Fermion Low Modes in Lattice QCD: Topology, the η' Mass and Algorithm Development

Duo Guo

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
under the Executive Committee
of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2021

Abstract

Fermion Low Modes in Lattice QCD: Topology, the η' Mass and Algorithm Development

Duo Guo

Lattice gauge theory is an important approach to understanding quantum chromodynamics (QCD) due to the large coupling constant in the theory at low energy. In this thesis, we report our study of the topological properties of the gauge fields and we calculate m_η and $m_{\eta'}$ which are related to the topology of the gauge fields. We also develop two algorithms to speed up the inversion of the Dirac equation which is computationally demanding in lattice QCD calculations.

The topology of lattice gauge fields is important but difficult to study because of the large local fluctuations of the gauge fields. In chapter 2, we probe the topological properties of the gauge fields through the measurement of closed quark loops, field strength and low-lying eigenvectors of the Shamir domain wall operator. The closed quark loops suggest the slow evolution of topological modes during the generation of QCD configurations. The chirality of the low-lying eigenvectors is studied and the lattice eigenvectors are compared to the eigenvectors in the continuous theory. The topological charges are calculated from the eigenvectors and the results agree with the topological charges calculated from the smoothed gauge fields. The fermion correlators are also obtained from the eigenvectors.

The non-trivial topological properties of QCD gauge fields are important to the mass of the η and η', m_η and $m_{\eta'}$. Lattice QCD is an area where m_η and $m_{\eta'}$ can be calculated by using gauge fields that are sampled over different topological sectors. We calculate m_η and $m_{\eta'}$ in chapter 3 by including the fermion correlators and the topological charge density correlators. The errors of m_η and $m_{\eta'}$ are reduced to the percent level and the mixing angle between the octet, singlet states in the SU(3) limit and the physical eigenstates is calculated.

An algorithm that reduces communication and increases the usage of the local computational power is developed in chapter 4. The algorithm uses the multisplitting algorithm as a preconditioner in the preconditioned conjugate gradient method. It speeds up the inversion of the Dirac equation during the evolution phase.

In chapter 5, we utilize two lattices, called the coarse lattice and the fine lattice, that lie on the renormalization group trajectory and have different lattice spacings. We find that the low-mode space of the coarse lattice corresponds to the low-mode space of the fine lattice. Because of the correspondence, the coarse lattice can be used to solve the low modes of the fine lattice. The coarse lattice is used in the restart algorithm and the preconditioned conjugate gradient algorithm where the latter is called the renormalization group based preconditioned conjugate gradient algorithm (RGPCG). By using the near-null vectors as the filter, RGPCG could reduce the operations of the matrix multiplications on the fine lattice by 33% to 44% for the inversion of Dirac equation. The algorithm works better than the conjugate gradient algorithm when multiple equations are solved.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I am very grateful to my advisor, Robert Mawhinney for his valuable suggestions. The insightful and rigorous discussions with him are fundamental to the completion of the thesis.

I want to thank other excellent mentors including Norman Christ, Chulwoo Jung, Masaaki Tomii, Christ Kelly and Peter Boyle. I am also indebted to my fellow graduate students, including David Murphy, Jiqun Tu, Bigeng Wang, Tianle Wang and Yidi Zhao. I feel fortunate to work in such an excellent collaboration and to receive so many supports.

Finally I would like to express my gratitude to my family and my girlfriend for their love, support and companion throughout the years.

Chapter 1: Introduction

In this section we give a brief introduction about quantum chromodynamics (QCD) and lattice QCD. We will start from QCD and then discuss how QCD is discretized as lattice QCD. We will also describe the major steps by which the physical quantities are obtained through the framework of lattice QCD: the gauge field generation, the measurements of the correlation functions and the fitting of the physical quantities.

1.1 Quantum chromodynamics

Quantum chromodynamics (QCD) is believed to be the correct theory of the strong nuclear interaction. The theory describes the interaction of the quarks and gluons and the formation of mesons and hadrons. Technically, QCD is a non-Abelian SU(3) gauge theory. The Lagrangian is given by:

$$\mathcal{L}_{QCD} = -\frac{1}{4}F^a_{\mu\nu}F^{\mu\nu}_a + \sum_f \bar{\psi}_f(i\gamma_\mu D^\mu - m_f)\psi_f \tag{1.1}$$

where $F^a_{\mu\nu}$ is the gluon field strength tensor calculated from the gluon fields A^a_μ:

$$F^a_{\mu\nu} = \partial_\mu A^a_\nu - \partial_\nu A^a_\mu + g f^{abc} A^b_\mu A^c_\nu \tag{1.2}$$

A^a_μ are the spin-1 gluon fields which live in the adjoint representation of the SU(3) gauge group, indexed by $a, b, c...$. The strong coupling constant is g and f^{abc} are the SU(3) structure constants constrained by the Lie algebra of $\mathfrak{su}(3)$. Specifically, given a basis T^a of the Lie algebra, we have:

$$[T^a, T^b] = i f^{abc} T^c \tag{1.3}$$

In summary, the first term of the Lagrangian is about the interaction between the gluons.

The second term is about the interaction between quarks and the the interaction between quarks and gluons. ψ_f are spin-$\frac{1}{2}$ quark fields. They live in the fundamental representation of the SU(3) gauge group. As fermion fields, ψ_f satisfy Grassman algebra. Different ψ_f's anti-commute with each other. There are different flavors of quark fields indexed by $f, g, \ldots$. $\bar{\psi}_f$ are the anti-quark fields. Note that $\bar{\psi}_f$ are independent from ψ_f. D_μ are the covariant derivatives:

$$D_\mu = \partial_\mu + igA^a_\mu T^a \tag{1.4}$$

The QCD Lagrangian is invariant under the local SU(3) gauge transformation $V(x)$ as a gauge theory. To be specific, the gauge transformations of the quarks and gluons are:

$$\psi(x) \to V(x)\psi(x) \tag{1.5}$$

$$A^a_\mu(x) \to V(x)(A^a_\mu(x)T^a + \frac{i}{g}\partial_\mu)V^\dagger(x) \tag{1.6}$$

The physical quantities, in particular, correlation functions are calculated in a path integral fashion:

$$\langle O \rangle = \frac{1}{Z}\int [d\bar{\psi}][d\psi][dA_\mu] O e^{iS(A_\mu,\bar{\psi},\psi)} \tag{1.7}$$

where S is the action:

$$S = \int d^4x \mathcal{L}(x) \tag{1.8}$$

and Z is the partition function:

$$Z = \int [d\bar{\psi}][d\psi][dA_\mu] e^{iS(A_\mu,\bar{\psi},\psi)} \tag{1.9}$$

1.2 Lattice QCD

Unlike QED, the QCD coupling constant is large when the energy scale is below ~1 GeV. This makes perturbative calculations unreliable. Lattice QCD is a non-perturbative method to study QCD. It uses a finite space-time lattice to study the infinite and continuous four dimensional space. The physics is simulated numerically through the Monte Carlo method. With the development of modern computers, lattice QCD has become one of the most important approaches to study QCD.

There are two main steps in lattice QCD with the Monte Carlo method: 1) generating a number of gauge field samples according to the QCD action; 2) measuring the physical quantities with the gauge field samples. There are many theoretical and practical difficulties and we will discuss some of the main obstacles.

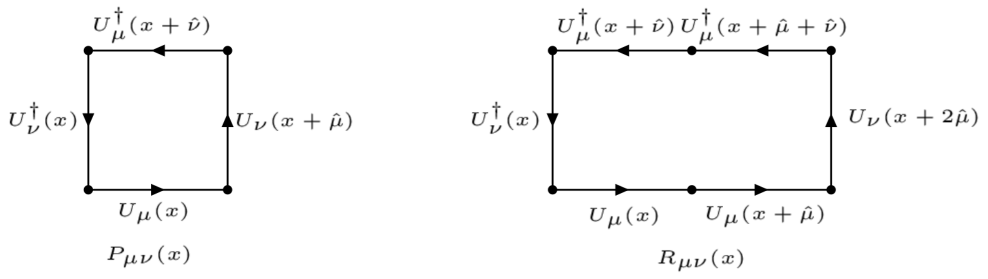
The first issue is that the integrand of the partition function is highly oscillating. To overcome the issue, the Wick rotation is performed:

$$x_0 \to -ix_4 \tag{1.10}$$

Then the Minkowski spacetime is transformed to the Euclidean spacetime and the integrand of the partition function is real. Therefore, one could view the integrand of the partition function as a probability distribution function for generating the gauge and fermion fields. After the gauge fields are generated, correlation functions and other physical quantities can be calculated through sample averaging. The gauge fields and fermion fields need to be discretized during the calculation and there are lots of difficulties. Those substantial challenges were solved through great efforts by the lattice QCD community and some of the solutions will be briefly explained in this chapter.

1.2.1 Discretization of the gauge fields

Although in the continuous theory the basic elements of the gauge fields are $A_\mu(x)$, in lattice QCD, it's easier to use the SU(3)-valued link variables $U_\mu(x)$ as the basic elements:



$U_\mu^\dagger(x+\hat{\nu})$
$U_\nu^\dagger(x)$
$U_\nu(x+\hat{\mu})$
$U_\mu(x)$
$P_{\mu\nu}(x)$
$U_\mu^\dagger(x+\hat{\nu})$ $U_\mu^\dagger(x+\hat{\mu}+\hat{\nu})$
$U_\nu^\dagger(x)$
$U_\nu(x+2\hat{\mu})$
$U_\mu(x)$ $U_\mu(x+\hat{\mu})$
$R_{\mu\nu}(x)$

Figure 1.1: Plaquette and rectangular link product

$$U_\mu(x) = \exp[iagA_\mu(x)] \tag{1.11}$$

The discretization of the gauge fields is straightforward. One simply put the link variables on the lattice sites.

With the link variables, one can construct other quantities on the lattice, for example, the plaquette:

$$P_{\mu\nu}(x) = \mathrm{tr}(U_\mu(x)U_\nu(x+\hat{\mu})U_\mu^\dagger(x+\hat{\nu})U_\nu^\dagger(x)) \tag{1.12}$$

One can also construct the rectangular link products like:

$$R_{\mu\nu}(x) = \mathrm{tr}(U_\mu(x)U_\mu(x+\hat{\mu})U_\nu(x+2\hat{\mu})U_\mu^\dagger(x+\hat{\mu}+\hat{\nu})U_\mu^\dagger(x+\hat{\nu})U_\nu^\dagger(x)) \tag{1.13}$$

The variables similar to the plaquette and the rectangular link products are invariant under the gauge transformation:

$$U_\mu(x) \to V(x)U_\mu(x)V^\dagger(x+\hat{\mu}) \tag{1.14}$$

where $V(x)$, $V^\dagger(x+\hat{\mu})$ are SU(3) matrices on the lattice. With the gauge invariant variables, the QCD action could be constructed as:

$$S = -\frac{\beta}{3}[(1-8c_1)\sum_{x,\mu<\nu} P_{\mu\nu}(x) + c_1 \sum_{x,\mu\neq\nu} R_{\mu\nu}(x)] \tag{1.15}$$

With this construction, in the continuum limit where the lattice spacing a goes to 0, the lattice action reduces to the continuous QCD action [1]. When $c_1 = 0$, the action is called the Wilson gauge action which is the simplest choice. When $c_1 = -0.331$, the action is called the Iwasaki gauge action [2]. The Iwasaki gauge action is widely used because it has a smaller a^2 dependence than the Wilson action. For a given lattice spacing, the gauge field is more smooth because of the small a^2 dependence. In addition, the Iwasaki gauge action approaches the same continuum limit as the Wilson action which allows good theoretical analysis using perturbative theory.

1.2.2 Discretization of the fermion fields

The discretization of the fermion fields relies heavily on the formulation of the Dirac operator. A direct finite difference realization of the differential operator will give 1 physical an 15 nonphysical modes in the continuum limit when the lattice spacing goes to 0 and will give non-physical results. There are multiple ways to solve the problem. For example, Wilson solved this problem by noticing that adding terms that vanish in the $a \to 0$ limit won't bring any trouble and the terms can be used to eliminate the nonphysical modes in the $a \to 0$ limit [3]. The Wilson Dirac operator is:

$$D_W(x,y) = (4+m)\delta_{x,y} - \frac{1}{2}\sum_{\mu}((1-\gamma_\mu\delta_{x+\mu,y}) + (1+\gamma_\mu)\delta_{x-\mu}, y) \tag{1.16}$$

People have also introduced staggered, twisted mass, overlap and domain wall fermions. In this thesis, the domain wall fermions will be used because this is the one that preserves the chiral symmetry the best.

We start by introducing briefly chiral symmetry. The continuous QCD action with massless quark fields has chiral symmetry under the transformation:

$$\psi \to e^{i\theta\gamma_5}\psi \tag{1.17}$$

$$\bar{\psi} \to \bar{\psi} e^{i\theta\gamma_5} \tag{1.18}$$

Nielsen and Ninomiya showed that it's impossible to construct a Hermitian, local, translationally invariant and chiral fermion in four dimension without lattice artifacts [4]. For example, with Wilson fermions, the action has bad chiral properties at finite lattice spacing.

The domain wall formulation [5] overcomes the issue by introducing a fifth dimension. The fifth dimension is also called the s direction and its size is given by L_s. With $L_s \to \infty$, the formulation has exact chiral symmetry. The physical chiral modes live on the $s = 0$ slice and $s = L_s - 1$ slice. To retrieve the four dimensional fermion fields, one projects out the left-handed fields on the $s = 0$ slice and right-handed fields on the $s = L_s - 1$ slice. With finite L_s, the domain wall formulation doesn't have exact chiral symmetry and the degree to which the chiral symmetry is broken is usually represented by the residual mass m_{res}. The residual mass depends on Ls according to the equation: $m_{res} = A\frac{e^{-CL_s}}{L_s} + B\frac{1}{L_s}$, where A, B and C depend on the spectrum of the transfer matrix H_{transfer} defined in [6].

The generic domain wall operator is:

$$D_{DWF}(x, s; x', s') = b_s D_W(x, x')\delta_{ss'} + \delta_{xx'}\delta_{ss'} + c_s D_W(x, x')L_{ss'} - \delta_{xx'}L_{ss'} \tag{1.19}$$

where D_W is the Wilson Dirac operator and $L_{ss'}$ is given by:

$$L_{ss'} = (L_+)_{ss'}P_R + (L_-)_{ss'}P_L \tag{1.20}$$

where

$$(L_+)_{ss'} = (L_-)_{s's} = \begin{cases} -m\delta_{L_s-1,s'} & s = 0 \\ \delta_{s-1,s'} & 1 \leq s \leq L_s - 1 \end{cases} \tag{1.21}$$

The parameters b_s and c_s are called the Mobius parameters. In the case that $b_s = 1$ and $c_s = 0$, the operator is usually called the Shamir domain wall operator [5]. When b_s and c_s are independent of s, $b_s - c_s = 1$ and $b_s + c_s = \alpha$, the operator is usually called the Mobius domain wall operator

[7] and α is called the Mobius scale. When b_s and c_s are complex parameters varying at different s, it's called the zMobius domain wall operator [8]. Through the overlap transformation, it can be shown that the chiral symmetry is preserved with $L_s \to \infty$ and $m \to 0$. The Mobius operator (when the Mobius scale is larger than 1) and the zMobius operator are more complex than the Shamir operator. However, they have better chiral property with given L_s compared to the Shamir operator in the sense that they have smaller m_{res}.

The domain wall formulation is computationally more expensive than most of the other formulations because of the extra dimension. However, due to good chiral properties, one can do fits according to the SU(2) or SU(3) chiral perturbation theory when the fermion masses are light. In [9][10], it's shown that the the chiral perturbation theory can be used for global fits to improve the precision and reliability of the results.

1.3 Gauge field generation

As discussed above, the first main step in lattice QCD is gauge field generation. The gauge fields are generated according to the distribution $\exp(-S)$. The Metropolis-Hastings algorithm is widely used to serve the purpose. The Metropolis-Hastings algorithm is a Markov chain method that is used to generate samples according to a certain distribution. Starting from a sample U_0, the probability to accept the next sample U_1 is:

$$P = \begin{cases} 1, & S(U_1) \leq S(U_0) \\ e^{S(U_0)-S(U_1)} & S(U_1) > S(U_0) \end{cases} \tag{1.22}$$

If U_1 is rejected, U_0 is duplicated as the next sample. It can be shown that the algorithm gives an ensemble of samples that follow the distribution $\exp(-S)$. To make the theory computationally viable, a proper algorithm to update the gauge links is needed.

To update the gauge field sample from U_i to U_{i+1}, the hybrid Monte Carlo (HMC) algorithm [11] is used. The hybrid Monte Carlo algorithm introduces the conjugate momenta $\pi_\mu(x)$ of the gauge fields $U_\mu(x)$ and a Hamiltonian system is constructed. Then the gauge fields are updated according to the Hamiltonian equations.

To be specific, the Hamiltonian is:

$$\mathcal{H}(\pi_\mu(x), U_\mu(x)) = \sum_{x,\mu} \frac{1}{2}\mathrm{Tr}(\pi^2_{x,\mu}) + S(U) \tag{1.23}$$

A fictitious evolution time τ (also called molecular dynamic time) is introduced and the equations of motion are:

$$\frac{d}{d\tau}U_\mu(x) = i\pi_\mu(x)U_\mu(x) \tag{1.24}$$

and

$$\frac{d}{d\tau}\pi_\mu(x) = -\partial^a_{x,\mu}S[U]t^a \tag{1.25}$$

By combining HMC and the Metropolis-Hastings algorithm, the gauge fields can be updated. However, the lattice QCD action also has fermions in it and the fermion term must be included. The fermion fields are Grassman variables and can't be incorporated directly in an easy way. In practice, one notices that the fermion term in the action can be integrated as:

$$\int [d\bar{\psi}][d\psi]e^{-\bar{\psi}D\psi} = \det(D) \tag{1.26}$$

The result is identical to a bosonic field ϕ with action:

$$\int [d\phi^\dagger][d\phi]e^{-\phi^\dagger D^{-1}\phi} = \det(D) \tag{1.27}$$

where ϕ is not a Grassman number. Now we can use ϕ in the simulation for the fermion action and generate gauge fields. However, D as a matrix is hard to be inverted. What's more, though det D is real, $\phi^\dagger D^{-1}\phi$ could be complex and the action would be unsuitable as a probability distribution. This is solved by taking:

$$\begin{aligned}\det D &= \det(D^\dagger D)^{1/2} \\ &= \int [d\phi^\dagger][d\phi] e^{-\phi^\dagger (D^\dagger D)^{-1/2}\phi}\end{aligned}$$

Lastly, the 1/2 power of the matrix is calculated through the rational approximation, i.e. the 1/2 power is approximated by the combination of several integer powers. This concludes the main techniques for gauge field generation.

