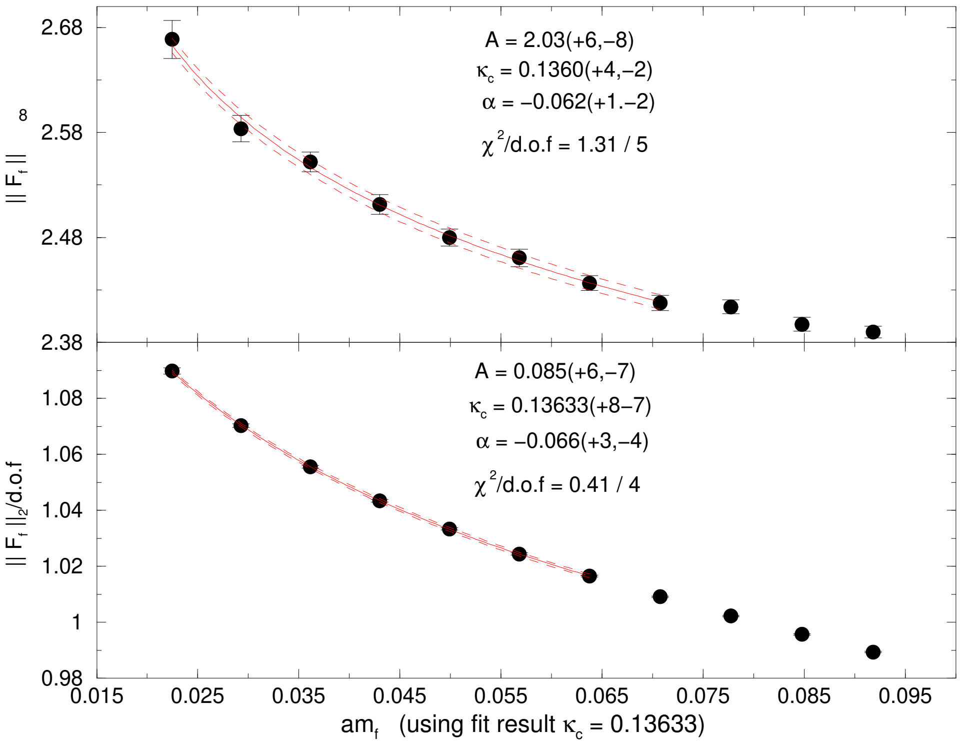

Figure 4.1: The 2-norm/d.o.f and the ∞-norm of the fermionic contribution to the MD force. F_f against am_f, fitted with ansatz (4.14).

The results of our test are shown in figure 4.1. We see that good fits to our data can be made, which reproduce κ_c from spectroscopy and that α is positive indicating an inverse relationship between the force and the fermion mass, supporting the hypothesis of [39].

4.3.2 Further investigation of the fermion force

To further investigate the onset of the instability, we computed the average gauge and fermion forces as well as ΔH along a single MD trajectory, using the same physical parameters and starting configuration as before. However, this time we varied the stepsize $\delta\tau$. The number of steps taken was adjusted to keep $\tau \approx 0.97$.

The results are plotted in figure 4.2. From the growth of ΔH in the top graphs of figure 4.2 it can be seen that the instability sets in between $\delta\tau = 0.0105$ and $\delta\tau = 0.011$. We can see that the instability seems to be accompanied by a growth in the fermion forces, and that the ∞–norm grows at a faster rate than the 2–norm.

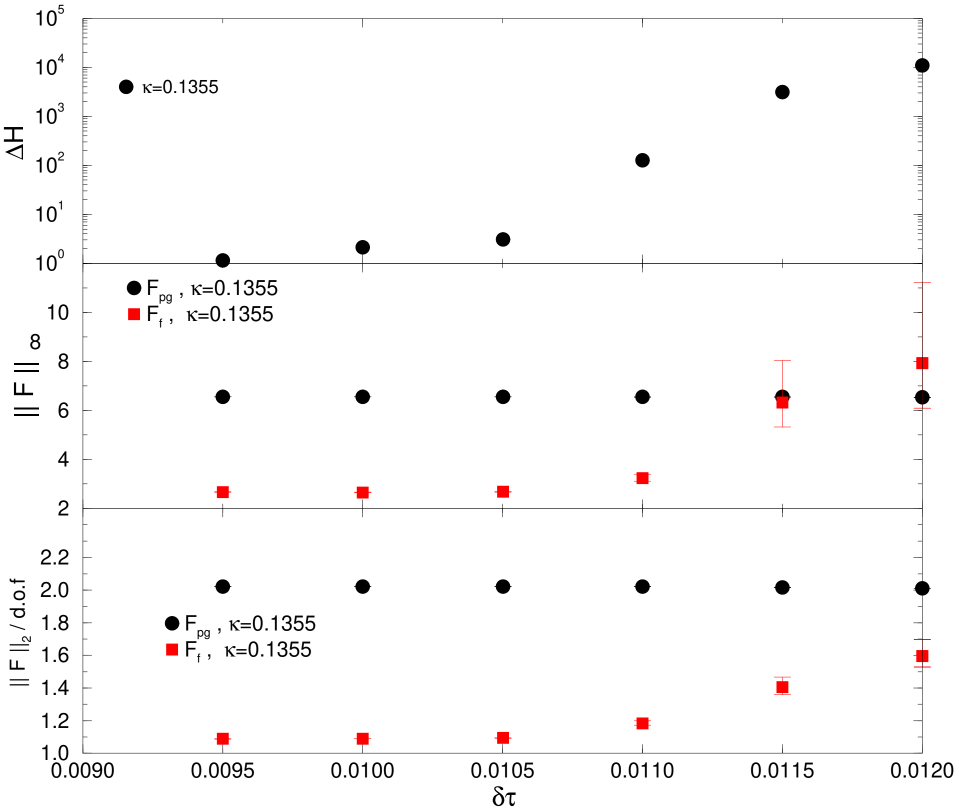

Figure 4.2: The gauge and fermionic components of the MD force, F_{pg} and F_f respectively, and ΔH plotted as a function of the stepsize $\delta\tau$.

In figure 4.2, the fermion force and the gauge force seem to be constant until the onset of the instability, after which the gauge force appears to remain steady and only the fermion force grows, which suggests to us that the instability is to do with the momentum update step. Again we note that the ∞–norm is several times the 2-norm averaged over the degrees of freedom, suggesting the presence of just a few unstable light modes.

Finally we investigated the behaviour of the force while varying κ, for two values of $\delta\tau$, one where the MD was stable ($\delta\tau = 0.01$) and one where we encountered an instability at $\kappa = 0.1355$,($\delta\tau = 0.012$). In figure 4.3 we see the results. We notice that the computation with $\delta\tau = 0.012$ appears to go unstable for $\kappa = 0.1355$, while the one with $\delta\tau = 0.01$ remains stable. When the instability sets in the fermion force ∞–norm increases accompanied by a sudden drop in the ∞–norm of the gauge force. We interpret this behaviour as the algorithm attempting to maintain energy conservation (but failing).

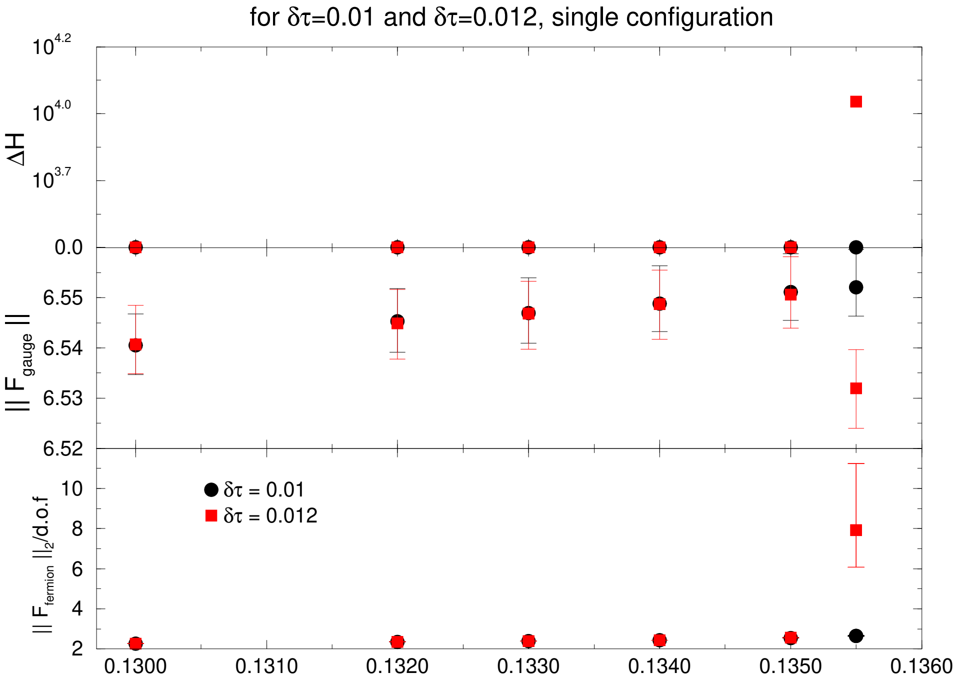

Figure 4.3: The ∞–norm of the gauge and fermion forces against κ, for values of $\delta\tau = 0.01$ and $\delta\tau = 0.012$.

These two results appear to tell us the following:

- There appears to be a critical value of the stepsize $\delta\tau_c$ for a given κ, so that for $\delta\tau > \delta\tau_c$ the MD becomes unstable.

- The onset of the instability is κ dependent, even though the solver residue r has been kept constant. For a given value of κ it may be possible to keep the instability under control by reducing $\delta\tau$.

- When instability sets in the fermion force grows. In particular the ratio of the ∞–norm of the force and the 2–norm of the force appears to suggest that there are just a few unstable "modes" responsible for the instability.

These results seem all seem consistent with the hypothesis of [39] which we now adopt as our working hypothesis for explaining the onset of instabilities in the MD. The assumption suggests the following picture:

When the term $F\delta\tau$ reaches a critical value in norm, the MD goes unstable. The instability may be controlled by tuning the algorithmic parameter $\delta\tau$. However as the norm of the fermionic contribution to the force appears to grow as $\kappa \to \kappa_c$, using lighter quarks implies the onset of the instability at smaller values of $\delta\tau$.

At this point we thought it desirable to map out the "safe region" of the $(\delta\tau, r)$ plane where, the MD remains stable for our given physical parameters, $(\beta = 5.2, \kappa = 0.1355, c = 2.0171)$.

4.4 Tuning Results

We carried out MD trajectories while varying values of $\delta\tau$ and r, using all of our 10 starting configurations. We kept the number of MD steps tuned so that the overall value of τ remained at $\tau = 0.97$. For these runs, we measured ΔH, N_{iters} as before to compute the cost function. We normalised the cost function with its value corresponding to parameters $\delta\tau = \frac{1}{180}$ and $r = 10^{-6}$ as this corresponds to the parameters used in our current production run.

Furthermore we have also carried out a trajectory with $r = 10^{-9}$ using double precision, to see if the instability was still present. Our results are plotted in figure 4.4.

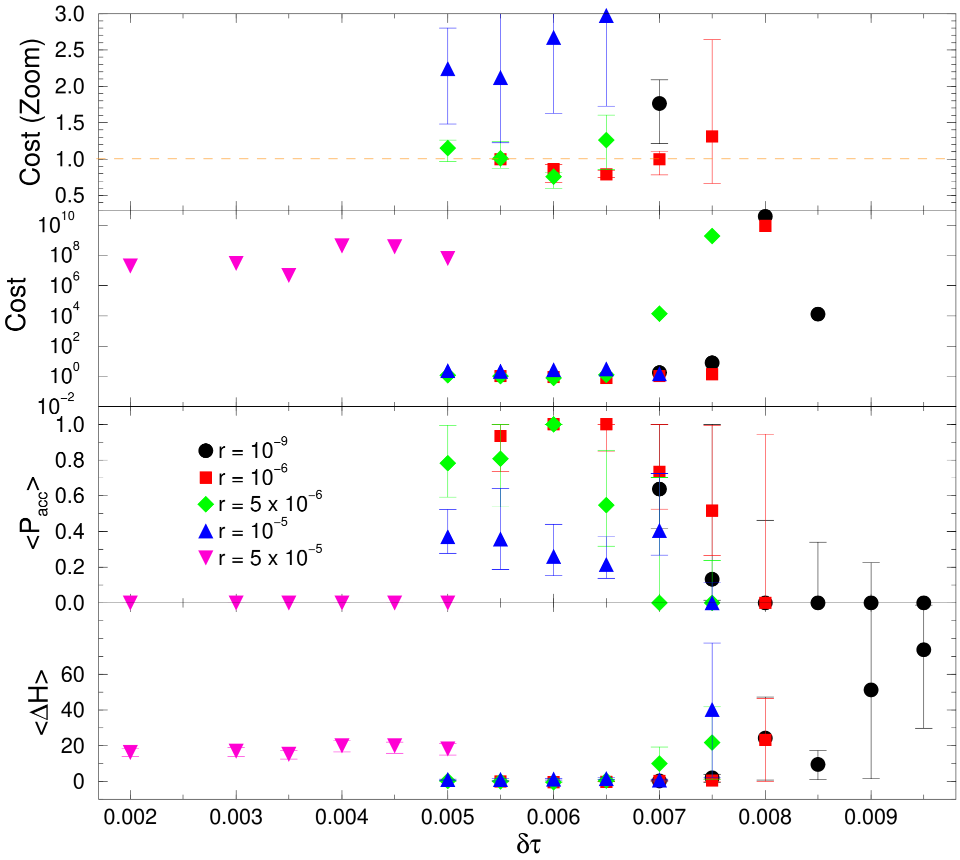

Figure 4.4: ΔH, $\langle P_{\rm acc}\rangle$ and the cost function as a function of $\delta\tau$, for a variety of r values.

Examining the graphs for $\langle \Delta H\rangle$ and $\langle P_{\rm acc}\rangle$ in figure 4.4 reveals that for $r = 5\times 10^{-5}$, we seem unable to obtain a reasonable acceptance rate even for step-sizes as small as $\delta\tau = 0.002$. The values of ΔH begin to grow large as $\delta\tau$ is increased, even for $r = 10^{-9}$ where the simulation is carried out in double precision, although the onset appears at slightly smaller values of $\delta\tau$ as the precision is reduced. It seems that a suitable upper bound for r is $r = 10^{-5}$ and a reasonable upper bound for $\delta\tau$, is about $\delta\tau \approx 0.007$.

Tuning is possible and small gains (again of the order of about 25%) can be made through judicious combination of r and $\delta\tau$. We note however that both tuning the step–size and the target residual appear to make the same amount of gain in terms of the cost function. We can see also an interplay between the two parameters, where reducing one needs to be counterbalanced by increasing the other and vice versa.

4.5 Conclusions for the chapter

We have identified instabilities in our MD evolution. We believe we can understand these in a generalisation of our hypothesis from free field theory. The hypothesis is that the instability sets in when a combination $F\delta\tau$ becomes suitably large in norm. This can occur through,

- The force growing large through the fermion mass becoming small
- The force being large due to inaccurate computation (varying r)
- The stepsize being increased keeping other parameters constant.

We have presented supporting evidence for this hypothesis and for our belief that the force diverges as $am_f \to 0$.

We believe this mechanism to adequately explain the results of our tuning exercise.

4.5.1 Conclusions for our tuning study

The chief results are that the instability seems to always be present when $r > 10^{-5}$ and for smaller values of r stability can be attained by tuning the step size $\delta\tau$.

In terms of tuning we believe the best procedure is to start off with values of r and $\delta\tau$ in the unstable region and to slowly lower them until stability is attained. Further decrease of r would bring with it a higher computational workload through the solves needing more iterations, and decreasing $\delta\tau$ would increase the workload through increasing the number of MD steps the algorithm needs to make.

We also find that at our current physical parameters, stability can be achieved when $r = O(10^{-6})$, a residual which it is possible to realize in a single precision solve. However this is close to the limits of single precision, and in future simulations, with lighter fermions, when the fermionic force is larger in norm one may need to carry out the solves in double precision to reach the region of stable MD. This result, combined with our earlier reversibility study seems to give a tentative answer to our question, that single precision computations can be safely carried out at our current physical parameters

4.5.2 Consequences for Future simulations

We would expect that as computational power grows, future simulations would aim to perform computations at lighter quark masses on larger volumes.

We expect that the effect of simulating at lighter quark masses is that a single precision computation of the fermionic force may not be adequate to keep the MD stable. However this problem can be circumvented by simply performing the solves for X and Y in double precision. The gauge fields may still be stored in single precision.

4.5.3 Consequences for Other Integration Schemes

Finally we mention a consequence of our results for the use of higher order integration schemes. Since these methods are expensive computationally, one probably would not be too tempted to use them anyway. However when instabilities set in one might wonder whether these could be side stepped by using a higher order integration scheme. Our answer to this issue is that using higher order integrators *would not* help in avoiding the instability. Our reasons are given below.

Consider a leapfrog algorithm that goes unstable at some value of $\delta\tau_{\max}$. Now consider a higher order scheme, making an update of the same length in time.

Now (2.51) indicates that the higher order scheme must make a step of length

$$|\delta\tau_2| = \left|\frac{\delta\tau_{\mathrm{max}}}{1 - \frac{2i}{s}}\right| \tag{4.15}$$

at some point to cancel the relevant error term. Since $i > 0$ and $s > 0$, we always have that $\delta\tau_2 > \delta\tau_{\mathrm{max}}$. Hence these higher order integrators need to make leapfrog updates for their central steps with a stepsize greater than $\delta\tau_{\mathrm{max}}$. From this we argue that a given higher order integrator will have at least one update step which is unstable, and we expect this to make the whole integration scheme unstable.

One may then be tempted not to use a higher order scheme, but to reduce the $O(\delta\tau^3)$ errors by applying the Sexton–Weingarten trick. Unfortunately we do not expect this method to avoid the instability either, as it functions by making many reduced length momentum updates using the gauge force (which is cheap to calculate), and somewhat fewer full length momentum updates using the fermion force. Unfortunately it is the fermion force and timestep combination in the fermionic momentum update that is responsible for the instability. As in the fermion force momentum updates neither the force or the timestep is changed we would expect that the use of these steps would drive the Sexton–Weingarten scheme unstable too.

Chapter 5

The Parallel Tempering Algorithm

5.1 Motivation

5.1.1 HMC and topological charge

It is believed, that the configuration space of both continuum and lattice QCD can be partitioned into several different sectors, each having different topological properties. In particular one can define in continuum QCD an observable over gauge configurations called the topological charge. This is an observable which, for any gauge configuration, should have an integer value, thus labelling the configuration as belonging to a given topological sector.

An algorithm for generating a physically representative ensemble of gauge configurations is required to sample different topological sectors with equal probability. Hence one would expect that the distribution of the topological charge over such an ensemble should be symmetric around a mean value of zero. If this condition is not met, one would expect an undue bias in the expectation values of some topologically sensitive physical observables, such as the proton mass [42].

It has been pointed out in [43, 44, 45] that during its progress through configuration space, the HMC algorithm can get trapped in individual topological sectors for a large number of trajectories. The expectation value of the topological charge over the resulting ensembles is neither zero nor is the distribution of the observable symmetric around the mean. Furthermore, being stuck in a given topological sector gives the topological charge an unduly long autocorrelation time. This difficulty is often referred to as the problem of "decorrelating topological charge".

We note that this problem appears more severe in the case of staggered fermions, as used in the investigations of [43, 44, 45], than for HMC with Wilson and Clover fermions. In particular the SESAM and TχL collaborations, who have employed Wilson fermions in their simulations, found that the distributions of the topological charge for their HMC ensembles are symmetric with a mean of zero [46, 47]. Recent, yet unpublished studies of the dynamical fermion data–sets of the UKQCD Collaboration support this view for Clover fermions.

One can speculate whether the reason for this difference in the decorrelation properties of the topological charge between different formulations of lattice fermions is due to the lower quark masses accessible when using the staggered prescription. If this is the only reason for the difference, it suggests that the problem is merely delayed for simulations with Wilson–like fermions until sufficiently light quark masses can be reached.

5.1.2 Tempering algorithms in Spin Glass Systems

Tempering algorithms were originally devised for the simulation of spin glass systems to solve a problem that is thought to be similar to the problem just discussed for lattice QCD. A spin glass system in its ordered phase may have many nearly degenerate free energy minima, separated by high free energy barriers. Traditional Monte Carlo simulation techniques for spin glass systems had difficulty surmounting these barriers and hence tended to get trapped in the minima. These free energy barriers were expected to become lower with increasing temperature.

In such situations the tempering approach was to change the value of the simulation parameter corresponding to the (inverse) temperature at certain points during the course of the simulation. It was envisaged that through a sequence of these changes the system could be brought to a temperature where the free energy barriers are not so high, and can be crossed. Eventually the system would be cooled back to its original temperature. In the end configurations at the temperature of interest would hopefully have good decorrelation properties.

The original tempering algorithm, Simulated Tempering [48], carried out exactly this approach. In this algorithm one system was simulated and occasionally it was suggested that its temperature parameter be changed to a different one, chosen from a fixed set of such parameters. The suggested parameter was accepted with a Metropolis acceptance test to ensure that the system remained in equilibrium. An alternative approach was the Parallel Tempering (PT) algorithm [4], originally called Exchange Monte Carlo, where the tempering scheme was implemented by carrying out many simulations concurrently and occasionally suggesting the exchange of configurations between subsystems. This technique may seem more efficient than simulated tempering if all the subsystems are of value in their own right and are not just used to decorrelate one particular ensemble of interest. Furthermore the simulated tempering algorithm introduced an extra parameter into the simulation action which had to be carefully tuned. In parallel tempering this constant is not present, making it more attractive than simulated tempering from a tuning point of view. However the remaining physical parameters still require some tuning in order to achieve reasonable acceptance rates.

5.1.3 Parallel Tempering in Lattice QCD

Tempering simulations, of both the Simulated and Parallel variety, have been performed in the case of lattice QCD by the Pisa group [49] following enthusiastic encouragement from [50]. As indicated earlier the problem of topological sectors in lattice QCD has been likened to the problem of free energy barriers in spin–glass models. The role of temperature for the spin glass systems was expected to be played by the quark mass in lattice QCD systems. Crossing between topological sectors, termed "tunnelling" seemed not to be a problem for simulations with relatively heavy quarks. It was perhaps for this reason that in [49] tempering was carried out by changing the fermion mass only.

In our study, we attempted tempering in both the independent couplings β and κ of our action. This was motivated by recent progress made in the technology of action matching [5, 6]. This technology has an important bearing on the acceptance rate of the parallel tempering algorithm as we will detail later.

We did not measure the topological charge in our simulations as we had no readily available code that we could integrate into our existing HMC code structure. We did however archive our configurations so that this measurement may be carried out in the future.

In the remainder of this chapter we formalise details of the PT algorithm. In particular we define what we mean by simulating several systems concurrently and treat the general PT algorithm as the combination of two kinds of Markov steps. We show that these Markov steps satisfy sufficient detailed balance relations for the whole scheme to converge to the correct equilibrium. We follow this description by a discussion of the acceptance rate of the algorithm and the relevance of action matching technologies.

Our discussions of the cost of the algorithm including our thoughts about how it affects autocorrelations of observables are presented in chapters 6 and 7, while our numerical computations will be detailed in chapter 8.

5.2 The Algorithm

As we have mentioned earlier, the PT algorithm simulates several systems concurrently. We shall refer to each such system as a *subensemble*. A simulation consists of N subensembles which we label by an integer $i = 1 \ldots N$.

5.2.1 Definition of a subensemble

Subensemble i has action S_i and coupling constants (β_i, κ_i, c_i) where these parameters have their usual meaning. For the action S_i we have used the usual clover improved pseudofermionic action

$$S_i(U,\phi) = -\beta_i \mathcal{W}(U) + \phi^\dagger Q_i(U)\phi \tag{5.1}$$

where we have written the usual gauge action term as $S_{pg}(U) = -\beta\mathcal{W}(U)$, with $\mathcal{W}(U) = \sum_{x,\mu\neq\nu} \mathrm{Re}\ TrU_{\mu\nu}(x)$ and have introduced the shorthand $Q_i = Q(\kappa_i, c_i; U)$ for the inverse fermion matrix. As we use HMC for some Markov steps in our algorithm, we also construct a Hamiltonian

$$H_i(U,\pi,\phi) = \frac{1}{2}\mathrm{Tr}\ \pi^2 + S_i(U,\phi) \tag{5.2}$$

for each subensemble i. A subensemble then has a state $a_i = (U_i, \pi_i, \phi_i)$, and a configuration space

$$\Gamma_i = \Gamma = \{U\} \otimes \{\pi\} \otimes \{\phi\}\ . \tag{5.3}$$

The corresponding equilibrium probability distribution of subensemble i is

$$P_i^{\mathrm{eq}}(U,\pi,\phi)\ \mathcal{D}U\ \mathcal{D}\pi\ \mathcal{D}\phi^\dagger\ \mathcal{D}\phi = \frac{1}{Z_i} e^{-H_i(U,\pi,\phi)}\ \mathcal{D}U\ \mathcal{D}\pi\ \mathcal{D}\phi^\dagger\ \mathcal{D}\phi \tag{5.4}$$

where

$$Z_i = \int \mathcal{D}U\ \mathcal{D}\pi\ \mathcal{D}\phi\ \mathcal{D}\phi^\dagger\ e^{-H_i(U,\pi,\phi)}\ . \tag{5.5}$$

To make notation easier we will drop the momenta, pseudofermions and their respective measures in future discussions, unless they are explicitly necessary. Furthermore, we will deal with actions S rather than Hamiltonian functions.

5.2.2 Joint Ensemble

The joint ensemble of N such subensembles as described above has configuration space

$$\Gamma^{\mathrm{PT}} = \prod_i^N \Gamma_i\ . \tag{5.6}$$

A configuration in this space will be denoted $a = (a_1, \ldots, a_N)$. The action of this joint ensemble is

$$S^{\mathrm{PT}}(a) = \sum_{i=1}^{N} S_i(a_i) \ . \tag{5.7}$$

The joint equilibrium probability distribution function is

$$P^{\mathrm{eq}}_{\mathrm{PT}}(a)\mathcal{D}\mu = \mathcal{D}\mu \prod_{i}^{N} P^{\mathrm{eq}}_i(a_i) \ , \tag{5.8}$$

where $\mathcal{D}\mu$ is the a measure on the joint phase space

$$\mathcal{D}\mu = \prod_{i=1}^{N} \mathcal{D}U_i \tag{5.9}$$

and $\mathcal{D}U_i$ is the measure on the configuration space copy of subensemble i.

The PT process is a Markov process in the configuration space Γ^{PT} which has (5.8) as its fixed point. This expresses what we meant earlier by demanding that the swapping of configurations/parameters must not affect the equilibrium of the subensembles. By this we mean precisely, that the overall Markov process should converge to have distribution (5.8). The individual ensembles may be coupled by the swapping, indeed such a process does induce cross correlations between the subensembles. At the end of a PT simulation however, the final probability distribution for the joint system still factorises, as if each simulation had been done independently.

5.2.3 The PT Markov Steps

We now define the two types of Markov steps from which the PT algorithm is constructed. They are :

1. **Markov Steps within Subensembles:** These steps are Markov update steps that one would normally use to simulate subensemble i on its own. The requirement on these steps is merely that they converge to P^{eq}, in other words that they are ergodic in subensemble i and have P^{eq}_i as their fixed point. In our simulations we employed HMC updates for these steps.