In practice, there are many other techniques to either accelerate the calculation or makes the gauge fields better approximation of the continuous fields. For example, in the action, sometimes a term called dislocation suppressed determinant ratio (DSDR) [6] is included to suppress the changes of the topological modes when the lattice spacing is large.

1.4 Measurements on the lattice

As mentioned before, after the gauge fields are generated, measurements can be done with the gauge fields and the average values will be the final results from the lattice. Some of the measurements can be done directly with the gauge fields. For example, the topological charge can be measured as[1]:

$$Q = \sum_x \frac{1}{32\pi^2} \epsilon_{\mu\nu\rho\lambda} \mathrm{Tr}(F_{\mu\nu}(x) F_{\rho\lambda}(x)) = \sum_x \rho(x) \tag{1.28}$$

where $F_{\mu\nu}(x)$ are the field strength tensors. Other quantities such as the average plaquette can also be directly measured with the the gauge fields. However, most of the interesting quantities involve fermions. For example, one might want to calculate the correlation function of the fermion fields between two points:

$$\langle \psi(x_1)\bar{\psi}(x_2)\rangle = \frac{1}{Z}\int [d\bar{\psi}][d\psi][dU] \psi(x_1)\bar{\psi}(x_2) e^{-\bar{\psi}D\psi + \dots} \tag{1.29}$$

[1]In practice, since the gauge fields are not smooth, the gauge fields have to be smoothed through techniques like the Wilson flow before the measurements of the topological charge to avoid lattice artifacts. This is explained in more details in chapter 2.

This can be evaluated as:

$$\langle \psi(x_1)\bar{\psi}(x_2)\rangle = \langle D^{-1}(x_1, x_2)\rangle \tag{1.30}$$

In practice, it's impossible to invert the whole Dirac matrix. Instead, one can solve the linear equation:

$$\sum_{x_2} D(x_1, x_2)\xi(x_2) = \eta(x_1) \tag{1.31}$$

$\eta(x_1)$ is generally referred as the source and one can set $\eta(x_1)$ to be a point source. Sometimes one also considers the propagation from a certain time slice, t. In that case η would be 1 where the coordinate of the time direction is t. After solving the linear equation above, the solution can be used for the purpose of the correlation function. Note that since we are using the domain wall formulation, proper projection is needed so that we get four dimensional physical fermion fields.

In QCD, since quarks are confined, most physical matrix elements involve at least four quark fields. Taking the pion correlation function as an example, the correlation function is:

$$\langle d(x)\gamma_5\bar{u}(x)u(y)\gamma_5\bar{d}(y)\rangle = \mathrm{tr}D_d^{-1}(x, y)\gamma_5 D_u^{-1}(y, x)\gamma_5 \tag{1.32}$$

In this case, two linear equations will be solved and then the solutions will be properly contracted to get the final correlation function. When 6 or more quarks are involved, there are more complex contractions and sometimes one needs to combine different contractions to get a single correlation function. In conclusion, when fermions are involved, one needs to solve the Dirac equations and do proper contractions to get the correlation function.

After getting the correlation function, one can extract information like particle masses from the correlation function. Theoretically, the correlators can be calculated in a Hamiltonian framework. For example, let's consider a field ϕ_1 at t_1 that propagates to t_2 and becomes ϕ_2. The fields are created by applying the corresponding operators to the vacuum. The theoretical correlation function can be calculated for a periodic lattice of time extent T as [12]:

$$\langle \phi_1(t_1)\phi_2(t_2)\rangle = \frac{1}{Z}\mathrm{Tr}(e^{-\hat{H}(T-t)}N[\hat{\phi}_1]e^{-\hat{H}t}N[\hat{\phi}_2]) \tag{1.33}$$

where $t = t_2 - t_1$, $N[\hat{\phi}_1]$ and $N[\hat{\phi}_2]$ are normally ordered operators defined in [12]. One can expand the equation in terms of the eigenstates of $\hat{H}$ and set the lowest energy to 0. The correlator is:

$$\begin{aligned}
\langle \phi_1(t_1)\phi_2(t_2)\rangle &= \frac{1}{Z}\sum_{i,j} e^{-E_j(T-t)}e^{-E_i t}\langle j|\hat{\phi}_1|i\rangle\langle i|\hat{\phi}_2|j\rangle \\
&= \frac{1}{Z}(e^{-E_1 t}\langle 0|\hat{\phi}_1|1\rangle\langle 1|\hat{\phi}_2|0\rangle + e^{-E_1(T-t)}\langle 1|\hat{\phi}_1|0\rangle\langle 0|\hat{\phi}_2|1\rangle + O(e^{-E_2 t} + e^{-E_2(T-t)})) \\
&\approx A(e^{-E_1 t} + e^{-E_1(T-t)})
\end{aligned}$$

In the last equation, we omit all the higher orders in terms of the energy. This shows that one can fit the energy E_1 from the correlation function. By combining different correlation functions, one can calculate particle masses and other physical quantities.

The approximation in the last step relies heavily on the energy levels. When the difference between different energy levels is small, there might be difficulties extracting the correct parameters from the correlation function. In addition, the noise is a concern in practical fitting. Besides analyzing the correlation function, the variance of the correlation function should be taken into consideration as an estimation of the noise. In the case that there are low energy states in the variance, the signal to noise ratio might be poor.

Chapter 2: Topological properties, closed quark loops and zero modes of the domain wall operator

The up quark, down quark and strange quark have small masses. In the massless limit, QCD has SU(3) flavor symmetry. Given the vacuum breaking of the symmetry, there should be nine Goldstone bosons with similar masses. However, the η' mass is very different. The reason is that the QCD vacuum has non-trivial topological properties which break the $\mathrm{U_A}(1)$ symmetry [13][14]. The symmetry-breaking explains why the mass of η' is heavy. As a result, the measurements of η' mass require a good sampling over the topological sectors for lattice QCD. However, the measurement of the topological properties on the lattice is difficult because the local fluctuations of the gauge fields are large. The common way to overcome the difficulty is to smear the fields. However, the measurements for η and η' are done without smearing. Also, direct measurements through gauge fields only give the global topological charge or topological charge density and don't give other aspects of the topological properties.

In this chapter, we study the topological properties of the lattice through direct measurements, closed quark loops and the zero modes of the domain wall operator. We will first introduce the famous index theorem and its relation with η and η'. We will then describe the lattices that are used in this chapter. Afterwards, We will talk about the results of the closed quark loops $\sum_{\vec{x}}\langle\bar{\psi}(\vec{x},t)\gamma_5\psi(\vec{x},t)\rangle$ and topological measurements to show the possible existence of long auto-correlation. The properties of the low-lying eigenvectors will be discussed. Lastly, we demonstrate that it's possible to construct quark bilinears from the eigenvectors. In the next chapter, we will calculate the mass of η and η'.

2.1 The index theorem

We will first give a brief introduction to the index theorem in the continuous theory, which states that the topological charge of the QCD vacuum equals the difference between the number of the chiral left-handed zero modes and the number of the chiral right-handed zero modes of the Dirac operator. We will follow Coleman's notation and process for the proof of the index theorem [15].

First of all, the topological charge is defined as the integration of the inner product of the field strength:

$$\nu = \frac{1}{32\pi^2}\int d^4x(F,\tilde{F}) \tag{2.1}$$

The topological charge and the Dirac operator are related through the chiral Ward identity. In QCD, in particular, it states:

$$-\frac{i}{16\pi^2}\int d^4x(F,\tilde{F}) = 2\langle\int d^4x\bar{\psi}m\gamma_5\psi\rangle \tag{2.2}$$

which means:

$$\begin{aligned} -2i\nu &= 2\langle\int d^4x\bar{\psi}m\gamma_5\psi\rangle \\ &= 2\frac{\int[d\psi][d\bar{\psi}]e^{-S}\int d^4x\bar{\psi}m\gamma_5\psi}{\int[d\psi][d\bar{\psi}]e^{-S}} \\ &= \frac{2m\sum_\lambda\int d^4x\psi^\dagger_\lambda\gamma_5\psi_\lambda\prod_{\lambda'\neq\lambda}(\lambda'-im)}{\prod(\lambda-im)} \end{aligned} \tag{2.3}$$

where we calculate the integration of the fermion fields to be the determinant of the fermion operator. In the equation, λ is the eigenvalue of the Dirac operator, ψ_λ is the corresponding eigenvector where the eigenvalue is used as the subscript of the eigenvector and m is the mass of the fermion. Since $i\not{D}$ is Hermitian, the eigenvalues are all real. In addition, $\not{D}$ anticommutes with γ_5, so for eigenvectors with non-zero eigenvalues, we have:

$$i\not{D}\gamma_5\psi_\lambda = -\lambda\gamma_5\psi_\lambda = -\lambda\psi_{-\lambda} \tag{2.4}$$

So the non-zero modes always appear in pairs. For one non-zero mode, one can apply γ_5 to the eigenvector and get another eigenvector that has an eigenvalue with the opposite sign. For zero modes, we have:

$$\gamma_5\psi_r = \chi_r\psi_r \tag{2.5}$$

For one gauge configuration, there could be n_+ zero modes where $\chi_r = 1$ and n_- zero modes where $\chi_r = -1$. χ_r is called the chirality of the mode.

Because of the properties of $\not{D}$, most terms in eq. (2.2) vanish because for non-zero modes:

$$\int d^4x\psi_\lambda^\dagger\gamma_5\psi_\lambda = \int d^4x\psi_\lambda^\dagger\psi_{-\lambda} = 0 \tag{2.6}$$

and we get the index theorem.

$$-2i\nu = 2i(n_+ - n_-) \tag{2.7}$$

This concludes the proof of the index theorem. Note that there are several aspects through which one can study the topology. In eq. (2.1), the topology can be measured directly through the gauge fields. In eq. (2.2) and eq. (2.4), the topological charge is related to the fermion correlator. In fact, this is the reason that the topological properties are important to η and η'. The correlator $\langle\bar{\psi}\gamma_5\psi\rangle$ appears in the correlation function of η and η'. In eq. (2.7), the topological charges are linked to the zero modes of the Dirac operator. We will study the topological properties through all three aspects.

2.2 Lattices and calculation details

We consider three ensembles in this chapter. The action for the three ensembles is the Iwasaki + DSDR (Dislocation Suppressing Determinant Ratio) gauge action [6]. The DSDR part suppresses

size	a^{-1}	m_π	total configurations	total eigenvectors	methods
12×32	1GeV	300MeV	70	3500	Lanczos
24×64	1GeV	140MeV	2	40	Ritz
24×64	2GeV	300MeV	2	60	Ritz

Table 2.1: Lattices and the eigenvectors

the changes of the topological charges which is important when the coupling is strong. The first ensemble is the $24^3 \times 64$, $a^{-1} \approx$ 1GeV ensemble [9] with $m_\pi \approx$ 140MeV. This is a physical ensemble in the sense that the physical quantities calculated from this ensemble are close to the real value. The second ensemble is the $24^3 \times 64$, $a^{-1} \approx 2$ GeV ensemble with $m_\pi \approx$ 300MeV. This ensemble has heavier pion mass but the lattice spacing is smaller. The small lattice spacing means that it's close to the continuum limit. The third ensemble is the $12^3 \times 32$, $a^{-1} \approx$ 1GeV ensemble with $m_\pi \approx$ 300MeV. The benefit from this ensemble is that the volume is small and the calculation is cheap. We calculate the topological charges through the gauge fields by the 5-loop improved method [16] or through field strength. For the eigenvectors, we use the implicitly restarted Lanczos algorithm to calculate eigenvectors for the 12ID ensemble and the Ritz algorithm to calculate the eigenvectors for the two 24ID ensembles. The information is listed in Table 2.1. The calculation methods will be discussed in more details in each section.

During the calculation of the eigenvectors, because the chiral symmetry is preserved in the massless limit, we use different input masses to study the mass dependence of the eigenvectors and try to push it to the massless limit. Because the finite fifth dimension brings the residual mass, we also vary the length of the fifth dimension, L_s, for the 12ID ensemble to study how the residual mass changes the chiral properties of the eigenvectors. For the two 24ID ensembles, we only calculate on a few configurations due to the computational limits.

2.3 Direct measurements of the topological charges and the closed quark loops

In this section, we show the direct measurements of the topological charges and the closed quark loops of the 24ID $a^{-1} \approx 1$GeV ensemble. In chapter 1, it's mentioned that the topological charge can be directly measured through the field strength. However, the gauge fields have large local fluctuations which affect correct measurements. The common way to overcome the obstacle is to smooth the fields by running the Wilson flow [17]. The equation for the Wilson flow is:

$$\frac{dU_t(x,\mu)}{dt} = -g_0^2\{\partial_{x,\mu} S_{\text{flow}}(U_t)\}U_t(x,\mu) \tag{2.8}$$

where $U_t(x,\mu)$ is the link variable at flow time t and it equals the original link variable when $t = 0$. S_{flow} is the flow action and it doesn't have to be the same as the action for generating the ensemble. g_0 is the coupling constant. With the Wilson flow, the gauge fields will change along the steepest descent direction toward the stationary point, or local low energy point of the action. The topological charge as a global property will not change with the Wilson flow. At the same time, the local lattice artifacts will be smoothed out. As a result, this is beneficial for the measurements of the global topological charge. In our measurements, the Wilson flow is run for 1000 steps and in each step, the step size is $dt = 0.05$. It's found that the topological charge measurements are stable after the Wilson flow.

The topological charge can then be measured through the gauge fields. The global topological charge is measured through the 5-loop improved (5Li) method [16]. The method measures the combination of 5 different kinds of loops to get rid of $O(a^2)$ and $O(a^4)$ corrections. The measurements of the topological charges are shown in Fig 2.1. Here the results are shown as a function of the molecular dynamic time (gauge field evolution time) to demonstrate the evolution of the topological charge. The average topological charges in every 20 MD units are also shown.

We would like to emphasize the influence of the action and the lattice spacing on the topological charge. The action has a DSDR term which suppresses the changes of the topological charges. However, the large lattice spacing $a^{-1} \approx 1$GeV promotes the changes of the topological charges.

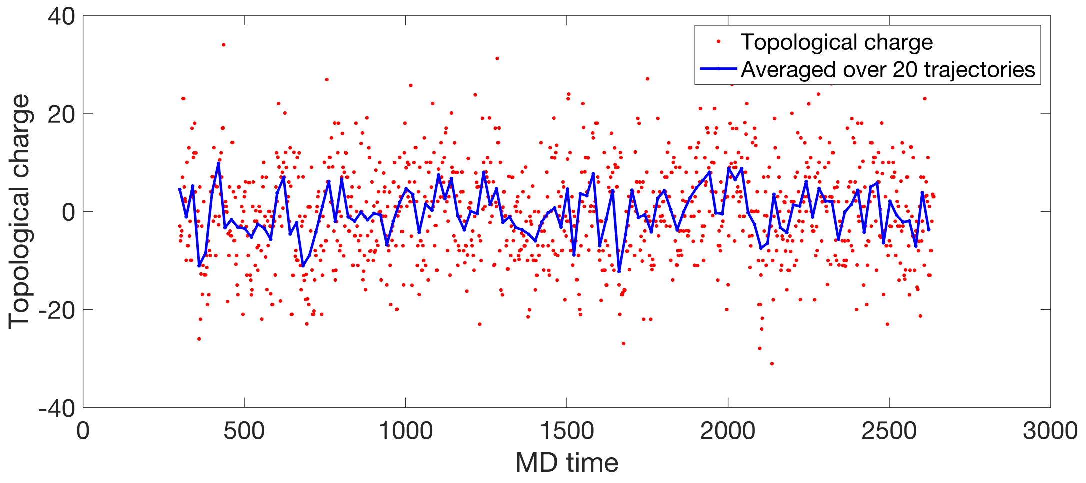


Figure 2.1: Topological evolution of the 24ID $a^{-1} = 1\text{GeV}$ ensemble, showing a good sampling over all topological sections.

The result will be a balance of the two effects. Overall, from Fig 2.1, it can be seen that the topological charges change by a large amount during the evolution and the average topological charge is close to zero. From this perspective, our gauge fields are well sampled over all global topological sectors and there is no topological freezing. However, as we look at the results for closed quark loops, we will find a more complicated situation.

The closed quark loops are important because they appear in the calculation of the η and η'. In the calculation of the η' mass, we must consider $\langle\bar{\psi}\gamma_5\psi(x)\bar{\psi}\gamma_5\psi(y)\rangle$, which leads to the disconnected terms like $D_{ll} = \langle\bar{\psi}\overline{\gamma_5\psi}(x)\bar{\psi}\overline{\gamma_5\psi}(y)\rangle$. We consider the closed quark loops on each time slice $\sum_{\vec{x}}\langle\bar{\psi}(\vec{x},t)\gamma_5\psi(\vec{x},t)\rangle$. During the measurement, we first set the gauge fields to Coulomb gauge. We then calculate $\sum_{\vec{x}}\langle\bar{\psi}(\vec{x},t)\gamma_5\psi(\vec{x},t)\rangle$ using a wall sink and a wall source. The results for the light quark mass are shown in Fig 2.2 as a function of the time slice. Error bars are shown with unbinned data and a binning of 10 (40 MD time units). When calculating errors, each bin is assumed independent. Note that since we have periodic boundary conditions, the results on each time slice should be the same by translational invariance. However, there is no translational invariance in time direction for our closed quark loop results. What's more, since the average topological charge should be zero, the values of closed quark loops should be zero. However, it can be seen that the

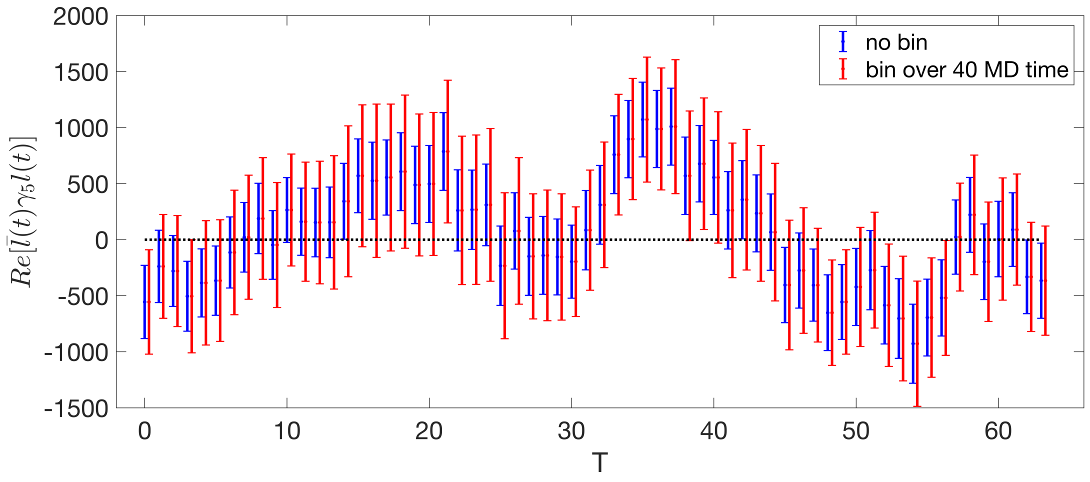


Figure 2.2: Closed quark loops for the light quark of the 24ID $a^{-1} = 1$GeV ensemble. This graph shows that the close quark loops don't average to 0 when there is no binning and persistent (in evolution time) topological objects might be the reason.

mean values for many time slices are many standard deviations from zero for unbinned results.