2. **Markov Steps between Subensembles:** These are our proposed swapping steps. We propose the swap of states a_i and a_j between subensembles i and j. Ignoring for now the other subensembles present, we write the PT state before the swap as the ordered pair (a_i, a_j). The swap step is the transition
$$(a_i, a_j) \to \begin{cases} (a_j, a_i) & \text{if the swap is accepted} \\ (a_i, a_j) & \text{if the swap fails .} \end{cases} \tag{5.10}$$
The swap is accepted with probability
$$P_{\text{acc}}((a_j, a_i) \leftarrow (a_i, a_j)) = \min\left(1, e^{-\Delta S}\right) \tag{5.11}$$
where
$$\Delta S = \{S_i(a_j) + S_j(a_i)\} - \{S_i(a_i) + S_j(a_j)\} \ . \tag{5.12}$$
We note that ΔS is the change in the joint action S^{PT} due to the swap. As the fields in subensembles other than i or j remain unchanged, the contribution from the actions of these subensembles cancel from ΔS.

   If one uses a Hamiltonian instead of an action, the change in joint Hamiltonian is
$$\Delta H = \{\frac{1}{2}\text{Tr } \pi_i^2 + \frac{1}{2}\text{Tr } \pi_j^2\} - \{\frac{1}{2}\text{Tr } \pi_i^2 + \frac{1}{2}\text{Tr } \pi_j^2\} + \Delta S \ , \tag{5.13}$$
and contributions from the momenta cancel whether they are swapped or not leaving $\Delta H = \Delta S$.

The swapping steps connect the otherwise unconnected configuration spaces Γ_i, thus ensuring ergodicity of the overall Markov process. However should we use no swap steps at all, the overall Markov process would still converge to the equilibrium distribution of (5.8), which is the equilibrium probability distribution of N independent simulations, providing we performed a sufficient number of internal updates in each subensemble.

We also point out that if we do not perform updates within the subensembles, we are doing no more than mixing and re–mixing an existing set of configurations. In this case the PT Markov Process is not ergodic. In summary, Markov updates within the subensembles are necessary, while swapping steps are not but if one performs no swaps one is not really performing a PT simulation.

5.2.4 Detailed Balance For Swap Steps

We now proceed to show that a swap step satisfies detailed balance with respect to the joint equilibrium distribution $P^{\rm eq}_{PT}\mathcal{D}\mu$. Once again we consider a pairwise swap of states a_i and a_j between subensembles i and j. The overall state of the joint ensemble before and after the swap step will be labelled a and a' respectively. As the swap proposal is inherently reversible the detailed balance condition is

$$P_{\rm acc}((a_j, a_i) \leftarrow (a_i, a_j))\ P^{\rm eq}_{\rm PT}(a)\ \mathcal{D}\mu_a = P_{\rm acc}((a_i, a_j) \leftarrow (a_j, a_i))\ P^{\rm eq}_{\rm PT}(a')\ \mathcal{D}\mu_{a'}\ . \tag{5.14}$$

It should be clear that the measure $\mathcal{D}\mu_a$ is completely unchanged by the swap apart from an inconsequential "reordering" in the product of (5.9). This has no effect on the measure so we can write that $\mathcal{D}\mu_a = \mathcal{D}\mu_{a'}$ and we are free to deal with the probability density functions alone in (5.14).

As fields in ensembles other than i or j are unchanged by the swap their equilibrium probability densities cancel from both sides of (5.14). If one uses Hamiltonians the contributions from all momentum fields cancel also, leaving

$$P_{\rm acc}((a_j, a_i) \leftarrow (a_i, a_j))\ e^{-\{S_i(a_i)+S_j(a_j)\}} = P_{\rm acc}((a_i, a_j) \leftarrow (a_j, a_i))\ e^{-\{S_i(a_j)+S_j(a_i)\}} \tag{5.15}$$

which the Metropolis acceptance formula of (5.11) satisfies by construction.

5.3 Swap Acceptance Rate

Following our discussion of detailed balance for the swaps it is a simple matter to follow a calculation similar to the one in section 2.3 to derive the acceptance rate of pairwise swaps between two subensembles i and j. We choose not to repeat the calculation here and give just the result

$$\langle P_{\rm acc}((a_j, a_i) \leftarrow (a_i, a_j))\rangle_{ij} = {\rm erfc}\left(\frac{1}{2}\sqrt{\langle \Delta S\rangle_{ij}}\right)\ . \tag{5.16}$$

The expectation value in the equation above is to be evaluated in the joint ensemble consisting of i and j only, which is equivalent to just averaging over the number of swap attempts made between the two subensembles.

In the course of our numerical simulations we have obtained excellent agreement with this formula for the swap acceptance rate as we will show in chapter 8.

5.4 Relevance of Action Matching Technology

We now discuss some relevant material from the technology of action matching. The original idea behind the technology was to tune so called approximate actions, which were supposed to be computationally cheap to simulate, so that they could be used to generate ensembles for some "true"action, the simulation of which would have been much more computationally demanding. The relevant formalism is outlined in [5] while applications in lattice QCD are detailed in [6].

The technology relies on comparing expectation values of observables measured in ensembles (subensembles in the case of PT) generated by simulations with two actions S_i and S_j, where the configuration spaces of these (sub)ensembles is the same.

The authors of [5] then define three tuning conditions for the actions S_i and S_j:

1. **Acceptance Rate Prescription :** This tuning prescription involves maximising the acceptance rate of an exact algorithm where trial configurations are generated for inclusion into an ensemble for action S_i. These configurations are then subjected to an acceptance test to determine whether or not they can be included into subensemble j.

2. **Distance Norm Prescription :** This condition requires minimising a distance norm between S_i and S_j in the space of actions. In PT, the actions of all the subensembles have the same form with different parameters, so the previous statement is equivalent to minimising a distance in the parameter space of the actions.

3. **Observable Matching Prescription :** This last condition requires fixing

the expectation values of a set of observables $\mathcal{O}_l$, with $l < N$ to be the same in both subensembles i and j.

Before presenting a more detailed description of the three tuning prescriptions we first introduce some machinery from [5] for comparing expectation values of an observables between two (sub)ensembles.

5.4.1 Comparison of expectation values

Consider making a direct comparison between the expectation values of observable $\mathcal{O}$ in two subensembles i and j, with actions S_i and S_j respectively. The relation between the expectation values can be found as follows

$$\begin{aligned}
\langle \mathcal{O} \rangle_j &= \frac{1}{Z_j} \int \mathcal{D}U \; \mathcal{O}(U) \; e^{-S_j(U)} \\
&= \frac{1}{Z_j} \int \mathcal{D}U \; \mathcal{O}(U) \; e^{-S_j(U)} \; e^{S_i(U) - S_i(U)} \\
&= \frac{1}{Z_j} \int \mathcal{D}U \; \mathcal{O}(U) \; e^{\Delta_{ij}} e^{-S_i(U)} \\
&= \frac{Z_i \left\langle \mathcal{O} e^{\Delta_{ij}} \right\rangle_i}{Z_j}
\end{aligned} \tag{5.17}$$

where in the third step we introduced $\Delta_{ij} = S_i(U) - S_j(U)$. Using similar reasoning it is easy to see that the partition function Z_j can be similarly expressed as

$$Z_j = \int \mathcal{D}U \; e^{-S_j(U)} = \int \mathcal{D}U \; e^{\Delta_{ij}} \; e^{-S_i(U)} = Z_i \left\langle e^{\Delta_{ij}} \right\rangle_i \; . \tag{5.18}$$

Hence one can write

$$\langle \mathcal{O} \rangle_j = \frac{\left\langle \mathcal{O} e^{\Delta_{ij}} \right\rangle_i}{\left\langle e^{\Delta_{ij}} \right\rangle_i} \; . \tag{5.19}$$

It is possible to rewrite (5.19) so that $\langle \mathcal{O} \rangle_j$ is given as $\langle \mathcal{O} \rangle_i$ and some additive correction:

$$\begin{aligned}
\langle \mathcal{O} \rangle_j &= \frac{\left\langle \mathcal{O} e^{\Delta_{ij}} \right\rangle_i}{\left\langle e^{\Delta_{ij}} \right\rangle_i} \\
&= \frac{\left\langle \mathcal{O} e^{\Delta_{ij}} \right\rangle_i}{\left\langle e^{\Delta_{ij}} \right\rangle_i} + \langle \mathcal{O} \rangle_i - \langle \mathcal{O} \rangle_i
\end{aligned}$$

$$\begin{aligned} &= \langle \mathcal{O} \rangle_i + \frac{\left\langle \mathcal{O} e^{\Delta_{ij}} \right\rangle_i}{\left\langle e^{\Delta_{ij}} \right\rangle_i} - \langle \mathcal{O} \rangle_i \frac{\left\langle e^{\Delta_{ij}} \right\rangle_i}{\left\langle e^{\Delta_{ij}} \right\rangle_i} \\ &= \langle \mathcal{O} \rangle_i + \langle (\mathcal{O} - \langle \mathcal{O} \rangle_i) \, C_i(\Delta_{ij}) \rangle_i \ , \end{aligned} \tag{5.20}$$

with

$$C_i(\Delta_{12}) = \frac{1}{\langle \Delta_{ij} \rangle_i} e^{\Delta_{ij}}, \qquad i = 1, 2. \tag{5.21}$$

Typically, Δ_{ij} is an extensive quantity growing as $O(V)$, V being the system volume. To predict the expectation value of an observable in ensemble j from that in i involves computing covariances with $C_i(\Delta_{ij})$ which grow as $O(e^V)$. The fluctuations of the correction terms are also expected to be $O(e^V)$ and so the number of samples of $\mathcal{O}$ required for a reasonable estimate of the required covariances must also grow exponentially with the system volume. Hence computing correlations with $C_i(\Delta_{ij})$ is not numerically feasible.

One can however express $C_i(\Delta_{ij})$ in a Taylor series,

$$C_i(\Delta_{ij}) = 1 + \left(\Delta_{ij} - \langle \Delta_{ij} \rangle_i\right) + \frac{1}{2}\left[\left(\Delta_{ij} - \langle \Delta_{ij} \rangle_i\right)^2 + \langle \Delta_{ij} \rangle_i^2 - \left\langle \Delta_{ij}^2 \right\rangle_i\right] + \text{h.o.t} \tag{5.22}$$

from which

$$\langle \mathcal{O} \rangle_j - \langle \mathcal{O} \rangle_i = \left\langle (\mathcal{O} - \langle \mathcal{O} \rangle_i) \left(\Delta_{ij} - \langle \Delta_{ij} \rangle_i\right) \right\rangle_i + \text{h.o.t} \tag{5.23}$$

The first order term in the difference between the expectation values is now only $O(V)$, the second order term is $O(V^2)$ and in general the n-th order term is $O(V^n)$. As a numerical simulation is necessarily carried out at a finite volume it is hoped that at least the first few terms of this expansion can be computed with a finite (although possibly large) amount of work. We now turn to the three tuning prescriptions.

5.4.2 Acceptance Rate Prescription

Originally this prescription was designed for a procedure to which we will refer as Exact Simulation with an Approximate Action (ESAA). In such a scheme, one chooses some theory with approximate action S_a which has efficient[1] techniques available for its simulation. One then generates configurations for the equilibrium distribution of S_a. To decide whether these configurations can be included into an ensemble corresponding to the desired "true" action S_t, one subjects them to a suitable accept/reject step.

We require, that the configuration space of the theory with action S_a be identical to that of the theory corresponding to S_t. Furthermore we require that the method for generating configurations for the ensemble of S_a be a Markov process which satisfies detailed balance with respect to the equilibrium probability distribution $P_a^{\rm eq}(U)$ of S_a. We can then look at a step of such a method, as reversibly generating a trial configuration U' from an initial configuration U, taking the role of $P_c(U' \leftarrow U)$ in a Markov process for the simulation with action S_t.

The detailed balance condition for the accept/reject step in terms of the probability density functions is

$$P_{\rm acc}(U' \leftarrow U)P_c(U' \leftarrow U)P_t^{\rm eq}(U) = P_{\rm acc}(U \leftarrow U')P_c(U \leftarrow U')P_t^{\rm eq}(U') \ . \quad (5.24)$$

We do not write down the measures here as they are the same on both sides of this equation.

A suitably chosen Metropolis accept/reject step will satisfy this detailed balance equation constraint by construction. Using the fact that P_c satisfies detailed balance with respect to $P_a^{\rm eq}(U)$,

$$\frac{P_c(U \leftarrow U')}{P_c(U' \leftarrow U)} = \frac{e^{-S_a(U)}}{e^{-S_a(U')}} \ , \quad (5.25)$$

we can write the Metropolis acceptance probability as

$$\begin{aligned} P_{\rm acc}(U' \leftarrow U) &= \min\left(1, \frac{P_c(U \leftarrow U')e^{-S_t(U')}}{P_c(U' \leftarrow U)e^{-S_t(U)}}\right) \\ &= \min\left(1, \frac{e^{-S_a(U)}e^{-S_t(U')}}{e^{-S_a(U')}e^{-S_t(U)}}\right) \\ &= \min\left(1, e^{-S_a(U)+S_t(U)+S_a(U')-S_t(U')}\right) \end{aligned} \quad (5.26)$$

[1] In this case efficient = computationally cheap

$$\begin{aligned} &= \min\left(1, e^{-\Delta_{at}|_U + \Delta_{at}|_{U'}}\right) \\ &= \min\left(1, e^{-\delta}\right), \end{aligned} \tag{5.27}$$

where we make the the definitions

$$\Delta_{at}|_U = S_a(U) - S_t(U) \tag{5.28}$$

and

$$\delta = \Delta_{at}|_U - \Delta_{at}|_{U'} , \tag{5.29}$$

to follow the notation of [5]. From our previous discussions of acceptance rates it should be clear that the acceptance rate for such an algorithm is

$$\langle P_{\mathrm{acc}}(U' \leftarrow U)\rangle_{at} = \mathrm{erfc}\left(\frac{1}{2}\sqrt{\langle\delta\rangle_{at}}\right) \tag{5.30}$$

where the expectation value of δ is evaluated in the joint ensemble of actions S_a and S_t.

We note that the quantity δ corresponds exactly to ΔS of (5.12) in a PT swap update step between subensembles i and j with actions $S_i = S_a$ and $S_j = S_t$. Also, if $P_c(U' \leftarrow U)$ corresponds to a Markov step where configuration U' is chosen directly from a heatbath for the distribution of S_a,

$$P_c(U' \leftarrow U) \ \propto \ e^{-S_a(U')} \tag{5.31}$$

then the detailed balance condition of (5.24) becomes identical to the one for swapping configurations U' and U between the previously mentioned two PT subensembles as can be seen by examining (5.14). Both these statements imply that the accept/reject step in the ESAA algorithm and the accept/reject step of a PT swap update between two corresponding subensembles are equivalent. Indeed one can look at a twin subensemble PT simulation as a modified version of the ESAA algorithm, in the special case where the actions $S_i = S_a$ and $S_j = S_t$ have the same form, but S_i is computationally cheaper to simulate with than S_t. For example S_t could correspond to a lattice QCD simulation with light quarks and S_a to one with heavy quarks where one needs fewer solver iterations to compute the fermion force and the system energy than in the light fermion case.

In the ESAA approach, however only one (sub)ensemble is generated, that of $S_t = S_j$ and hence one cannot evaluate $\langle\delta\rangle_{at}$ directly. However if $P_c(U' \leftarrow U)$

is of the form (5.31) one can express $\langle\delta\rangle_{at}$ as an expectation value in just the ensemble of S_t,

$$\begin{aligned}\langle\delta\rangle_{at} &= \langle\Delta_{at}|_U - \Delta_{at}|_{U'}\rangle_{at} \\ &= \langle\Delta_{at}\rangle_a - \langle\Delta_{at}\rangle_t \\ &= \left\langle(\Delta_{at} - \langle\Delta_{at}\rangle_t)^2\right\rangle_t + O(\Delta_{at}^3) \\ &= \sigma_t^2(\Delta_{at}) + O(\Delta_{at}^3) \qquad (5.32)\end{aligned}$$

where in the third step the cumulant expansion, defined in (5.23), has been used. In the case of PT, the situation is more symmetric as one generates subensembles for both actions.

We now return to the language of PT, writing ΔS instead of δ and we shall stop using the subscripts a and t. In this language we write

$$\langle\delta\rangle_{ij} \approx \sigma_i^2(\Delta_{ij}) \qquad (5.33)$$

in other words, to first order in Δ_{ij} it does not matter which ensemble the variance is computed in.

We can also see from (2.66) in our computation of the HMC acceptance rate that,

$$\langle\Delta S\rangle_{ij} = \frac{1}{2}\sigma_{ij}^2(\Delta S)\ . \qquad (5.34)$$

Thus we can relate the variances $\sigma_{ij}^2(\Delta S)$, $\sigma_i^2(\Delta_{ij})$ and $\sigma_j^2(\Delta_{ij})$ and the expectation value $\langle\Delta S\rangle_{ij}$ to first order as

$$\sigma_i^2(\Delta_{ij}) \approx \sigma_j^2(\Delta_{ij}) \approx \frac{1}{2}\sigma_{ij}^2(\Delta S) = \langle\Delta S\rangle_{ij}\ . \qquad (5.35)$$

The tuning condition for maximising the acceptance rate in the ESAA and PT approaches can be then be re-stated as minimising the variance of Δ_{ij} in either subensemble or minimising the variance of ΔS in the joint ensemble of subensembles i and j.

5.4.3 Distance Norm Prescription

We now turn to the second tuning prescription mentioned earlier, that of minimising a distance norm in the space of actions. It has been shown in [51] that a measure defined by an action S_i allows the construction of an inner product between gauge invariant functions, say F and G, of the degrees of freedom as

$$(F, G) = \left\langle F^{\dagger} G \right\rangle_i , \tag{5.36}$$

turning the space of gauge invariant functions into a Hilbert space. Thus one can define the distance between any two such functions as

$$d(F, G) = \left\langle (F - G)^{\dagger} (F - G) \right\rangle \tag{5.37}$$

As the actions are gauge invariant functions of the gauge fields, we can then consider a distance between two actions S_i and S_j to be defined as

$$d(S_i, S_j) = \left\langle (S_i - S_j)^2 \right\rangle \tag{5.38}$$

or more generally, noting that the addition of constant factors to the action results only in a change of normalisation for a given path integral the definition of the distance can be written as

$$d(S_i - \langle S_i \rangle , S_j - \langle S_j \rangle) = \left\langle (\Delta - \langle \Delta_{ij} \rangle)^2 \right\rangle = \sigma^2 \left(\Delta_{ij} \right) . \tag{5.39}$$

Hence the distance between the two actions is given by the variance of Δ_{ij}. The variances can be evaluated in either subensemble i or j. The relation to $\sigma^2_{ij}(\Delta S)$ is given by (5.35).

These statements then imply that to first order in Δ_{ij} minimising the distance norm in parameter space and maximising the acceptance rate of the swap attempts of PT simulations are equivalent.

5.4.4 Observable Matching Prescription

The physical motivation for this last tuning condition, in the words of [5], is that in some cases at least, two actions with different couplings can be taken to be different regularisations of the same continuum theory. In a PT algorithm, where all the actions have the same form, this is almost always the case.[2]

[2]The obvious exception to this rule is the case when $\kappa = 0$ and one is dealing with the quenched approximation for which the continuum limit is known to be different from dynamical fermion actions

Given an approximate action S_i with n independent coupling constants, one needs n independent "renormalisation" conditions to match the corresponding theory to that of another action S_j. For example one can fix the values of some number $m \leq n$ of observables $\mathcal{O}_l$ in both measures, thus demanding

$$\langle \mathcal{O}_l \rangle_j = \langle \mathcal{O}_l \rangle_i \qquad l = 1 \ldots m \tag{5.40}$$

for each observable $\mathcal{O}_l$.

Using the cumulant expansion of (5.23) we can write to first order

$$\langle \mathcal{O}_l \rangle_j = \langle \mathcal{O}_l \rangle_i + \Big\langle \left(\mathcal{O}_l - \langle \mathcal{O}_l \rangle_i \right) \Big(\Delta_{ij} - \langle \Delta_{ij} \rangle_i \Big) \Big\rangle_i \tag{5.41}$$

and it is immediately clear that for (5.40) to be satisfied one must have

$$\Big\langle \left(\mathcal{O}_l - \langle \mathcal{O}_l \rangle_i \right) \Big(\Delta_{ij} - \langle \Delta_{ij} \rangle_i \Big) \Big\rangle_i = 0 \tag{5.42}$$

We note that this prescription is not equivalent to the other two discussed previously except in some special cases which are detailed in [5].

The utility of this latter prescription is that it allows one to find curves of constant observables in parameter space. One could consider, for example, a curve along which the observable r_0 was kept constant. Simulations, corresponding to points on such a curve would have equal physical volumes in the asymptotic scaling region. However, if the simulations are not in the scaling region, the value of a determined from r_0 is not expected to be unique. In particular, a different value for a could be obtained, in principle, from the mass of the vector particle m_ρ. In our simulations, we used r_0 to set the lattice scale merely for the reason that this observable is quite easy to determine computationally.

The benefits of such a "constant lattice spacing" scheme are expected to appear when one needs to perform extrapolations during the course of analysing the simulation results. It is known, for example that changing the parameter κ in lattice QCD simulations can have quite noticeable effects on the lattice spacing. Thus in traditional simulations where one changes κ only, extrapolations of observables in κ can be hampered by finite volume effects. If one can set up curves of constant lattice spacing (as specified by r_0 say) one would expect to have these finite volume effects better under control. We refer the reader to [6] for further details of this approach.

5.5 Minimising the distance

So far the discussion of matching has been rather abstract. In this section we discuss how to tune some actions of interest in lattice QCD. In the discussion to follow, we will deal with Wilson fermions only ($c_i = 0$), for which the tuning exercise is an explicit problem. Clover fermions, introduce an added dependence on β in the inverse fermion matrix Q as the clover coefficient is a function of β. This complicates matters, making the tuning process once again an implicit problem. For a discussion of how to tune in the presence of a non–zero clover coefficient we refer the reader to [6].

As usual we shall focus on two subensembles, this time labelled subensemble 1 and subensemble 2. For notational simplicity we introduce another piece of notation from [5]:

$$\tilde{\mathcal{O}} = \mathcal{O} - \langle \mathcal{O} \rangle \ . \tag{5.43}$$

We discuss the tuning of parameters for two particular actions. One is the "Trace Log" action

$$S_i^{\mathrm{Trln}}(U) = -\beta_i \mathcal{W} - \mathrm{Tr}\ \ln Q_i^{-1} = -\beta_i \mathcal{W} + \mathrm{Tr}\ \ln Q_i \tag{5.44}$$

and the other is the usual pseudofermion action

$$S_i^{\mathrm{p.f.}}(U, \phi, \phi^\dagger) = -\beta_i \mathcal{W} + \phi^\dagger Q_i \phi \tag{5.45}$$

The discussion of tuning these particular actions is pertinent as our action matching computations employed S^{Trln} while our PT simulations used $S^{\mathrm{p.f.}}$. One of the points to be made by the following calculations is that although tuning S^{Trln} is equivalent to tuning $S^{\mathrm{p.f.}}$ to first order in a Taylor expansion, the minimum value of $\sigma^2(\Delta_{ij})$ is larger when one uses pseudofermions than in the case where one does not.

The tunable parameters for both of these actions are $\Delta\beta = \beta_2 - \beta_1$ and $\Delta\kappa = \kappa_2 - \kappa_1$. We adopt the procedure where we fix $\Delta\kappa$ to some desired quantity, and then tune $\Delta\beta$, although it is also possible to tune $\Delta\kappa$ for a fixed $\Delta\beta$.

5.5.1 Tuning the Trace Log action

We now consider tuning the action S^{Trln}. As $\Delta\kappa$ is fixed the tuning prescription is to minimise the variance $\sigma_i^2(\Delta_{12})$ with respect to $\Delta\beta$, in either subensemble $i = 1, 2$. The quantity Δ_{12} is given by

$$\Delta_{12} = \Delta\beta\mathcal{W} + \Delta T \tag{5.46}$$

where we define

$$\Delta T = -\mathrm{Tr}\ \ln Q_1^{-1} + \mathrm{Tr}\ \ln Q_2^{-1} = \mathrm{Tr}\ \ln Q_1 - \mathrm{Tr}\ \ln Q_2\ . \tag{5.47}$$

The variance of this quantity is then

$$\begin{aligned} \sigma_i^2(\Delta_{12})|_{\mathrm{Trln}} &= \left\langle (\Delta\beta\widetilde{\mathcal{W}} + \widetilde{\Delta T})^2 \right\rangle_i \\ &= \Delta\beta^2 \left\langle \widetilde{\mathcal{W}}^2 \right\rangle_i + 2\Delta\beta \left\langle \widetilde{\mathcal{W}}\widetilde{\Delta T} \right\rangle_i + \left\langle \widetilde{\Delta T}^2 \right\rangle. \end{aligned} \tag{5.48}$$

To minimise the variance we need to solve the equation

$$\frac{d\sigma_i^2(\Delta_{12})}{d(\Delta\beta)} = 0 \tag{5.49}$$

Carrying out the differentiation with respect to $\Delta\beta$ gives,

$$\frac{d\sigma_i^2(\Delta_{12})}{d(\Delta\beta)} = 2\left(\Delta\beta \left\langle \widetilde{\mathcal{W}}^2 \right\rangle_i + \left\langle \widetilde{\mathcal{W}}\widetilde{\Delta T} \right\rangle_i\right) \tag{5.50}$$

leaving one to solve

$$\Delta\beta \left\langle \widetilde{\mathcal{W}}^2 \right\rangle_i + \left\langle \widetilde{\mathcal{W}}\widetilde{\Delta T} \right\rangle_i = 0 \tag{5.51}$$

which immediately gives

$$\Delta\beta = -\frac{\left\langle \widetilde{\mathcal{W}}\widetilde{\Delta T} \right\rangle_i}{\left\langle \widetilde{W}^2 \right\rangle_i} \tag{5.52}$$

as the required tuning condition.

We would like to raise some obvious points from the calculation above. Looking at the definition of $\sigma^2(\Delta_{12})$ in (5.48) we see that it is a quadratic polynomial in $\Delta\beta$. Hence it has only one turning point, and this turning point is a minimum as the coefficient of the $\Delta\beta^2$ term is a variance which is a positive definite quantity. The minimum value of the variance therefore is

$$\min \sigma_i^2(\Delta_{12})|_{\mathrm{Tr\ ln}} = \left\langle \widetilde{\Delta T}^2 \right\rangle_i = \sigma_i^2(\Delta T)\ . \tag{5.53}$$

This value is dependent on $\Delta\kappa$ only and cannot be tuned away by varying $\Delta\beta$. We note also that $\sigma^2(\Delta T)$ is a positive definite quantity. Hence the minimum value of $\sigma^2(\Delta T)$ is zero which is realised when actions S_1 and S_2 are the same.

5.5.2 Tuning the Pseudofermion Action

The calculation for tuning action $S^{\mathrm{p.f}}$ is a little more complicated than the previous one due to the presence of the pseudofermions. For this calculation we write the variance as

$$\sigma_i^2(\Delta_{12}) = \left\langle \Delta_{12}^2 \right\rangle_i - \left\langle \Delta_{12} \right\rangle_i^2 \tag{5.54}$$

and deal with each term separately.

We begin by noting that

$$\Delta_{12} = \Delta\beta\mathcal{W} + \phi^\dagger(Q_1 - Q_2)\phi \ . \tag{5.55}$$

Hence,

$$\left\langle \Delta_{12} \right\rangle_i = \Delta\beta \left\langle \mathcal{W} \right\rangle_i + \left\langle \phi^\dagger(Q_1 - Q_2)\phi \right\rangle_i \ . \tag{5.56}$$

In (5.56) above, we can carry out the integration over the pseudofermion fields in the second term so that we need consider only expectation values over gauge fields.

$$\left\langle \phi^\dagger(Q_1 - Q_2)\phi \right\rangle_i^\phi = \left\langle \mathrm{Tr}\ (Q_i^{-1}(Q_1 - Q_2)) \right\rangle_i = \left\langle \Delta\mathcal{T}_i \right\rangle_i \ , \tag{5.57}$$

where for notational simplicity we define

$$\Delta\mathcal{T}_i = \mathrm{Tr}\ (Q_i^{-1}(Q_1 - Q_2)) \ . \tag{5.58}$$

We then find that

$$\left\langle \Delta_{12} \right\rangle_i^2 = \Delta\beta^2 \left\langle \mathcal{W} \right\rangle_i^2 + 2\Delta\beta \left\langle \mathcal{W} \right\rangle_i \left\langle \Delta\mathcal{T}_i \right\rangle_i + \left\langle \Delta\mathcal{T}_i \right\rangle_i^2 \ . \tag{5.59}$$

For $\left\langle \Delta_{12}^2 \right\rangle_i$ we have,

$$\left\langle \Delta_{12}^2 \right\rangle_i = \Delta\beta^2 \left\langle \mathcal{W}^2 \right\rangle_i + 2\Delta\beta \left\langle \mathcal{W}\phi^\dagger(Q_1 - Q_2)\phi \right\rangle_i + \left\langle \phi^\dagger(Q_1 - Q_2)\phi\phi^\dagger(Q_1 - Q_2)\phi \right\rangle_i \ . \tag{5.60}$$

Again we can then carry out the integration over the pseudofermion fields to obtain

$$\left\langle \mathcal{W}\phi^\dagger(Q_1 - Q_2)\phi\right\rangle_i^\phi = \langle \mathcal{W}\Delta\mathcal{T}_i\rangle_i \tag{5.61}$$

and

$$\left\langle \phi^\dagger(Q_1 - Q_2)\phi\phi^\dagger(Q_1 - Q_2)\phi\right\rangle_i^\phi = \left\langle \mathrm{Tr}\ (Q_i^{-1}(Q_1 - Q_2)Q_i^{-1}(Q_1 - Q_2)\right\rangle_i + \left\langle \Delta\mathcal{T}_i^2\right\rangle_i . \tag{5.62}$$

Collecting all the terms we can now write the required variance as

$$\begin{aligned} \sigma_i^2(\Delta_{ij})|_{\mathrm{p.f}} &= \Delta\beta^2\left\langle \widetilde{\mathcal{W}}^2\right\rangle_i + 2\Delta\beta\left\langle \widetilde{\mathcal{W}}\widetilde{\Delta\mathcal{T}_i}\right\rangle_i + \left\langle \widetilde{\Delta\mathcal{T}_i}\right\rangle_i^2 \\ &+ \left\langle \mathrm{Tr}\ (Q_i^{-1}(Q_1 - Q_2)Q_i^{-1}(Q_1 - Q_2))\right\rangle_i . \end{aligned} \tag{5.63}$$

Differentiating with respect to $\Delta\beta$ gives

$$\frac{d\sigma_i^2(\Delta_{ij})|_{\mathrm{p.f}}}{d(\Delta\beta)} = 2\Delta\beta\left\langle \widetilde{\mathcal{W}}^2\right\rangle_i + 2\left\langle \widetilde{\mathcal{W}}\widetilde{\Delta\mathcal{T}_i}\right\rangle \tag{5.64}$$

resulting in the tuning condition

$$\Delta\beta = -\frac{\left\langle \widetilde{\mathcal{W}}\widetilde{\Delta\mathcal{T}_i}\right\rangle}{\left\langle \widetilde{\mathcal{W}}^2\right\rangle_i} . \tag{5.65}$$

It can be shown, (see appendix) that to first order in a Taylor expansion, $\Delta T = \Delta\mathcal{T}_i$ implying that, the tuning conditions (5.52) and (5.65) are approximately the same. However, we find that

$$\min\sigma^2(\Delta_{12})|_{\mathrm{p.f}} = \left\langle \widetilde{\Delta\mathcal{T}_i}^2\right\rangle_i + \left\langle \mathrm{Tr}\ (Q_i^{-1}(Q_1 - Q_2)Q_i^{-1}(Q_1 - Q_2))\right\rangle_i \tag{5.66}$$

which is clearly different from (5.53) due to the second term on the right hand side of (5.66) to which we shall refer as the connected term. Thus, for fixed values of $\Delta\beta$ and $\Delta\kappa$, distances in the space of parameters are larger if one uses the $S_{\mathrm{p.f.}}$ than if one were to use S_{Trln}. Consequentially, the minimum distance attainable for a given $\Delta\kappa$, is larger in the case when one uses $S_{\mathrm{p.f}}$ than in the case where one used S_{Trln}. Hence, for a given $\Delta\kappa$, a higher maximum swap acceptance rate is available when one uses the S_{Trln} than when one uses $S_{\mathrm{p.f}}$.

5.6 Lessons learned from Action Matching

We now summarise the main conclusions for PT from the technology of action matching:

- In a PT simulation tuning the parameters of a pair of subensembles i and j to maximise the acceptance rate for swaps of configurations between them is equivalent (to first order in Δ_{ij}) to minimising the distance norm between S_i and S_j in the space of actions, or equivalently in the PT parameter space.
- The distance is defined as $\sigma_i^2(\Delta_{ij})$ in either subensemble or one can think of it as $\frac{1}{2}\sigma_{ij}^2(\Delta S)$ in the joint ensemble of subensembles i and j.
- The distances in the space of actions/parameters are extensive quantities and grow linearly with the system volume.
- It is better to consider the fundamental degrees of freedom for subensemble i to be $a_i = (U_i)$ than $a_i = (U_i, \pi_i, \phi_i)$. It is also better to use the S^{Trln} when swapping configuration between subensembles than $S^{\mathrm{p.f.}}$. This is because of the connected contribution to $\sigma^2(\Delta_{ij})$ arising from the pseudofermions, which increases the minimum distance attainable in the parameter space (and decreases the maximum swap acceptance rate) compared to the case of using S^{Trln}.

The most depressing of these conclusions is perhaps the third one. More clearly stated the implication of it is that for a fixed pair of points in the space of parameters, the distance between these points grows linearly with system volume. Using the acceptance rate formulae for PT/ESAA, (5.16) and (5.30) one would expect a corresponding exponential decrease in the swap acceptance rate. Hence to attain reasonable acceptance rates at large volumes, the parameters of the subensembles may have to be very close to each other in the parameter space indeed.

Chapter 6

The Cost of Parallel Tempering

6.1 Initial Considerations

In this chapter we wish to consider the cost of the Parallel Tempering algorithm. In particular we would like to discuss whether or not PT is cost–effective. The discussion has two major aspects which we will now consider in turn.

6.1.1 Criteria for Cost Effectiveness

Firstly we have to define what we mean by PT being cost effective. For this we need some notion of cost for a PT algorithm and some basis for comparing this cost with that of other algorithms. This latter issue is made nontrivial by the fact that traditional Monte Carlo algorithms usually produce just a single configuration ensemble, whereas PT generates configurations in all its subensembles.

We shall discuss two viewpoints regarding the cost effectiveness of PT. In the first one all the PT subensembles are considered to be of interest in the sense that if one did not have PT available, one would carry out independent HMC simulations corresponding to each of the subensembles. The criterion for cost–effectiveness in this case will involve the comparison of the total cost of the PT simulation with the combined cost of the set of corresponding HMC simulations.

In the second viewpoint one assumes that one is interested only in one of the PT subensembles. The criterion of cost–effectiveness then involves comparing the cost of the PT simulation to that of a single HMC simulation with the same parameters as the PT subensemble of interest.

In both cases we shall find that comparison of cost between the PT and HMC algorithms reduces to the consideration of the autocorrelations for the two methods.

6.1.2 Actual behaviour of PT Costs

The next point in the discussion is to consider how autocorrelations in a PT simulation actually behave. In particular one is interested in whether PT can actually meet the cost–effectiveness criteria.

In this discussion we shall first consider the case of PT when none of the swap attempts succeeds. In this situation one is effectively dealing with a set of independent HMC simulations with corresponding autocorrelation times. A reduction in these autocorrelation times can come through successful swap steps. We shall argue that this reduction can come about in two ways:

- The successful swap steps induce a mixing of configurations between subensembles. We shall refer to this as a mechanical reduction in autocorrelation times due to mixing.
- In the zero swap case the PT autocorrelation times correspond to those of HMC simulations. Different subensembles may have different values of these HMC autocorrelation times. We would expect that the swapping would effect some kind of averaging of the HMC autocorrelation times between subensembles, speeding up slow ones and slowing down fast ones.

The worst case scenario is when all the PT subensembles have the same autocorrelation time for a given observable in the zero swap case, and one can rely only on the mechanical reduction in these autocorrelation times due to the mixing induced by the swaps. We shall investigate this scenario by considering a PT system composed of two subensembles.

Before proceeding to the discussion outlined above we note at this point that a PT simulation is made up of two kinds of Markov steps, those within a given subensemble – the internal steps – and those that connect two subensembles – the swapping steps. We mentioned in the previous chapter that one has considerable freedom in combining these two kinds of Markovian updates into an overall algorithm. Hence discussions of cost–effectiveness for PT are not meaningful until one specifies the exact combination of the updates one wishes to use.

In light of the above points this chapter is organised as follows. Firstly we shall review the topic of autocorrelations and how they affect simulation cost for a Markov process generating a single configuration ensemble. We then fix the particular PT scheme which we use for the remainder of this thesis. Having done this we proceed to analyse the cost of such a PT simulation and present our conditions for it to be cost effective. Finally we shall discuss whether or not PT can meet these criteria by analysing the autocorrelations in a twin subensemble PT system.

6.2 Autocorrelations and Simulation Cost

6.2.1 Cost of a single ensemble simulation

Let us now consider a Monte Carlo algorithm of the traditional type, for generating a single gauge configuration ensemble. We consider such an algorithm to consist of a sequence of fundamental Markov updates. A fundamental update is one which produces a gauge configuration that is stored as part of the ensemble. For example in HMC a fundamental update is made up of a momentum refresh step followed by an MD trajectory of length τ ending in an accept/reject step. Such a procedure then produces a sequence of gauge configurations $\{U_t\}$ where t is the sequence number of the fundamental update that produced the configuration. We shall assume that successive configurations in this sequence are correlated.

Let us now suppose that the ensemble consists of N_c configurations and we wish to estimate the expectation value $\langle\mathcal{O}\rangle$ of some observable $\mathcal{O}$, by computing the sample mean $\bar{\mathcal{O}}$. The statistical error $\epsilon_{\mathcal{O}}$ on the estimate $\bar{\mathcal{O}}$ is then

$$\epsilon_{\mathcal{O}} = \sqrt{\frac{2\mathcal{A}_{\mathcal{O}}+1}{N_c}\sigma^2(\mathcal{O})} \tag{6.1}$$

where $\mathcal{A}_{\mathcal{O}}$ is the integrated autocorrelation time for $\mathcal{O}$ and $\sigma^2(\mathcal{O})$ is the intrinsic variance of $\mathcal{O}$ that one would measure in an ensemble of independent configurations.

The definition of $\mathcal{A}$ used above is from [52]. This particular definition is attractive in the sense that in the case of a sequence of independent configurations $\mathcal{A}_{\mathcal{O}} = 0$. The quantity $\mathcal{A}_{\mathcal{O}}$ itself is defined in this notation as

$$\mathcal{A}_{\mathcal{O}} = \sum_{1}^{\infty} C_{\mathcal{O}}(t) \tag{6.2}$$

where

$$C_{\mathcal{O}}(t) = \frac{1}{\sigma^2(\mathcal{O})}\left\langle(\mathcal{O}(t+t_0)-\langle\mathcal{O}\rangle)(\mathcal{O}(t_0)-\langle\mathcal{O}\rangle)\right\rangle \tag{6.3}$$

is the normalised autocorrelation function for the observable over the ensemble. Here $\mathcal{O}(t)$ is the value of the observable evaluated on configuration U_t and the expectation value is over all pairs of configurations in the sequence separated by t Markov updates. From here on we shall drop the subscripts $\mathcal{O}$ on $\mathcal{A}$ and $C(t)$, as the observable dependence is explicit in (6.3). Since we do not consider the so called exponential autocorrelation time in the ensuing discussion we shall freely refer to the integrated autocorrelation time of an observable simply as its autocorrelation time.

We note that the definitions in (6.2) and (6.3) both implicitly assume that the configuration ensemble contains a countably infinite number of samples. In the case of a numerical simulation however one has only N_c configurations available and hence one is reduced to constructing estimators for $C(t)$ and $\mathcal{A}$ as described in [53][1] and Appendix A. We remark that generally, the signal on these estimators

[1]This reference, uses as slightly different definition of the integrated autocorrelation time from the one used here. The difference between the two definitions is a constant additive factor of $\frac{1}{2}$.

is very noisy and a large number of configurations are needed for a reasonable estimate.

We can then define the computational cost for a simulation using this Markov process as the average CPU time needed to produce a sufficient number of configurations for the statistical error on $\bar{\mathcal{O}}$ to reach some pre–determined value ϵ. The number of configurations needed is

$$N_c = \frac{2\mathcal{A}+1}{\epsilon^2}\sigma^2(\mathcal{O}) \tag{6.4}$$

and hence the average CPU time required is

$$\mathcal{T}(N_c) = N_c\mathcal{T}(1) \ , \tag{6.5}$$

where $\mathcal{T}(1)$ is the average CPU time required to execute one fundamental Markov update for the algorithm.

6.3 PT Algorithm specification

Having reviewed the effects of autocorrelations on the cost of traditional simulations we now proceed to discuss the particular PT algorithm we wish to consider for the remainder of this thesis.

Our PT system will consist of N subensembles, labelled by $i = 1 \ldots N$ as before. The update scheme will proceed by carrying out an HMC update, consisting of a single trajectory of unit length in each subensemble followed by a set of pairwise swap attempts between subensembles i and $i+1$ for $i = 1 \ldots N-1$. The average swap acceptance rate between subensembles i and $i+1$ shall be denoted p_i. This combination of internal and swapping transitions will be referred to as one compound Markov step or a compound update. We then regard such a compound update as the fundamental update of our PT algorithm.

6.4 Cost of PT Simulations

6.4.1 CPU Time for fixed number of compound updates

The average CPU time used by a PT simulation, such as the one above, to carry out N_c compound steps can be written as

$$\mathcal{T}^{\mathrm{PT}}(N_c) = N_c \left(\sum_i^N \mathcal{T}_i^{\mathrm{H}}(1) + \sum_i^{N-1} \mathcal{T}_i^{\mathrm{S}} \right) . \tag{6.6}$$

Here $\mathcal{T}_i^{\mathrm{H}}(1)$ is the average time taken by one single HMC step in subensemble i, and $\mathcal{T}_i^{\mathrm{S}}$ is the average time taken to complete a swap step between subensembles i and $i+1$.

This quantity has upper bound

$$\max \mathcal{T}^{\mathrm{PT}}(N_c) = N_c \left\{ N \max_i \mathcal{T}_i^{\mathrm{H}}(1) + (N-1) \max_i \mathcal{T}_i^{\mathrm{S}}) \right\} . \tag{6.7}$$

In this case $\max_i \mathcal{T}^{\mathrm{H}}(1)$ is the average time taken by an HMC step in the subensemble with parameters corresponding to the highest computational workload. This is usually the subensemble with the lowest fermion mass (highest κ) given that all the other parameters are the same for the HMC steps in all the subensembles.

A swap attempt between two subensembles involves 2 computations of the system action in each one. In lattice QCD this means a maximum of 4 solves for the vectors X and/or Y of (2.73) in the fermionic contribution to the action. Hence $\max_i \mathcal{T}_i^{\mathrm{S}}$ is the average time taken for the swap of configuration between the subensembles corresponding to the smallest and second smallest quark mass.

Considering that a typical HMC trajectory, on a lattice with a realistic volume and interesting physical parameters usually involves $O(100)$ timesteps, and consequentially $O(100)$ solves for vectors X and Y, the computational overhead of the swap steps of up to 4 solves per swap attempt is negligible in comparison.

We can then say that the upper bound for the average CPU time needed for N_c compound steps of our PT algorithm with N subensembles is approximately

$$\max \mathcal{T}^{\mathrm{PT}}(N_c) \approx N N_c \max_i \mathcal{T}_i^{H}(1) . \tag{6.8}$$

6.4.2 CPU Time for a specified level of error

We now consider the CPU time needed by a PT simulation so that an estimate of the expectation value of some observable converges to within a given statistical error in all subensembles.

Let the observable of interest have integrated autocorrelation time $\mathcal{A}_i^{\mathrm{PT}}$ in subensemble i of a PT simulation. For the purposes of future discussion we will assume that the labels of the subensembles are chosen in a fashion such that $\mathcal{A}_1^{\mathrm{PT}} \geq \mathcal{A}_2^{\mathrm{PT}} \geq \ldots \geq \mathcal{A}_N^{\mathrm{PT}}$, so that $\mathcal{A}_1^{\mathrm{PT}} = \mathcal{A}_{\mathrm{max}}^{\mathrm{PT}}$ is the autocorrelation time of the observable in the "slowest" subensemble.

Clearly the number of compound updates needed in a PT simulation is then controlled by the subensemble with autocorrelation time $\mathcal{A}_{\mathrm{max}}^{\mathrm{PT}}$, as this is the subensemble for which the number of configurations required to reach a given level of statistical error is the largest.

Hence the CPU time needed to reach a given statistical error ϵ in all the subensembles can be written as

$$\mathcal{T}^{\mathrm{PT}}(\epsilon) = \frac{(2\mathcal{A}_{\mathrm{max}}^{\mathrm{PT}} + 1)}{\epsilon^2} \sigma^2(\mathcal{O}) \sum_i^N \mathcal{T}_i^{\mathrm{H}}(1) \tag{6.9}$$

where we have neglected the overhead from swapping steps. This quantity is bounded above by

$$\max \mathcal{T}^{\mathrm{PT}}(\epsilon) = N \frac{(2\mathcal{A}_{\mathrm{max}}^{\mathrm{PT}} + 1)}{\epsilon^2} \sigma^2(\mathcal{O}) \max_i \mathcal{T}_i^{\mathrm{H}}(1)\ . \tag{6.10}$$

This bound is attained if the computational cost of HMC simulations in all subensembles is the same. We shall assume that this is the case in all the following discussion and simply write $\mathcal{T}_i^{\mathrm{H}}(1) = \mathcal{T}^{\mathrm{H}}(1)$.

6.5 Comparing the cost of PT and HMC

We now wish to compare the costs of PT simulations to the cost of HMC simulations to identify criteria for cost effectiveness. We consider separately the two cases outlined earlier in section 6.1.1.

6.5.1 Case 1: All the PT subensembles are of interest

In this case the cost of a PT simulation needs to be compared to the cost of *an equivalent set* of HMC simulations. Hence we now consider N HMC simulations which we label with an integer $i = 1 \ldots N$. We will consider HMC simulation i to have the same parameters as PT subensemble i. Hence we would expect that an HMC update in simulation i will take the same amount of CPU time, $\mathcal{T}^{\mathrm{H}}(1)$, on average as an HMC update in the corresponding PT subensemble.

However in a PT algorithm one carries out swapping whereas in the equivalent set of HMC simulations one does not. Hence the integrated autocorrelation functions are not equal between corresponding HMC simulations and PT subensembles. We label the integrated autocorrelation functions in the reference HMC simulations as $\mathcal{A}_i^{\mathrm{H}}$.

The ratio of the cost of the PT simulation to the combined costs of the reference HMC simulations is then

$$\frac{\mathcal{T}^{\mathrm{PT}}(\epsilon)}{\sum_i \mathcal{T}_i^{\mathrm{H}}(\epsilon)} = \frac{(2\mathcal{A}_{\max}^{\mathrm{PT}}+1)N\mathcal{T}^{\mathrm{H}}(1)}{\sum_{i=1}^{N}(2\mathcal{A}_i^{\mathrm{H}}+1)\mathcal{T}^{\mathrm{H}}(1)} . \tag{6.11}$$

This ratio is bounded above and below as

$$\frac{2\mathcal{A}_{\max}^{\mathrm{PT}}+1}{2\mathcal{A}_{\max}^{\mathrm{H}}+1} \le \frac{\mathcal{T}^{\mathrm{PT}}(\epsilon)}{\sum_i \mathcal{T}_i^{\mathrm{H}}(\epsilon)} \le \frac{2\mathcal{A}_{\max}^{\mathrm{PT}}+1}{2\mathcal{A}_{\min}^{\mathrm{H}}+1} , \tag{6.12}$$

where $\mathcal{A}_{\max}^{\mathrm{H}}$ and $\mathcal{A}_{\min}^{\mathrm{H}}$ correspond to the maximum and minimum values of $\mathcal{A}_i^{\mathrm{H}}$ respectively.

To achieve a gain in cost by using PT instead of the equivalent set of HMC simulations one must have that

$$\frac{\mathcal{T}^{\mathrm{PT}}(\epsilon)}{\sum_i \mathcal{T}_i^{\mathrm{H}}(\epsilon)} < 1 \tag{6.13}$$

which, if one assumes that the ratio attains its upper bound, implies that one requires that $\mathcal{A}_{\max}^{\mathrm{PT}}$ fulfil the condition

$$\frac{2\mathcal{A}_{\max}^{\mathrm{PT}}+1}{2\mathcal{A}_{\min}^{\mathrm{H}}+1} < 1 \tag{6.14}$$

This condition can be satisfied if

$$\frac{\mathcal{A}_{\max}^{\mathrm{PT}}}{\mathcal{A}_{\min}^{\mathrm{H}}} < 1. \tag{6.15}$$

Hence we adopt condition 6.15 as our criterion for a PT simulation to be considered cost–effective when all the subensembles are of interest.

6.5.2 Case 2: Only one subensemble is of interest

We now consider an alternate viewpoint which regards only one subensemble of PT as interesting, in the sense that the other subensembles of a PT simulation serve only the purpose of assisting the decorrelation of observables in the interesting one. We will assume that the subensemble of interest is the slowest one, with autocorrelation function $\mathcal{A}^{\mathrm{PT}}_{\mathrm{max}}$. In this case one needs to compare the cost of PT to a single HMC computation carried out at the parameters of this subensemble. We assume that a single update of this HMC calculation takes $\mathcal{T}^{\mathrm{H}}(1)$ units of CPU time on average as usual and that the integrated autocorrelation time, for the observable of interest in this reference HMC simulation is $\mathcal{A}^{\mathrm{H}}$.

The ratio of the costs of the PT and HMC simulations is then

$$\frac{\mathcal{T}^{\mathrm{PT}}(\epsilon)}{\mathcal{T}^{\mathrm{H}}(\epsilon)} = \frac{(2\mathcal{A}^{\mathrm{PT}}_{\mathrm{max}}+1)N\mathcal{T}^{\mathrm{H}}(1)}{(2\mathcal{A}^{\mathrm{H}}+1)\mathcal{T}^{\mathrm{H}}(1)} = \frac{(2\mathcal{A}^{\mathrm{PT}}_{\mathrm{max}}+1)N}{2\mathcal{A}^{\mathrm{H}}+1} \; . \tag{6.16}$$

The criterion for a gain via the use of PT is then that

$$\frac{2\mathcal{A}^{\mathrm{PT}}_{\mathrm{max}}+1}{2\mathcal{A}^{\mathrm{H}}+1} < \frac{1}{N} \; . \tag{6.17}$$

However, for our cost criterion we will use the condition

$$\frac{\mathcal{A}^{\mathrm{PT}}_{\mathrm{max}}}{\mathcal{A}^{\mathrm{H}}} < \frac{1}{N} \; . \tag{6.18}$$

as we think it more intuitive. If this condition is satisfied then

$$\frac{2\mathcal{A}^{\mathrm{PT}}_{\mathrm{max}}+1}{2\mathcal{A}^{\mathrm{H}}+1} < \frac{1}{N} + \frac{1}{2\mathcal{A}^{\mathrm{H}}+1} \tag{6.19}$$

and hence the difference between conditions (6.17) and (6.18) is an additive factor of $\frac{1}{2\mathcal{A}^{\mathrm{H}}+1}$, which is negligibly small for a sufficiently large value of $\mathcal{A}^{\mathrm{H}}$.

6.6 Relating PT and HMC autocorrelation times

Having established the criteria for PT to be cost effective, we now need to consider whether PT can actually meet these conditions or not. To do this we need to relate the autocorrelation times in the PT subensembles to those of a corresponding set of independent HMC simulations.

We consider initially a variant of our PT algorithm where all our swap attempts fail. Clearly in this case the autocorrelation times in the PT subensembles will correspond to those of an equivalent set of HMC simulations. If once again reference HMC simulation i has autocorrelation time $\mathcal{A}_i^{\mathrm{H}}$ for our observable of interest, the autocorrelation times in the zero swap acceptance variant of the PT algorithm will be $\mathcal{A}_i^{\mathrm{PT}} = \mathcal{A}_i^{\mathrm{H}}$.

Looking at (6.11) we can see that in this case it can actually cost more to perform a PT simulation than an equivalent set of HMC ones. This is because the number of updates in a PT simulation is governed by the subensemble with $\mathcal{A}_{\mathrm{max}}^{\mathrm{PT}}$. Hence the PT scheme can only be terminated when the estimate of the expectation value of the observable of interest has converged to the required accuracy in this subensemble, implying that faster subensembles must undergo more updates than is truly necessary. On the other hand each HMC simulation can individually be terminated as soon as the estimate of the expectation value in question reaches required level of statistical error.

To ease discussion from here on we will make a notational convention wherein we refer to the HMC autocorrelation time of a given subensemble. By such a phrase we will refer to the autocorrelation time of the observable of interest in the HMC simulation corresponding to the subensemble, or alternatively, the autocorrelation time of the observable in the subensemble for the case when none of the swaps succeed.

Let us now assume that some of the swap steps are accepted between the PT subensembles. In this case we would expect that the autocorrelation time in PT subensemble i would be some function of the HMC autocorrelation times of the reference simulations and the probabilities of swaps between subensembles being accepted,

$$\mathcal{A}_i^{\mathrm{PT}} = \mathcal{A}_i^{\mathrm{PT}}(\mathcal{A}_1^{\mathrm{H}}, \ldots, \mathcal{A}_N^{\mathrm{H}}; p_1, \ldots, p_{N-1}) \ , \tag{6.20}$$

where p_i is the swap acceptance probability between subensembles i and $i+1$. We expect such behaviour from two effects of the swapping:

Firstly we have the mechanical mixing of configurations between the subensembles. This is expected to improve autocorrelations times in each subensemble compared to their HMC ones.