The most likely reason that the mean values don't fall to zero is that the autocorrelations are long during the evolution. To illustrate this, we give three pieces of evidence. The first evidence

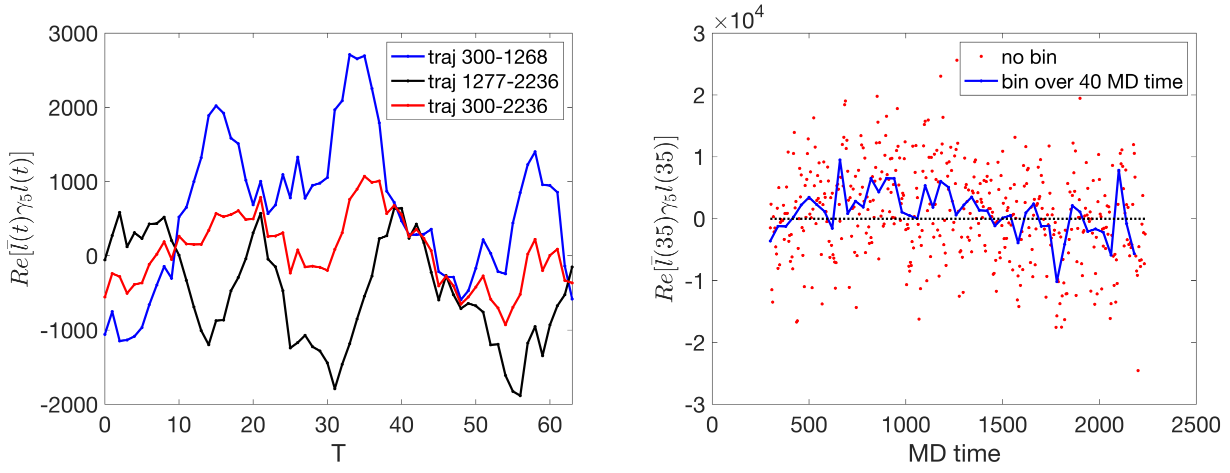


Figure 2.3: Left: The closed quark loops with part of the trajectories; right: The evolution of the closed quark loops for one time slice. The deviation from zero shows the possible existence of topological objects over long MD time.

comes from binned results. If one bins the results for closed quark loops over 40 MD time units, it is found that the error bars grow substantially and most mean values fall to zero (Fig 2.2). Also, the error bars stay the same if one bins the results over 80 MD time units, which means that the error bars are stable for 40 MD time units. Additional evidence for long autocorrelations is that if one uses only part of the trajectories (Fig 2.3), one finds that the mean values are very far away from zero, which shows that part of the trajectories doesn't provide enough statistics and one can deduce that the autocorrelation length is long. Lastly, one can look at the evolution of one single time slice in the molecular dynamic time. If one bins the results over 40 MD time units, one finds that the values could be above zero for a very long MD time. In Fig 2.3, the results for T = 35 is shown and one can see that it is constantly above zero for MD time 700 to 1200.

Hence, one sees that there is tension between the results of the closed quark loops which probe local topological fluctuations on the configurations and the results for the global topological charge measured after the substantial Wilson flow. The global topological charge tells one that the gauge fields are sampled well but the closed quark loops tell one that there are long autocorrelations. Binning to 40 or 80 MD time units gives a better estimation of errors, but O(500) MD time unit autocorrelations still appear for certain local topological quantities. This is important because correct local fluctuations are necessary to get localized correlators to produce the correct masses.

2.4 Lattice Dirac operator and the Dirac operator plus the mass term

With the domain wall operators, it's almost impossible to get a massless Dirac operator because the finite fifth dimension gives a small yet non-zero residual mass. As a result, to understand the domain wall operators, we have to compare to the Dirac operator plus the mass term in the continuous theory. The operator $\gamma_5(\not{D} + m)$ is a Hermitian operator with mass. Here we use the $\not{D}$ in the Euclidean space. By squaring the operator, one finds that the eigenvalues λ_H are related to the eigenvalues λ of the massless operator by:

$$\lambda_H^2 = \lambda^2 + m^2 \tag{2.9}$$

The anti-commutator with γ_5 is

$$\{\gamma_5(\not{D} + m), \gamma_5\} = 2m \tag{2.10}$$

If we define the chirality of an eigenvector as $\langle\psi|\gamma_5|\psi\rangle$, the chirality of the zero modes of the operator $\not{D}$ is 1 or -1 and the the chirality of the non-zero modes of the operator $\not{D}$ is 0 as discussed in section 1. The chirality of the eigenvector of the operator with the mass term can be calculated:

$$\langle\psi^H_{\pm\sqrt{\lambda^2+m^2}}|\gamma_5|\psi^H_{\pm\sqrt{\lambda^2+m^2}}\rangle = \frac{m}{\lambda_H} = \frac{m}{\pm\sqrt{\lambda^2+m^2}} \tag{2.11}$$

where we use the eigenvalues as the subscript for the eigenvectors. Note that the mass introduces a scale on which the chirality depends. For all eigenvectors with $\lambda_H \approx m$, $\lambda \ll m$, the chirality is close to ± 1. This is very different from the massless results where the non-zero modes all have 0 chirality.

In fact, one can even get the eigenvectors of the Dirac operator with the mass term from the eigenvectors of the massless Dirac operator $i\not{D}$. The eigenvectors are related by a rotation:

$$\begin{aligned}
\psi^H_{\sqrt{\lambda^2+m^2}} &= \frac{1}{\sqrt{2}}(e^{i\frac{\theta}{2}}\psi_\lambda + e^{-i\frac{\theta}{2}}\psi_{-\lambda}) \\
\psi^H_{-\sqrt{\lambda^2+m^2}} &= \frac{1}{\sqrt{2}}(e^{i\frac{\theta}{2}}\psi_\lambda - e^{-i\frac{\theta}{2}}\psi_{-\lambda})
\end{aligned} \tag{2.12}$$

where θ is given by $e^{i\theta} = \frac{m+i\lambda}{\sqrt{m^2+\lambda^2}}$. The eigenvectors on the left hand side are the eigenvectors of the operator with a mass term and the eigenvectors on the right hand side are the eigenvectors of the massless Dirac operator. In general, for two operators with mass m_1 and m_2, we have:

$$\begin{aligned}
\langle\psi^H_{\sqrt{\lambda^2+m_1^2}}|\psi^H_{\sqrt{\lambda^2+m_2^2}}{}'\rangle &= \cos(\theta_2 - \theta_1) \\
\langle\psi^H_{\sqrt{\lambda^2+m_1^2}}|\psi^H_{-\sqrt{\lambda^2+m_2^2}}{}'\rangle &= i\sin(\theta_2 - \theta_1)
\end{aligned} \tag{2.13}$$

For the Shamir domain wall fermions, the Hermition operator is $D^H_{DWF} = \gamma_5 R_5 D_{DWF}(m)$. The eigenvalue Λ should have the following form [18] for the low lying eigenvectors:

$$\Lambda^2 = n^2(\lambda^2 + (\delta m + m)^2) \tag{2.14}$$

Here, m is the input mass. n, δm and λ are the scale factor, the residual mass and the eigenvalue in the massless continuum. In the equation, Λ can be calculated by calculating the eigenvalues on the lattice. The input mass is known. However, n, δm and λ are unknown and have to be fitted. We will be able to confirm the relation through the quality of the fitting.

As described, we calculated the eigenvectors of $D^H_{DWF} = \gamma_5 R_5 D_{DWF}(m)$ through the Lanczos algorithm and the Ritz algorithm. The algorithms are used to calculate the eigenvectors when direct calculation is hard. With the Ritz algorithm, $\frac{\langle z, Az\rangle}{\langle z, z\rangle}$ is minimized to get the eigenvector with the lowest eigenvalue. A minimization method that is similar to the conjugate gradient algorithm is used [19]. Afterwards, a few low-lying eigenvectors are calculated in a similar fashion by keeping them orthogonal to the previously calculated eigenvectors. The Lanczos algorithm is described in the appendix and we introduce the algorithm here briefly. The Lanczos algorithm shares some ideas with the power method but the idea of Krylov space is utilized. In the power method, through iterative application of the matrix to a vector, the low-mode components of the vector are smaller and smaller and the eigenvector with the largest eigenvalue will be left in the end. The eigenvector with the lowest eigenvalue can be obtained by proper modifications. The Lanczos algorithm can calculate multiple low-lying eigenvectors by utilizing the Krylov space. However, the Lanczos algorithm itself is not stable. Numerical errors due to the precision limit will accumulate and destroy the final results. In [20], proper restarts are utilized to avoid numerical instability. This is called the implicitly restarted Lanczos algorithm and the algorithm is used in our calculation. Furthermore, Chebyshev polynomials are used to change the spectrum. This means that a polynomial of the operator instead of the operator itself is used in the algorithm. The technique is also called preconditioning. The spectrum of the low eigenvalues are more isolated with the Chebyshev preconditioning.

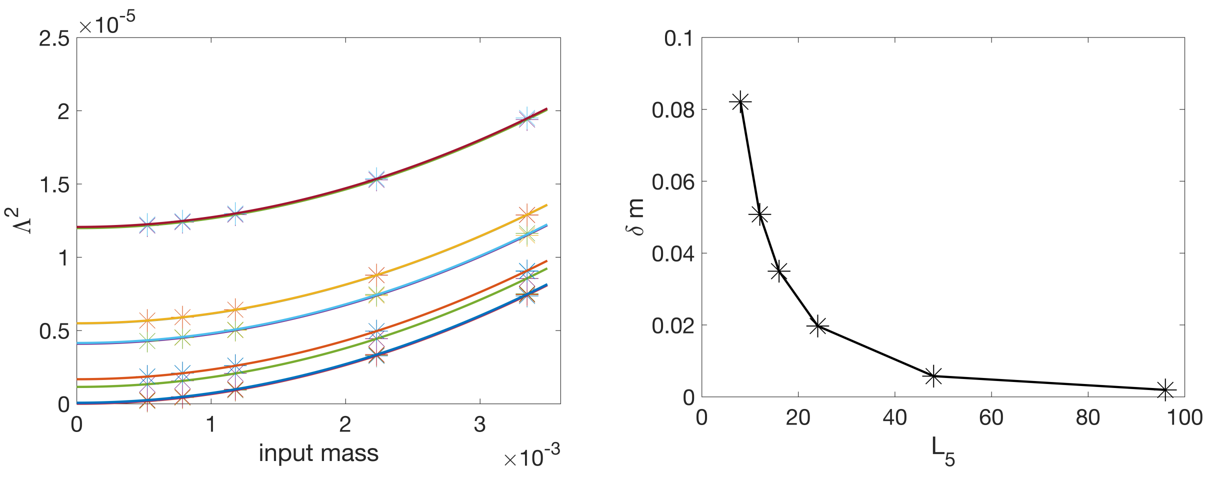


Figure 2.4: Left: eigenvalues of the lattice operator for different input masses, 24ID $a^{-1} \approx 2$GeV ensemble, configuration 800. The lines are fitting results according to eq. (2.13); Right: fitted residual mass as a function of L_s, 12ID ensemble, configuration 700. The result shows that the residual mass decreases when L_s increases.

The fits of the eigenvalues are shown in Fig 2.4. The points are the calculated eigenvalues on the lattice for the different input masses. For each input mass, the 10 lowest eigenvalues are plotted (some of the eigenvalues are very close). We fit eq. (2.13) for the eigenvalues of the different input masses. The same n^2 and δm are used for all the eigenvalues. The curves are then plotted according to eq. (2.13). It can be seen that eq. (2.13) describes the eigenvalues very well. We also calculated eigenvectors when different L_s are used and the residual masses are fitted. The residual masses are plotted against L_s in the right graph of Fig 2.4. The residual mass decreases inversely with L_s which agrees with the understanding of the domain wall operator. The graphs are for the configurations listed in the caption but the results are similar for all the ensembles and configurations.

The anti-commutation relation for the lattice Hermition Dirac operator is: $\{D_{DWF}, \Gamma_5\} = 2m_f Q^{(w)} + 2Q^{(mp)} \equiv 2Q$, where $Q^{(w)}$ and $Q^{(mp)}$ are defined in [18]. Γ_5 is 1 for $s >= L_s/2$

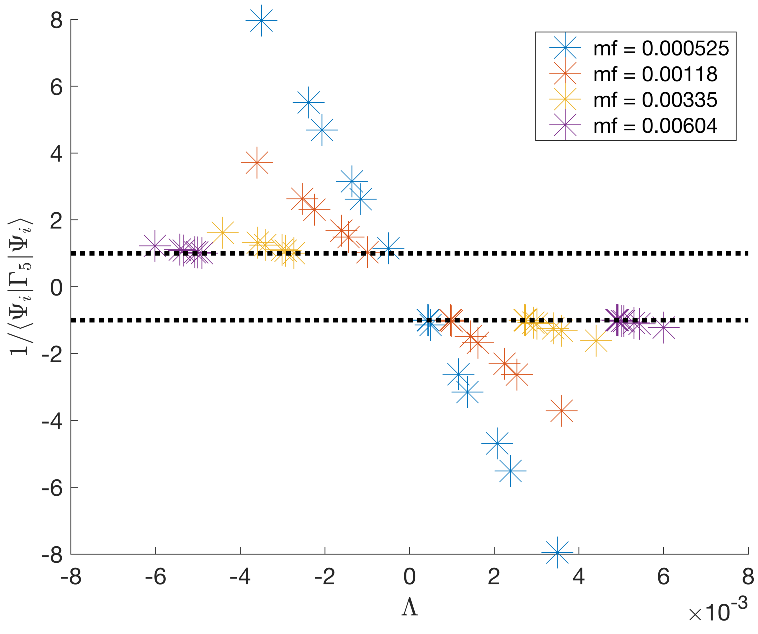


Figure 2.5: $1/\langle\Psi_i|\Gamma_5|\Psi_i\rangle$ as a function of eigenvalues, 24ID $a^{-1} \approx$ 2GeV ensemble, configuration 800. This demonstrates that the chirality of a lattice eigenvector depends on the eigenvalue according to eq. (2.14)

and -1 for $s < L_s/2$. This means that the chirality of the eigenvectors follows:

$$\langle\Psi_i|\Gamma_5|\Psi_i\rangle = \frac{\langle\Psi_i|Q|\Psi_i\rangle}{\Lambda_i} \tag{2.15}$$

When the numerator on the right hand side is constant for different eigenvectors, the chirality of the eigenvectors will be anti-proportional to the eigenvalues. This is the analog of eq. (2.10) for the Dirac operator with the mass term in the continuum. In Fig 2.5, we show $1/\langle\Psi_i|\Gamma_5|\Psi_i\rangle$ against the eigenvalues. The linear relationship shows that $\langle\Psi_i|Q|\Psi_i\rangle$ is indeed constant for different eigenvectors. Under this condition, when the mass increases, there are more eigenvectors with chirality close to 1 because there are more eigenvalues that are close to the mass. This explains why there are multiple negative and positive chiral modes for the heavy mass. We emphasize that this is very different from the continuous massless case where only zero modes have chirality 1 or -1. The results are similar for the other ensembles and configurations.

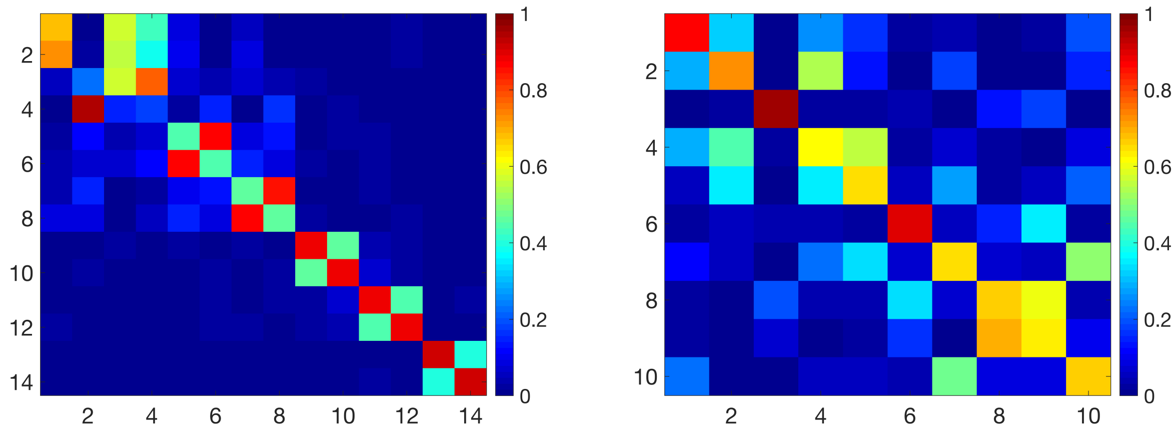


Figure 2.6: Left: Inner product between the eigenvectors with m = 0.000525 and m = 0.00604 for the 24ID, $a^{-1} \approx 2$GeV ensemble, config 800; right: Inner product between eigenvectors with m = 0.01 and m = 0.02 for the 12ID ensemble, config 700. The graph shows that the eigenvectors mix in pairs when the fermion mass is changed and the lattice is close to the continuum limit. The topological modes are hard to isolate when the fermion mass is unneglectable compared to the eigenvalues.

We are also able to study how the eigenvectors themselves change when the mass is changed and we compare the results to eq. (2.11) and (2.12) in the continuum limit. In Fig 2.6, we show the inner product between eigenvectors for different masses. The x and y axis are the index of the eigenvectors of the different input masses. A good agreement with eq. (2.11) and eq. (2.12) is found for 24ID $a^{-1} \approx 2$GeV. This means that the eigenvectors mix when the mass is changed and the high chirality of an eigenvector doesn't mean that it's a topological mode. To find a true topological modes, we have to have zero fermion mass. For the 12ID $a^{-1} \approx 1$GeV, the agreement is poor which means that for the coarse lattice, it's further away from the continuum limit. The results also show how mass can twist the eigenvectors and why true zero modes are hard to obtain with non-zero mass. Again, the results are consistent for different configurations.

For finite L_s, the residual mass makes it hard to get the true zero modes as in the massless continuous case. To understand the L_s effect, we calculate the eigenvectors at different L_s. In

Fig 2.7, we show the matrix elements $\langle \Psi_i | \Gamma_5 | \Psi_j \rangle$ where the diagonal elements are the chirality. For small L_s, it can be seen that there are both positive and negative modes because of the large residual mass. When L_s is increased, these modes gradually disappear. When $L_s = 192$, there is one distinct zero modes. However, large mixing can still be seen which means that it's very hard to get rid of the residual mass effect.