Secondly we can have a situation where one subensemble has a shorter HMC autocorrelation time than another. In this case, by mixing the configurations between the subensembles, we would expect that the resulting autocorrelation times of the subensembles will be weighted averages of their original HMC autocorrelation times. The weighting would presumably depend on the swap acceptance rate. In particular we would expect that the faster of the two subensembles will have its PT autocorrelation time increased, and the slow one to have its PT autocorrelation time decreased over its original HMC one.

We note that the first of the two effects is more advantageous when one wishes to use all the PT subensembles for later analysis as the mixing is expected to improve the autocorrelation times of all the subensembles. The latter effect on the other hand is more useful when one wishes to consider only a single subensemble, the one whose autocorrelation time is decreased. In practice we expect these two effects to be combined in some complicated way.

6.7 Autocorrelations in Twin Subensemble PT system

We now proceed to discuss improvements in autocorrelation time of a generic observable $\mathcal{O}$ due solely to mechanical mixing. We shall consider a PT system consisting of two subensembles only. We will assume that both the subensembles have the same HMC autocorrelation time $\mathcal{A}^{\mathrm{H}}$. We note that in this case we would also expect from symmetry that the final autocorrelation times $\mathcal{A}_i^{\mathrm{PT}}$ will be equal in both subensembles also and we shall drop the subscript i.

The assumption of equal HMC autocorrelation times is rather restrictive, but pertinent to our numerical calculations as we have only carried out PT simulations where this condition was fulfilled. This was not by design, but rather by misfortune as were unable to reach sufficient distances in paramter space for the HMC autocorrelation times to change a great deal and what is more our estimates of the HMC autocorrelation times carry large error. The details of our numerical calculations will be clarified in chapter 8.

We have several other reasons for discussing the equal autocorrelation time scenario. Firstly we believe it to be a worst case scenario, when one can rely only on mechanical mixing of configurations between subensembles. Secondly the calculations is model independent, in the sense that it does not depend on the particular physical system being simulated, nor does it make any assumptions about the observables in question. However, the combinatorics involved in the calculation become very difficult in the case where more than one subensemble is used and/or where the HMC autocorrelation times of the subensembles are different from each other.

We have managed to construct a more flexible, but less general matrix model which is only valid under more stringent assumptions. We leave the discussion of this model to chapter 7

6.7.1 Calculation of the autocorrelation function

We now proceed to calculate the normalised autocorrelation function for the twin subensemble system just described.

In the compact notation of (5.43) the definition of the autocorrelation function for the HMC simulation is

$$C^{\mathrm{H}}(t) = \frac{1}{\sigma^2(\mathcal{O})} \left\langle \tilde{\mathcal{O}}(t+t_0)\tilde{\mathcal{O}}(t_0) \right\rangle \ . \tag{6.21}$$

For an ensemble of N_c configurations this becomes

$$C^{\mathrm{H}}(t) = \frac{1}{\sigma^2(\mathcal{O})(N_c - t)} \sum_{t_0=1}^{N_c-t} \widetilde{\mathcal{O}}(t+t_0)\widetilde{\mathcal{O}}(t_0) \ . \tag{6.22}$$

When calculating $C^{\mathrm{PT}}(t)$ we must bear in mind that in the t compound steps that occurred between generating configurations $U(t+t_0)$ and $U(t_0)$, upon which observables $\mathcal{O}(t+t_0)$ and $\mathcal{O}(t_0)$ were measured, there have been t swap attempts between the subensembles which may or may not have been accepted. Since the number of successful swaps must have been either even or odd, we split $C^{\mathrm{PT}}(t)$ into two parts

$$C^{\mathrm{PT}}(t) = \frac{1}{\sigma^2(\mathcal{O})} \left(\mathcal{S}_e + \mathcal{S}_o\right) \tag{6.23}$$

where

$$\mathcal{S}_e = \frac{1}{N_e} \sum_{\text{even}} \widetilde{\mathcal{O}}(t+t_0)\widetilde{\mathcal{O}}(t_0) \tag{6.24}$$

$$\mathcal{S}_o = \frac{1}{N_o} \sum_{\text{odd}} \widetilde{\mathcal{O}}(t+t_0)\widetilde{\mathcal{O}}(t_0) \ . \tag{6.25}$$

By the label "even" in (6.24) we mean that the only terms contributing to $\mathcal{S}_e$ are those where an even number of swap attempts has succeeded out of the t that were attempted. Likewise by "odd" in (6.25) we mean that the only terms which contribute to the sum are those where an odd number of swaps succeeded out of the t tries. The N_e and N_o are just suitable normalisation constants.

After an odd number of successful swaps, a configuration must necessarily be in a subensemble that is different from the one it was in at simulation time t_0. Since the HMC steps in different subensembles are independent, we make the assumption that cross correlations between subensembles are negligible. The configurations in the terms contributing to S_o are separated by an odd number of successful swaps, and hence originate from different subensembles. Thus, by virtue of the above assumption $\mathcal{S}_o$ sums to zero.

We can write the remaining term $\mathcal{S}_e$ as

$$\mathcal{S}_e = P^e \sum_{t_0=0}^{N_c-t} \widetilde{\mathcal{O}}(t+t_0)\widetilde{\mathcal{O}}(t) \tag{6.26}$$

where P^e is the probability of an even number of swap attempts being accepted from t trials given by

$$P_e = \sum_{i}^{t} C_i^t (1-p)^{t-i} p^i, \quad i \text{ is even}, \tag{6.27}$$

where we remind the reader that p is the swap acceptance rate, which is the average probability that any single swap attempt succeeds. The normalisation constant N_e has been absorbed into the definition of P^e.

In the equation above, C_i^t is the Binomial Coefficient,

$$C_i^t = \frac{t!}{n!(t-n)!} \tag{6.28}$$

which is equal to the number of ways of choosing i successful swaps out of t tries.

The PT autocorrelation can then be written as

$$C^{\mathrm{PT}}(t) = \frac{\mathcal{S}_e}{\sigma^2(\mathcal{O})} = \frac{P^e}{\sigma^2(\mathcal{O})} \sum_{t_0=0}^{N_c-t} \tilde{\mathcal{O}}(t+t_0)\tilde{\mathcal{O}}(t) = P^e C^{\mathrm{H}}(t)\ . \tag{6.29}$$

To carry out the sum in (6.27) we appeal to the Binomial theorem, which states that for two quantities a and b,

$$(a+b)^t = \sum_{i=0}^{t} C_i^t a^{t-i} b^i. \tag{6.30}$$

However in (6.30) the index i runs over both even and odd valued integers, whereas we wish to carry out the sum only over the terms where i is even. To do this we use a consequence of (6.30):

$$\begin{aligned} \frac{(a+b)^t + (a-b)^t}{2} &= \frac{1}{2}\sum_{i=0}^{t} C_i^t a^{t-i} b^i + \sum_{i=0}^{t} C_i^t a^{t-i}(-b)^i \\ &= \frac{1}{2}\sum_{i=0}^{t} C_i^t a^{t-i}(b^i + (-b)^i). \end{aligned} \tag{6.31}$$

As $(-b)^i = -(b)^i$ when i is odd, it can be seen that terms corresponding to odd i do not contribute to (6.31). Hence we can use (6.31) to carry out the sum in (6.27) and evaluate P^e. We set $q = 1-p$ and obtain

$$P_e = \sum_{i=0}^{t} C_i^t (1-p)^{t-i} p^i \qquad i \text{ is even}$$

$$
\begin{aligned}
&= \sum_{i=0}^{t} C_i^t q^{t-i} p^i \\
&= \frac{1}{2}\left\{(q+p)^t + (q-p)^t\right\} \\
&= \frac{1}{2}\left\{1 + (1-2p)^t\right\} \qquad (6.32)
\end{aligned}
$$

Hence we can write

$$C^{\mathrm{PT}}(t) = \frac{1}{2}\left\{1 + (1-2p)^t\right\} C^{\mathrm{H}}(t)\ . \qquad (6.33)$$

We consider three cases for the behaviour of C^{PT}:

i) $p = 0$: In this case $C^{\mathrm{PT}}(t) = C^{\mathrm{H}}(t)$. This is clearly what one would expect.

ii) $0 < p \leq \frac{1}{2}$: In this situation we have $C^{\mathrm{PT}}(t) \in [\ \frac{1}{2}C^{\mathrm{H}}(t), C^{\mathrm{H}}(t)\ )$. We see a reduction in the PT autocorrelation time over that of the HMC autocorrelation time of at most a factor of 2.

iii) $\frac{1}{2} < p \leq 1$: In this case the term $(1-2p)^t$ becomes oscillatory depending on whether t is even or odd. In particular when $p = 1$ the term in question oscillates between -1 and $+1$. This is an indication of the fact, that when all the swaps are accepted it is not possible to choose an even number of successful swaps in an odd number of trials.

6.7.2 Calculation for PT integrated autocorrelation time

At the expense of making an additional assumption about $C^{\mathrm{H}}(t)$ one can attempt to calculate the corresponding integrated autocorrelation time $\mathcal{A}^{\mathrm{PT}}$. This additional assumption is to model $C^{\mathrm{H}}(t)$ by an exponential decay function with decay constant k

$$C^{\mathrm{H}}(t) = \exp\{-kt\} \qquad , k > 0\ . \qquad (6.34)$$

This assumption can be motivated from considering the exponential autocorrelation time $\mathcal{A}_{\mathrm{exp}}$ which is defined in chapter 1, and is defined in a relation similar to the one above.

As an auxiliary result we first calculate $\mathcal{A}^{\mathrm{H}}$. We start with the definition

$$\mathcal{A}^{\mathrm{H}} = \sum_{t=1}^{\infty} C^{\mathrm{H}}(t) = \sum_{t=0}^{\infty} C^{\mathrm{H}}(t) - 1. \tag{6.35}$$

Hence

$$\mathcal{A}^{\mathrm{H}} + 1 = \sum_{t=0}^{\infty} (\exp\{-k\})^t . \tag{6.36}$$

The sum on the right hand side is just a geometric series with common ratio $r = \exp\{-k\}$. Since k is positive, we have that $r < 1$. One can then carry out the sum in (6.36) to obtain,

$$\mathcal{A}^{\mathrm{H}} + 1 = \frac{1}{1 - \exp\{-k\}} , \tag{6.37}$$

whence

$$\mathcal{A}^{\mathrm{H}} = \frac{\exp\{-k\}}{1 - \exp\{-k\}} . \tag{6.38}$$

A useful feature of the above relation is that it can also be inverted to give

$$\exp\{-k\} = \frac{\mathcal{A}^{\mathrm{H}}}{\mathcal{A}^{\mathrm{H}} + 1} . \tag{6.39}$$

The computation of the integrated autocorrelation time for either PT subensemble then proceeds as follows:

$$\begin{aligned}
\mathcal{A}^{\mathrm{PT}} &= \sum_{t=1}^{\infty} C^{\mathrm{PT}}(\tau) \\
&= \frac{1}{2} \sum_{t=1}^{\infty} \left\{ 1 + (1-2p)^t \right\} C^{\mathrm{H}}(t) \\
&= \frac{1}{2} \sum_{t=1}^{\infty} C^{\mathrm{H}}(t) + \frac{1}{2} \sum_{t=1}^{\infty} [(1-2p)\exp\{-k\}]^t
\end{aligned} \tag{6.40}$$

We can see that the first sum in (6.40) is just the definition of the HMC integrated autocorrelation time. The second term is again a geometric series with common ratio $r' = (1-2p)\exp\{-k\}$. Since $|r'| < 1$, this geometric series can also be summed to obtain,

$$\sum_{t=1}^{\infty} [(1-2p)\exp\{-k\}]^t = \frac{1}{1-(1-2p)\exp\{-k\}} - 1 = \frac{1+2p\mathcal{A}^{\mathrm{H}}}{\mathcal{A}^{\mathrm{H}}(1+2p)} \tag{6.41}$$

where in the last step we have used (6.39). Substituting the above back into (6.40) and simplifying, one finally obtains,

$$\mathcal{A}^{\mathrm{PT}} = \frac{\mathcal{A}^{\mathrm{H}} \left[1 + p(\mathcal{A}^{\mathrm{H}} - 1) \right]}{1 + 2p\mathcal{A}^{\mathrm{H}}}. \tag{6.42}$$

One can then readily evaluate the ratio of PT and HMC integrated autocorrelation times as

$$\frac{\mathcal{A}^{\rm PT}}{\mathcal{A}^{\rm H}} = \frac{1 + p(\mathcal{A}^{\rm H} - 1)}{1 + 2p\mathcal{A}^{\rm H}} \; . \tag{6.43}$$

We remark on the following features of (6.43):

i) When $p = 0$, one obtains $\frac{\mathcal{A}^{\rm PT}}{\mathcal{A}^{\rm H}} = 1$. This is what one expects as it is the zero swap case discussed earlier.

ii) For a fixed $p \in (0, \frac{1}{2})$ increasing $\mathcal{A}^{\rm H}$ from 0 has the effect that the ratio in (6.43) approaches the value of $\frac{1}{2}$ rapidly from above. If one considers all the subensembles as interesting, then the simulation cost is effective for sufficiently large $\mathcal{A}^{\rm H}$. One gains less than a factor of 2 in cost compared to the zero swap case case, however, implying that strictly speaking the simulation is not cost effective if one is interested in only one subensemble. Nonetheless, the break even ratio of $\frac{\mathcal{A}^{\rm PT}}{\mathcal{A}^{\rm H}} = \frac{1}{2}$ is approached rapidly with increasing $\mathcal{A}^{\rm H}$. Hence for observables with sufficiently long autocorrelation times the difference between the costs of the PT and HMC simulations are negligible.

iii) If $p = \frac{1}{2}$ the ratio is exactly $\frac{1}{2}$ for all values of $\mathcal{A}^{\rm H}$. In this case one always gains by performing PT if all the subensembles are of interest and always breaks even for all values of $\mathcal{A}^{\rm H}$ when only one subensemble is of interest.

iv) For $p \in (\frac{1}{2}, 1)$ the ratio approaches $\frac{1}{2}$ rapidly from below with increasing $\mathcal{A}^{\rm H}$. In this case one gains over performing independent HMC simulations irrespective of whether one is interested in just one or all of the PT subensembles. However when one is interested in just one subensemble, the cost ratio approaches the break–even cost rapidly with increasing $\mathcal{A}^{\rm H}$, implying that the greatest improvements for PT should be manifest in observables which have short HMC autocorrelation times in the first place.

6.8 Summary of Chapter

We now summarise the main results and conclusions of this chapter.

We began by discussing the cost of a simulation in terms of the integrated autocorrelation time of some interesting observable. We then fixed the PT algorithm under discussion by specifying the combination of Markov steps used for its construction.

We followed this by a discussion of the cost of a PT simulation. We considered the cost for a fixed number of compound steps and also the cost of reaching a given level of statistical error for the observable of interest in all the subensembles. We found that the latter case was controlled by the subensemble with the largest autocorrelation time, $\mathcal{A}^{\mathrm{PT}}_{\mathrm{max}}$ for the observable in question.

We then discussed two different ways for looking at cost effectiveness for PT simulations. We first considered the case where all the PT subensembles were intrinsically of interest. To do this we had to compare the cost of the PT simulation with a corresponding set of reference HMC simulations. In this case we found that for PT to be cost–effective we required that the inequality $\frac{\mathcal{A}^{\mathrm{PT}}}{\mathcal{A}^{\mathrm{H}}_{\mathrm{min}}} < 1$ should hold, where $\mathcal{A}^{\mathrm{H}}_{\mathrm{min}}$ is the integrated autocorrelation time for the slowest reference HMC system.

We then considered the case where only one subensemble was considered interesting. In this case we found that for PT to be cost–effective the autocorrelation time $\mathcal{A}^{\mathrm{PT}}$ of the interesting observable had to be factor of N smaller than $\mathcal{A}^{\mathrm{H}}$, the autocorrelation time of the same observable in the corresponding HMC simulation.

Finally we proceeded to investigate whether or not PT can meet these criteria. To this end, we first briefly discussed how we expect the swapping step of PT to improve autocorrelation times compared to an equivalent set of HMC simulations. We carried out a calculation for the autocorrelation time of an observable in the simple case when the PT system was comprised of two subensembles having equal HMC autocorrelation times. In this case we found that the ratio of autocorrelation times $\frac{\mathcal{A}^{\mathrm{PT}}}{\mathcal{A}^{\mathrm{H}}}$ tended quickly to the value of $\frac{1}{2}$ for increasing $\mathcal{A}^{\mathrm{H}}$ at a fixed value of the swap acceptance rate $p > 0$. For $p < \frac{1}{2}$ this asymptotic value was approached from above whereas in the case $p > \frac{1}{2}$ it was approached from below.

From this latter analysis we conclude that if all the subensembles are of interest in a PT simulation, one can almost always gain over a corresponding set of HMC simulations just through the mechanical improvement in PT autocorrelation times due to mixing. In the case when only one subensemble is important one is usually close to the break even point as $\frac{N\mathcal{A}^{\mathrm{PT}}}{\mathcal{A}^{\mathrm{H}}} \approx 1$ for sufficiently large $\mathcal{A}^{\mathrm{H}}$.

We also mentioned briefly that we do not consider the viewpoint in which all the subensembles of a PT system are of interest to be a realistic one for systems of sufficiently large volume. We know from action matching considerations that to achieve reasonable acceptance rates on the swap attempts the parameters of the subensembles have to be very close to each other in the parameter space. In this case, if the autocorrelation times are smooth functions of the parameters, we expect the HMC autocorrelation times of our PT subensembles to be approximately the same.

Nonetheless, we shall continue our discussions of cost effectiveness of PT in chapter 7. There we shall present a more flexible matrix model, and demonstrate a situation when great gains can be made with PT. Although the model is only valid under very stringent assumptions, (fulfilled by harmonic oscillator and free field systems), we shall make the leap of faith that its gross features may apply to other systems. We shall also exhibit a system where the autocorrelation time is not a smooth function of the simulation parameters but changes rather abruptly, and for which the conditions for gain by parallel tempering can be met.

Chapter 7

Matrix Model of PT Autocorrelations

7.1 Introductory Remarks

In chapter 6 we presented a calculation for the autocorrelation time of a twin ensemble parallel tempering system, where the HMC autocorrelation times of the two subensembles in question were assumed to be equal. We have shown there, that the maximum attainable gain in terms of the ratio of the PT to HMC autocorrelation times was a factor of two, at the computational cost of performing twice the work of a single HMC simulation. Hence, if both subensembles were of interest one could gain up to a factor of two, whereas if only one subensemble was of interest, it appeared that the computational overhead of simulating the second subensemble overshadowed any gain that could be made in terms of the improvements in autocorrelation time.

This earlier calculation was maximally conservative, and dealt only with improvements in autocorrelation times that came from the so called mechanical mixing of configurations between the two subensembles. It should be looked upon as a worst case algorithmic analysis. Another limitation of the previous calculation is that it is difficult to transpose to the case when there are more than two subensembles in the PT system under consideration. However, the calculation does have the advantage that it is observable independent.

We now present a different model for autocorrelations in PT systems which has the advantage that it can be applied in principle to PT systems consisting of an arbitrary number of subensembles, each of which in principle can have different HMC integrated autocorrelation times. Furthermore, it predicts some cross correlations between the tempered subensembles, which one should take into account when combining the results from them, say for the purpose of some extrapolation.

We shall also show, that in the case when the untempered autocorrelation times in a two subensembles of a twin subensemble system are the same, the *matrix model* which we shall shortly present gives identical results to the calculation of chapter 6. This is a useful consistency check on the matrix model.

We now proceed to outline how the matrix model is formulated, and discuss the assumptions in its construction. Thereafter we shall apply it to the PT autocorrelation times of the magnetisation for the cases of twin and triple subensemble harmonic oscillator simulations, for which the model is exact. The reasons for this choice of system will be made clear during the formulation of the model and the discussion of its underlying assumptions.

7.2 Formulation of the Model

Let us suppose that our PT simulation produces infinitely long time sequences, $\{\mathcal{O}_i(t)\}$, for some observable $\mathcal{O}$, in N_s subensembles, where the subscript i labels the subensembles and t labels the simulation time. In this case autocorrelations in the system can be expressed via the autocorrelation matrix[1]

$$C_{ij}(t) = \frac{\left\langle \tilde{\mathcal{O}}_i(|t|+t_0)\tilde{\mathcal{O}}_j(t_0)\right\rangle}{\left\langle \tilde{\mathcal{O}}_i(t_0)\tilde{\mathcal{O}}_j(t_0)\right\rangle}, \tag{7.1}$$

giving the correlation of the observable in subensemble i with itself in subensemble j with for a shift of t time units between the two time series. The expectation

[1] Hence the name *matrix model*

values are over the t_0 time indices, and we have used the notation defined in (5.43) where $\widetilde{\mathcal{O}} = \mathcal{O} - \langle \mathcal{O} \rangle$.

If $\widetilde{\mathcal{O}}_i(|t| + t_0)$ can be written in terms of some *linear update operator* $\hat{U}_{ij}(|t|)$, and the $\widetilde{\mathcal{O}}_j(t_0)$ as

$$\widetilde{\mathcal{O}}_i(|t| + t_0) = \sum_j \hat{U}_{ij}(|t|)\widetilde{\mathcal{O}}_j(t_0), \tag{7.2}$$

then the expression for the correlation matrix can be written as

$$C_{ij}(t) = \frac{1}{N_{ij}} \sum_k \hat{U}_{ik}(|t|) \left\langle \widetilde{\mathcal{O}}_k(t_0)\widetilde{\mathcal{O}}_j(t_0) \right\rangle \tag{7.3}$$

where the $\hat{U}_{ik}(|t|)$ is independent of the observables and can hence be pulled outside expectation values, and N_{ij} is defined as

$$N_{ij} = \left\langle \widetilde{\mathcal{O}}_i(t_0)\widetilde{\mathcal{O}}_j(t_0) \right\rangle \ . \tag{7.4}$$

Assuming that the simulations are independent at the outset we can write

$$N_{ij} = N_{ii}\delta_{ij}, \tag{7.5}$$

which when substituted into (7.3) gives

$$\begin{aligned} C_{ij}(t) &= \frac{1}{N_{ij}} \sum_k \hat{U}_{ik}(|t|) N_{jk}\delta_{kj}\delta_{ij} \\ &= \hat{U}_{ij}(|t|)\frac{N_{jj}}{N_{jj}}\delta_{ij} \\ &= \hat{U}_{ij}(|t|)C_{ij}(0) \ , \end{aligned} \tag{7.6}$$

where $C_{ij}(0) = \delta_{ij}$ once again simply expresses that initially the simulations are considered independent.

Here we have met the first major assumption of our matrix model, that our observables depend linearly on some time evolution operator. The calculation of chapter 6 did not make this assumption. This linearity demand is quite restrictive. Consider for example, the case of lattice QCD when the observable in question is just a single link variable $U_\mu(x)$ on the lattice, disregarding for the moment the fact that it not gauge invariant. The time evolution of the field in the HMC step is in general some non linear transformation and hence the model is not applicable to lattice QCD. However, as described in chapter 4, in the case of simulating the harmonic oscillator, the MD update of the gauge and momentum fields is a linear transformation of the fields as the time evolution operator is independent of the fields in this case. Hence we can attempt to apply the model to the case of the tempered harmonic oscillator simulations, in the hope that it has some heuristic bearing for lattice QCD and reveals some qualitative features of the PT algorithm.

We now make our second major assumption, to which we shall refer to as the factorisation property. This is the assumption that the HMC autocorrelation times factorise as

$$C_i^H(t) = [C_i^H(1)]^t \ . \tag{7.7}$$

This assumption is needed so that, when there are successful swaps in a PT simulation we shall be able to piece together the autocorrelation function of an ensemble as a product of autocorrelations from several subensembles. If we choose to model our autocorrelation times as simple exponential decays, this assumption is certainly fulfilled. Indeed though it is not immediately evident, deep down this assumption is also made in the calculation in chapter 6.

Since the combination of HMC and swapping steps in a PT simulation is more or less arbitrary, the time evolution operator $\hat{U}(t)$ does not necessarily factorise. However when one repeatedly executes compound Markov steps as in our simulations the operator does factorise as

$$\hat{U}(t) = \left(\hat{U}(1)\right)^t \ , \tag{7.8}$$

where we have dropped the subensemble indices and have written the equation as one involving *matrices* ($\hat{U}(t)$ and $\hat{U}(1)$). The rest of our matrix model consists of constructing a suitable candidate for the matrix $\hat{U}(1)$

7.2.1 Modelling the HMC Updates

We have already assumed that each subensemble has its own HMC autocorrelation function $C_i^{\mathrm{H}}(t)$ which factorises in the way described earlier. In this case we can model the HMC evolution of the subensembles with the diagonal matrix

$$\hat{U}^{\mathrm{H}}(t) = \mathrm{diag}\left(C_1^{\mathrm{H}}(t), C_2^{\mathrm{H}}(t), \ldots, C_{N_s}^{\mathrm{H}}(t)\right) \ . \tag{7.9}$$

Using the factorisation property we can write

$$\hat{U}^{\mathrm{H}}(t) = \left(\hat{U}^{\mathrm{H}}(1)\right)^t = \mathrm{diag}\left(C_1^{\mathrm{H}}(1), C_2^{\mathrm{H}}(1), \ldots, C_{N_s}^{\mathrm{H}}(1)\right)^t \tag{7.10}$$

7.2.2 Modelling the swapping

We now wish to model the swapping part of the PT process. We denote the swap acceptance rate between subensemble i and $i+1$ by p_i as before. The swapping will mix elements of the correlation matrix in proportion to the swap acceptance rate. We model the effect of a swapping by an update operator $\hat{U}^{\mathrm{S}}$ which with probability p_i swaps rows i and $i+1$ of the correlation matrix and with probability $1-p_i$ has no effect on the correlation matrix. Hence it performs the rôle of mechanical mixing of correlations between subensembles.

This effect of the swap update matrix derives originally from considering the evolution of the fields in the simulation as we shall now demonstrate. Consider two harmonic oscillators with coordinates ϕ_1 and ϕ_2. These can then be written as the column vector $\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix}$. The effect of the swap step in PT would be to swap the elements of the vector with probability $p_1 = p$,

$$\begin{aligned}
\hat{U}^{\mathrm{S}}\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} &= (1-p)\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} + p\begin{pmatrix} \phi_2 \\ \phi_1 \end{pmatrix} && (7.11) \\
&= \left[(1-p)\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + p\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\right]\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} && (7.12) \\
&= \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} && (7.13)
\end{aligned}$$

The correlation matrix can then be found as the expectation value of terms such as

$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix}(\phi_1, \phi_2) = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}\begin{pmatrix} \phi_1\phi_1 & \phi_1\phi_2 \\ \phi_2\phi_1 & \phi_2\phi_2 \end{pmatrix}. \tag{7.14}$$

In particular for the case of the magnetisation in a harmonic oscillator, these will be the *only* terms, as the disconnected pieces of the autocorrelation matrix will vanish as $\langle \phi_i \rangle = 0$. Hence we believe that modelling the effects of swapping on the autocorrelation functions is sensible.

In general, the effects of swapping, on average, between subensembles i and $i+1$ can formally be written as

$$\hat{U}_i^{\mathrm{S}}(p_i) = (1 - p_i) I_{N_s} + p_i P_{N_s}^{i,i+1} \tag{7.15}$$

where I_{N_s} is the N_s dimensional unit matrix, and $P_{N_s}^{i,i+1}$ is the (row permutation) matrix which will swap rows i and $i+1$ of an N_s dimensional square matrix.

7.2.3 Putting it together

Combining the swap update steps and the HMC update steps we can now write down the main equation of the matrix model,

$$C_{ij}(t) = \left\{ \left(\prod_{i=1}^{N_s - 1} \hat{U}_i^{\mathrm{S}}(p_i) \right) \hat{U}^{\mathrm{H}}(1) \right\}^t C_{ij}(0) \tag{7.16}$$

with the assumption that

$$C_{ij}(0) = I_{N_s} \ . \tag{7.17}$$

One could construct more complicated expressions for $C_{ij}(t)$ by allowing swapping between a generic pair of subensembles i and j rather than just between i and $i+1$, however we are sticking here to our definition of the PT algorithm as given in chapter 6.

As a final note before applying the model we reiterate that as it effectively models the autocorrelation of the fields ϕ_i for the harmonic oscillator, we expect the model to hold accurately only in this case. However we expect, that just as the harmonic oscillator was found to be relevant to lattice QCD in chapter 4, the gross features of this model will have some bearing also to PT in more complicated systems than just the harmonic oscillator.

7.3 Twin Ensemble Tempered Oscillator

We shall now apply the matrix model to the case of a twin subensemble, tempered harmonic oscillator simulation. In this case there is only one acceptance probability $p_1 = p$. We proceed by constructing the matrices needed by the model. Thereafter our first application will be to derive the results of chapter 6, in the case when the magnetisation has equal untempered autocorrelation times in both subensembles. Following this we shall apply the model to the case when the untempered autocorrelation times are different in the two ensembles, and identify the criteria when gains can be made by tempering the oscillator system.

From here on, to ease notation we shall drop the superscript H on the HMC autocorrelation functions $C_i^{\mathrm{H}}(t)$ and the HMC integrated autocorrelation times $\mathcal{A}_i^{\mathrm{H}}$. The number of indices on these quantities shall be sufficient to remove any ambiguity. Hence, from now on C_i ($\mathcal{A}_i$) shall refer to the HMC autocorrelation functions (integrated autocorrelation times), while the PT variants shall always have two indices to denote their position in the relevant autocorrelation matrix.

7.3.1 The Autocorrelation Matrix

To start with we assume that the subensembles are initially uncorrelated so that

$$C_{ij}(0) = I_2 \ . \tag{7.18}$$

The individual HMC autocorrelation functions of the subensembles are $C_1^{\mathrm{H}}(t)$ and $C_2^{\mathrm{H}}(t)$ respectively. The HMC update matrix of the autocorrelations is then

$$\hat{U}^{\mathrm{H}} = \begin{pmatrix} C_1(1) & 0 \\ 0 & C_2(1) \end{pmatrix} \tag{7.19}$$

and the swap update matrix of the autocorrelations is

$$\hat{U}_1^{\mathrm{S}}(p) = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix} \ . \tag{7.20}$$

The compound update matrix is therefore

$$\hat{U}_{ij}(1) = \begin{pmatrix} (1-p)C_1(1) & pC_2(1) \\ pC_1(1) & (1-p)C_2(1) \end{pmatrix} \ . \tag{7.21}$$

Hence

$$\begin{pmatrix} C_{11}(t) & C_{12}(t) \\ C_{21}(t) & C_{22}(t) \end{pmatrix} = \begin{pmatrix} (1-p)C_1(1) & pC_2(1) \\ pC_1(1) & (1-p)C_2(1) \end{pmatrix}^t . \tag{7.22}$$

7.3.2 Equal HMC Autocorrelation Functions

If the subensembles have the same HMC autocorrelation function, $C(t) = C_1(t) = C_2(t)$, as was assumed in the calculation of chapter 6, the HMC update term becomes

$$\hat{U}^{\mathrm{H}}(1) = C(1)I_2 \tag{7.23}$$

and the correlation matrix becomes

$$C_{ij}(t) = C(t) \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}^t \tag{7.24}$$

The swap update matrix can be diagonalised as

$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1-2p \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{7.25}$$

and it can be raised to the power t, giving

$$\begin{pmatrix} C_{11}(t) & C_{12}(t) \\ C_{21}(t) & C_{22}(t) \end{pmatrix} = \frac{C(t)}{2} \begin{pmatrix} 1+(1-2p)^t & 1-(1-2p)^t \\ 1-(1-2p)^t & 1+(1-2p)^t \end{pmatrix} . \tag{7.26}$$

The diagonal elements $C_{11}(t) = C_{22}(t)$ correspond to the autocorrelation functions within the subensembles, and $C_{12}(t) = C_{21}(t)$ correspond to the cross correlations induced between the subensembles. It is clear that

$$C_{11}(t) = C_{22}(t) = \frac{1}{2}\left\{1 + (1-2p)^t\right\} C(t) \tag{7.27}$$

just as in (6.33), however the combinatorics of the calculation in chapter 6 were taken care of in an elegant manner by raising the swap update matrix to the power t.

7.3.3 Unequal HMC Autocorrelation Times

To progress any further we have to start making assumptions about the HMC autocorrelation functions $C_1(t)$ and $C_2(t)$. In section 6.7.1 we chose to model the HMC autocorrelation functions with simple exponentials. We shall do the same here and use

$$C_i(t) = \exp\{-k_i t\} \qquad i = 1, 2\ . \tag{7.28}$$

We have also shown that we can write

$$\exp\{-k_i\} = C_i(1) = \frac{\mathcal{A}_i}{\mathcal{A}_i + 1}\ . \tag{7.29}$$

In this case, the autocorrelation matrix becomes

$$\begin{aligned} C_{ij}(t) &= \begin{pmatrix} C_{11}(t) & C_{12}(t) \\ C_{21}(t) & C_{22}(t) \end{pmatrix} = \left\{ \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix} \begin{pmatrix} \frac{\mathcal{A}_1}{\mathcal{A}_1+1} & 0 \\ 0 & \frac{\mathcal{A}_2}{\mathcal{A}_2+1} \end{pmatrix} \right\}^t \\ &= \begin{pmatrix} (1-p)\frac{\mathcal{A}_1}{\mathcal{A}_1+1} & p\frac{\mathcal{A}_2}{\mathcal{A}_2+1} \\ p\frac{\mathcal{A}_1}{\mathcal{A}_1+1} & (1-p)\frac{\mathcal{A}_2}{\mathcal{A}_2+1} \end{pmatrix}^t . \end{aligned} \tag{7.30}$$

The determinant of $\hat{U}_{ij}(1)$ is

$$\det \hat{U}_{ij}(1) = \begin{pmatrix} (1-p)\frac{\mathcal{A}_1}{\mathcal{A}_1+1} & p\frac{\mathcal{A}_2}{\mathcal{A}_2+1} \\ p\frac{\mathcal{A}_1}{\mathcal{A}_1+1} & (1-p)\frac{\mathcal{A}_2}{\mathcal{A}_2+1} \end{pmatrix} = \frac{\mathcal{A}_1\mathcal{A}_2(1-2p)}{(\mathcal{A}_1+1)(\mathcal{A}_2+1)}\ , \tag{7.31}$$

hence

$$|\det \hat{U}_{ij}(1)| < 1 \qquad \text{for} \qquad \mathcal{A}_1, \mathcal{A}_2 > 1, \quad 0 \le p \le 1, \tag{7.32}$$

implying that the matrix geometric sequence

$$\mathcal{A}_{ij} + I_2 = \sum_{t=0}^{\infty} \hat{U}_{ij}(1)^t \tag{7.33}$$

is expected to converge to and to have the value

$$\mathcal{A}_{ij} + I_2 = \left(I_2 - \hat{U}_{ij}(1)\right)^{-1}, \tag{7.34}$$

when the required inverse exists. In fact the inverse exists and can be found explicitly. Carrying out the inversion gives

$$\mathcal{A}_{ij} = \begin{pmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} \\ \mathcal{A}_{21} & \mathcal{A}_{22} \end{pmatrix} \tag{7.35}$$

with diagonal elements

$$\mathcal{A}_{11} = \frac{\mathcal{A}_1\,[(\mathcal{A}_2-1)\,p+1]}{1+p(\mathcal{A}_1+\mathcal{A}_2)} \tag{7.36}$$

$$\mathcal{A}_{22} = \frac{\mathcal{A}_2\,[(\mathcal{A}_1-1)\,p+1]}{1+p(\mathcal{A}_1+\mathcal{A}_2^{\mathrm{H}})} \tag{7.37}$$

and off diagonal elements

$$\mathcal{A}_{12} = \frac{p\mathcal{A}_2(\mathcal{A}_1+1)}{1+p(\mathcal{A}_1+\mathcal{A}_2)} \tag{7.38}$$

$$\mathcal{A}_{21} = \frac{p\mathcal{A}_1(\mathcal{A}_2+1)}{1+p(\mathcal{A}_1+\mathcal{A}_2)}\,. \tag{7.39}$$

The diagonal elements are the integrated autocorrelation times of the PT subensembles in the sense that $\mathcal{A}_{11} = \mathcal{A}_1^{\mathrm{PT}}$ and $\mathcal{A}_{22} = \mathcal{A}_2^{\mathrm{PT}}$. The off diagonal elements could perhaps be called integrated cross–correlation times.

7.3.4 Investigation of Model predictions

Suppose we are in the happy (but hypothetical) situation, that subensemble 2 has an HMC autocorrelation time $\mathcal{A}_2^{\mathrm{H}} = 0$, in other words that the HMC manages to produce a completely decorrelated time series of the field ϕ and the sole purpose of subensemble 2 is to decorrelate subensemble 1. The cost metrics of the chapter 6 can then be written as

$$\frac{\mathcal{A}_{11}}{\mathcal{A}_1} = \frac{1-p}{1+p\mathcal{A}_1} \tag{7.40}$$

$$\frac{\mathcal{A}_{22}}{\mathcal{A}_2} = \frac{1+p(\mathcal{A}_1-1)}{1+p\mathcal{A}_1}\,. \tag{7.41}$$

For a fixed nonzero acceptance probability, $0 < p \leq 1$, $\frac{\mathcal{A}_{11}}{\mathcal{A}_1} \to 0$ as $\mathcal{A}_1 \to \infty$. The rate of decrease of $\frac{\mathcal{A}_{11}}{\mathcal{A}_1}$ grows as p increases. When $p = 1$, we have that $\frac{\mathcal{A}_{11}}{\mathcal{A}_1} = 0$ implying that the PT perfectly decorrelates subensemble 1. We note that for the no acceptance scenario ($p = 0$) both cost metrics are unity.

Assuming now that $\mathcal{A}_2 \neq 0$, we write $\mathcal{A}_1 = r\mathcal{A}_2$, for some non–negative real number r. The cost functions become

$$\frac{\mathcal{A}_{11}}{\mathcal{A}_1} = \frac{(\mathcal{A}_2-1)p+1}{(1+r)p\mathcal{A}_2+1} \tag{7.42}$$

$$\frac{\mathcal{A}_{22}}{\mathcal{A}_2} = \frac{(r\mathcal{A}_2 - 1)p + 1}{(1+r)p\mathcal{A}_2 + 1} . \tag{7.43}$$

For fixed non–zero values of p and $\mathcal{A}_2$, we have that the cost function $\frac{\mathcal{A}_{11}}{\mathcal{A}_1}$ decreases with increasing r and that the rate of decrease increases as p is increased. We show a plot of $\frac{\mathcal{A}_{11}}{\mathcal{A}_1}$ against the acceptance rate p and ratio r in figure 7.1. To be able to plot the cost as a three dimensional surface we have set $\mathcal{A}_2 = 20$. We think this is a reasonable value for $\mathcal{A}_2$. It is certainly in line with our autocorrelation estimates of chapter 8. To aid interpreting figure 7.1 we show also the $\frac{\mathcal{A}_{11}}{\mathcal{A}_1} = \frac{1}{2}$ plane, where break–even is achieved when only subensemble 1 is of interest. The

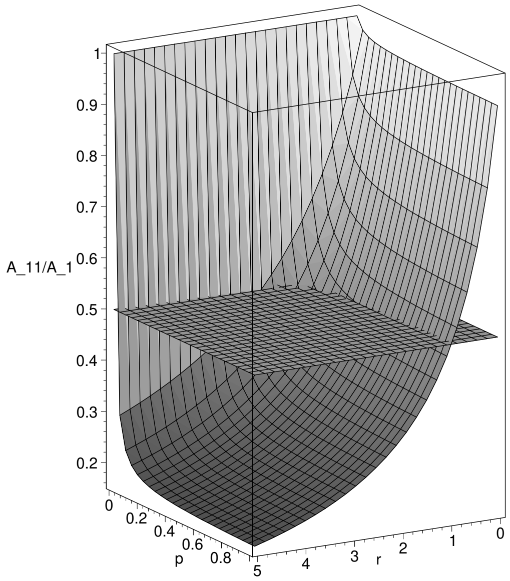

Figure 7.1: The metric $\frac{\mathcal{A}_{11}}{\mathcal{A}_1}$ of (7.42), against acceptance rate p and ratio $r = \frac{\mathcal{A}_1}{\mathcal{A}_2}$ with $\mathcal{A}_2 = 20$. The plane $\frac{\mathcal{A}_{11}}{\mathcal{A}_1} = \frac{1}{2}$ is also displayed.

behaviour of $\frac{\mathcal{A}_{11}}{\mathcal{A}_1}$ in figure 7.1 seems to be dominated by the ratio r. When $r < 1$, in other words, when $\mathcal{A}_2 > \mathcal{A}_1$, break–even is not achieved even for large values of p. When $r = 1$ break–even is achieved as discussed before. When $r > 1$, the cost ratio falls below the break–even level, even for modest values of the acceptance rate p. The conclusion is that the gain in subensemble one is greatest when $\mathcal{A}_1 \gg \mathcal{A}_2$, as we have speculated in chapter 6.

In figure 7.2, we show the cost ratio $\frac{\mathcal{A}_{22}}{\mathcal{A}_2}$, of the second subensemble, against the parameters r and p, once again with $\mathcal{A}_2 = 20$, and once again showing the plane

$\frac{\mathcal{A}_{22}}{\mathcal{A}_2} = \frac{1}{2}$. We can see that $\frac{\mathcal{A}_{22}}{\mathcal{A}_2}$ behaves in a complimentary manner to $\frac{\mathcal{A}_{11}}{\mathcal{A}_1}$. Hence

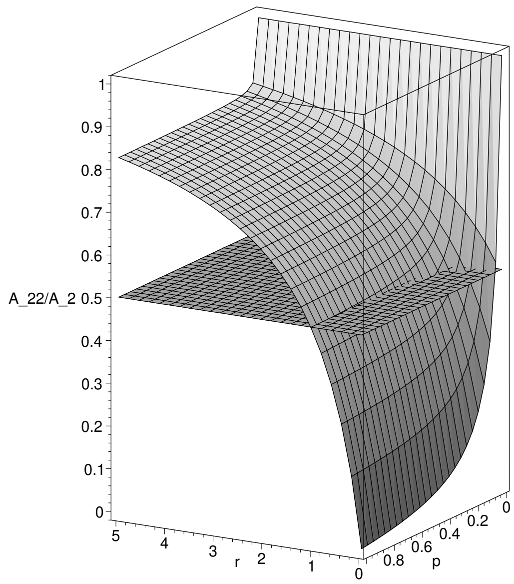

Figure 7.2: The metric $\frac{\mathcal{A}_{22}}{\mathcal{A}_2}$ of (7.42), against acceptance rate p and ratio $r = \frac{\mathcal{A}_1}{\mathcal{A}_2}$ with $\mathcal{A}_2 = 20$. The plane $\frac{\mathcal{A}_{22}}{\mathcal{A}_2} = \frac{1}{2}$ is also displayed.

for the case of $\frac{\mathcal{A}_{22}}{\mathcal{A}_2}$, when $r > 1$, (when $\mathcal{A}_1 > \mathcal{A}_2$), break–even is not achieved. This is the effect of mixing in the longer autocorrelations of subensemble 1 into subensemble 2. This complementarity is also immediate on comparison of (7.42) and (7.43). Hence speed up of subensemble 1 is at the price of a slow down of subensemble 2 in the case when $\mathcal{A}_1 > \mathcal{A}_2$ and vice versa when $\mathcal{A}_1 < \mathcal{A}_2$.

We can see here that in the two subensemble system the gain from PT depends chiefly on the ratio of the untempered integrated autocorrelation times. When this ratio is large, there will be a large gain in one subensemble with a corresponding "loss" in the other. We can also see that if the ratio of autocorrelation times is sufficiently large and one can obtain a non zero acceptance rate between the subensembles, one does not necessarily require more than two subensembles to maximise the gain from PT.

7.4 Three Subensemble Oscillator

We now proceed to discuss the matrix model for the three subensemble tempered oscillator. Although the construction of the autocorrelation matrix $C_{ij}(t)$ is a straightforward generalisation of the two subensemble case, the algebra gets very complicated. Furthermore the number of parameters jumps from 3 to 5 and so analysis also becomes nontrivial. Here we shall content ourselves with the construction of the model, and its predictions in the maximally conservative case that the untempered autocorrelation times are equal in all three subensembles and where the swap acceptances are also all equal.

7.4.1 The Autocorrelation Matrix

For a three subensemble system the HMC Update matrix is a straightforward extension of the two subensemble one,

$$\hat{U}^{\mathrm{H}}(1) = \begin{pmatrix} C_1(1) & 0 & 0 \\ 0 & C_2(1) & 0 \\ 0 & 0 & C_3(1) \end{pmatrix}. \tag{7.44}$$

One now has three available swapping matrices (for the swaps of subensembles 1 and 2, 1 and 3, 2 and 3), however, restricting ourselves to our earlier algorithmic definition we need consider only two of these, swapping between subensembles 1 and 2 with acceptance probability p_1 and swapping between subensembles 2 and 3 with acceptance probability p_2. The relevant swapping matrices are

$$\hat{U}^{\mathrm{S}}_1 = \begin{pmatrix} 1-p_1 & p_1 & 0 \\ p_1 & 1-p_1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{7.45}$$

$$\hat{U}^{\mathrm{S}}_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1-p_2 & p_2 \\ 0 & p_2 & 1-p_2 \end{pmatrix}. \tag{7.46}$$

The autocorrelation matrix $C_{ij}(t)$ is constructed in the usual fashion as

$$C_{ij}(t) = \left(\hat{U}^{\mathrm{S}}_2 \hat{U}^{\mathrm{S}}_1 \hat{U}^{\mathrm{H}}(1)\right)^t C_{ij}(0) = \left(\hat{U}^{\mathrm{S}}_2 \hat{U}^{\mathrm{S}}_1 \hat{U}^{\mathrm{H}}(1)\right)^t I_3 \tag{7.47}$$

where $C_{ij}(0) = I_3$ is the three dimensional unit matrix, which comes from the assumption of "independent" simulations. Carrying out the matrix algebra one finds

$$C_{ij}(t) = \begin{pmatrix} (1-p_1)C_1(1) & p_1 C_2(1) & 0 \\ (1-p_2)p_1 C_1(1) & (1-p_2)(1-p_1)C_2(1) & p_2 C_3(1) \\ p_1 p_2 C_1(1) & p_2(1-p_1)C_2(1) & (1-p_2)C_3(1) \end{pmatrix}^t . \tag{7.48}$$

This correlation matrix is now substantially more difficult to work with than the one in the twin subensemble case. One can attempt to find the integrated autocorrelation matrix $\mathcal{A}_{ij}$ proceeding in a manner similar to before. One begins by setting

$$C_1(1) = \exp\{-k_1\} = \frac{\mathcal{A}_1}{\mathcal{A}_1 + 1} \tag{7.49}$$

$$C_2(1) = \exp\{-k_2\} = \frac{\mathcal{A}_2}{\mathcal{A}_2 + 1} \tag{7.50}$$

$$C_3(1) = \exp\{-k_3\} = \frac{\mathcal{A}_3}{\mathcal{A}_3 + 1} , \tag{7.51}$$

whereupon $C_{ij}(t)$ becomes

$$C_{ij}(t) = \begin{pmatrix} \frac{(1-p_1)\mathcal{A}_1}{\mathcal{A}_1+1} & \frac{p_1\mathcal{A}_2}{\mathcal{A}_2+1} & 0 \\ \frac{(1-p_2)p_1\mathcal{A}_1}{\mathcal{A}_1+1} & \frac{(1-p_2)(1-p_1)\mathcal{A}_2}{\mathcal{A}_2+1} & \frac{p_2\mathcal{A}_3}{\mathcal{A}_3+1} \\ \frac{p_1p_2\mathcal{A}_1}{\mathcal{A}_1+1} & \frac{(1-p_1)p_2\mathcal{A}_2}{\mathcal{A}_2+1} & \frac{(1-p_2)\mathcal{A}_3}{\mathcal{A}_3+1} \end{pmatrix}^t . \tag{7.52}$$

The inverses required for $\mathcal{A}_{ij}$ exists but is complicated and we shall not detail their individual elements. However when we make worst case assumption of chapter 6 that $\mathcal{A}_1 = \mathcal{A}_2 = \mathcal{A}_3 = \mathcal{A}$ and that $p_1 = p_2 = p$, the matrix elements simplify considerably. The diagonal elements of $\mathcal{A}_{ij}$ are

$$\mathcal{A}_{11} = \frac{\mathcal{A}(\mathcal{A}^2p^2 + 3(1-p)p\mathcal{A} + (1-p))}{3\mathcal{A}^2p^2 + (4-p)p\mathcal{A} + 1} \tag{7.53}$$

$$\mathcal{A}_{22} = \frac{\mathcal{A}(\mathcal{A}^2p^2 + 2(1-p)p\mathcal{A} + (1-p)^2)}{3\mathcal{A}^2p^2 + (4-p)p\mathcal{A} + 1} \tag{7.54}$$

$$\mathcal{A}_{33} = A_{11} , \tag{7.55}$$

and the off diagonal elements are

$$\mathcal{A}_{12} = \frac{(1+\mathcal{A}p)(\mathcal{A}+1)\mathcal{A}p}{3\mathcal{A}^2p^2 + (4-p)p\mathcal{A} + 1} \tag{7.56}$$

$$\begin{aligned}
\mathcal{A}_{13} &= \frac{p^2\mathcal{A}^2(\mathcal{A}+1)}{3\mathcal{A}^2p^2+(4-p)p\mathcal{A}+1} && (7.57)\\
\mathcal{A}_{21} &= \frac{\mathcal{A}p(1+p(\mathcal{A}-1))(\mathcal{A}+1)}{3\mathcal{A}^2p^2+(4-p)p\mathcal{A}+1} && (7.58)\\
\mathcal{A}_{23} &= \frac{(1+\mathcal{A}p)(\mathcal{A}+1)\mathcal{A}p}{3\mathcal{A}^2p^2+(4-p)p\mathcal{A}+1} && (7.59)\\
\mathcal{A}_{31} &= \frac{p^2\mathcal{A}(\mathcal{A}+1)^2}{3\mathcal{A}^2p^2+(4-p)p\mathcal{A}+1} && (7.60)\\
\mathcal{A}_{32} &= \mathcal{A}_{21} && (7.61)
\end{aligned}$$

It can be seen from (7.53) and (7.55) that $\frac{\mathcal{A}_{11}}{\mathcal{A}_1}$ and $\frac{\mathcal{A}_{33}}{\mathcal{A}_3}$ are identical. This is probably due to the fact that configurations originating in subensemble 2 swap either into subensemble 1 or 3 on a symmetric fashion, whereas configurations from subensemble 1 or three can swap only to subensemble 2. We note that the symmetry of the two subensemble model is no longer evident although some traces remain, such as the identical integrated autocorrelation times $\mathcal{A}_{11}$ and $\mathcal{A}_{33}$ as well as the identical integrated cross correlations $\mathcal{A}_{32} = A_{21}$.

The cost functions for the subensembles can then be found as

$$\begin{aligned}
\frac{\mathcal{A}_{11}}{\mathcal{A}_1} &= \frac{\mathcal{A}^2p^2+3(1-p)p\mathcal{A}+(1-p)}{3\mathcal{A}^2p^2+(4-p)p\mathcal{A}+1} && (7.62)\\
\frac{\mathcal{A}_{22}}{\mathcal{A}_2} &= \frac{\mathcal{A}^2p^2+2(1-p)p\mathcal{A}+(1-p)^2}{3\mathcal{A}^2p^2+(4-p)p\mathcal{A}+1} && (7.63)\\
\frac{\mathcal{A}_{33}}{\mathcal{A}_3} &= \frac{\mathcal{A}_{11}}{\mathcal{A}_1}\,, && (7.64)
\end{aligned}$$

For a fixed, nonzero p, all three of these cost ratios will approach $\frac{\mathcal{A}_{ii}}{\mathcal{A}_i} = \frac{1}{3}$ as $\mathcal{A} \to \infty$. Depending on the value of p this limit is approached from either above and below. This result is consistent with our supposition of chapter 6, that for a system of N_s subensembles, where the subensembles have equal autocorrelation times, the maximum gain that can be achieved through mechanical mixing of configurations is a factor of $\frac{1}{N}$.

7.5 Conclusions for the Chapter

We now recollect the results of the matrix model when applied to the twin subensemble tempered harmonic oscillator. We shall make the bold leap of faith, that although quantitatively the model is only applicable to systems satisfying our earlier assumptions, we can carry over its gross qualitative features regarding the mixing of HMC autocorrelation times to other systems such as lattice QCD and various spin glass systems.

After constructing the model and discussing the underlying linearity assumption we applied the matrix model to tempered harmonic oscillator systems consisting of two and three subensembles. We discuss the conclusions for each in turn.

Firstly we have shown that in the two subensemble system we could recover the results of chapter 6 for the case when the untempered autocorrelation times of the two subensembles were equal. Since the calculation in section 6 does not make the strong assumptions which we have used in this chapter, namely our linearity assumption and that the observables in question have to be linear variables of the fields in the simulation, we expect it to be more general than our matrix model and thus to constrain it. Hence the agreement between the two models is not unexpected.

We then applied the matrix model to the case of unequal untempered autocorrelation times, and found that not only could we reach break–even, but that we could make quite significant gains in one subensemble to the detriment of autocorrelations in the other. The amount of gain (loss) depended mostly on the ratio $r = \frac{\mathcal{A}_1}{\mathcal{A}_2}$ of the untempered autocorrelation times. Hence one can look at the equal untempered autocorrelation time case in a different light, as the balance point where neither subensemble will "improve", but neither will deteriorate either.

Our investigation of the three subensemble system was quite cursory. Although in principle the construction of the relevant matrices is straightforward, in practice the complexity of the matrix determinants, inverses and even just the matrix elements becomes unwieldy in the general situation when all the untempered autocorrelation times and acceptance probabilities are distinct, although the software package Maple could manipulate the expressions with ease. We considered here only the worst case scenario when all three untempered autocorrelation times as well as the acceptance probabilities were equal. The results indicate that in this maximally conservative scenario, the maximum gain is a factor of 3, the break–even value, compared to the single ensemble case which is what one would expect naively.