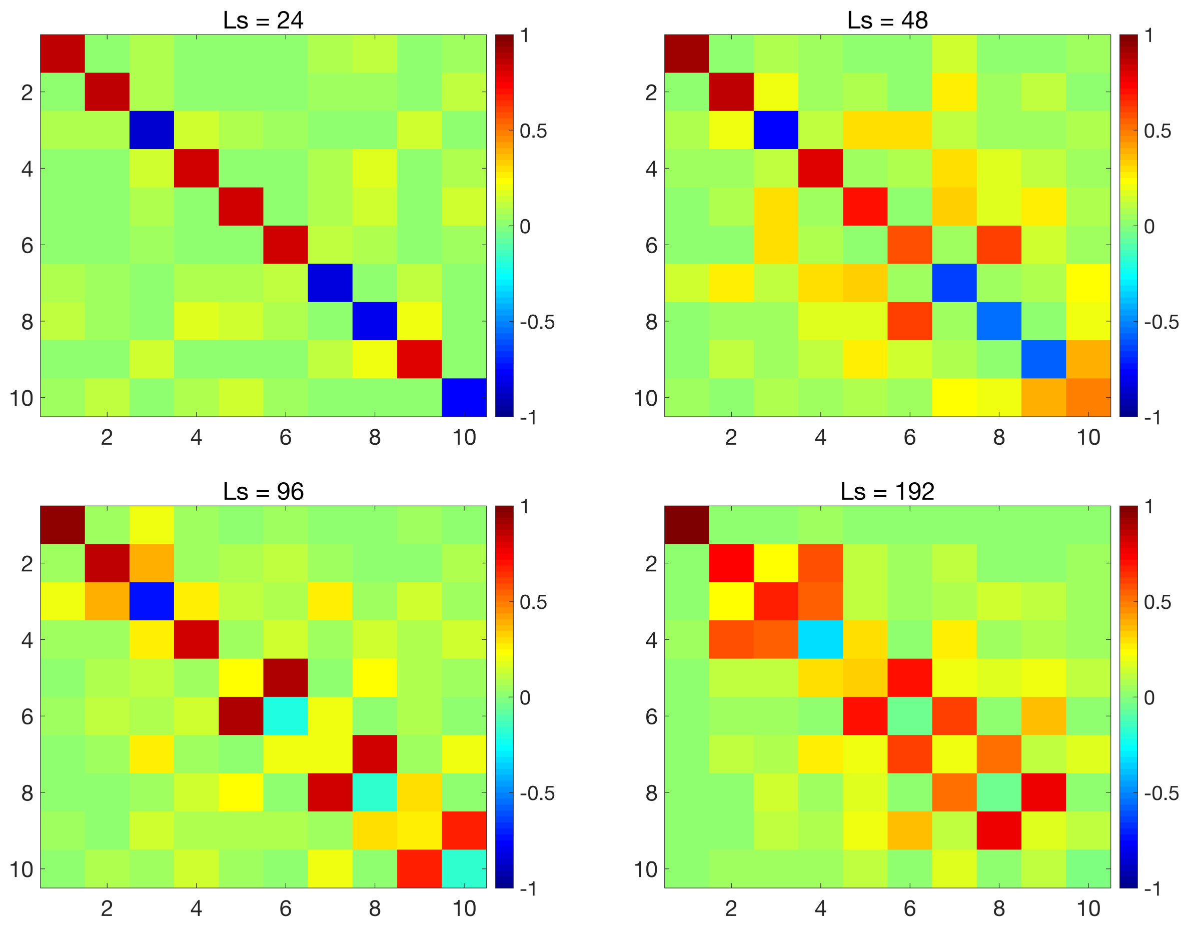


Figure 2.7: $\langle \Psi_i | \Gamma_5 | \Psi_j \rangle$ for different L_s, 12ID ensemble, configuration 700. The graphs show that zero modes that are close to massless limit could only be obtained when large L_s is used.

Lastly, we calculate the net topological charge by counting the net chiral modes. We consider chiral modes ($|\langle \Psi_i | \Gamma_5 | \Psi_i \rangle| >= 0.9$) and round the chirality to ± 1. For the results from the field strength ($F\tilde{F}$), we first run the Wilson flow with the flow time $t = 5.3$. The results are in Fig 2.8 and one can see that the results agree with each other. However, for the eigenvectors, the measurements

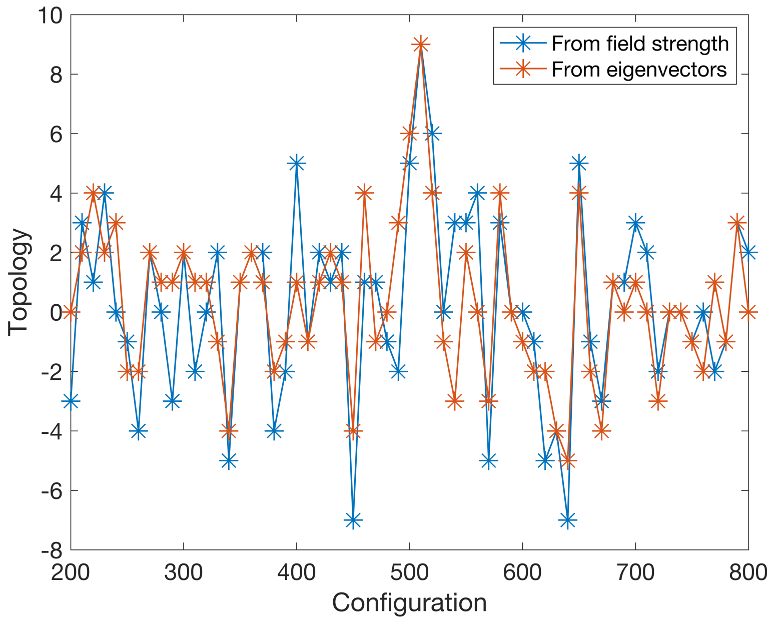


Figure 2.8: Topological charge calculated from the eigenvectors and the field strength. The results are for 12ID ensemble, configuration 200 to 800 with a separation of 10. Without smearing the gauge fields, we are able to obtain the topological charge through the eigenvectors.

are done on the original lattices without smearing.

In summary, the eigenvalues and the chirality of the domain wall operator are described by the Dirac operator with the mass term in the continuum. We are able to understand how the eigenvectors, eigenvalues and chirality change when the mass changes on the lattice. With finite mass, the chirality of the low modes are close to 1 as long as the eigenvalues are close to the mass which is very different from the continuum massless limit. The finite L_s brings in a mass scale and the massless limit is hard to reach. However, given a large enough L_s, one or a few true zero modes could be obtained which are isolated topological modes. Lastly, the global topological charge can be measured through the low-lying eigenvectors and the results are consistent with the measurements through the gauge fields. Again, we emphasize that smearing is not applied for the measurements from the eigenvectors.

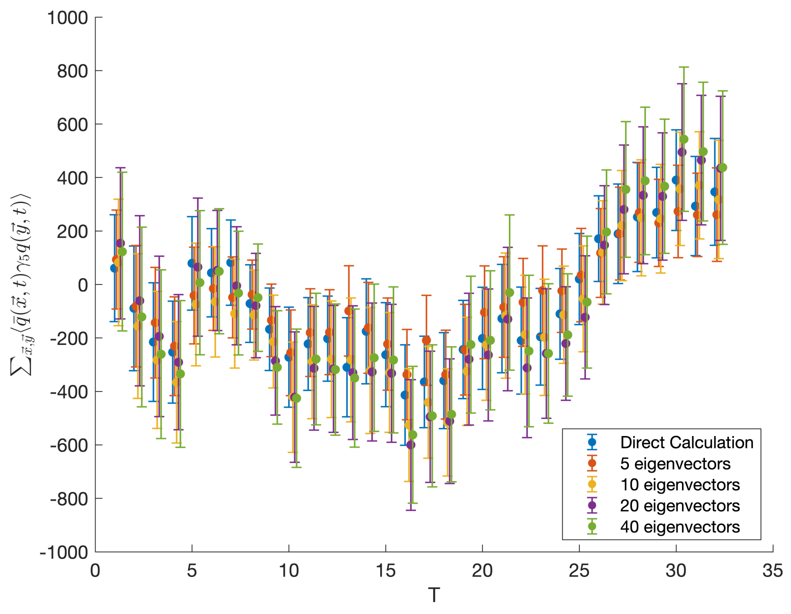


Figure 2.9: The quark closed loops obtained through the eigenvectors and the direct calculation. The results are for 12ID ensemble, configuration 200 to 800 with a separation of 10

2.5 From eigenvectors to the quark closed loops

To show that the eigenvectors and near-zero modes are closely related to the topological properties as well as η and η', we also consider the quark closed loops $\sum_{\vec{x},\vec{y}}\langle\bar{q}(\vec{x},t)\gamma_5 q(\vec{y},t)\rangle$ on the time slice in eq. (2.2). Note that this contributes significantly to the calculation of m_η and $m_{\eta'}$. The direct calculation is obtained by applying the inverse of $\not{D}$ to a wall source. However, one can also obtain the results by using the near-zero eigenvectors. The idea is very similar to the low-mode approximation. The formula is:

$$\sum_{\vec{x},\vec{y}}\langle\bar{q}(\vec{x},t)\gamma_5 q(\vec{y},t)\rangle = \sum_n \frac{\mathrm{Tr}(\sum_{\vec{x},\vec{y}}(\Psi_n^\dagger(\vec{x},t)\Psi_n(\vec{y},t)))}{\Lambda_n} \tag{2.16}$$

We use different numbers of eigenvectors and the results are shown in Fig 2.9. Note that the results from the eigenvectors approximate the direct calculation very well. It's interesting that we can approximate the direct calculation with only 5 eigenvectors. This shows that it's really topological zero modes that are involved in the quark closed loop and $m_{\eta'}$.

Chapter 3: The mass of η and η'

In this chapter, we will calculate m_η and $m_{\eta'}$. The calculation is closely related to the last chapter because the quark closed loops and topological charge density enter into the calculation. We would like to point out that η and η' are interesting in several aspects. First of all, as mentioned, it's tightly related to the topological properties of the QCD vacuum. With light quark masses (up quark, down quark and strange quark), there should be nine conserved axial-vector currents which bring nine Goldstone bosons with similar masses. However, η' has very different mass [21]. The breaking of the symmetry can only be explained by the non-trivial topological structure of the QCD vacuum [13][14]. Lattice QCD is one of the ways that the relationship can be explored. We will connect the topological density correlators with the mass of η and η' in this chapter. The quark closed loops, which is demonstrated in last chapter to be closely related to the topology, will also be used in the calculation.

There is another reason that η and η' are interesting in lattice QCD calculation. As states with the same quantum numbers, η and η' are mixtures of the pseudoscalar singlet and octet states of the SU(3) symmetric limit [22]. Lattice QCD is one of the most important ways to calculate the mixing angle. In this chapter, the mixing angle will be calculated.

We will begin this chapter by introducing some of the techniques used in the calculation. Then we will discuss the fermion correlators that will be used in the fits of η and η'. The topological charge density correlators can also be used in the fits for η and η' and it's discussed in the fourth section. Due to computational limitation, we measured on two sets of trajectories of gauge configurations and the intervals of the measurements are different. To combine the data, we study the autocorrelation of the data. Lastly, we show the results for η and η' as well as the mixing angle.

In this chapter, the focus is the 24ID $a^{-1} = 1\text{GeV}$ ensemble. This is a physical ensemble in the sense that the measured pion, kaon and omega masses are the same as the physical masses

[9]. Therefore, our measurements could be taken as a realistic study of the physical η mass and η' mass up to finite lattice spacing corrections. Results from different ensembles with different lattice spacings would be required for a continuum limit determination.

3.1 Jackknife resampling and other techniques in the calculation

As described in chapter 1, the physical masses are calculated by fitting to the correlators. To get good fits, some techniques are introduced to reduce and measure the errors. The first one is the jackknife resampling [23]. As discussed, we measure the same correlators on different gauge configurations. It's very likely that there will be one or a few outliers. The outliers are part of the samples but may give strange results after the fitting. The jackknife technique smooths the outliers and allows for error estimation for non-linear functions of measured values.

To describe it in details, suppose that we have samples: $\{X_1, X_2, ...X_n\}$. The jackknife resampling transforms the samples according to the formula:

$$Y_i = \frac{\sum_{j\neq i} X_j}{n-1} \tag{3.1}$$

Instead of working with the original samples, now we work with the new samples $\{Y_1, Y_2, ...Y_n\}$. After the resampling, the standard deviation for the new samples can be estimated through double-jackknife. For example, to calculate the standard deviation of Y_1, one create another ensemble where the elements are:

$$Z_i = \frac{\sum_{j\neq_{i,1}} X_j}{n-2} \tag{3.2}$$

Then the standard deviation of Y_1 will be $\sqrt{\sum(Z_i - \bar{Z})/(n-1)(n-2)}$. After fitting with the new ensemble $\{Y_1, Y_2, ...Y_n\}$, the true error is the standard deviation multiplied by $\sqrt{n-1}$. One can easily verify the factors with a set of Gaussian variables and the jackknife resampling will give correct expected values for the mean and the variance.

In the case that the samples are not independent from each other, one needs to consider the autocorrelation between the samples. To make the samples independent, the correlated samples can be binned together. Binning is one of the simplest techniques to resolve the autocorrelation between the samples.

Fitting is an important step to get the masses. The standard algorithm used to do the fits is the Levenberg-Marquardt (LM) algorithm [24]. The algorithm is an iterative method that can minimize χ^2 which is defined as:

$$\chi^2 = (\mathbf{y} - \mathbf{f}(\mathbf{x}, \beta))\mathbf{\Sigma}^{-1}(\mathbf{y} - \mathbf{f}(\mathbf{x}, \beta)) \tag{3.3}$$

where $\mathbf{y}$ is the vector of the measured correlators in our case, $\mathbf{f}$ is the fitting function, $\mathbf{x}$ contains the known arguments, β is the vector of the parameters to be fitted and $\mathbf{\Sigma}$ is the covariance matrix. Taking the derivative, one finds that the parameters should be updated by δ according to the equation:

$$\mathbf{J}^T\mathbf{\Sigma}^{-1}\mathbf{J}\delta = \mathbf{J}^T\mathbf{\Sigma}^{-1}(\mathbf{y} - \mathbf{f}(\mathbf{x}, \beta)) \tag{3.4}$$

where $\mathbf{J}$ is the Jacobian matrix of $\mathbf{f}$ with respect to β. This is the Gauss-Newton algorithm. The LM algorithm adds a damp parameter λ and changes the equation to:

$$(\mathbf{J}^T\mathbf{\Sigma}^{-1}\mathbf{J} + \lambda\mathbf{I})\delta = \mathbf{J}^T\mathbf{\Sigma}^{-1}(\mathbf{y} - \mathbf{f}(\mathbf{x}, \beta)) \tag{3.5}$$

The damping parameter λ is actually adjusted in every iteration. With a small λ, the algorithm behaves similarly to the Gauss-Newton algorithm. With a large λ, the algorithms behaves similarly to the gradient descent method. With the damping parameter, the algorithm is more robust than the Gauss-Newton algorithm, especially when the functions are not well behaved.

Lastly, sometimes the covariance matrix $\mathbf{\Sigma}$ in eq. (3.4) can be replaced by diag($\mathbf{\Sigma}$) when the correlations between different y's are small. However, in the case that there are correlations, taking the full covariance matrix can reduce the error and improve the reliability. In the case that the dimension of $\mathbf{y}$ is large, the estimation of the covariance of $\mathbf{y}$ could be inaccurate. In the situation, thinning could be used to reduce the dimension of $\mathbf{y}$, i.e. only part of $\mathbf{y}$ is used in the fits. The thinning technique is also useful in terms of reducing χ^2.

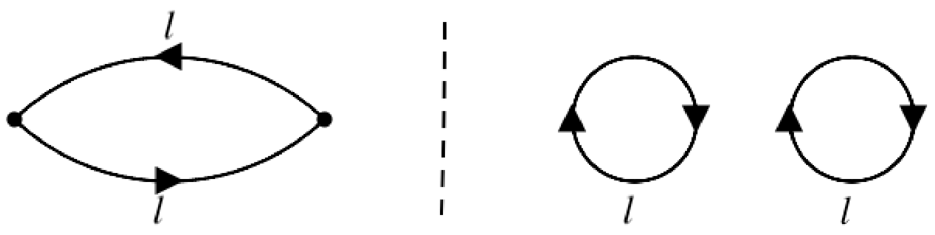


Figure 3.1: The diagrams for the connected and disconnected correlators

3.2 Fermion correlators and the results from fermion correlators

m_η and $m_{\eta'}$ can be calculated from the fermion correlators in the way that's described in chapter 1. We follow the procedure in [25] for the fermion correlators. To do the calculation, one can write down the states of η and η'. With exact SU(3) flavor symmetries, we have:

$$|8\rangle = \frac{1}{\sqrt{6}}|\bar{u}\gamma_5 u + \bar{d}\gamma_5 d - 2\bar{s}\gamma_5 s\rangle \tag{3.6}$$

$$|1\rangle = \frac{1}{\sqrt{3}}|\bar{u}\gamma_5 u + \bar{d}\gamma_5 d + \bar{s}\gamma_5 s\rangle \tag{3.7}$$

where $|8\rangle$ is the octet and $|1\rangle$ is the singlet. Since the symmetry is broken, η and η' are mixtures of the octet and the singlet. If one calculates the correlators, there will be many similar terms. With $|l\rangle = \frac{1}{\sqrt{2}}|\bar{u}\gamma_5 u + \bar{d}\gamma_5 d\rangle$ and $|s\rangle = |\bar{s}\gamma_5 s\rangle$, one can calculate the correlators using the light state and strange state. The light state and the strange state are both SU(2) singlets. In fact, this is preferred because in lattice calculation, the mass of the up quark and the mass of the down quark are usually the same because both quarks are very light. The correlators for the light and strange states have both disconnected and connected parts [25]:

$$\begin{pmatrix} \langle l(t)l^\dagger(0)\rangle & \langle s(t)l^\dagger(0)\rangle \\ \langle l(t)s^\dagger(0)\rangle & \langle s(t)s^\dagger(0)\rangle \end{pmatrix} = \begin{pmatrix} C_{ll} - 2D_{ll} & -2\sqrt{2}D_{ls} \\ -\sqrt{2}D_{ls} & C_{ss} - D_{ss} \end{pmatrix} \tag{3.8}$$

The connected correlator C_{ll} and disconnected correlator D_{ll} are explained in the diagrams of Fig 3.1. The other correlators are similar. The connected correlators are calculated by inverting the

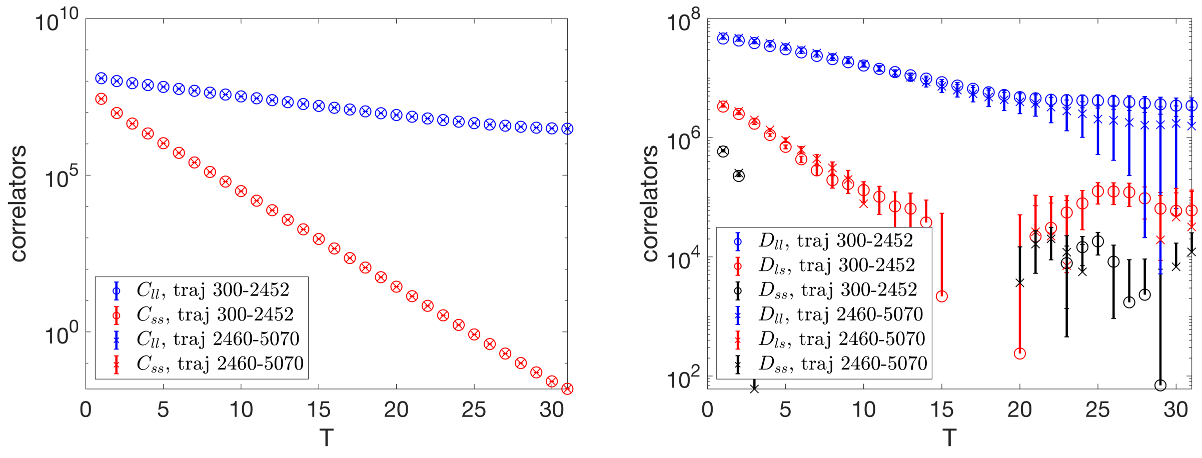


Figure 3.2: Fermion correlators. Left: C_{ll} and C_{ss}; right: D_{ll}, D_{ls}, D_{ss}

.

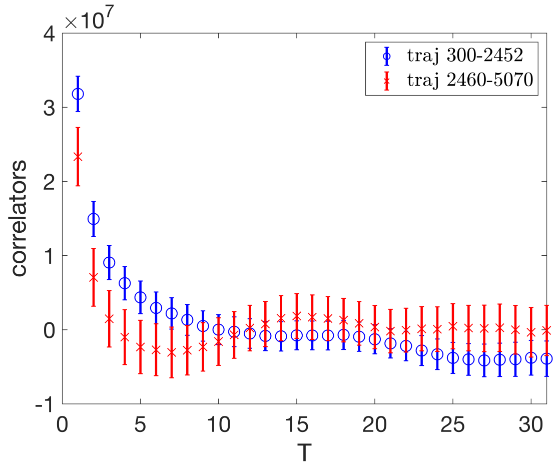
Dirac equation as described in chapter 1. In our case, we first fix the gauge to be Coulomb gauge and then use wall source and wall sink. The disconnected parts are calculated directly from the closed quark loops and that's how topology enters into m_η and $m_{\eta'}$.

$$D_{ab}(\tau) = \frac{1}{T}\sum \langle q_a(t)\gamma_5 q_a(t)\rangle\langle q_b(t+\tau)\gamma_5 q_b(t+\tau)\rangle \tag{3.9}$$

To get m_η and $m_{\eta'}$, one can do the rotation [25]:

$$\begin{pmatrix} |\eta\rangle \\ |\eta'\rangle \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}\begin{pmatrix} |l\rangle \\ |s\rangle \end{pmatrix} \tag{3.10}$$

where α is the mixing angle that relates η and η' to the SU(2) eigenstate and the strange state. Note that α is not the angle (which is usually represented by θ) that relates η and η' to the singlet state and octet state. The relation between θ and α is: $\alpha = \theta + 54.7°$ [22]. Given all the correlators, one can do simultaneous fits to get the m_η and $m_{\eta'}$ [25]. The correlators are calculated for the physical 24ID ensemble. The quark closed loops are also calculated on each time slice. For trajectory 300-2452, the measurements are done for every 4th configuration. Due to the resource limitation, for trajectory 2460-5070, the measurements are done for every 10th configuration. The related



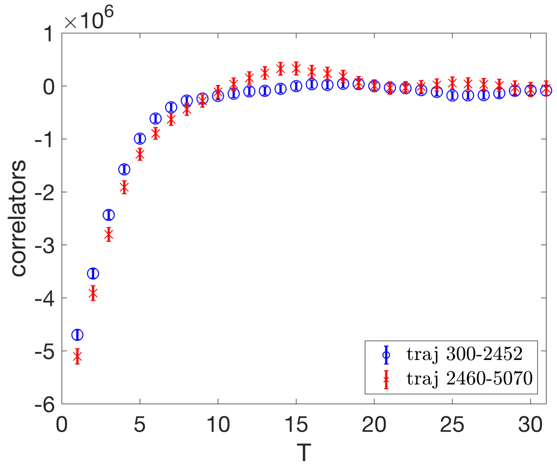


Figure 3.3: Fermion correlators. Left: $\langle l(t) l^\dagger(0) \rangle$; right: $\langle s(t) l^\dagger(0) \rangle$

.

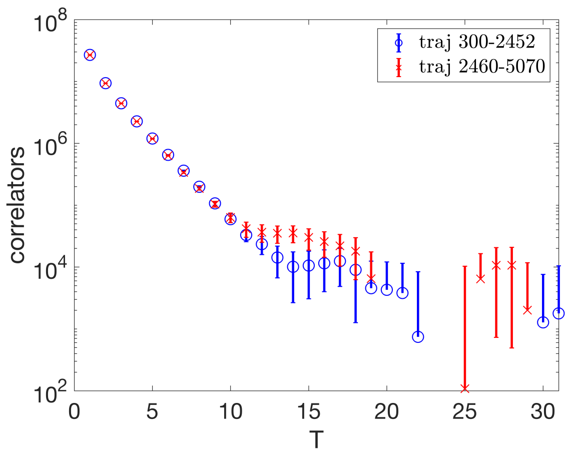


Figure 3.4: Fermion correlators, $\langle s(t) s^\dagger(0) \rangle$

correlators are plotted in Fig 3.2 to Fig 3.4. The largest differences between the two sets happen for D_{ll}, $\langle l(t) l^\dagger(0) \rangle$. This is because the signal for D_{ll} is formed by averaging all the topological samples and the error reduces slowly. This is different from the pion correlators C_{ll} where the signal to noise ratio is strong because the pion is the lightest particle.

With the fermion correlators, one can already calculate m_η and $m_{\eta'}$. In Fig 3.5, the results of m_η and $m_{\eta'}$ are shown. The labels in x-axis show the fit range $[t_{min}, t_{max}]$ of the fermion correlators. Here t is the separation between the source and the sink of the fermion fields. For η', when $t_{min} = 2$,

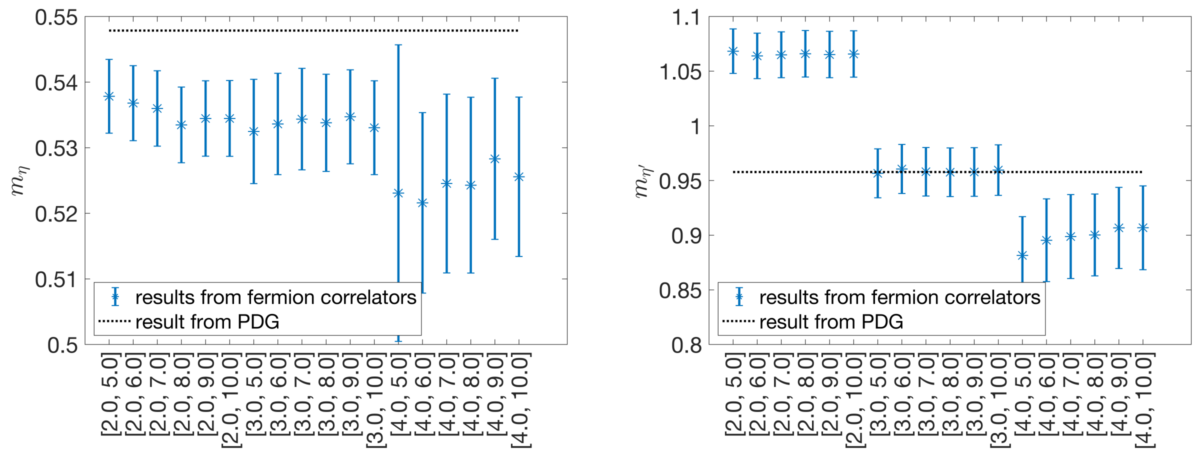


Figure 3.5: The results from the fermion correlators. Left: the mass of η; right: the mass of η'. The dashed lines show the results from PDG [22]. The labels in x-axis show the fit range $[t_{min}, t_{max}]$ of the fermion correlators.

.

the results don't agree with the physical value. This is because there is contamination from high energy states that will only disappear when the separation between the source and the sink is large. Given that the results are relatively stable when $t_{min} > 2$, we will use the fermion correlators in the range $3 \leq t \leq 10$.

However, one can increase the precision and accuracy by adding the topological charge density correlators. In the next section, we will introduce how one can use the topological charge density correlators to calculate m_η and $m_{\eta'}$. In the last section, we will compare the results calculated from fermion correlators and the results calculated from fermion correlators and topological charge density correlators.

3.3 Topological charge density correlators

The topological charge density couples to the pseudoscalar singlet meson channel. By adding an additional correlator into the fit, the error could be reduced. In addition, the topological charge density is easy to calculate because it can be directly calculated from the gauge fields and no matrix inversion is needed. In this section, we will first show how m_η and $m_{\eta'}$ are related to the topological charge density. We will describe how the correlators are calculated and how we deal with the systematic effects from the Wilson flow.

3.3.1 The relation between topological charge density correlators and m_η, $m_{\eta'}$

The effective Lagrangian for the meson singlet, octet and topological charge density is [26]:

$$\mathcal{L}_{eff} = -i \int d^4x C\phi_0 Q + D(\phi_0, \phi_8) \tag{3.11}$$

where C is a constant, Q is the topological charge density and D is the kinetic term for the mesons, represented by the singlet ϕ_0 and octet ϕ_8. Given the Lagrangian, one can calculate that the topological charge density correlator is related to m_η and $m_{\eta'}$ by [27]:

$$\langle q(x)q(y)\rangle = C_1\delta(r) - C_2\frac{K_1(m_{\eta'r})}{r} - C_3\frac{K_1(m_\eta r)}{r} \tag{3.12}$$

where K_1 is the modified Bessel function, $r = |x - y|$ and C_1, C_2, C_3 are constants. Therefore, with good signals, one can fit m_η and $m_{\eta'}$ with the topological density correlator. During the fits, it's found that if both η and η' states are included in the fits, the results are not stable. Since the fits can not differentiate two states, we only include the η' state during the fits for the topological charge density correlators. This is supported by the fact that the mixing angle θ is small and the SU(3) singlet state is mainly composed of the η' state. However, the current results will be improved if two states could be differentiated.

3.3.2 Calculating the topological charge density correlators

The topological density $q(x)$ is first calculated from the field strength as described in chapter 1. To calculate the correlators, the Fourier transformation of $q(x)$ is calculated to be $\mathcal{F}(q(x))$. Since the Fourier transformation of the correlator is the product of the Fourier transformation of the original operator, one can get the correlators by the equation:

$$\langle q(x)q(y)\rangle = \tilde{\mathcal{F}}(\mathcal{F}(q(x_1))^*\mathcal{F}(q(x_2))) \tag{3.13}$$

After the correlators are calculated, all correlators with the same $r^2 = (x-y)^2$ are averaged to improve the signal.

3.3.3 The effect of the Wilson flow

The data of the topological charge density correlators are generally very noisy and proper smearing is required. We use the Wilson flow as described in the previous chapter. For the Wilson flow, the flow time gives a smearing scale. For a small smearing range, it's expected that the correlators are not affected severely. In particular, it's pointed out that the change of the Wilson flow to the topological charge density correlators is (up to corrections of order $|x-y|^{-2}$) [27]:

$$\frac{\delta\langle q(x)q(y)\rangle}{\langle q(x)q(y)\rangle} \propto e^{-(\frac{|x-y|}{\sqrt{(8t)}}-m_{\eta'}\sqrt{8t})^2}\frac{m_{\eta'}\sqrt{8t}^{\frac{3}{2}}}{|x-y|^2} \tag{3.14}$$

So as long as the Wilson flow doesn't change the x,y dependency of the correlators, one can use the flowed results to get $m_{\eta'}$ and m_η. The correlators are plotted in Fig 3.6 for the 24ID physical ensemble. The correlators are calculated for every two configurations for trajectory 300 to trajectory 5070.

One can actually calculate the exact influence of the Gaussian smearing to the propagators. First, we notice that the Wilson flow is approximately a Gaussian smearing of the operator q with the smearing range t. Given that the correlator $\langle q(x)q(y)\rangle \equiv c$ is the multiplication of two q's and q is smeared by the range t, c is smeared by $2t$. So the smeared topological charge density correlator is related to the unsmeared topological charge density correlator by:

$$c'(x) = A\int d^4x' c(x) e^{-\frac{(x-x'^2)}{4t}} \tag{3.15}$$

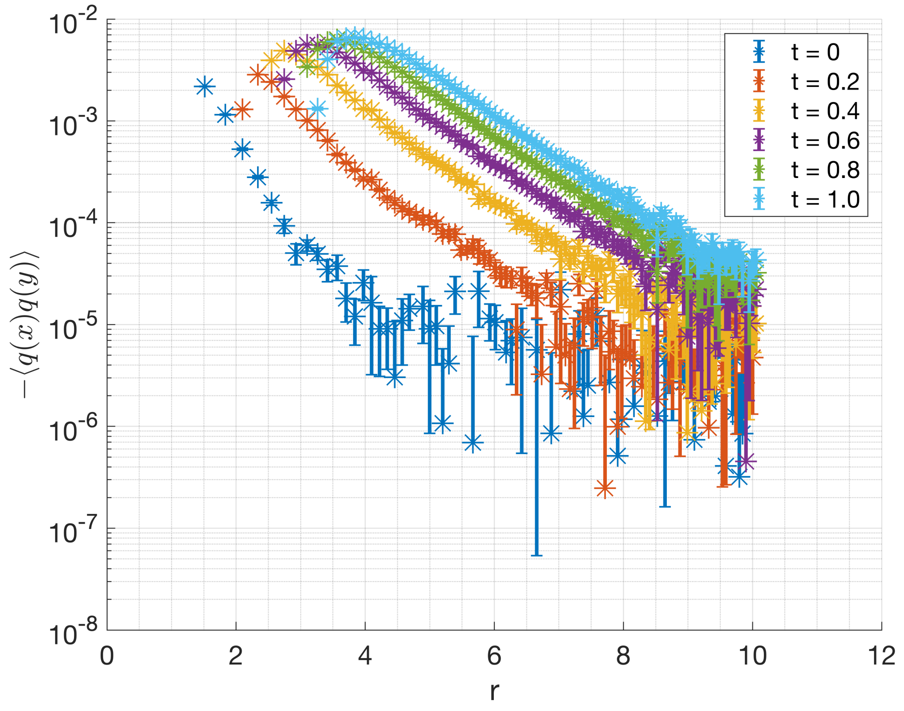


Figure 3.6: Topological charge density correlators for the 24ID physical ensemble as a function of t, the Wilson flow time.

This can be seen as the convolution between $c(x)$ and a Gaussian function e^{-x^2}. The Fourier transformation of the convolution is simply the multiplication of the Fourier transformation of the individual function. The unsmeared topological charge density correlator is calculated through the Fourier transformation of the propagator $\frac{1}{p^2+m^2}$, i.e. $c(x) = \mathcal{F}(\frac{1}{p^2+m^2})$. So we have:

$$\mathcal{F}(c'(x)) = C\frac{e^{-tp^2}}{p^2+m^2} \tag{3.16}$$

If we want to get $c'(x)$, we can apply inverse Fourier transformation to the equation. By replacing $\frac{1}{p^2+m^2}$ with an exponential integral (Schwinger's proper time integral), one can obtain the following result for $c'(x)$:

$$c'(r,t,m,A) = Ae^{tm^2}[\frac{m}{r}K_1(mr) - \int_0^t dl\frac{1}{4l^2}e^{-\frac{r^2}{4l}-lm^2}] \tag{3.17}$$

The formula has an integral but it can still be used to fit the data. An example is plotted in Fig 3.7 for $m = 1$, $t = 0.6$. In fact, one can fit the parameter t to the correlators. From the calculation, the fitted flow time should be twice the Wilson flow time. In Fig 3.8, the fitted flow time is plotted against the Wilson flow time and it's confirmed. In the graph, the blue data are fitted values by including the topological charge density correlators only. The red data are fitted values by including both the topological correlators as well as the fermion correlators. Both results show that the fitted flow time is roughly twice the Wilson flow time. This verifies our analysis of the influence of the Wilson flow.