Since the twin ensemble calculation indicates that substantial gains may be made in the situation when the ratio r is large, it is tempting to ask whether physical systems exists when one can attain reasonable acceptance rates between subensembles which have wildly different autocorrelation times. Such a situation can be achieved in spin glass simulations for which the method was originally devised. In figure 7.3 we show the autocorrelation time of the magnetisation plotted against the temperature for the Edwards-Anderson Spin Glass system investigated in [4][2]. We can see in figure 7.3 that as the temperature of the system was lowered below a critical value of $T_c \approx 1.5$, the untempered autocorrelation time of the magnetisation grows dramatically from $\mathcal{A} \approx 0$ to $\mathcal{A} \approx 3000$, even though the change in temperature is quite small so that one can simulate a subensemble on either side of T_c and still have these subensembles close to each other in terms of distances in the parameter space. In this situation it is possible to get a reasonable acceptance rate between the two subensembles so that the one with $T < T_c$ and the one with $T > T_c$ have vastly different autocorrelation times. In this situation clearly one expects to gain with PT.

It is not immediately clear whether the behaviour of topological charge in the case of lattice QCD is similar to the behaviour of the magnetisation in the Edwards-Anderson spin–glass system. It is expected that topological charge autocorrelations should grow long in the region of parameter space where chiral symmetry is restored, as for the case of chiral fermions, the topological charge is a conserved quantity. However the abruptness of the change in the autocorrelation times of the topological charge is not well known at present.

We finish off this chapter by pointing out, that tempering does introduce cor-

[2]We reproduce here the plot of [4] which was available on the World Wide Web

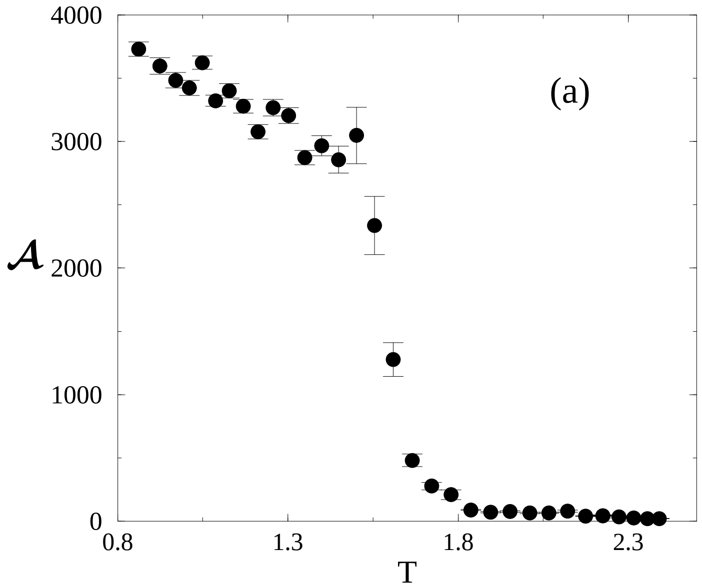

Figure 7.3: Untempered Autocorrelation time, A, of the magnetisation in the Edwards-Anderson Spin–Glass system as a function of temperature T (Reproduced from [4])

relations between subensembles, as can be seen by looking at the off diagonal elements of the autocorrelation matrix C_{ij} and of the integrated autocorrelation matrix $\mathcal{A}_{ij}$. It is not immediately clear how these cross–correlations enter into expectation values computed from the data in the subensembles. However, if all the subensembles are of interest, say for the purpose of some chiral extrapolation, it would perhaps be prudent to use techniques which account for correlations between the datasets such as correlated χ^2 fitting. We expect that the data correlation matrix for such fits can be estimated using jackknife or bootstrap techniques as is being done in current day analysis codes. The degree by which these cross correlations would increase statistical errors, in other words the amount of expected gain taken away by these cross correlations, is not yet addressed by our studies.

Chapter 8

Numerical Investigations of PT

In this short chapter we outline our numerical investigations of the parallel tempering algorithm. We shall proceed by highlighting some noteworthy features of our PT simulation program. Thereafter we shall present our simulation parameters. Finally we shall discuss the results of our computations.

8.1 Design and Implementation of our PT simulation Code

Our parallel tempering program was constructed using code from the software code–base of the UKQCD collaboration. We have implemented a program which allowed a reasonably general combination of HMC and swapping steps. This code was run on a Cray T3E-900 computer housed at the University of Edinburgh, which is a MIMD[1] massively parallel supercomputer. In particular we have used many subroutines from the GHMC code [38] of the collaboration to perform the HMC updates of our PT subensembles, which are heavily optimised to take maximum advantage of the facilities offered by the T3E.

[1] Multiple Instruction Multiple Data

8.1.1 Serial Parallel Tempering

In the initial design of our program we considered distributing individual PT subensembles to specific groups of processors on the Cray T3E, so that HMC updates could be made in several subensembles concurrently. However we have abandoned this idea after considering load balancing issues.

The load balancing problem arose from the consideration that HMC steps in different subensembles require in principle a different amount of CPU time to complete. Hence before performing a swap step between two subensembles the program would have to wait for the HMC steps in both subensembles to complete. If separate groups of processors were used for the two subensembles, this may have entailed that members of one set may become idle, waiting for the computation on the other set to finish before the swap attempt could proceed.

This problem was overcome by performing HMC updates in serial before attempting swapping steps. This way HMC steps in each subensemble could take advantage of all the processors available. The workload for performing an HMC step is known to be generally well balanced across the processors of the system. Hence by employing this "serial parallel tempering" strategy we did not need to worry about issues of load balancing.

The decision to employ this serial approach also proved beneficial when considering software integration issues, as the UKQCD software code–base was written to take advantage of all available processors in its allocated machine partition. The serial approach therefore meant that the amount of maintenance on the subroutines of the UKQCD code required for their integration into our PT program could be kept to a minimum.

Apart from load balancing and software reuse issues, the choice of performing only one HMC update at a time also simplified the design of the overall PT code, by turning a potentially difficult concurrent algorithm into a simple state machine[2] the structure of which is shown in figure 8.1.

[2]This is not strictly speaking a Finite State Automaton in the rigorous sense, since the state machine in this case requires memory

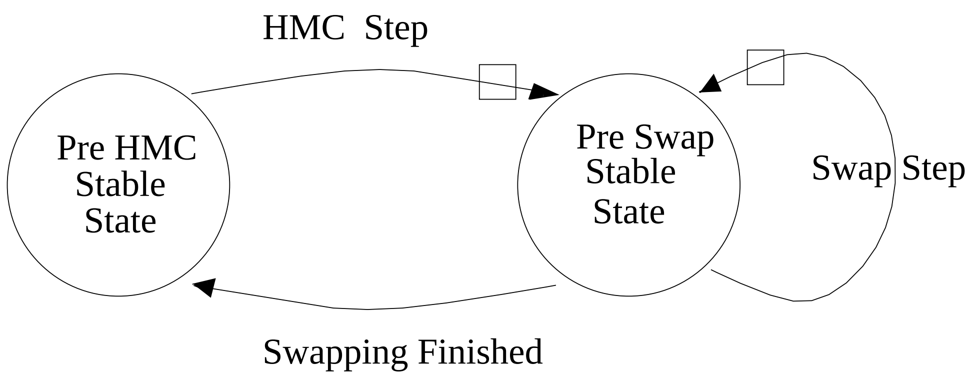

Figure 8.1: The PT state machine. Arrows represent Markov Steps. A box at the end of a Markov step indicates where a checkpoint needs to be made.

For the purposes of figure 8.1 we consider the PT simulation to comprise of two kinds of stable states. The so called Pre–HMC stable state is the state prior to an HMC step in one of the subensembles. The HMC step is a transition which takes the system to a Pre–Swap stable state. The system can then execute swap steps after each of which it returns to a Pre–swap stable state ready to perform another swap. Once all the desired swaps have been performed the system returns to a Pre–HMC stable state and can then repeat this cycle for a different subensemble. The stable states are points at which the simulation can write a checkpoint, which is complete description of the system state. In case of a machine crash the system can be restarted from such a checkpoint.

In our implementation, each subensemble could be controlled through its own independent input parameter file. This way, one had independent control of all the simulation parameters for each subensemble. Thus, in principle, one was free to make HMC steps of differing lengths in the different subensembles, or to use different step–sizes, solver residues and so forth. The format of these parameter files was compatible with the UKQCD GHMC code, which was used to equilibrate the individual subensembles prior to employing PT.

8.1.2 Swap Plan Matrix

After subensemble i has completed its HMC update, the decision to propose a swap of configurations between subensembles j and $j+1$, ($j = 1 \dots N-1$) was made depending on the value, M_{ij}, of a matrix of Boolean values which we called the *swap plan matrix*. The program would perform the swap step if the value of M_{ij} was *true* otherwise it would not.

The plan matrix gave us considerable flexibility (although not total freedom) in how we mixed our two update schemes. In general however we usually adopted a strategy where all the elements of M were set to *false*, except in the last row corresponding to $i = N$. This scheme corresponds exactly to the PT algorithm in the last chapter in which one performs HMC updates in all the subensembles which are then followed by a string of pairwise swap steps between subensembles j and $j+1$ for $j = 1, \dots, N-1$.

8.1.3 Code Verification and Validation

To verify the correct operation of our PT code we carried out test simulations where we performed no swap steps at all. We then carried out reference HMC tests starting from the same original configurations as the PT simulations. As the individual subensembles had their own (pseudo)random number generators, and we used exactly the same random number seed between corresponding subensembles and HMC test simulations, we could ensure that the configurations from the PT tests were bit-identical with the results of the corresponding HMC ones. We also carried out tests to ensure the correctness of the swap steps and the checkpointing mechanism.

In order to validate the program, we computed the expectation value of the plaquette operator in one subensemble of our PT simulations and compared these with the corresponding value in a reference HMC simulation. This data is presented later on in this chapter.

8.2 Simulation Parameters

We carried out five PT simulations in total, labelled S1, S2, S3, S4 and S5 respectively. Each of these simulations comprised of two sub-ensembles. In what follows we shall often refer to subensembles 1 and 2 of these simulations as the first and second subensembles of the simulation respectively. The lattice volume used for these simulations was $V = 8^3 \times 16$ sites. The physical parameters used and the number of compound PT steps taken for each simulation is shown in table 8.1. In all the cases a HMC step in the PT subensembles consisted of a single

Simulation	(β_1, c_1, κ_1)	(β_2, c_2, κ_2)	No. of Steps
HMC	$(5.2, 2.0171, 0.133)$	N/A	6000
$S1$	$(5.2, 2.0171, 0.133)$	$(5.2060, 2.01002, 0.13280)$	6000
$S2$	$(5.2, 2.0171, 0.133)$	$(5.2105, 2.00471, 0.13265)$	6000
$S3$	$(5.2, 2.0171, 0.133)$	$(5.2150, 1.99940, 0.13250)$	6000
$S4$	$(5.2, 2.0171, 0.133)$	$(5.2, 2.0171, 0.13280)$	1000
$S5$	$(5.2, 2.0171, 0.133)$	$(5.2, 2.0171, 0.13265)$	1000

Table 8.1: Parameters of the PT simulations and reference HMC computation with $V = 8^3 \times 16$. The number of fundamental updates used in the simulations are also shown.

trajectory of unit length. Other HMC parameters such as the MD timestep, the type of solvers used and the desired solver residues were chosen to be identical in all the subensembles.

We also had data from previous HMC simulations with parameters ($\beta = 5.2, c = 2.0171, \kappa = 0.1330$) on lattices of volume $V = 8^3 \times 16$ and $V = 8^3 \times 24$ sites. The results from the reference run at $V = 8^3 \times 16$ were used to validate the PT simulation code. Furthermore, adequate statistics were available to compare the integrated autocorrelation time of the plaquette operator from this reference simulation with the results from the first sub-ensembles of the PT ones. For the second sub–ensembles no reference HMC simulations were available and hence no such a comparison could be made.

A pictorial illustration of the region of parameter space explored by our simulations is shown in figure 8.2. In this figure the red diamonds represent the parameters of the second subensembles of the matched PT simulations whereas the blue squares correspond to the second subensembles of the unmatched ones. The parameters of the first subensemble and the reference HMC simulation are shown by a black circle[3] .

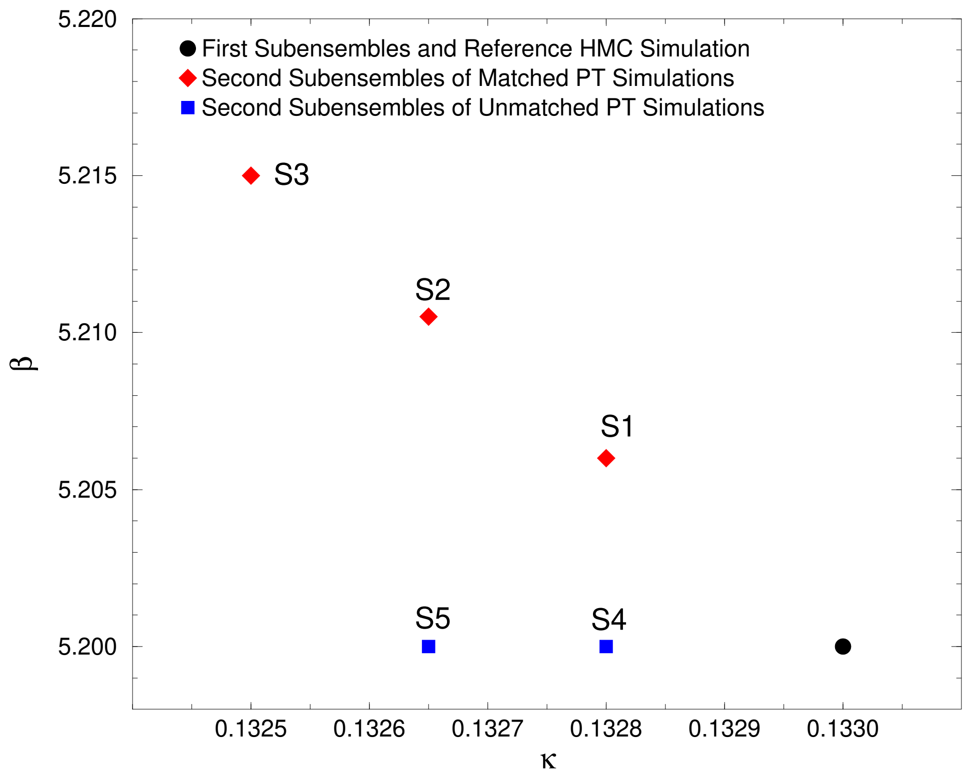

Figure 8.2: The parameter space explored by our PT simulations

8.2.1 Background to the Selection of Parameters

At the outset of this study, the complications of a nonzero clover coefficient were not yet fully appreciated by us, and it was hoped that the non-linear dependence of the tuning conditions on β from the clover term was not severe. In the case of

[3]In this depiction we ignored the clover coefficient.

a zero clover coefficient, matching the plaquette between two subensembles and minimising the distance between them is equivalent to first order and we hoped, wrongly it has to be admitted, that even with the non-zero clover coefficient the two tuning prescriptions would be reasonably close to each other. In practice we were unable to make unbiased estimates of some of the quantities required to minimise the distance between the actions with our Lánczos based technology [6]. We could only make unbiased estimates of the correlators required to match the plaquette. We also tried matching the action itself, in the observable matching prescription, but in this case we needed to use biased estimators for some of the quantities.

The actual matching procedure used was the one described in [6], for matching the plaquette and matching the full action. We matched both the plaquette and the action separately for every value of $\Delta\kappa$, although for matching the action some biased estimators were used. Both matching prescriptions gave the same values of $\Delta\beta$ within statistical errors.

Given our chosen value of $\Delta\kappa$, the required correlations ($\left\langle \widetilde{\mathcal{W}}\widetilde{\Delta T}\right\rangle, \left\langle \widetilde{\mathcal{W}}^2\right\rangle$, etc) were evaluated on configurations of the $V = 8^3 \times 24$ reference HMC simulation, which were generated at the parameters of the first sets of the PT simulations. The value of $\Delta\beta$ corresponding to our chosen $\Delta\kappa$ was then calculated by using

$$\Delta\beta = \frac{d\beta}{d\kappa}\Delta\kappa \tag{8.1}$$

$$\frac{d\beta}{d\kappa} = -\frac{\left\langle \widetilde{\mathcal{O}}\frac{\partial\widetilde{T}}{\partial\kappa}\right\rangle}{\left\langle \widetilde{\mathcal{O}}(\widetilde{\mathcal{W}} + \frac{\partial\widetilde{T}}{\partial c}\frac{dc}{d\beta})\right\rangle}\,, \tag{8.2}$$

where $\mathcal{O}$ took the value $\mathcal{O} = \mathcal{W}$ when matching the plaquette and $\mathcal{O} = S_i$ when matching the action. The quantity $c(\beta)$, has been parameterised as a rational polynomial in [12]. Hence $\frac{dc(\beta)}{d\beta}$ could be written down, without having to measure any correlations.

Our initial studies were concerned with investigating the acceptance rate of simulations with $\Delta\kappa$. Furthermore we had hoped at the outset that the matching would give us reasonable acceptance rates for quite large values of $\Delta\kappa$. We chose the parameters of the first subensembles to be at $(\beta = 5.2, c = 2.0171, \kappa = 0.1330)$

as we have already carried out reference HMC simulations at these parameters and this not only gave us an HMC dataset for comparison with our PT studies, but we knew the approximate speed at which configurations were generated and had found this to be suitable for our test simulations. We initially tried making the parameters of the first subensembles to be ($\beta = 5.2, c = 2.0171, \kappa = 0.1340$) where a reference simulation was also available but found that the speed of configuration generation was too slow for us.

Having decided on the parameters for our first subensembles, we chose initially a rather large $\Delta\kappa = -7.5 \times 10^{-4}$ for simulation $S3$. However having found our acceptance rates to be quite low, we systematically lowered $\Delta\kappa$ to obtain the parameters for $S2$ and $S1$, in an attempt to investigate the behaviour of the acceptance rate with $\Delta\kappa$. In all three cases, $\Delta\beta$ was determined with the tuning methodology described above. The parameters for simulations $S4$ and $S5$ were chosen to be unmatched simulations with $\Delta\kappa$ corresponding to $S1$ and $S2$ respectively, so that the effect of the matching on the acceptance rate for fixed $\Delta\kappa$ could be investigated.

8.2.2 Observables And Data Analysis

In each simulation we measured the expectation value of the change in the joint action of the PT system, $\langle \Delta S \rangle_{12}$, after every swap attempt. We also noted whether a swap attempt was successful or not, thus allowing a direct determination of the swap acceptance rate $\langle P_{\text{acc}} \rangle_{12}$

We calculated the average plaquette $U_{\mu\nu}$ in both subensembles of each simulation following every PT compound step. We used this data to calculate the expectation value, $\langle U_{\mu\nu} \rangle_1$, and its corresponding integrated autocorrelation time $\mathcal{A}_1$ in the first subensembles of the simulations.

Errors on $\langle \Delta S \rangle_{12}$ and $\langle P_{\text{acc}} \rangle$ were computed using the bootstrap method. For $\langle U_{\mu\nu} \rangle_1$ the errors quoted here were estimated using corresponding value of the integrated autocorrelation time and (6.1). Error estimates on the integrated autocorrelation times themselves were computed using the relevant equations in

[53].

We point out that the autocorrelation times quoted here are somewhat different from those of our published paper [54]. The reason for this is twofold. Firstly, in this thesis we use the definition of the integrated autocorrelation time from [52] whereas in our paper we used the definition of [53]. Secondly, for inclusion into this thesis the autocorrelation data has been reanalysed and it was found that the results presented in our paper were slightly underestimated, although the error bars are so large that consistency with [54] is still maintained. These differences in the quoted autocorrelation times do not affect the conclusions of either [54] or this thesis.

8.3 Results

8.3.1 Swap Acceptance Rate and Action Matching

Our measurements of the swap acceptance rate, $\langle P_{\rm acc}\rangle$, and $\langle \Delta S\rangle_{12}$, the expectation value of the change in the joint action, are shown in table 8.2. To facilitate our forthcoming discussion of action matching we show also the $\Delta\beta = \beta_2 - \beta_1$ and $\Delta\kappa = \kappa_2 - \kappa_1$ for each simulation. We note that in this convention $\Delta\kappa$ is always negative.

Simulation	$\Delta\beta(\times 10^{-3})$	$\Delta\kappa(\times 10^{-4})$	$\langle \Delta S\rangle_{12}$	$\langle P_{\rm acc}\rangle_{12}$
S1	6	-2.0	1.23(2)	0.43(1)
S2	10.5	-3.5	3.76(4)	0.17(1)
S3	15	-7.5	7.64(6)	0.051(2)
S4	0	-2.0	0.91(4)	0.49(1)
S5	0	-3.5	2.29(7)	0.26(2)

Table 8.2: $\langle \Delta S\rangle_{12}$ and $\langle P_{\rm acc}\rangle_{12}$ of the swapping attempts in the simulations, $\Delta\beta$ and $\Delta\kappa$ are also shown.

In figure 8.3 we plot the measured swap acceptance rates against the corresponding measurements of $\langle \Delta S \rangle_{12}$. We indicate by a dashed line the formula (5.16). We note that no fitting was used in the production of this plot. We can see excellent agreement between the measured acceptance rates and (5.16).

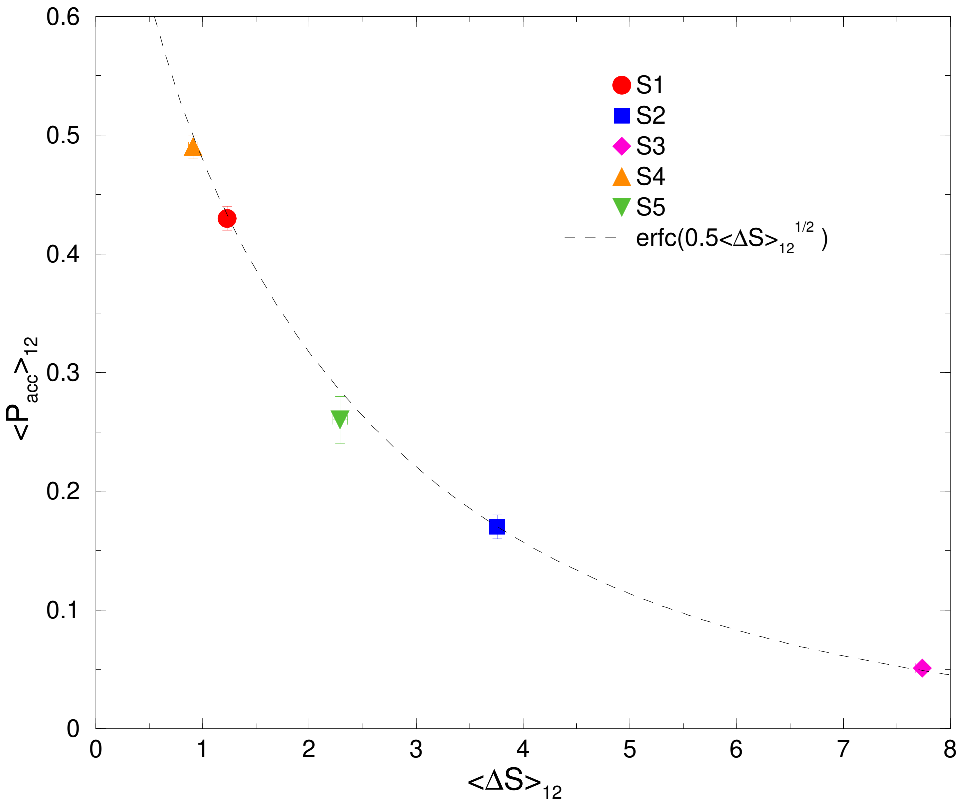

Figure 8.3: The swap acceptance rate of the simulations. We indicate also the analytic formula of (5.16).

We note also from figure 8.3 that the swap acceptance rate of simulation $S4$ is greater than that of $S1$ and likewise simulation $S5$ has a higher swap acceptance rate than $S2$. It can be seen in table 8.2 that simulations $S4$ and $S1$ have the same value of $\Delta\kappa$ as do simulations $S5$ and $S2$. We remind the reader that action matching was employed to tune $\Delta\beta$ in simulations $S1$ and $S2$, while for the simulations $S4$ and $S5$, $\Delta\beta = 0$ and tempering was carried out in κ only.

It is with some slight disappointment that we find the unmatched simulations $S4$ and $S5$ to have higher swap acceptance rates than their matched counterparts. We believe there are two causes for this seeming anomaly. Firstly, the action matching was carried out using the trace log action of (5.44) whereas the PT simulations have employed the pseudofermionic action (5.45). It may therefore be tempting to attribute the lower acceptance rates of simulations $S1$ and $S2$ to the connected term in the distance norm of the pseudofermionic action.

On the other hand, if we follow the analysis of chapter 5, we would still expect that the matched simulations should have higher acceptance rates than the unmatched ones, as the connected term corresponds to a $\Delta\kappa$ dependent shift between the distance norms of the pseudofermionic and trace log actions as illustrated in (5.66). Hence, if the distance norm for the trace log action is minimised, we would expect the distance for the pseudofermion action to be larger, but still to be minimised.

However, the discussion in chapter 5 does not take account of the clover coefficient in the two actions, which is a genuine headache for action matching. In our simulations, we used the observable matching tuning prescription to match our actions, where the observable $\mathcal{O}$ to keep constant between a pair of subensembles was the expectation value of the action itself. In the special case where the action to be tuned is a linear function of the tunable parameters, this condition is equivalent to first order in Δ to minimising the distance norm and maximising the swap acceptance rate between the two subensembles, as is shown in [5] where the approximate action to be tuned consists of a sum of Wilson loops. However, in our simulations the action to be tuned is a non–linear function of β due to the presence of the clover term in the fermion matrix. In this case, matching the actions in this manner is not equivalent to maximising the swap acceptance or minimising the distance norm.

We still expected this treatment to give reasonable swap acceptance rates, since our experience of matching indicated that most of the fluctuations in Δ came from the plaquette term of the action. Hence we expected small contributions to $\Delta\beta$ from correlations depending on the clover term. With hindsight, it is probably fair to say we were incorrect in this assumption.

As a final point we would like to mention that the acceptance rates for simulations $S2$ and $S5$, both of these having $\Delta\kappa = -3.5 \times 10^{-4}$, were 17% and

26% respectively, which we consider quite low. For simulation $S3$ the acceptance rate was about 5% with a corresponding $\Delta\kappa = 7.5 \times 10^{-4}$. Ideally, for all the subensembles of a simulation to be useful, we would like to have a high acceptance rate (of at least 50%) in situations where $\Delta\kappa \approx 5 \times 10^{-4}$. Achieving this seems unlikely even at such such small volumes as we have used in our study, and we expect it to be even less likely at larger volumes.

8.3.2 The plaquette operator

The measurements of the plaquette operator, $\langle U_{\mu\nu}\rangle_1$ in subensemble 1 of our simulations are shown in column 2 of table 8.3. In the third and fourth columns of this table we show the corresponding integrated autocorrelation times $\mathcal{A}_1$ and cost ratios $\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}}$ respectively. We also show the corresponding results from our reference HMC simulation at $V = 8^3 \times 16$. The errors on $\langle U_{\mu\nu}\rangle_1$ include the effects of autocorrelations. The errors on the ratio $\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}}$ in column 4 of the table were obtained by combining the errors of the relevant entries in column 3 in quadrature.