3.4 The autocorrelation of the correlators

In section 3.2, we mentioned that due to resource limitation, we measured fermion correlators at different intervals for trajectory 300 to 2452 and trajectory 2460 to 5070. We measured for every 4 configurations for trajectory 300 to 2452 and we measured for every 10 configurations for trajectory 2460 to 5070. To combine the measurements, we need to understand the autocorrelation time of the correlators [28]. Here the time means molecular dynamic time or evolution time. In the case that the autocorrelation time is very short, for example, 1, since each configuration is

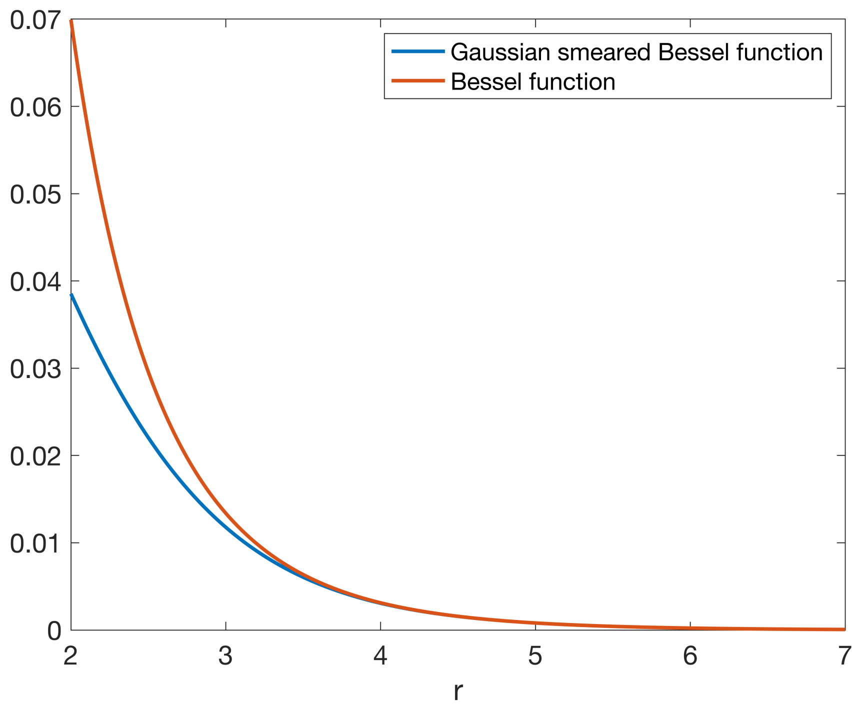


Figure 3.7: Comparison between a Bessel function and a Gaussian smeared Bessel function. $m = 1$, $t = 0.6$

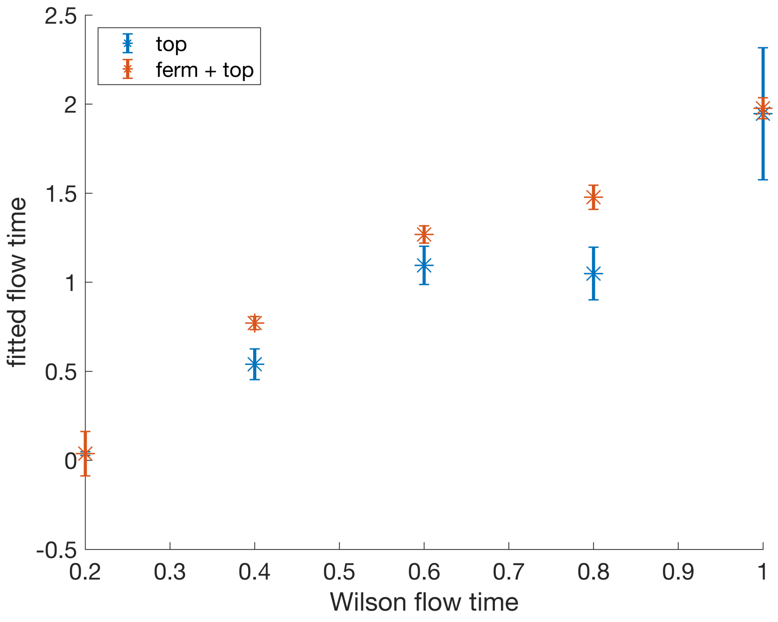


Figure 3.8: Fitted flow time against the Wilson flow time. The blue data are obtained when only the topological charge density correlators are included. The red data are obtained when both fermion correlators and topological charge density correlators are included. The graph demonstrates that with the Wilson flow time t, the topological charge density correlator is smeared with range $2t$.

independent from each other, we don't need to care that we measured at different intervals. We can simply treat each measurement as independent measurement. In the case that the autocorrelation time is long, for example, 20, then we need to treat the measurements for trajectory 300 to 2452 and the measurements for trajectory 2460 to 5070 differently. For example, we might bin every 5 measurements for trajectory 300 to 2452 and bin every 2 measurements for trajectory 2460 to 5070. We have to measure the autocorrelation time to know how we should combine the data together.

In order to find the autocorrelation time, we measure the autocorrelation function. Given measurements $\{x_i\}$, the normalized autocorrelation function is defined as:

$$c_i = \frac{\sum_{j=0}^{j=N-i}(x_j - \bar{x})(x_{i+j} - \bar{x})}{\sum_{j=0}^{j=N}(x_j - \bar{x})(x_j - \bar{x})} \tag{3.18}$$

where $\bar{x}$ is the average. The integrated autocorrelation time is calculated as:

$$\tau = 1 + \sum_{i=1}^{i<k, c_k<0} 2c_i \tag{3.19}$$

The summation stops when $c_k < 0$ because we don't want to include the noise. However, the formula above is slightly biased. The measured integrated autocorrelation time would be larger than the real integrated autocorrelation time when the measurements are independent from each other. In the case that the integrated autocorrelation time is 1, the equation above would give a result larger than or equal to 1. So it's likely that τ given by the above equation would be larger than the real integrated autocorrelation time.

In Table 3.1, the integrated autocorrelation time is calculated for different correlators. All the numbers in the table are in molecular dynamic units. For example, for trajectory 300-2452, C_{ll}, the measurements are done for every 4 configurations, so after measuring the integrated autocorrealtion time, the raw result is multiplied by 4. Similarly, for trajectory 2460-5070, the measurements are done for every 10 configurations, so the raw result is multiplied by 10. This applies to $C_{ll}, C_{ss}, D_{ll}, D_{ls}, D_{ss}, \langle l(t)l^\dagger(0)\rangle, \langle l(t)s^\dagger(0)\rangle, \langle s(t)s^\dagger(0)\rangle$. For topological correlators, the mea-

	τ for traj 300-2452	τ for traj 2460-5070
Top correlator, t=0.2	2.50(0.23)	2.42(0.22)
Top correlator, t=0.4	3.11(0.41)	3.20(0.54)
Top correlator, t=0.6	3.94(0.60)	4.51(0.82)
Top correlator, t=0.8	4.74(0.91)	5.54(0.98)
Top correlator, t=1	5.21(0.94)	5.99(1.16)
C_{ll}	6.50(0.22)	11.63(1.32)
C_{ss}	7.83(2.32)	12.72(1.36)
D_{ll}	6.49(0.20)	11.34(0.15)
D_{ls}	6.54(0.37)	12.65(1.65)
D_{ss}	7.88(1.12)	12.10(1.53)
$\langle l(t)l^{\dagger}(0)\rangle$	6.56(0.22)	11.38(0.17)
$\langle l(t)s^{\dagger}(0)\rangle$	6.54(0.37)	12.65(1.65)
$\langle s(t)s^{\dagger}(0)\rangle$	7.84(0.98)	12.33(1.56)

Table 3.1: Integrated autocorrelation time for different correlators in molecular dynamic unit

surements are done for every 2 configurations for trajectory 300-2452 and for trajectory 2460-5070, so the raw results are multiplied by 2 in both cases. For fermion correlators, the results are the average results of $t = 3$ to $t = 10$, which is the range that will be used in the fits. For topological correlators, the results are the average results of $r = 3$ to $r = 10$. The numbers in the brackets are the standard deviations of τ calculated from different t or r.

The topological correlators have small integrated autocorrelation time at short flow time. At large flow time, the field is smeared at a larger range and the integrated autocorrelation time increases. However, the integrated autocorrelation time is always smaller than 6.

For fermion correlators, the measurements for trajectory 2460-5070 are roughly independent (done for every 10 configurations) because the calculated integrated autocorrelation length is close to 10. Note that the smallest integrated autocorrelation time that can be measured is 10 when the

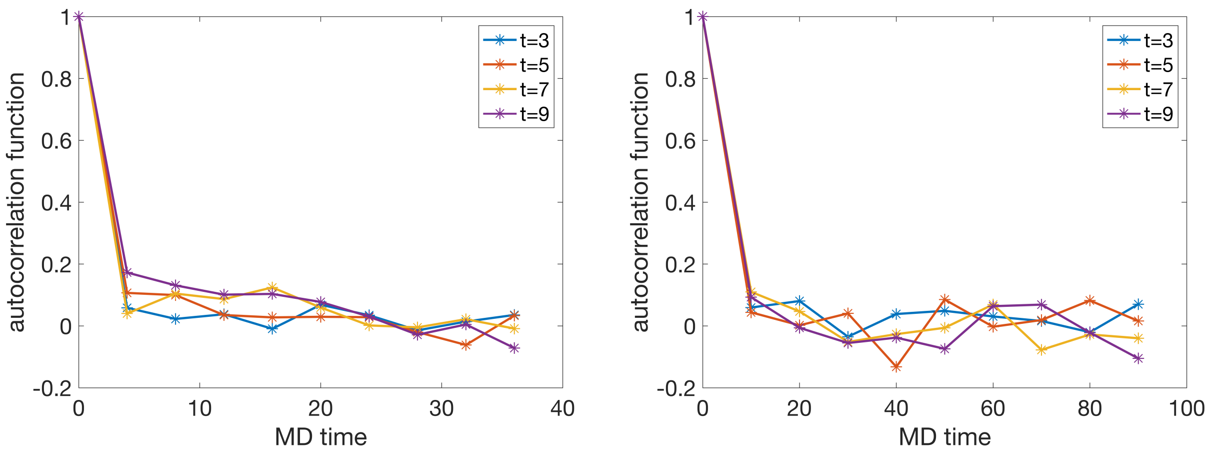


Figure 3.9: Autocorrelation functions for C_{ss}. Left: autocorrelation functions for trajectory 300 to trajectory 2452; right: autocorrelation functions for trajectory 2460 to trajectory 5070. The graphs show that one cannot use the linear extrapolation to calculate the integrated autocorrelation time when the separation between the measurements is large. The most accurate calculation of the integrated autocorrelation time is obtained when the interval between individual measurement is small

.

measurements are separated by 10. The measurements for trajectory 300-2452 have integrated autocorrelation time 6.5-7.88. This seems inconsistent with the integrated autocorrelation time in trajectory 2460-5070. The reason can be explained by the fact that eq. (3.19) tends to give a value larger than 1 when the data have zero autocorrelation. For trajectory 2460-5070, by looking at the autocorrelation function plotted in Fig 3.9, we can see that that the autocorrelation function drops to a small value at molecular dynamic time 10. However, due to a linear interpolation, eq. (3.19) gives a value that's around 12. For the autocorrelation function plotted for trajectory 300-2452, the autocorrelation function is positive but small for MD units 4-10. This means that the linear interpolation used for trajectory 2460-5070 is actually problematic and eq. (3.19) tends to overestimate the integrated autocorrelation length when the separation between each measurements are large. Therefore, the integrated autocorrelation time for fermion correlators should be 6-8 molecular dynamic units. However, it should be noted that although the integrated autocorrelation time is 6-8 molecular dynamic units, it doesn't mean that there is no small autocorrelation when the separation is larger than 8.

To make sure the above observation is correct, we also measured the integrated autocorrelation time for trajectory 300-2452 with separation 8 and we measured the integrated autocorrelation time for trajectory 300-2452 with binning. Then the integrated autocorrelation time of trajectory 300-2452 agrees with the integrated autocorrelation time of trajectory 2460-5070. This again demonstrates that if the measurements are done with a large separation, the measured integrated autocorrelation length tends to be large and the most accurate integrated autocorrelation time is calculated when the measurements are done with a small separation.

Since the integrated autocorrelation time is 6-8 molecular dynamic units for fermion correlators, we bin the measurements of 300-2452 so that the separation between each measurements is 8 molecular dynamic units.

3.5 m_η and $m_{\eta'}$

Now we present the final results of the measurements for η and η'. For clarity, we summarize our fitting methods:

1) We use correlated fits. For topological correlators, we thin the data so that the separation between r_i and r_{i+1} is larger than 0.5. In this way, the size of the covariance matrix is smaller and we have a better estimation of the covariance matrix.

2) We bin the fermion measurements of trajectory 300-2452 so that the separation between each data is 8 molecular dynamic units. For the fermion measurements of trajectory 2460-5070, we treat each measurements as independent. The topological correlators are binned so that the separation is the same as the fermion correlators.

3) The topological correlators and fermion correlators are fitted together through simultaneous fits.

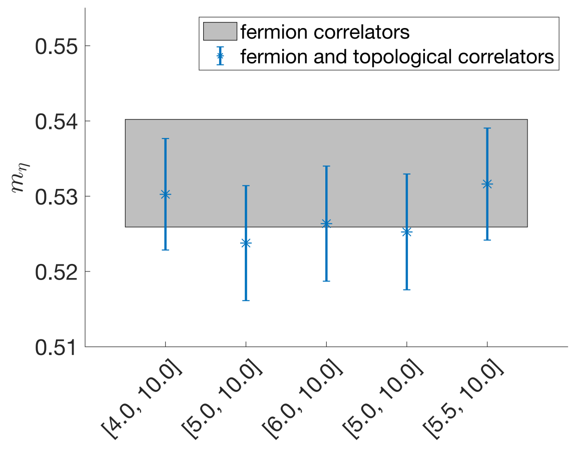
4) Jackknife resampling is used to smooth the data.

5) For the topological correlators, we only include the η' state.

The results are plotted in Fig 3.10. For pure fermionic fits, the error for m_η is 0.0072 and the error for $m_{\eta'}$ is 0.023. When we also include the topological correlators, the error for m_η is around 0.0076 and the error for $m_{\eta'}$ is around 0.018-0.021. Since the p-value calculated from χ^2 is the largest for flow time 0.2, we choose the results from flow time 0.2 as our final results. In Table 3.2, we compare our results with the Particle Data Book [22] and the previous results from [25] by the RBC-UKQCD Collaboration where only fermion correlators are taken into consideration. We also include the mixing angle θ.

It can be seen that the error of $m_{\eta'}$ is decreased by a factor of 7 compared to [25], bringing the error to 2.1%. In addition, m_η is also closer to the value from PDG. The mixing angle from our results is quite different from the results in [25] but with a much better precision.

There are several reasons that the results are different. The most important reason is that the results are obtained from different ensembles. The ensemble in our work is a physical ensemble



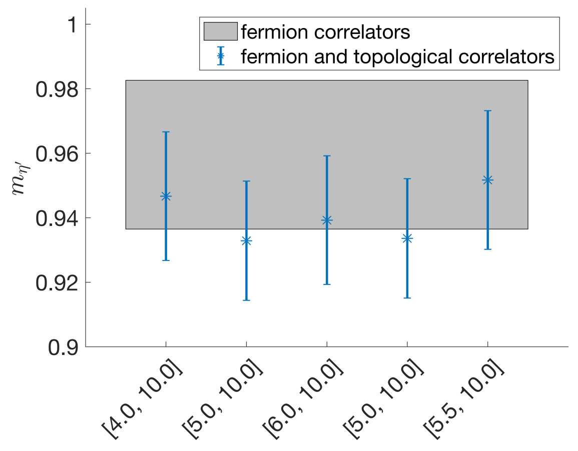


Figure 3.10: Fitted results for the mass of η (left) and η' (right). The results are in lattice unit and the lattice spacing is approximately 1GeV. On both graphs, the shaded area denotes the range for the mass obtained by using the fermion correlators only. The five error bars on each graph are obtained by using both the fermion correlators and the topological charge density correlators. The difference in the five results is that the Wilson flow time is separately 0.2, 0.4, 0.6, 0.8 and 1.0. The parameters on the x-axis denotes the range of the topological charge density correlators that's used in the fit.

.

	This work	PRL in 2010 [25]	PDG [22]
m_η	530(7)	573(6)	547.862(17)
$m_{\eta'}$	947(20)	947(142)	957.78(6)
θ	$-21.5(1.2)^\circ$	$-14.1(2.8)^\circ$	-10° to -20°

Table 3.2: The results compared between this work, previous results from [25] by the RBC-UKQCD and PDG [22]

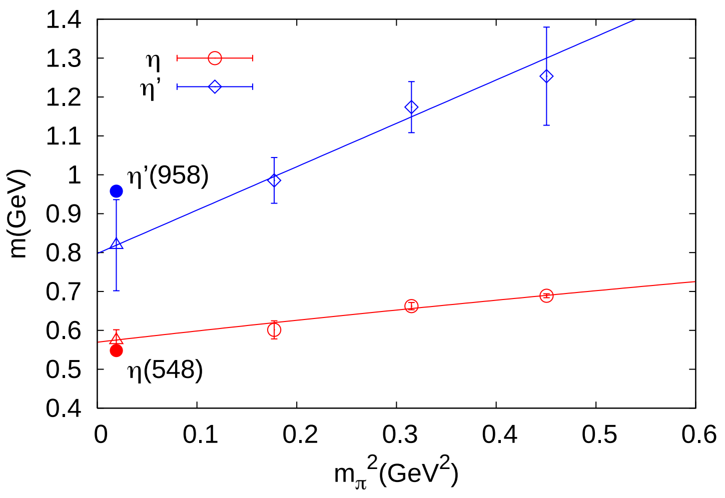


Figure 3.11: Extrapolation of m_η and $m_{\eta'}$ to the physical light quark mass. Quoted from [25]

with $a^{-1} \approx 1$GeV. The lattices in [25] have the lattice size $16^3 \times 32 \times 16$ and the lattice spacing is $a^{-1} \approx 1.73$GeV. In addition, while the light quark mass and heavy quark mass are chosen in the current ensemble so that the pion mass is quite close to 140MeV, [25] calculated the results for different choices of quark masses and extrapolated the results to the physical light quark mass limit (Fig 3.11). This is the main reason that the results in this work are different. Methodologically, the most significant difference is that we included the topological correlators which decreases the error of η' by 18% while [25] only used fermion correlators.

However, to further improve the results from the current work, we should include more ensembles to extrapolate the results to the continuum and infinite volume limit. Adding more ensembles will help understanding the systematic errors and getting more accurate results. It would also be beneficial to compare the mixing angle from different ensembles. Nevertheless, we have improved the measurements greatly by including the topological correlators.

Chapter 4: The multisplitting preconditioned conjugate gradient algorithm

The development of lattice QCD depends highly on the improvements of the computers and the algorithms. One of the most resource demanding process is the Dirac matrix inversion. This is needed when fermions are involved. The Dirac matrix is a large, sparse square matrix. The number of rows can easily go to a few billion. With such a large matrix, the standard Gauss-elimination won't work. However, the matrix is sparse because the matrix is a discretization of the fermion derivative operator. The standard algorithm for the inversion of a large sparse matrix is the conjugate gradient (CG) algorithm. However, the conjugate gradient algorithm is slow when the condition number of the matrix is large where the condition number is determined by the ratio of the maximum and minimum eigenvalues: $\lambda_{max}/\lambda_{min}$. To make the lattice QCD simulation work with large lattices and small fermion mass, the algorithm must be improved.

To be specific, there are two situations that the inversion is needed, the gauge evolution phase and the measurement phase. In the measurement phase, the Dirac matrix that's associated with one gauge field would be solved many times. Therefore, one can do some calculations in advance to improve the condition number of conjugate gradient. For example, one can calculate some eigenvectors for the Dirac matrix. Then during the measurements, one can use the eigenvectors to improve the condition number. This turns out to be very effective. The lattice QCD groups have been generating eigenvectors using the implicitly restarted Lanczos algorithm [20] for the measurements. Since the eigenvectors would be used many times, the cost of the eigenvector generation is justified. Other algorithms such as EigCG are also used where the inexact eigenvectors are calculated during the conjugate gradient algorithm.

The methods mentioned above work only in the measurement phase. During the evolution, there is usually only one or a few inversions for the Dirac matrix associated with one gauge field. Therefore, generating the eigenvectors is not worthwhile.

The topic of this chapter, the multisplitting preconditioned conjugate gradient is an algorithm suitable for the inversion during the evolution. The algorithm benefits from the properties of the modern computers and supercomputers. Due to the large size of the lattices in QCD, most of the calculation is done on a supercomputer composed of many nodes where each node is a computer. The modern computers have processors that can process very fast. However, since the calculation is done on many nodes, communication among the nodes is needed and the speed of the communication is very slow compared to the processing speed of each node. The multisplitting preconditioned conjugate gradient utilizes this property of the supercomputers by doing more computation on each node locally and avoiding inter-node communication.

The algorithm, as the name, is originated from two algorithms, the multisplitting algorithm and the preconditioned conjugate gradient algorithm. The multisplitting algorithm is able to solve the inversion by doing more local computation. This is used in the preconditioned conjugate gradient algorithm to reduce the condition number. In particular, the multisplitting algorithm reduces the condition number by solving the high modes of the matrix locally. In next chapter, we will show another algorithm that will solve the low modes of the matrix in the preconditioner.

In this chapter, we will first introduce the conjugate gradient algorithm and the preconditioned conjugate gradient algorithm. In the second section, the multisplitting algorithm will be discussed. We will also talk about the complexity caused by the boundary terms during the implementation. Lastly, we will show how the multisplitting preconditioned conjugate gradient algorithm can be used to solve the Dirac equation.

4.1 The conjugate gradient algorithm and the preconditioned conjugate gradient algorithm

The conjugate gradient algorithm is an algorithm based on the Krylov space for a positive definite Hermitian matrix. The Krylov space is critical to an iterative algorithm. With a vector r and a matrix A, the Krylov space is defined as: $\{r, Ar, A^2r, ...\}$. The space appears naturally because in an iterative algorithm, the matrix will be multiplied to a vector repeatedly. The conjugate gradient algorithm minimizes $|e_n|_A^2$ for the linear equation $Ax = b$ in the Krylov subspace, where $e_n = x_n - x$

and $|e_n|_A^2 = \langle e_n|A|e_n\rangle$. The algorithm is shown in Algorithm 1.

Algorithm 1: The conjugate gradient algorithm

```
**Result:** solution for $Ax = b$
$r_0 = b - Ax_0$
$p_0 = r_0$
$k = 0$
**while** $|r_{k+1}| > \epsilon$ **do**
    $\alpha_k = \frac{|r_k|^2}{\langle p_k|A|p_k\rangle}$
    $x_{k+1} = x_k + \alpha_k p_k$
    $r_{k+1} = r_k - \alpha_k A p_k$
    $\beta_k = \frac{|r_k+1|^2}{|r_k|^2}$
    $p_{k+1} = r_{k+1} + \beta_k p_k$
    $k = k + 1$
**end**
```

The algorithm has several properties. For example, $\langle r_i r_j\rangle = 0$ for $i \neq j$. Also, $\langle p_i|A|p_j\rangle = 0$ for $i \neq j$. The property says that p_i and p_j are conjugate with respect to A. This is the origin of the word "conjugate".

As mentioned above, the algorithm minimizes $|e_n|_A^2$ in the Krylov space. From this property, one can deduce the convergence rate of the algorithm:

$$\begin{aligned} |e_n|_A^2 &= \min_{p_n} |p_n(A)e_0|_A^2 \\ &\leq \min_{p_n} \max_{\lambda} |p_n(\lambda)|^2 |e_0|_A^2 \\ &\leq 2(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1})^n |e_0|_A^2 \\ &\approx 2(1 - \frac{2}{\sqrt{(\kappa)}})^n |e_0|_A^2 \end{aligned} \tag{4.1}$$

where p_n is an order n polynomial, λ is the eigenvalue of the matrix A and κ is the condition number defined as: $\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$. Now we see clearly that the convergence rate of the algorithm depends highly on the largest and the smallest eigenvalues. In the situation where the eigenvalues are clustered to several values, the algorithm could converge faster than the limit because there are low-order polynomials that can fit the spectrum well [29]. In the case of lattice QCD, the common behavior is that the algorithm converges fast initially and then the convergence rate approaches the limit.

Since the conjugate gradient algorithm can be slow when the condition number is large. "Preconditioning" is invented to reduce the condition number. It means that a variation of the matrix instead of the original matrix is used in the algorithm to reduce the condition number. In particular, note that the solution of $Ax = b$ is equivalent to the solution of:

$$(E^{-1}A(E^{-1})^T)E^T x = E^{-1}b \tag{4.2}$$

Therefor, one can solve for $E^T x$ with matrix $E^{-1}A(E^{-1})^T$ and then find x. By choosing a proper E, the condition number of A can be reduced and the convergence rate can be improved. In practice, it's found that the conjugate gradient algorithm for $E^{-1}A(E^{-1})^T$ can be simplified as the preconditioned conjugate gradient algorithm by using the matrix $M = EE^T$. The matrix M, sometimes M^{-1}, is called the preconditioner. The algorithm is shown below.

The convergence rate is now related to the condition number of $M^{-1}A$. The benefit from the convergence rate will justify the cost of M^{-1} when the convergence rate is fast enough and the cost of M^{-1} is small enough. Note that the algorithm still requires M to be symmetric and positive definite.

In practice, the preconditioning matrix M could change between different iterations. The algorithm could still work with a slight modification. Instead of calculating β by $\frac{\langle z_{k+1}|r_{k+1}\rangle}{\langle z_k|r_k\rangle}$. One could calculate $\beta = \frac{\langle (z_{k+1}-z_k)|r_{k+1}\rangle}{\langle z_k|r_k\rangle}$. The convergence rate could improve dramatically when the modification is applied. This is called the flexible preconditioned conjugate gradient. The formula

Algorithm 2: The preconditioned conjugate gradient algorithm

```
Result: solution for $Ax = b$
$r_0 = b - Ax_0$
$z_0 = M^{-1} r_0$
$p_0 = z_0$
$k = 0$
while $|r_{k+1}| > \epsilon$ do
    $\alpha_k = \frac{\langle r_k | z_k \rangle}{\langle p_k | A | p_k \rangle}$
    $x_{k+1} = x_k + \alpha_k p_k$
    $r_{k+1} = r_k - \alpha_k A p_k$
    $z_{k+1} = M^{-1} r_{k+1}$
    $\beta_k = \frac{\langle z_{k+1} | r_{k+1} \rangle}{\langle z_k | r_k \rangle}$
    $p_{k+1} = z_{k+1} + \beta_k p_k$
    $k = k + 1$
end
```

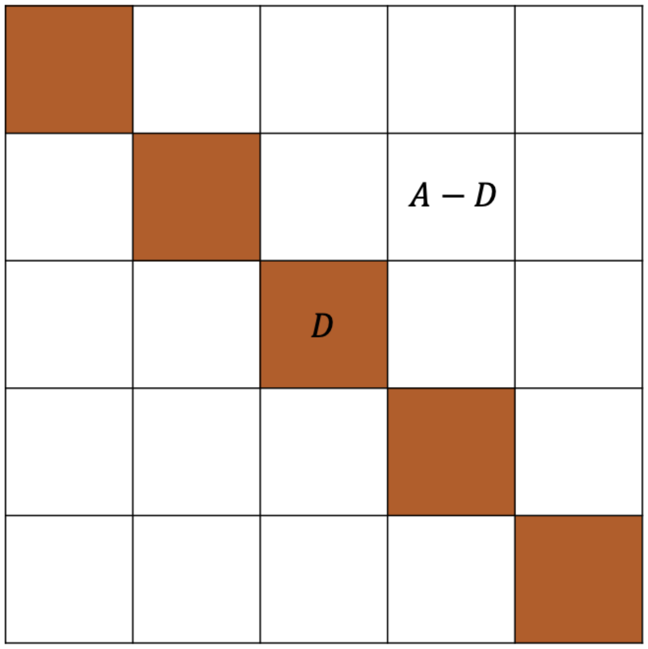


Figure 4.1: Matrix split in the Jacobi algorithm

$\beta = \frac{\langle (z_{k+1}-z_k)|r_{k+1}\rangle}{\langle z_k|r_k\rangle}$ is called the Polak–Ribière formula and the formula $\beta = \frac{\langle z_{k+1}|r_{k+1}\rangle}{\langle z_k|r_k\rangle}$ is called the Fletcher–Reeves formula.

In this chapter, we will use the multisplitting algorithm as the preconditioner for the conjugate gradient algorithm. The multisplitting algorithm will be introduced in the next section.

4.2 The multisplitting algorithm

The multisplitting algorithm is proposed in [30] for solving large linear systems in parallel. To illustrate the multisplitting algorithm, it's useful to start from the Jacobi algorithm which can be used for the inversion of a strictly diagonally dominated matrix. The Jacobi algorithm splits the matrix A to two parts, the diagonal part D and the rest $A - D$ (Fig 4.1). Then starting from x_0, the solution is updated as:

$$x_{k+1} = D^{-1}(b - (A - D)x_k) \tag{4.3}$$

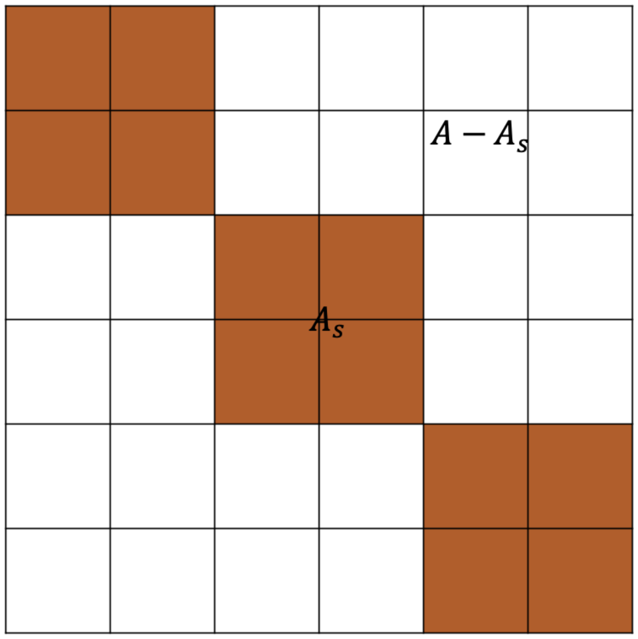


Figure 4.2: Matrix split in the multisplitting algorithm

The Jacobi algorithm has a very interesting property that it solves the high modes first. This property will be used again in the next chapter and we will explore more properties of the Jacobi algorithm. The multisplitting algorithm is very similar to the Jacobi method except that the matrix A is split into blocks. In the Jacobi method, the diagonal elements are taken out as D. In the multisplitting algorithm, the matrix A is split into blocks of sub-matrices and the sub-matrices that are at the diagonal position are taken out. To make the terminology clear, let's call the sub-matrices that are at the diagonal position A_s (Fig 4.2). Then the algorithm updates the solution according to the equation:

$$x_{k+1} = A_s^{-1}(b - (A - A_s)x_k) \tag{4.4}$$

The multisplitting algorithm, similar to the Jacobi method, also requires the diagonal elements to be positive. This algorithm is suitable for lattice QCD calculation to reduce the communication because of the way that the calculation is split to different nodes. During the lattice QCD calculation, the lattice is split in spacetime directions to the computation nodes. On each node there is a sub lattice. The Dirac matrix multiplication is first done on the sub lattice. Then the communication happens between the nodes and the final calculation is done. Therefore, it's natural to use the Dirac matrix multiplication on each node as the diagonal sub-matrices A_s. By solving $A_s^{-1}b$ locally, there will be less communication. This is the reason that the multisplitting algorithm is suitable for the Dirac inversion.

4.3 The complexity of the boundary terms

The lattice community has been using the even-odd preconditioning for the Dirac matrix to reduce the size of the problem. The detailed implementation of the Dirac matrix makes the implementation of the multisplitting algorithm non-trivial because of the boundary terms. To explain it in detail, it's best to explain the even-odd preconditioning first [31]. A lattice site is called even if $(x + y + z + t) \mod 2$ is 0 and a lattice site is called odd if $(x + y + z + t) \mod 2$ is 1. The Dirac matrix can then be rearranged so that the matrix is composed of four sub-matrices, D_{ee}, D_{eo}, D_{oe} and D_{oo}. The sub-matrix D_{ee} connect the even sites to even sites and the other sub-matrices are similar. Since D_{ee} and D_{oo} can be analytically inverted, the linear equation can be decomposed through the Schur decomposition:

$$\begin{pmatrix} D_{ee} & D_{eo} \\ D_{oe} & D_{oo} \end{pmatrix} \begin{pmatrix} \psi_e \\ \psi_o \end{pmatrix} = \begin{pmatrix} D_{ee} & D_{eo} \\ 0 & D_{oo} \end{pmatrix} \begin{pmatrix} 1 - D_{ee}^{-1} D_{eo} D_{oo}^{-1} D_{oe} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ D_{oo}^{-1} D_{oe} & 1 \end{pmatrix} \begin{pmatrix} \psi_e \\ \psi_o \end{pmatrix} = \begin{pmatrix} \phi_e \\ \phi_o \end{pmatrix} \tag{4.5}$$

The solution can be gotten by solving the equivalent equation:

$$(1 - D_{ee}^{-1} D_{eo} D_{oo}^{-1} D_{oe})\psi_e \equiv D_{prec}\psi_e = D_{ee}^{-1}(\phi_e - D_{eo} D_{oo}^{-1}\phi_o) \equiv \hat{\phi}_e \tag{4.6}$$

After ψ_e is solved, the odd part ψ_o can be gotten by:

$$\psi_o = D_{oo}^{-1}(\phi_o - D_{oe}\psi_e) \tag{4.7}$$

Since the conjugate gradient algorithm requires the matrix to be Hermitian and positive definite, the usual way to solve eq. (4.6) is to multiply $D^{\dagger}_{prec}$ on both sides and solve the normal equation:

$$D^{\dagger}_{prec} D_{prec} \psi_e = D^{\dagger}_{prec} \hat{\phi}_e \tag{4.8}$$

By even-odd preconditioning, the size of the problem is reduced by a factor of two. However, it means that the matrix A in the multisplitting algorithm should be $D^{\dagger}_{prec} D_{prec}$. Finding the proper sub-matrices of $D^{\dagger}_{prec} D_{prec}$ needs extra care. For example, one can not use the sub-matrices D_{ee} or D_{eo} to construct the sub-matrices of $D^{\dagger}_{prec} D_{prec}$ in a naive way. In fact, The application of D_{prec} would introduce some components that are outside of the local lattice. These components are called the boundary terms. The boundary terms have to be stored in the memory and have to be included properly when $D^{\dagger}_{prec}$ is applied. Our tests show that the inclusion of these terms is critical to the convergence of the multisplitting algorithm.

4.4 The multisplitting preconditioned conjugate gradient algorithm

Although the multisplitting algorithm is suitable for parallel calculation and is good at reducing the communication and utilizing the computational power of the local node, the convergence rate of the algorithm is usually slow. The key point of making the algorithm beneficial is to combine the algorithm with the preconditioned conjugate gradient algorithm. The idea relies on the fact that the multisplitting algorithm solves the high modes of the Dirac operator efficiently. This is because the high modes are mostly local. When the sub-matrices are inverted, the high modes are solved. Working as a preconditioner, λ_{max} is reduced when the high modes are solved. The convergence rate, which depends on the condition number $\lambda_{max}/\lambda_{min}$, will be improved.

Specifically, the preconditioned conjugate gradient solves z with the preconditioner M:

$$z_k = M^{-1} r_k \tag{4.9}$$

With the multisplitting preconditioned conjugate gradient algorithm, the step is:

$$z_k = A_{ms}^{-1} r_k \tag{4.10}$$

where the action of A_{ms}^{-1} means that we solve the equation $Az_k = r_k$ with the multisplitting algorithm. One certainly cannot solve eq. (4.10) to very high precision with the multisplitting algorithm. In fact, only one iteration of the multisplitting algorithm is used so that there is only one communication for the multisplitting step. In addition, when the sub-matrices are solved (eq. (4.10)), they are not solved to high precision. The sub-matrices are solved by the conjugate gradient algorithm. To avoid confusion, we call the conjugate gradient iterations for the sub-matrices as inner iterations. The inner iteration number is also not large since the local computation power is limited. In Fig 4.3, the residuals of the multisplitting algorithm with different numbers of inner iteration number are shown. The matrix inversion is done for a $32 \times 64 \times 12$ ID lattice with 128 KNL nodes. It can be seen that with only 3 local iterations, the number of iterations is reduced by 33% with means that the local high modes are solved. The fact that the local sub-matrices don't need to be solved exactly is also important for the algorithm to work.

The algorithm actually shares the idea of the domain decomposition method. In the domain decomposition method, a problem is split into smaller problems on the subdomains and then the solutions on the subdomains are coordinated. The process is iterated to find the true solution. In [32], the idea is used as a preconditioner for the generalized conjugate residual algorithm for the Wilson Dirac operator. In this work the idea is used for the domain wall Dirac operator. The domain decomposition method is also called the additive Schwarz method and was used for the inversion of the Dirac equation in [32][33][34].

The implementation of the algorithm is in fact highly non-trivial. In [35], the implementation is described for supercomputers with GPUs. Techniques like kernel fusion are used to utilize the tensor cores in GPUs. The detailed implementation requires careful utilization of the various units of the computers. The final speedup is found to be as high as 1.22x on SUMMIT. Since the author

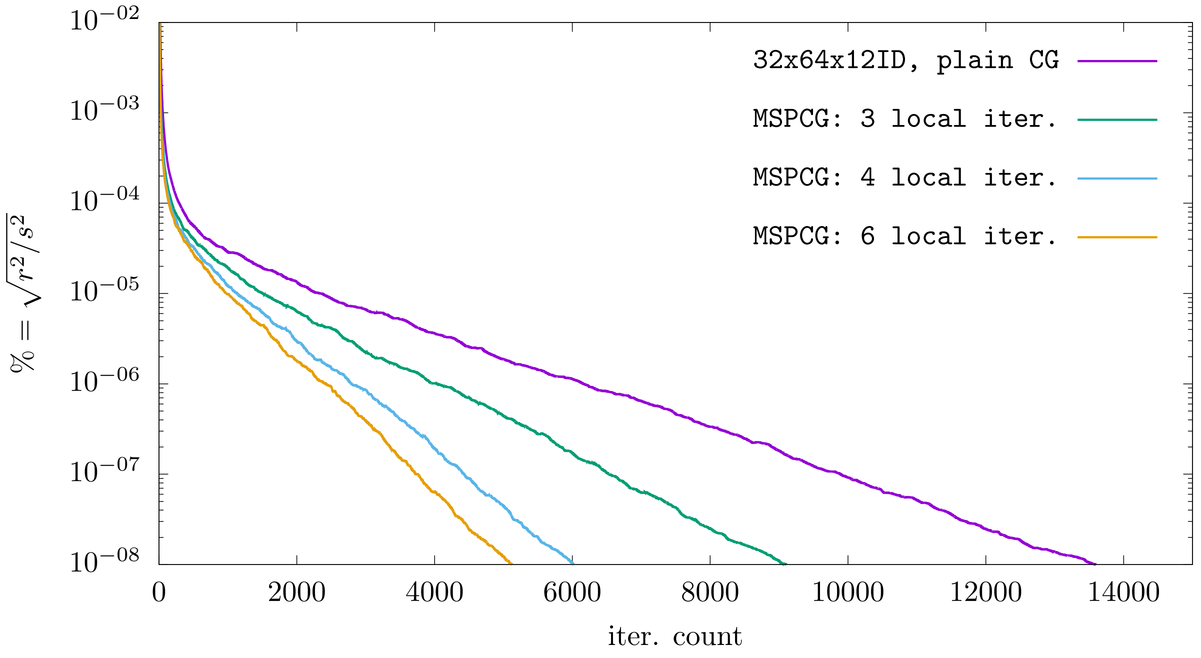


Figure 4.3: Residual for the multisplitting algorithm

mainly worked on the algorithmic part, the detailed implementation is out of the scope for the discussion.