Simulation	$\langle U_{\mu\nu}\rangle_1$	$\mathcal{A}_1$	$\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}}$
HMC	0.5197(3)	25(7)	1.0(4)
S1	0.5195(2)	14(4)	0.5(2)
S2	0.5197(3)	23(9)	0.9(4)
S3	0.5202(3)	26(9)	1.0(5)
S4	0.5204(6)	14(9)	0.5(4)
S5	0.5210(8)	18(11)	0.7(5)

Table 8.3: Estimates of the expectation value $\langle U_{\mu\nu}\rangle_1$, autocorrelation time $\mathcal{A}_1$, and cost ratio $\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}}$ for the plaquette in subensemble 1 of the simulations. We show the HMC results for reference.

In figure 8.4 we illustrate graphically the results from column 2 of table 8.3 by plotting $\langle U_{\mu\nu}\rangle_1$ from the PT simulations against the corresponding values of $\Delta\kappa$ between the subensembles. We show also the value $\langle U_{\mu\nu}\rangle$ from the reference

HMC simulation. The results from the PT simulations are represented by data points in this figure, while the one from HMC is represented by a solid line for $\langle U_{\mu\nu} \rangle$ with dashed lines indicating the upper and lower error bounds. We observe

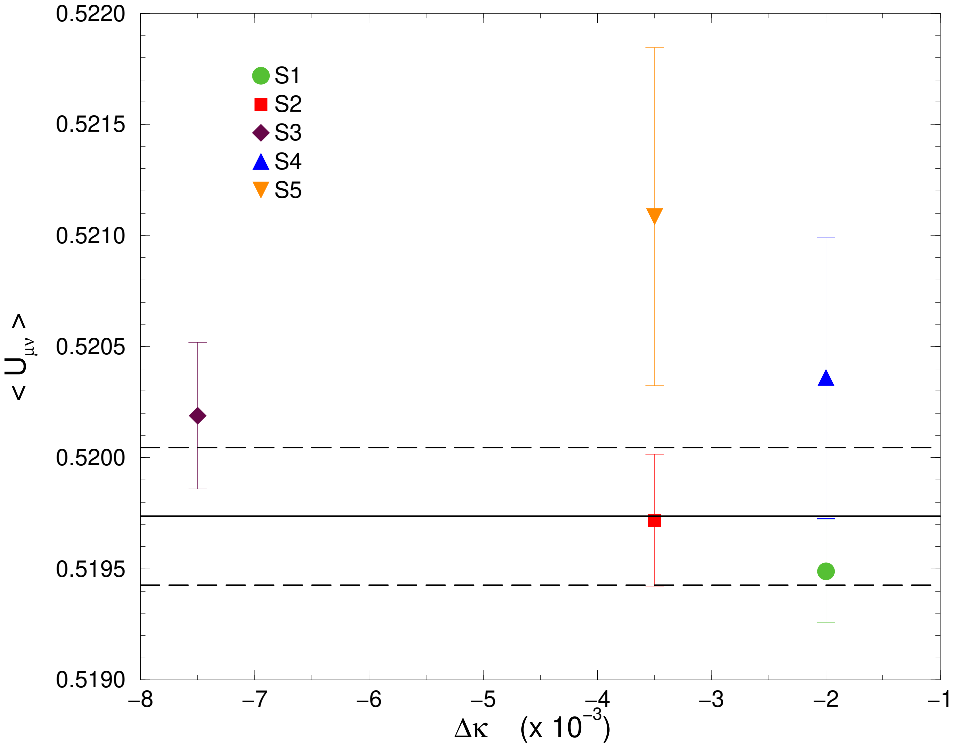

Figure 8.4: Values of $\langle U_{\mu\nu} \rangle_1$ in the first subensembles of the PT simulations (data points) and for the referene HMC simulation (the solid line is the mean value; the dashed lines are the upper and lower error bounds).

from figure 8.4, that within statistical errors, the estimates of $\langle U_{\mu\nu} \rangle_1$ from the first subensembles of our PT simulations are consistent with the result from the reference HMC simulation. The growth of the errors bars, from simulation $S1$ to $S3$ is due to the increase in autocorrelation times between these simulations. The large errors for simulations $S4$ and $S5$ on the other hand are due to the low statistics in these simulations compared to those of $S1 - S3$. This consistency between the PT and HMC results is a reassuring validation check on our PT simulation code.

8.3.3 Autocorrelation Times and Simulation Cost

The cost ratios $\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}}$ from column 4 of table 8.3 are plotted against the swap acceptance rate of the corresponding PT simulation in figure 8.5. The prediction of our twin subensemble calculation from (6.43) is also shown in this figure with a dashed line. The central values of the cost ratio for simulations $S1$ and $S4$ lie

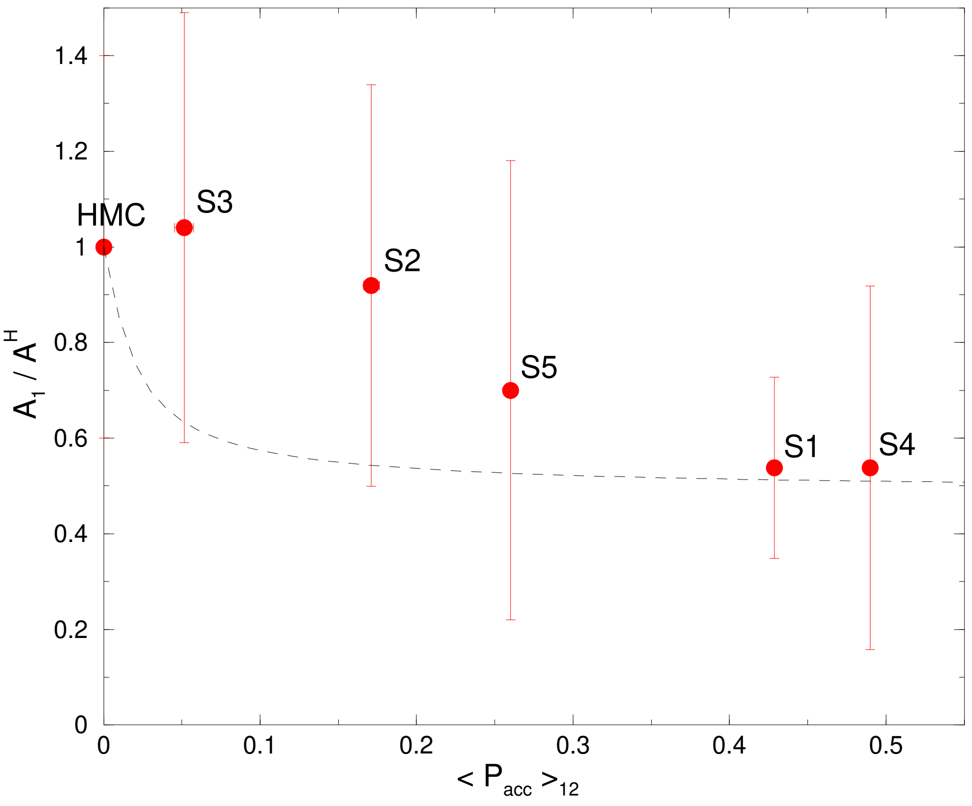

Figure 8.5: The cost ratio $\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}}$ for the PT simulations against swap acceptance rate $\langle P_{\mathrm{acc}} \rangle_{12}$. The points correspond to the simulations. The line is the prediction of the twin subensemble model (6.43).

close to the prediction of (6.43). We remark that these were the simulations with the two highest swap acceptance rates, and in both cases $\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}} \approx 0.5$. Although the rest of the data are also consistent with the predictions of (6.43) we cannot draw any definite conclusions regarding our twin subensemble model due to the very large errors on the data.

In table 8.4 we show estimates for the integrated autocorrelation time of the plaquette for several reference HMC simulations that have been carried out as part of projects not discussed in this thesis. We indicate also the system volume and input parameters β, κ and c used in these simulations, as well as the number of HMC trajectories over which the autocorrelation analysis was carried out.

V	β	c	κ	No. Traj.	$\mathcal{A}$
$8^3 \times 16$	5.200	1.99	0.1335	1500	18(10)
$8^3 \times 24$	5.200	2.02	0.1340	5500	35(10)
$8^3 \times 24$	5.232	1.98	0.1335	3800	26(10)

Table 8.4: Estimates of the integrated autocorrelation time for the plaquette on reference HMC simulations

Our purpose in showing these results is to in some way support our assumption of the last chapter, that the integrated autocorrelation time remains roughly constant between simulations separated by such small shifts in β, κ and c in parameter space. Clearly the values of $\mathcal{A}$ presented in this table corroborate this view within the errors indicated, however we stress again, that errors are very large.

8.4 Conclusions from the Numerical Results

In light of the data presented above we draw the following conclusions from our PT simulations:

The agreement in the values of the plaquette operator between the PT simulations and the reference HMC simulation indicates that we have no obvious programming errors in our simulation code. These results provide a reasonable validation of our code.

We have observed excellent agreement between the swap acceptance rates measured in our PT simulations and the predictions of (5.16).

We have not been able to optimise the swap acceptance rate of the PT simulations through action matching. In fact the acceptance rates of the unmatched simulations ($S4$ and $S5$) are somewhat higher than their matched counterparts ($S1$ and $S2$). We believe that this is our fault due to using an inappropriate matching condition rather than an inherent difficulty with the action matching approach.

We have noted that for $\Delta\kappa = -3.5 \times 10^{-4}$ between two PT subensembles we have been able to obtain acceptance rates of $O(20\%)$. The highest acceptance rate observed in all our simulations was approximately 50%, with a corresponding $\Delta\kappa = -2.0 \times 10^{-4}$. We note that with such small values of $\Delta\kappa$ we cannot adopt the optimistic point of view that all our subensembles are intrinsically interesting and therefore require a factor of 2 improvement in the autocorrelation times of our observables in the subensemble of interest, to make PT cost–effective compared to HMC.

We have exhibited autocorrelation times from some reference HMC simulations with similar parameter values to our PT simulations. From this data we can infer, within the very large errors, that in the region of parameter space that we have explored, the autocorrelation time of the plaquette is constant, and at least with reference to this observable, the results of our twin subensemble calculation of chapter 6 should apply.

We have measured the integrated autocorrelation times of the plaquette in the first subensemble of each PT simulation and compared these with the autocorrelation time of the same observable in a reference HMC simulation. We found that for large values of the swap acceptance rate (simulations $S1$ and $S4$), the estimated autocorrelation times were in line with the predictions of the twin subensemble model insofar as a gain of a factor of 2 in the plaquette autocorrelation time could be made in subensemble 1 of the PT simulations compared with HMC. For the other simulations, the estimates of the cost ratio were above the predictions of our twin subensemble calculation. However the errors on the estimates of the autocorrelation time are rather large and so the behaviour of the data we have just highlighted may be devoid of statistical significance.

We note that our discussion is of course observable dependent. Our study of the plaquette operator may be criticised as one would not expect the plaquette to couple too strongly to the light modes of the fermion matrix. Hence it could be claimed that our observation stating that the autocorrelation times of operators are reasonably constant over the distances accessible in parameter space is also observable dependent. In particular, as the fermion mass varies inversely with κ, one might expect quite large changes in the quark mass with small changes in κ close to the critical value κ_c. It could then be argued that we should have really investigated the autocorrelation times of operators that are more strongly coupled to the light fermionic modes, as in this case we may arrive at a situation where such an observable may have quite different HMC autocorrelation times for the two PT subensembles and one would not need to rely on the mechanical mixing of configurations between subensembles as the primary means of improving autocorrelation times in PT.

However, in our simulations the plaquette was the only observable that we could measure on every compound step of the PT simulation. We have attempted the measurement of the correlation function of the pseudoscalar meson at zero momentum for autocorrelation analysis in some of our simulations. This exercise turned out to be unfruitful, however, since we only saved our gauge configurations after every 10th compound step of the algorithm, due to lack of sufficient data storage facilities. We were unable to find nonzero autocorrelation times on such a thinned out sample set.

In summary, our results with respect to the plaquette operator in the first subensembles of our PT simulations and in a reference HMC simulation indicate that our PT simulation code is working correctly. Our results for the acceptance rates of the PT swap attempts give us confidence in our understanding of this particular feature of the PT algorithm. To obtain a reasonable swap acceptance rate between two PT subensembles, the parameters of these need to be very close in the space of parameters indicating that most of the improvements in autocorrelation times in PT simulations, compared to other methods, come from the mechanical mixing of configurations between the PT subensembles. We believe that our estimates of the cost ratio $\frac{\mathcal{A}_1}{\mathcal{A}^{\mathrm{H}}}$ go some small way to support this claim.

However, due to the large errors on our autocorrelation data we cannot make any conclusive statements about the cost–effectiveness of the PT algorithm.

Chapter 9

Summary of Results and Conclusions

We now recollect the material presented in this thesis and summarise our results and conclusions. After the introductory material of the first two chapters, the discussion presented split into two parts, these being the work on the finite precision effects and the tuning of the HMC algorithm and the discussion of the Parallel Tempering algorithm. We summarise each in turn.

9.1 Finite Precision and Tuning HMC

In chapters 3 and 4 of this thesis we presented our investigations into the effects of finite precision in Hybrid Monte Carlo simulations. We focused on the subject of reversibility violations in finite precision and on tuning the MD solver residue. In the course of these studies we discovered that the discrete integration of the MD equations of motion via the leap–frog algorithm can become unstable at values of simulation parameters that are close to those used in present day gauge production.

We stress that these effects are only problematic when finite precision is used in our simulations. Unstable Molecular Dynamics in exact arithmetic is still area preserving, reversible and the magnitude of the energy change along a trajectory can in principle be controlled by varying the MD step–size. In finite precision however, instability in the molecular dynamics can result in large values of ΔH with correspondingly small HMC acceptance rates. Our results have shown that this effect may not always be controlled by tuning the MD solver residue or the step size.

In chapter 3 we investigated the reversibility of the UKQCD GHMC code in single precision (32-bit) mode. We identified lack of associativity in the gauge and momentum updates of the second order leap–frog integration scheme as the source of reversibility violations in finite precision. We have demonstrated, through calculations on lattices of small volume, that our implementation of these steps violates reversibility at the level of machine precision per single step. We then examined the global accumulation of reversibility violations in quenched systems and verified that these scale with the system volume in the way we expect. We have discussed briefly the issue of chaos in the MD evolution of HMC simulations and presented the leading Liapunov exponent as an indicator of the growth of rounding errors along an MD trajectory. Finally we have attempted to tune the MD solver residue to minimise our chosen cost metric and found that the MD evolution became unstable at values of the target MD solver residue of around $r = 10^{-5}$.

We computed the leading Liapunov exponents during this tuning study to monitor the accumulation of rounding errors as the solver target residue was decreased. We found that we were unable to obtain reliable estimates of these exponents for values of r when the MD was in the region of transition from being stable to becoming unstable. Once the instability has set in, the values of the Liapunov exponents seemed to stabilise, indicating that the chaos in the MD evolution now came from our particular leap–frog scheme rather than from the underlying continuous time equations of motion for lattice QCD.

In chapter 4 we linked the instability in the MD evolution to a combination of the fermionic forces present in the system and the step–size in the leap–frog update of the momentum fields. We illustrated this connection by examining the case of free field theory and making an assumption that the fermionic modes of lattice QCD could be likened to a collection of harmonic oscillator modes. Such an assumption suggested that the fermionic force in lattice QCD should vary in an inverse manner with the fermion mass in lattice units. We supplied numerical evidence to support this hypothesis, by measuring the fermionic force along MD trajectories for several values of the hopping parameter κ. We managed to fit the fermionic force with an ansatz of the form $F = A(am_{\rm f})^{\alpha}$, where F is the magnitude of fermionic force, $am_{\rm f} = \frac{1}{2}(\frac{1}{\kappa} - \frac{1}{\kappa_c})$ is the fermion mass, κ is the hopping parameter and κ_c is its critical value corresponding to massless quarks. The fits produced negative values of α and predicted correctly the value of κ_c which were known from independent studies, thus supporting our hypothesis.

We then attempted to map the $(\delta\tau, r)$ plane of parameters to identify the region where our current HMC simulations would be safe from instability in the MD. We have found upper limits of $\delta\tau_{\rm max} \approx 10^{-2}$ and $r_{\rm max} \approx 10^{-5}$, although as we have pointed out, these limits are expected to be strongly dependent on $am_{\rm f}$.

We believe our work has several important consequences for future HMC simulations. Firstly we noted that as $\kappa \to \kappa_c$ the fermionic force would diverge implying that $\delta\tau_{\rm max}$ will correspondingly decrease. Simultaneously, we would expect that as the magnitude of the fermionic force grows, the value of $r_{\rm max}$ needed for a sufficiently accurate computation of this force may decrease to such a level, that one may not be able to reach it in single precision calculations, implying that the solves needed to compute the fermionic force may have to be carried out in double precision in future simulations using very light fermions. Finally we concluded that higher order integration schemes would not be able to delay the onset of instability, as, for a given value of $\delta\tau$ these schemes must always make a step of length $\delta\tau' > \delta\tau$ during their update of the fields.

The update scheme of Sexton and Weingarten is not expected to be able to avoid the instability either. This scheme attempts to reduce the discretisation error of the basic leap–frog method through making more gauge force momentum updates than the leap–frog algorithm, for the same number of fermion force momentum updates. According to our hypothesis, the source of the instability lies in the combination of the fermionic forces and the length of the step–size on the fermion force momentum update of the leapfrog scheme, which the method of Sexton and Weingarten does not change.

9.1.1 Furthers studies in this area

We believe our work in the tuning of the HMC algorithm will assist others in the carrying out of dynamical fermion simulations using this algorithm in the future. Such tuning studies in the past have usually been carried out on small lattices and using heavy quarks due to reasons of their high computational cost.

Unfortunately it is not always possible to extrapolate the results of small volume studies to interesting systems with large volumes and light quark masses. We believe we have made a step in the right direction with the work presented in this thesis, by using lattices of the same size as used by our current large scale dynamical fermion simulations and with physical parameters corresponding to our lightest available quarks. However our statistics are very low; for most of our results we have used only 10 configurations for reasons of cost. We would welcome further systematic studies on such large volumes with higher statistics. In particular it would be useful to know at what rate the upper bounds on r and $\delta\tau$ change as the quark masses decrease. It would also be interesting to have an estimate on the upper bound of κ, beyond which solves need to be carried out in double precision to keep the molecular dynamics stable.

Furthermore, it has been suggested [55], that reversibility and area preservation may hold exactly, albeit stochastically, in the presence of rounding errors, providing that these errors are random and are symmetrically distributed around a mean value of 0. The counter argument to this suggestion is that rounding errors are not random, as the rules for rounding are strictly defined by the relevant IEEE standard. In this case, the reversibility violations due to rounding errors will introduce a bias into the simulation results. Such a bias, however, does not constitute a problem as long as it is negligible compared to statistical errors for a given simulation. Therefore, it would be interesting to investigate at what level of precision would such a bias have observable effects on simulation results. Such a project would require a computer implementation of IEEE style floating point numbers and arithmetic with variable precision. This implementation could then be used to simulate computationally cheap theories, such as the harmonic oscillator and lattice free field theory, while varying the precision of the numbers and arithmetic. One could then make more precise statements about the apparent randomness of rounding errors, their symmetry properties and about the size of bias they introduce into simulation results.

9.2 The Parallel Tempering Algorithm

In chapters 5-8 of this thesis we discussed the Parallel Tempering algorithm. Our work in this area concerned itself with two particular aspects:

- understanding the swap acceptance rates of the algorithm and tuning the simulation parameters to maximise these, using the technology of action matching
- investigating the cost–effectiveness of the PT approach when compared to the Hybrid Monte Carlo algorithm.

Chapters 5, 6 and 7 examined both these aspects, and presented the relevant tuning and cost–effectiveness criteria as well as models of PT autocorrelations while chapter 8 presented numerical results from our simulations.

In chapter 5 we presented the PT algorithm, showed that it satisfied the required detailed balance relation and deduced a formula for the swap acceptance rate between two PT subensembles. We then discussed action matching and its relevance to PT simulations. In particular we showed that two of the three action matching conditions, the acceptance rate prescription and the distance norm prescription, bore a direct impact on the tuning of parameters in PT simulations. We illustrated these ideas by deriving expressions for the distance in parameter space between two PT subensembles using the trace log and pseudofermion actions of (5.44) and (5.45) for the case of unimproved Wilson fermions. We showed that the distances in parameter space are always smaller when one uses the trace log action than if one were to use the pseudofermion action, due to a connected term in the distance norm of the latter. This implies that PT simulations using the trace log action in the swap attempt may be able to obtain higher swap acceptance rates than if the pseudofermion action is used for a given $\Delta\kappa$.

In chapter 6 we discussed the question of cost–effectiveness for a PT simulation. Due to the fact that unlike traditional Monte Carlo methods, PT produces gauge configurations in several subensembles we had to identify clearly what is meant by saying that the algorithm is cost effective. In particular we examined two viewpoints. In the first instance we considered the case when all the subensembles of a PT simulation are of interest. In this case we found that even quite small improvements of autocorrelation times in a PT simulation, compared with an equivalent set of HMC simulations could produce a computational gain. The exact criterion was, that the slowest PT subensemble had to have a smaller autocorrelation time than the fastest reference HMC ensemble. This condition is embodied by (6.15).

The other point of view considered was when only one of the PT subensembles was interesting in its own right and the other subensembles served only to reduce autocorrelation times in this subensemble compared to a reference HMC simulation. In this case the cost–effectiveness criterion was that the autocorrelation in the PT subensemble of interest had to be a factor of N smaller than that of the corresponding HMC simulation as is manifest in (6.17) and (6.18).

We then identified mechanisms by which PT can achieve smaller autocorrelation times than HMC. The first of these was the mechanical mixing of configurations between subensembles and the second one was the "mixing of autocorrelation times" between subensembles due to swapping. The former of these was amenable to immediate analysis and we presented the results from considering a simple twin subensemble PT system with equal HMC autocorrelation times.

In chapter 7 we presented our matrix model of PT Autocorrelations. Having discussed the construction of the model and its underlying assumptions we applied it to the case of the magnetisation in twin subensemble harmonic oscillator PT simulations with unequal HMC autocorrelation times and to three subensemble harmonic oscillator PT simulations with equal HMC autocorrelation times and equal swap acceptance rates. We identified the situation under which PT can offer substantial cost gains for the tempered oscillator, and have extrapolated these results with the assumptions that the gross features of the model remain even in the case of other systems. Finally we have exhibited the original system for which PT was devised and showed that this fulfilled the conditions for cost gains to be made with PT.

Our numerical results of chapter 8 seem to confirm our understanding of the swap acceptance steps in PT. However, they indicate that we did not succeed in maximising our swap acceptance rates using the technology of action matching. We believe the fault lies with us for using an inappropriate tuning condition, rather than with the idea of tuning through action matching. Our measurements of the integrated autocorrelation times for the plaquette operator are not inconsistent with the results of the twin ensemble model of chapter 6 (or 7), however the errors on these estimates are too large to draw definite conclusions.

The chief conclusion from the material presented is that the swap acceptance rate between pairs of subensembles in a PT simulation is expected to decrease exponentially with the system volume. Hence we believe that the point of view that all the PT subensembles may be of intrinsic interest is not realistic at our chosen parameter values. In a large volume simulation the distances accessible in parameter space are expected to be so small that the only useful purpose of the extra PT subensembles is to assist in the decorrelation of observables in the interesting subensemble. If the autocorrelation time of the observable of interest is a smooth, slowly varying function of the parameters, we would expect that the HMC autocorrelation times of the observables in all the PT subensembles should be almost equal for such a large volume simulation. In this case we would expect that the improvement of PT autocorrelation times above their HMC ones should come mainly from mechanical mixing of configurations between subensembles. For a high swap acceptance rate we expect at most a factor of N improvement over the corresponding HMC autocorrelation times which implies that PT can be competitive with HMC but in terms of computational cost no great gains are expected from PT.

On the other hand, if the gross features of our matrix model are applicable beyond the harmonic oscillator system, then one may be able to make substantial gains using PT in the situation where the integrated autocorrelation times of an observable undergo a dramatic increase (decrease) within a small variation of the simulation parameters. This appears to be the situation in the case of the Edwards–Anderson spin model, for which the method was originally devised. In lattice QCD a similar situation may be met near the finite temperature phase transition, where the autocorrelation time of the Polyakov loop is expected to change dramatically between the confined and deconfined phases.

It is possible that in the case of observables that are strongly coupled to light fermion modes, such as the topological charge, one is in a situation where the autocorrelation times in two subensembles are wildly different from each other. We believe that in this case one can gain reductions in the autocorrelation times of the desired observable in a PT subensemble over and above just the mechanical mixing of configurations as we have shown in chapter 7.

We welcome the recent results of [56] presented at the Lattice 99 symposium in Pisa and of [57] presented subsequently at the chiral fermion workshop at Dubna. These contributions discussing the same study indicated impressive improvements in the tunnelling rates of topological charge in PT simulations compared to HMC ones. However these articles did not present details as to the computational cost paid for their impressive tunnelling rates. Furthermore we believe that the tunnelling rate is not necessarily a sensible cost metric. One can imagine having two subensembles each with infinitely long untempered autocorrelation times. In this case the overall algorithm is clearly not ergodic and the tunnelling rates are bad. However one can mix configurations between the subensembles through swapping. In this case the tunneling rates will be good, however the overall algorithm will still not be ergodic. However we note that both these contributions acknowledge the problem of the expected decrease of swap acceptance rates with increasing system volume.

In summary we believe we have a good understanding of the workings of the parallel tempering algorithm. The question of cost effectiveness for lattice QCD however remains open.

9.2.1 Prospects and outlook

To answer the remaining question about the cost effectiveness of the PT approach for lattice QCD, improvements in simulation statistics of roughly a factor of 100 compared to the ones presented here are required, in order to reduce the relative errors on estimates of autocorrelation times and cost ratios to O(10%)[1]. To ascertain whether or not parallel tempering offers a computational gain with respect to HMC, such simulations should also preferably investigate observables that are strongly coupled to light fermionic degrees of freedom such as the topological charge, or the low lying eigenvalues of the fermion matrix. Hence by necessity these simulations would need to be carried out at light quark masses. Such a project would require computational resources that at the current time could be better employed by carrying out HMC simulations at interesting parameter values or by testing innovative formulations of lattice theories such as the ones recently proposed to allow the simulations of chiral fermions [58, 59], rather than by carrying out precision tests of the PT algorithm.

However useful results could be obtained by investigating the PT algorithm in simpler, computationally less demanding theories than lattice QCD. We have already made reference to free field theory results in which the cost of PT could be investigated in systems where different subensembles had different HMC autocorrelation times. We suggest also PT investigations of lattice QED in two dimensions. This theory is computationally much less expensive than lattice QCD in four dimensions, to the extent that the generation of configuration ensembles with very high statistics is within the power of workstations. Yet QED in two dimensions possesses notions of topological sectors analogous to QCD. One could investigate whether HMC had difficulties in tunnelling between topological sectors in this theory for light fermion masses. High statistics PT studies could also be carried out allowing precise determination of autocorrelation times, thus giving reasonable indications to the cost effectiveness of the algorithm.

At the outset of our own studies, we considered such an investigation instead of the one presented here, but the availability of the UKQCD software code base

[1]This estimate assumes that the errors on the autocorrelation times decrease as $\frac{1}{N}$

tipped the balance in favour of lattice QCD trials. With hindsight, we should perhaps have given more thought to investigating lattice QED in two dimensions.

The prospects for action matching are bright. In particular the approach of simulating lattice QCD through some approximate action and executing accept/reject steps to accept the resulting configurations into a QCD configuration ensemble is of great interest. Such simulation schemes, where the short range effects of fermions are dealt with explicitly through computation of some low lying eigenvalues of the fermion matrix and the long range effects are approximated via loop expansions are at this moment receiving critical testing [60]. Recent progress in the estimation of the fermionic term in the trace log action using Lánczos methods [61] makes prospects for such an approach even brighter.

9.3 Conclusion

In summary, the material presented in this thesis did but touch upon the tip of the algorithmic iceberg of efficient lattice QCD simulations. We hope to uncover more and more of this fascinating subject in forthcoming years[2].

[2]Hopefully without becoming shipwrecked in the process

Appendix A

Numerical Estimation of Autocorrelation Times

In practice, one encounters difficulties in estimating $\mathcal{A}$ as estimators of this quantity tend to be very noisy. We now describe the method of [53] which we used in the determination of our autocorrelation times.

We are interested in estimating the integrated autocorrelation of an observable $\mathcal{O}$. This is defined as

$$\mathcal{A} = \sum_{t=1}^{\infty} C(t) \tag{A.1}$$

for an infinite sequence of samples $\mathcal{O}(i)$. The autocorrelation function $C(t)$ is in turn defined as

$$C(t) = \frac{1}{C(0)} \left\langle (\mathcal{O}(|t| + t_0) - \langle \mathcal{O} \rangle)(\mathcal{O}(t_0) - \langle \mathcal{O} \rangle) \right\rangle_{t_0} , \tag{A.2}$$

where the expectation value is over all pairs $\mathcal{O}(|t| + t_0)$ and $\mathcal{O}(t_0)$. We note that $C(t)$ is normalised and even.

In the presence of a finite number of samples, say N, we must resort to calculating estimators for these quantities. A suitable estimator for $C(t)$ is

$$\hat{C}(t) = \frac{1}{\hat{C}(0)} \frac{1}{(N-t)} \sum_{i=0}^{N-t} \left(\mathcal{O}(i+t)\mathcal{O}(i) - \bar{\mathcal{O}}^2 \right) , \tag{A.3}$$

where $\bar{\mathcal{O}}$ is the arithmetic mean,

$$\bar{\mathcal{O}} = \frac{1}{N}\sum_{t=0}^{N}\mathcal{O}(t) \tag{A.4}$$

which is an estimator for $\langle\mathcal{O}\rangle$.

The natural estimator for $\mathcal{A}$ would be

$$\hat{\mathcal{A}} = \sum_{t=1}^{N-1}\hat{C}(t) \tag{A.5}$$

however as pointed out in [53, 62] this is not a good estimator for $\mathcal{A}$. It is claimed that this is because for $t \gg \mathcal{A}$, the sample autocorrelations $\hat{C}(t)$ have a poor signal to noise ratio. Furthermore it has been shown in [62] that the variance of this estimator does not go to zero as $t \to \infty$. Instead the authors of [53] suggest that the sum be cut-off at some suitable window size Λ so that

$$\hat{\mathcal{A}}(\Lambda) = \sum_{t=1}^{\Lambda}\hat{C}(t) \ . \tag{A.6}$$

The variance of this estimator is [53]

$$\sigma^2(\hat{\mathcal{A}}(\Lambda)) \approx \frac{2(2\Lambda+1)}{N}\hat{\mathcal{A}}(\Lambda)^2 \tag{A.7}$$

for $\mathcal{A} \ll \Lambda \ll N$ which was the formula we used to estimate errors on our measurements.

However in this case, the cut-off Λ introduces a bias into the estimate of

$$\mathrm{bias}(\hat{\mathcal{A}}(\Lambda)) \approx \sum_{t>\Lambda}\hat{C}(t) \tag{A.8}$$

As the decay of $C(t)$ is expected to be exponential one would presume that if $\Lambda \gg \mathcal{A}$ the bias in the results would be small. However, Λ should be small enough to filter out the "noise" from the samples of $\hat{C}(t)$ at large t. Hence one requires a choice of Λ such that $\mathcal{A} \ll \Lambda \ll N$, which is a trade off between bias in $\hat{\mathcal{A}}(\Lambda)$ and the noise. In particular, in [53] it was suggested that Λ should be chosen to be the smallest integer so that $\Lambda \geq c\mathcal{A}(\Lambda)$ for some positive coefficient c. Several possible values for c were then suggested from considerations about the exponential fall off of $C(t)$.

We have found in the course of our analyses, that using the different values of c in [53] gave differing estimates for $\mathcal{A}$. Hence we have opted for a "sliding" window scheme, where we varied the constant c until our estimates of $\mathcal{A}$ plateauxed. We then chose the smallest value of c that gave us an estimate of $\mathcal{A}$ in the plateau region.

Appendix B

Statistical Analysis of Data

Once a simulation has been carried out and the gauge configurations and interesting observables are available, the final phase of a simulation involves the analysis of the data. This kind of analysis tends to involve

- the calculation of ensemble averages with some sensible estimate of errors, discussed in B.0.1
- the fitting of results to some known theoretical model, in order to extract hadron masses, decay constants and so forth, detailed in B.0.4

We note that there can be three major kinds of error one is faced with. They are respectively statistical, systematic and numerical errors. The discussion in B.0.1 refers exclusively to the treatment of statistical errors. Systematic errors in a simulation can be estimated mostly through repeating the simulation in a different manner from the one which has been employed originally, or by comparing the result of a given simulation to those obtained by other independent simulations. Numerical errors are mostly of concern in particular parts of a simulation such as carrying out a molecular dynamics step in the QCD phase space or in the convergence of a solver when calculating hadron correlation functions.

For the most part though, unless stated otherwise, discussions of errors hereafter will refer to statistical errors, which we expect to be the predominant ones in Monte Carlo simulations.

B.0.1 Error Estimation and sub-sampling methods

When a sample of data is not normally distributed, the usual formula for the the sample variance does not necessarily give the true variance of the data. In this case, the usual formula for the standard deviation of the means of an observable is not guaranteed to give the 68% confidence limit that one generally attributes to σ.

Re-sampling methods attempt to simulate the distribution of mean for a sample of data. The general technique is to create a number of re-sampled datasets from the original one and compute the quantities of interest on each one. The distribution of the quantities can then be studied.

The two most popular re-sampling methods are the Jackknife and the Bootstrap methods. In the next two sections we describe both, from an operational point of view. The reader is referred to [63] for detailed description of these methods.

B.0.2 The Jackknife Method

The Jackknife Method is a sub-sampling method in which the subsample datasets are created by removing (cutting – jack-knifing) items from the original one.

Let us consider a dataset $\mathcal{D}$ having N elements d_i where $i = 1 \dots N$. The i-th Jackknife dataset $\mathcal{D}_i$ is defined to contain all the elements d_j of $\mathcal{D}$ with $j \neq i$. Let $\mathcal{O}$ be the value of some interesting quantity (such as an arithmetic mean) computed from dataset $\mathcal{D}$ and $\mathcal{O}_i$ be the the quantity of interest computed from dataset $\mathcal{D}_i$. The jackknife error σ^J_P is defined as

$$\sigma^J_{\mathcal{O}} = \sqrt{\frac{N-1}{N}\sum_1^N \left(\mathcal{O}_i - \bar{\mathcal{O}}\right)^2} \tag{B.1}$$

with $\bar{\mathcal{O}}$ begin the arithmetic mean of the $\mathcal{O}_i$

$$\bar{\mathcal{O}} = \frac{1}{N}\sum_i^N \mathcal{O}_i \ . \tag{B.2}$$

The measured quantity is then quoted as $\mathcal{O} \pm \sigma^J_{\mathcal{O}}$.

B.0.3 The Bootstrap Method

Like the Jackknife method, the bootstrap method is also a sub-sampling method. Here the subsample datasets are created by picking data items from the original one with repetition.

More specifically, if the original dataset is $\mathcal{D}$ with N data items d_i as before, one can create a *bootstrap dataset* $\mathcal{D}^B$ by picking randomly with repetition N elements from $\mathcal{D}$.

Let us suppose one is interested in estimating an error on some quantity $\mathcal{O}$ computed from dataset $\mathcal{D}$. The bootstrap method proceeds by creating N_B bootstrap datasets $\{\mathcal{D}^B_i,\ i = 1 \ldots N_B\}$. The quantity of interest is computed from each bootstrap dataset resulting in set of *bootstrap quantities* $\{\mathcal{O}_i,\ i = 1 \ldots N_B\}$, where $\mathcal{O}_i$ is the quantity measured on bootstrap dataset $\mathcal{D}^B_i$.

Let us suppose that the desired confidence level in the error estimates is C. The errors are computed by finding the upper and lower quantities $\mathcal{O}_U$ and $\mathcal{O}_L$ respectively such that

$$\frac{\#(\mathcal{O}_i < \mathcal{O}_L)}{N} = \frac{1-C}{2} \text{ and } \frac{\#(\mathcal{O}_i > \mathcal{O}_U)}{N} = \frac{1+C}{2} \tag{B.3}$$

where $\#(\text{Condition on } \mathcal{O}_i)$ is the number of bootstrap quantities $\mathcal{O}_i$ fulfilling the condition. This ensures that the fraction of bootstrap quantities within the upper and lower limits is approximately C of the total number of bootstrap quantities.

The result and the bootstrap errors can be quoted as $\mathcal{O}^{+\epsilon_U}_{-\epsilon_L}$ with

$$\epsilon_U = \mathcal{O}_U - \mathcal{O} \tag{B.4}$$

$$\epsilon_L = \mathcal{O} - \mathcal{O}_L \tag{B.5}$$

In this thesis, $C = 0.68$, corresponding to the standard 68% confidence limits.

Suppose one is interested in the errors on some derived quantity $\mathcal{O}^D$ which is related to other quantities of interest $\mathcal{O}^j$ computed from a dataset $\mathcal{D}$ according to the relationship

$$\mathcal{O}^D = f(\mathcal{O}^j) \tag{B.6}$$

The bootstrap procedure is to create N_B bootstrap datasets as before. On each dataset i one can measure the observables $\mathcal{O}_i^j$ and evaluate the bootstrap quantity

$$\mathcal{O}_i^D = f(\mathcal{O}_i^j) \tag{B.7}$$

resulting in the set of bootstrap quantities $\{\mathcal{O}_i^D\}$. One then proceeds to calculate error bounds as before.

B.0.4 Fitting Parameters to Data

Consider the problem of fitting a model to data. Generally we have available N measurements of a quantity $y_n(x_i), n = 0...N$ at a set of independent variables $\{x_i\}$. We want to fit some function d of the *average* of the $y_n(x_i)$ to some *model function* of the independent variable $\phi(x, \mathbf{a})$, where $\mathbf{a}$ is a set of parameters which is yet to be determined. Let us for this purpose denote by y_i the average of the data at $x = x_i$,

$$y_i = \frac{1}{N}\sum_{n=1}^{N} y_n(x_i) \tag{B.8}$$

and by d_i the result of applying the function d to y_i, in other words $d_i = d(y_i)$.

We assume that the y_n came originally from a probability distribution with probability density function $f(y, \hat{\mathbf{a}})$. Here the $\hat{\mathbf{a}}$ are true parameters that we are trying to find. We define the *likelihood function* $\mathcal{L}(y, \mathbf{a})$ for some parameter set $\mathbf{a}$ as the joint probability that the data items y_n were drawn from the distribution $f(y_n, \mathbf{a})$ so that

$$\mathcal{L}(\mathbf{a}) = \prod_n f(y_n, \mathbf{a}) \tag{B.9}$$

The method of maximum likelihood fitting assumes that $\mathcal{L}$ will be maximised, when the parameter set $\mathbf{a}$ is the same as the true parameter set $\hat{\mathbf{a}}$. It transpires that maximising $\mathcal{L}$ is equivalent to minimising the χ^2 statistic.

The χ^2 statistic of the data is defined as

$$\chi^2(\mathbf{a}) = \sum_{i,j\in\{x_i\}} [d_i - \phi(i,\mathbf{a})]\, C_{ij}^{-1}\, [d_j - \phi(j,\mathbf{a})] \ . \tag{B.10}$$

The matrix C_{ij} is the data correlation matrix

$$C_{ij} = \frac{\mathrm{Cov}(d_i, d_j)}{\sqrt{\mathrm{Cov}(d_i, d_i)\mathrm{Cov}(d_j, d_j)}} \tag{B.11}$$

and

$$\mathrm{Cov}(d_i, d_j) = \langle (d_i - \langle d_i \rangle)(d_j - \langle d_j \rangle) \rangle \tag{B.12}$$

is the covariance of d_i with d_j.

Since in practice there is only one value of d_i available from the dataset $\{y_n(x_i)\}$, the question arises as to how one can compute the elements of the covariance matrix. Usually this problem is solved by employing either the jackknife or the bootstrap methods to subsample the dataset, and the expectation values can be evaluated on the subsamples. In particular, jackknife covariances are given by

$$\mathrm{Cov}^J(d_i, d_j) = \frac{N-1}{N} \sum_k \left(d_i^k - \bar{d}_i\right)\left(d_j^k - \bar{d}_j\right) \tag{B.13}$$

where the index k refers to the k-th jackknife sample and $\bar{d}_i$ refers to the arithmetic mean of d_i over the jackknife samples.

We note in passing that if the data are independent then the data correlation matrix is the unit matrix, and the problem of minimising the χ^2 statistic reduces to the problem of least–squares fitting.

B.0.5 Errors on the fitted parameters

Depending on the method one uses to perform the fit, one may or may not have available at the end of the χ^2 minimisation process the *parameter covariance matrix* of the $\mathbf{a}$. If the parameters $\mathbf{a}$ are uncorrelated, than the diagonal elements of this matrix can be used as estimates for the variance of the parameter.

In a grand scheme of a bootstrap or a jackknife analysis, one can perform the fitting process on each jackknife, or bootstrap sample (possibly using a secondary bootstrap or jackknife to evaluate the data correlation matrix). The bootstrap and jackknife errors can then be found as described earlier in section B.0.1.

B.0.6 Goodness of Fit

Some idea of the goodness of fit can be obtained from the χ^2 probability $Q(\chi^2, \nu)$ function, where ν is the number of degrees of freedom, that is the number of data over which the fit is performed minus the number of parameters. The probability Q itself is defined as $Q(\chi^2, \nu) = 1 - P(\chi^2, \nu)$ and P is the *incomplete Gamma function.*

The Q value is the probability that the observed χ^2 should exceed the value of χ^2 used as an input to Q by chance, even for a correct fit model for a given dataset. Thus if $Q(\chi^2, \nu)$ is very small for a value of ν and a value of χ^2 which is a minimum obtained during a fit, it seems to indicate that an observed χ^2 for the dataset and the model is unlikely to be larger than the input one, in other words, the χ^2 that was input is unlikely to be a true minimum.

Thus the Q value can be used as a quantitative indicator of the goodness of the fit [64]. Here however we make some reservations:

- In [64] it was claimed that the Q-value assumes that the measurement errors on the d_i in the fit are normally distributed. This is not necessarily the case in lattice QCD simulations.

- Several examples and circumstances are mentioned in [64] as to why a small Q value may still be acceptable and why a Q value close to unity may still not be acceptable.

- In lattice QCD, the fit-model is often predetermined already for the observable. It is known already, how certain correlation functions ought to decay for example. Thus we are not always in the position suggested in [64] where the choice of a fitting model may need to be determined by comparing its goodness of fit with those of others, since the model is already known.

Having stated all the above, it must be added that there are times when different fit models are applied to the same data, such as multiple exponential fit functions instead of single exponential functions to control the effects from higher energy states in some correlation functions. However in such cases it is perhaps more important to look for consistency and stability in the fit results than to hawkishly watch the Q-value. At the same time a reasonable Q-value may add an extra feeling of comfort in the fit results.

Appendix C

Some Proofs

C.1 Proof of $\Delta T \approx \Delta \mathcal{T}_i$

We now proceed to prove an earlier claim, that

$$\Delta \mathcal{T}_i \approx \Delta T \tag{C.1}$$

where

$$\begin{aligned} Q_i &= Q(\kappa_i), \\ \Delta T &= -\mathrm{Tr}\ \ln Q_1^{-1} + \mathrm{Tr}\ \ln Q_2^{-1} \\ &= \mathrm{Tr}\ \ln Q_1 - \mathrm{Tr}\ \ln Q_2 \qquad \text{(C.2)} \\ \Delta \mathcal{T}_i &= \mathrm{Tr}\ (Q_i^{-1}(Q_1 - Q_2)) \qquad \text{(C.3)} \end{aligned}$$

We will prove this statement separately for $i = 1$ and $i = 2$ as the the matrix $\Delta \mathcal{T}_i$ has different forms in each case:

$$\Delta \mathcal{T}_i = \begin{cases} 1 - Q_1^{-1} Q_2, & \text{for } i = 1 \\ Q_2^{-1} Q_1 - 1 & \text{for } i = 2 \end{cases} . \tag{C.4}$$

C.1.1 Case $i = 1$

We proceed from the definition of ΔT:

$$\Delta T = \mathrm{Tr}\ \ln Q_1 - \mathrm{Tr}\ \ln Q_2$$

$$\begin{aligned} &= -\mathrm{Tr}\ (\ln Q_2 - \ln Q_1) \\ &= -\mathrm{Tr}\ \ln Q_1^{-1} Q_2 \end{aligned} \tag{C.5}$$

We re-write the argument of the logarithm as

$$\Delta T = -\mathrm{Tr}\ \ln\left(1 - (1 - Q_1^{-1}Q_2)\right) \tag{C.6}$$

Now we write the Taylor series for the logarithm as

$$\ln(1-x) = -x + O(x^2)\ . \tag{C.7}$$

We finish the proof by setting

$$x = 1 - Q_1^{-1}Q_2 = Q_1^{-1}(Q_1 - Q_2) \tag{C.8}$$

which gives the final result,

$$\begin{aligned} \Delta T &= -\mathrm{Tr}\ \ln(1-x) \\ &= -\mathrm{Tr}\ (-x + O(x^2)) \\ &= \mathrm{Tr}\ (x - O(x^2)) \\ &= \mathrm{Tr}\ \left(Q_1^{-1}(Q_1 - Q_2)\right) + \mathrm{h.o.t} \qquad (C.9) \\ &\approx \Delta\mathcal{T}_1 \qquad (C.10) \end{aligned}$$

as required.

C.1.2 Case $i = 2$

We proceed from the definition of ΔT :

$$\begin{aligned} \Delta T &= \mathrm{Tr}\ \ln Q_1 - \mathrm{Tr}\ \ln Q_2 \\ &= \mathrm{Tr}\ (\ln Q_1 + \ln Q_2^{-1}) \\ &= \mathrm{Tr}\ \ln Q_2^{-1} Q_1 \end{aligned} \tag{C.11}$$

We then rewrite the argument of the logarithm as

$$\Delta T = \mathrm{Tr}\ \ln(1 + Q_2^{-1}Q_1 - 1) \tag{C.12}$$

To finish the proof, we use again the Taylor expansion of the logarithm,

$$\ln(1+x) = x - \frac{x^2}{2} + O(x^3) \; . \tag{C.13}$$

We set

$$x = Q_2^{-1} Q_1 - 1 = Q_2^{-1}(Q_1 - Q_2) \tag{C.14}$$

and hence, to first order in x,

$$\begin{aligned} \Delta T &= \mathrm{Tr}\ \ln(1+x) \\ &= \mathrm{Tr}\ \Big(x + O(x^2)\Big) \\ &= \mathrm{Tr}\ (Q_2^{-1} Q_1 - 1) + \mathrm{h.o.t} \\ &= \mathrm{Tr}\ Q_2^{-1}(Q_1 - Q_2) + \mathrm{h.o.t} && \text{(C.15)} \\ &\approx \Delta \mathcal{T}_2 && \text{(C.16)} \end{aligned}$$

Bibliography

[1] R. Burkhalter, Nucl. Phys. B (Proc. Suppl.) **73**, 3 (1999).

[2] R. D. Kenway, Nucl. Phys. B (Proc. Suppl.) **73**, 16 (1999).

[3] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, Phys. Lett. **B195**, 216 (1987).

[4] K. Hukushima and K. Nemoto, Exchange Monte Carlo Method and Application to Spin Glass Simulation, *cond–mat/9512035*.

[5] A. C. Irving and J. C. Sexton, Phys. Rev. **D55**, 5456 (1997).

[6] A. C. Irving *et al.*, Phys. Rev. **D58**, 114504 (1998).

[7] H. J. Rothe, *Lattice Gauge Theories* (World Scientific, Singapore, 1992).

[8] I. Montvay and G. Münster, *Quantum Fields on a Lattice* (Cambridge University Press, Cambridge, 1994).

[9] R. Gupta, Introduction to Lattice QCD, Lectures given at the LXVIII Les Houches Summer School, 28 .July–5. September 1987, `hep-lat/9807028`.

[10] N. B. Nielsen and M. Ninomiya, Phys. Lett. **B105**, 219 (1981).

[11] B. Sheikholeslami and R. Wohlert, Nucl. Phys. **B259**, 572 (1985).

[12] K. Jansen and R. Sommer, Nucl. Phys. **B530**, 185 (1998).

[13] Z. Sroczynski, Ph.D. thesis, University of Edinburgh, 1998.

[14] A. D. Kennedy, J. Kuti, S. Meyer, and B. J. Pendleton, J. Comp. Phys. **64(1)**, 133 (1986).

[15] N. Metropolis *et al.*, J. Chem. Phys. **21**, 1087 (1953).

[16] J. Garden, Ph.D. thesis, University of Edinburgh, In preparation.

[17] C. M. Maynard, Ph.D. thesis, University of Edinburgh, 1998.

[18] C.-J. D. Lin, Ph.D. thesis, University of Edinburgh, 1998.

[19] G. Bali and K. Schilling, Phys. Rev. **D47**, 661 (1993).

[20] C. R. Allton *et al.*, Phys. Rev. **D60**, 034507 (1999).

[21] R. Sommer, Nucl. Phys. **B411**, 839 (1994).

[22] M. Campostrini and P. Rossi, Nucl. Phys. **B329**, 753 (1990).

[23] M. Creutz and A. Gocksch, Phys. Rev. Lett. **63-1**, 9 (1989).

[24] J. C. Sexton and D. H. Weingarten, Nucl. Phys. **B380**, 665 (1992).

[25] S. Gupta, A. Irbäck, F. Karsch, and B. Petersson, Phys. Lett. **B242**, 437 (1990).

[26] M. Creutz, Phys. Rev. **D38**, 1228 (1988).

[27] A. D. Kennedy and P. Rossi, Nucl. Phys. **B327**, 782 (1989).

[28] A. D. Kennedy, Exact exponential mapping for su(3)., Private Communication, 1990.

[29] C. Moler and C. van Loan, SIAM Review **20**, 801 (1978).

[30] A. D. Kennedy, Parallel Computing **25**, 1311 (1999).

[31] Y. Saad, *Iterative Methods for Sparse Linear Systems* (PWS–Publishing, U.S.A, 1996), No. ISBN 0-534-94776-X.

[32] H. van der Vorst, Lecture Notes on Iterative Methods, available at `http://www.math.ruu.nl/people/vorst/lecture.html`.

[33] M. R. Hestenes and E. Stiefel, J. Res. Nat. Bur. Standards **49**, 409 (1952).

[34] H. van der Vorst, SIAM J. Sc. Stat. Comp. **12**, 409 (1992).

[35] S. M. Pickles, Ph.D. thesis, University of Edinburgh, 1998.

[36] M. Göckeler *et al.*, Nucl. Phys. Proc. Suppl. **B53**, 312 (1997).

[37] I. S. Association, IEEE Standard for Binary Floating-Point Arithmetic, Available from IEEE Standards Association, 1985.

[38] Z. Sroczynski, S. M. Pickles, and S. P. Booth, Nucl. Phys. B (Proc. Suppl.) **63A-C**, 949 (1998).

[39] I. Horváth, A. D. Kennedy, and R. G. Edwards, Nucl. Phys. **B484**, 375 (1997).

[40] C. Liu, A. Jaster, and K. Jansen, Nucl. Phys. **B524**, 603 (1998).

[41] R. Gupta, G. W. Kilcup, and R. S. Sharpe, Phys. Rev. **D38-4**, 1278 (1984).

[42] B. Allés, G. Boyd, M. D'Elia, and A. D. Giacomo, Nucl. Phys. Proc. Suppl. **63**, 239 (1998).

[43] B. Allés *et al.*, Phys. Lett. **B389**, 107 (1996).

[44] B. Allés, G. Boyd, M. D'Elia, and A. D. Giacomo, New results on topological susceptivility in SU(3) gauge theory, Talk presented at the 28th International Conference in High Energy Physics in Warsaw, *hep-lat/9610009.*

[45] G. Boyd *et al.*, Nucl. Phys. Proc. Suppl **53**, 544 (1997).

[46] B. Allés *et al.*, Phys. Rev. **D58**, 071503 (1998).

[47] T. Lippert *et al.*, Nucl. Phys. Proc. Suppl. **73**, 521 (1999).

[48] E. Marinari and G. Parisi, Europhys. Lett. **19**, 451 (1992).

[49] G. Boyd, Nucl. Phys. Proc. Suppl. **60A**, 341 (1998).

[50] E. Marinari, Optimized Monte Carlo Methods, Lectures given at the 1996 Budapest Summer School on Monte Carlo Methods, *cond–mat/9612010.*

[51] J. C.Sexton and D. H. Weingarten, Phys. Rev. **D55**, 4025 (1997).

[52] I. Horváth and A. D. Kennedy, Nucl. Phys. **B510**, 367 (1998).

[53] N. Madras and A. D. Sokal, J. Stat. Phys. **50**, 109 (1998).

[54] B. Joó *et al.*, Phys. Rev. **D59**, 114501 (1999).

[55] J. C.Sexton, A. C. Irving, B. Joó, and B. Pendleton, Reversibility of Hybrid Monte Carlo, In preparation.

[56] E.-M. Ilgenfritz, W. Kerler, and H. Stüben, Decorrelation of the topological charge in tempered Hybrid Monte Carlo simulations of QCD, To appear in the Nucl. Phys. Proc. Suppl. in the Proceedings of Lattice 99, the XVII-th International Symposium on Lattice Field Theory, held at Pisa, 1999, *hep-lat/9908022*.

[57] H. Stüben, Decorrelation of the topological charge in tempered simulations of full QCD, writeup of talk presented at the workshop "Lattice Fermions and Structure of the vacuum", held in Dubna, Russia, Oct. 5-9, 1999, *hep-lat/9912047*.

[58] H. Neuberger, Phys. Lett. **B417**, 141 (1998).

[59] F. Niedermayer, Nucl. Phys. Proc. Suppl. **73**, 105 (1999).

[60] A. Duncan, E. Eichten, and H. Thacker, Phys. Rev. **D59**, 014505 (1999).

[61] E. Cahill, A. C. Irving, C. Johnston, and J. C. Sexton, Numerical Stability of Lánczos Methods, To appear in Nucl. Phys. Proc. Suppl. in the proceedings of Lattice 99 - XVII International Symposium on Lattice Field theory, *hep-lat/9909131*.

[62] M. B. Priestley, *Spectral Analysis and Time Series* (Academic Press Inc., London, 1981).

[63] B. Efron and R. J. Tibshirani, *An Introduction to the Bootstrap* (Chapman and Hall, London, 1993).

[64] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C* (Cambridge University Press, Cambridge, 1992).