Chapter 5: Renormalization group based preconditioned conjugate gradient algorithm

In the last chapter, the multisplitting preconditioned conjugate gradient algorithm is developed. The preconditioner reduces the condition number by solving the high modes. In this chapter, we develop different algorithms which reduce the condition number by solving the low modes with preconditioners based on renormalization group ideas.

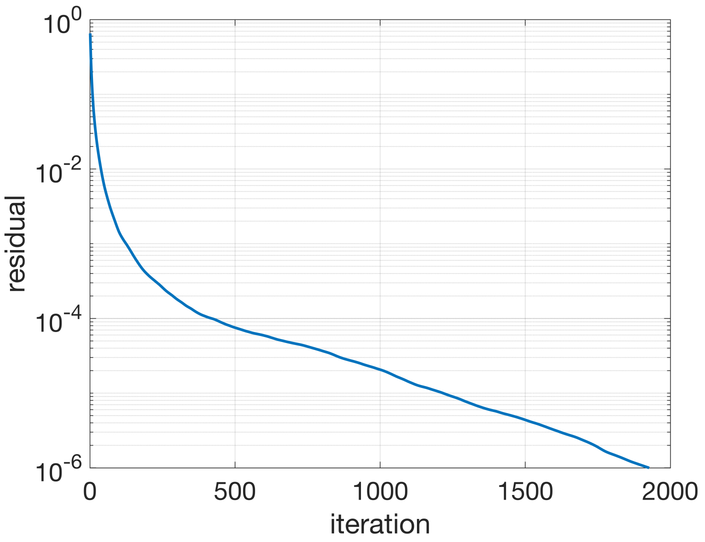


Figure 5.1: The residual as a function of the iteration number using the conjugate gradient algorithm

To understand why this approach is promising, we can take a look at Fig 5.1 where the residual is plotted against the number of iterations needed to reach the precision 10^{-6} (iteration number) using the conjugate gradient algorithm. The equation to be solved is $Dx = y$ where D is the Mobius domain wall operator on a $24^3 \times 64 \times 12$ lattice. We solve the modified equation $D^\dagger Dx = D^\dagger y$. The residual is defined as $|Ax - b|/|b|$ where $A = D^\dagger D$ and $b = D^\dagger y$. Fig 5.1 plots the typical behavior of the residual for the widely used conjugate gradient algorithm in lattice QCD. As discussed in the last chapter, the convergence rate of the conjugate gradient algorithm depends on the condition number $\lambda_{max}/\lambda_{min}$ but the convergence rate could be faster if the eigenvalues are clustered. Initially, the convergence rate in Fig 5.1 is fast because the operator has lots of large eigenvalues clustered together and the conjugate gradient could solve the high modes fast. When the low modes come into play, the convergence rate approaches the limit which is determined by $\lambda_{max}/\lambda_{min}$. This is why the convergence rate is slow and constant after iteration 400. Note that the overall iteration number is greatly affected by the low modes.

As the lattice size grows, the lattice spacing decreases and the fermion mass decreases, the convergence rate of the conjugate gradient algorithm is affected by the low modes substantially. This is a critical difficulty for lattice QCD with a large lattice size. Therefore, the algorithms that solve the low modes effectively are of great interest to the lattice QCD community. In this chapter, we consider two lattices, the original fine lattice and the blocked coarse lattice, that lie on the same renormalization group trajectory but with lattice spacings that differ by a factor of 2. We explore how closely the low-mode space of the coarse lattice corresponds to the low-mode space of the fine lattice with the goal of using the coarse lattice to solve the low modes of the fine lattice. By working on a coarse lattice with a smaller size, the inversion becomes much easier. When the low modes are solved, the convergence rate of the algorithm can be improved greatly.

We will first introduce how the fine lattice is blocked based on a renormalization group approach. Then we will give a general description about the various ideas and methods used in this chapter. We will demonstrate how the fermion vectors are transformed from the coarse lattice to the fine lattice and vice versa. Afterwards, we will show that the low modes of the fine lattice can be represented by the low modes of the coarse lattice. An algorithm based on restarting the conjugate gradient and the preconditioned conjugate gradient algorithm will be discussed and a summary will be given in the last section.

There is one difference between this chapter and the last chapter that's worth pointing out. In the last chapter, the even-odd preconditioning is used to solve the Dirac equation $Dx = y$.

In this chapter, when solving $Dx = y$, we apply $D^\dagger$ on both sides and work with the equation $D^\dagger Dx = D^\dagger y$. Therefore, the matrix in the algorithm will be $D^\dagger D$. We would like to point out the difference to avoid any confusion.

5.1 The blocking scheme based on the renormalization group

In [35], a blocking scheme is developed to transform the gauge fields from a fine lattice to a coarse lattice based on the renormalization group idea. Since this is the starting point of the algorithms in this chapter, we introduce the blocking scheme briefly.

The RBC-UKQCD Collaboration generated several ensembles with the Iwasaki DSDR action at different lattice spacings. The ensembles use 2+1 flavor Mobius domain wall fermions with various quark masses. Since the ensembles have similar actions and good chiral properties, fits based on the chiral perturbation theory can be done to the results from these ensembles to reach the continuum limit. The general fit formula is:

$$X(m, a^2) = X_0(1 + f(m) + c_X a^2) \tag{5.1}$$

where $X(m, a^2)$ are the results from different ensembles, X_0 is the result for the chiral and continuum limit, $f(m)$ is a function given by the chiral perturbation theory about the mass dependence and c_X denotes the lattice spacing dependence. The results from different ensembles are related through this formula and it is found that the scaling errors are very small for the ensembles. The small scaling errors suggest that the actions for different lattice spacings lie on a renormalization group trajectory. Based on the idea, [35] found a blocking kernel that carries the transformation on the renormalization group trajectory.

The study started with a pair of Iwasaki DSDR ensembles with Mobius domain wall fermions that are generated separately. The size of the first ensemble is $24^3 \times 64 \times 12$ with lattice spacing $a^{-1} \approx 2$GeV. This ensemble is called the fine ensemble with action S_f. The size of the second ensemble is $12^3 \times 32 \times 12$ with lattice spacing $a^{-1} \approx 1$GeV. This ensemble is called the coarse

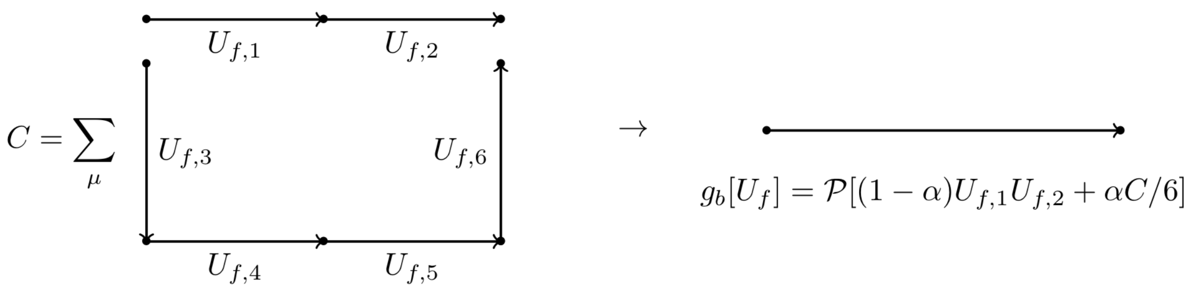


Figure 5.2: APE-like blocking kernel. C is the sum of the staples and P is a projection matrix that maps the sum of SU(3) matrices to a SU(3) matrix. Quoted from [35].

ensemble with action S_c. The parameters β, m_l and m_h are tuned so that the physical observables m_π, m_K, m_Ω are similar on the two ensembles. f_π and f_K are also found to be similar. The good scaling behavior is similar to the other Iwasaki DSDR ensembles used by the RBC-UKQCD Collaboration and it suggests that the two ensembles lie on the renormalization group trajectory. The goal was to find a blocking kernel so that a coarse ensemble can be directly gotten from the fine ensemble instead of being generated separately. One can confirm that a proper blocked coarse ensemble is obtained by comparing the blocked coarse ensemble to a distinct coarse ensemble generated by standard hybrid Monte Carlo. The action of the blocked coarse ensemble is called S_c^b.

[35] found that a kernel that is similar to the APE smearing can work (Fig 5.2). The kernel produces a coarse link from a pair of fine links plus staples. The coefficient for the staples is α. [35] determined α by using the demon algorithm and compared the coefficients of several terms between the action S_c^b and the action S_c. After determining α, one can confirm whether the blocked coarse ensemble is in the renormalization group trajectory by comparing the observables from S_c^b and S_c. [35] compared m_π, m_K, m_Ω, f_π, f_K and other observables. The results are quoted from [35] in Table 5.1. It's found that the difference between observables calculated from S_c and S_c^b are at the percent level. Therefore, a blocking scheme has been found which transforms a fine lattice to a coarse lattice that is on the renormalization group trajectory.

This is very important because it means that the blocked ensemble will have similar properties

	$\langle O\rangle_f$	$\langle O\rangle_c$	$\langle O\rangle_c^b$
size	$24^3 \times 64 \times 12$	$12^3 \times 64 \times 12$	$12^3 \times 64 \times 12$
β	1.943	1.633	-
am_l	0.000787	0.008521	0.007494
am_h	0.019896	0.065073	0.064150
a^{-1}(GeV)	2.001(18)	1.015(16)	1.010(16)
am_{res}	0.004522(12)	0.007439(86)	0.00847(21)
m_π(MeV)	300(3)	307(5)	308(8)
m_K(MeV)	491(5)	506(8)	507(11)
m_Ω(MeV)	1557(71)	1652(27)	1685(52)
f_π(MeV)	138(2)	147(2)	151(3)
f_K(MeV)	155(2)	166(3)	169(4)

Table 5.1: Observables from the fine, coarse and blocked coarse ensembles. The close values show that the ensembles lie on the renormalization group trajectory. The results are quoted from [35].

as the fine ensemble. However, we would like to point out that using the blocked coarse lattice to solve the low modes of the fine lattice needs a strong correspondence between the lattices. To utilize the blocked ensemble successfully, there has to be a configuration to configuration correspondence, not just the correspondence between the physical quantities.

Throughout the chapter, we will use the fine ensemble and the blocked coarse ensemble from [35]. For the rest of the chapter We will call the lattices in the blocked coarse lattices as coarse lattices for convenience because we will only use the blocked coarse ensemble.

5.2 An overview of the approaches

In this section, we give a brief overview of the methods that will be used in this chapter. The details of the methods and the results will be explained in the rest of the chapter.

The most obvious character in this chapter is that we utilize two lattices, one fine lattice and one coarse lattice, and we will use the coarse lattice to solve the low-mode part of the fine lattice. The idea is also known as the multigrid method. Currently there are other people [36][37] who are also using the idea to solve the lattice Dirac equation. There is a significant difference between our approach and the methods in [36][37]. In [36][37], the coarse gauge fields are not constructed and the operator on the coarse lattice is obtained by restricting the operator on the fine lattice. In our case, we have the gauge fields on the coarse lattice by following the renormalization group trajectory. The operator on the coarse lattice is constructed from the gauge fields on the coarse lattice. Therefore, there is more physical meaning in our approach.

Since we are using two lattices, the fermion vector has to be transformed between the coarse lattice and the fine lattice. We will call the operator that transforms the fermion vectors from the coarse lattice to the fine lattice as the interpolation operator I and the operator that transforms the fermion vectors from the fine lattice to coarse lattice as the restriction operator R. This will be explained in the next section.

The successful utilization of the coarse lattice depends on a correspondence between the low-mode space of the coarse lattice and the low-mode space of the fine lattice. This is studied in section

4. We will demonstrate that although there is not a clear one-to-one correspondence between the low modes of the coarse lattice and the low modes of the fine lattice, the low modes of the coarse operator and the fine operator span the same subspace.

We will then use different approaches to utilize the coarse lattice: the restart algorithm and the preconditioned conjugate gradient algorithm. For the restart algorithm, we first solve the high-mode part of the equation in the fine lattice and then we transfer the problem to the coarse lattice to solve the low-mode part. The result from the coarse lattice is added to the result from the fine lattice as a correction. Since the transfer between the coarse and fine lattice is not exact, errors appear when the correction from the coarse lattice is added. Therefore, the equation is solved again on the fine lattice. This process is shown in Fig 5.3. The spike shows that the residual increases when the coarse correction is added to the fine lattice. However, we can see that the total number of iterations is reduced compared to the conjugate gradient algorithm (CG). In section 5, we will try multiple restarts to harvest the benefit. There are other difficulties for this approach. For example, although there is correspondence between the coarse operator and the fine operator, the coarse operator has to be scaled so that the magnitude of the coarse correction is correct. In addition, during the interpolation, the errors appear because high modes are introduced. To reduce the high modes, the overlap transformation and other filtering techniques are used. These issues will also be explained in section 5.

For the preconditioned conjugate gradient algorithm, we use two kinds of preconditioners to reduce the conditioner number. In section 6, we use the preconditioner:

$$M^{-1} = 1 + bPI(\sum_{i}^{N} |\psi_{2h,i} >< \psi_{2h,i}| \frac{1}{\lambda_{2h,i}})RP \tag{5.2}$$

where $\psi_{2h,i}$ are eigenvectors on the coarse lattice, $\lambda_{2h,i}$ are eigenvalues on the coarse lattice and P is the operator that filters out the high modes. $\sum_i^N |v_{2h,i} >< v_{2h,i}| \frac{1}{\lambda_{2h,i}}$ is the inverse of the low-mode part of the coarse operator. Since the low modes of the coarse operator corresponds to the low modes of the fine operator, the second term approximately solves the low modes. Thus, the

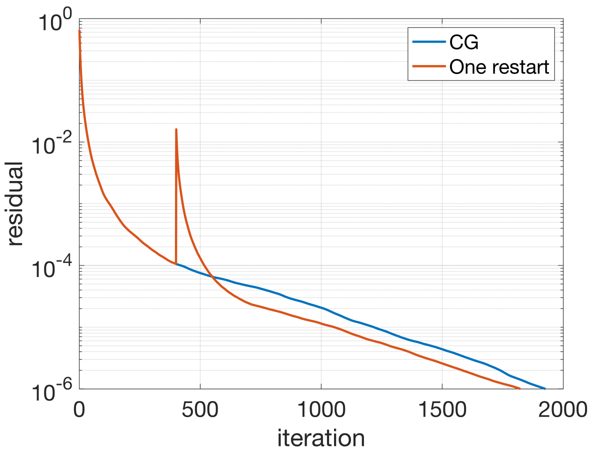


Figure 5.3: An example result for the restart algorithm. The spike happens when the coarse correction is added. However, the convergence is improved because some of low modes are solved through the coarse lattice. The overlap transformation is used to reduce the high-mode contamination and the norm of the coarse correction is adjusted. These issues will be discussed in detail in section 5.

condition number $\lambda_{max}/\lambda_{min}$ is decreased and the algorithm would be faster. Various approaches are tried in section 5. One example result is plotted in the left graph of Fig 5.4.

However, the filter P tends to be expensive. To make it less expensive, in section 7, a different preconditioner is used where the filter also serves to solve the high modes of the fine lattice. Various filters are tried in section 7 and an example is shown in Fig 5.4. This method tends to have more operations of $D^\dagger D$ compared to the method in section 6 since operations of $D^\dagger D$ are needed in the filter. However, the number of the total iteration is much smaller because the preconditioner changes not only λ_{min} but also λ_{max} and the condition number is reduced greatly. Because the total iteration number is reduced, the number of restrictions and interpolations is smaller. This is a benefit compared to the approach in section 5. However, if we take the number of operations of $D^\dagger D$ as the metric, the method might not be beneficial because $D^\dagger D$ is needed in the preconditioner.

In this chapter, when solving $Ax = b$, the source b is taken as a point source. However, we tested a few situations where b is a random vector and the performance is similar. For most situations, we solve the equation to a residual of 10^{-6} where the residual is defined as $|b - Ax|/|b|$.

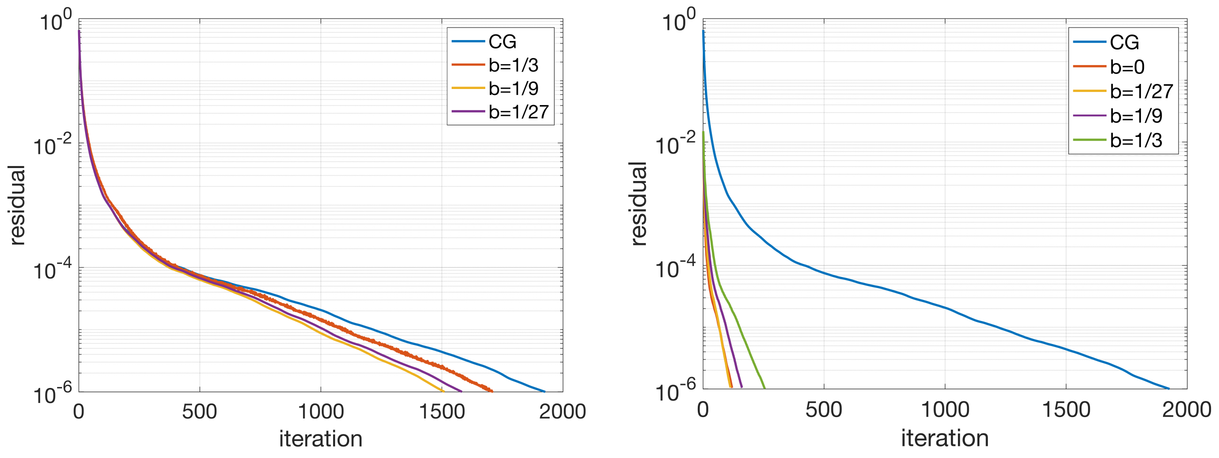


Figure 5.4: Left: an example showing that the preconditioner in eq. (5.2) can increase the convergence rate; Right: an example where we use the 50 Jacobi iterations in the preconditioner and the iteration number for the preconditioned conjugate gradient is reduced greatly; however, the number of operations of $D^\dagger D$ is larger because $D^\dagger D$ is needed for the Jacobi iteration.

Based on the convergence rate, we believe the algorithms in this chapter will perform better when the equations are solved to a higher precision.

While the domain wall operator has 5 dimensions, our discussion here focuses on the 4-d parts, which are related by the renormalization group blocking that we've described. The fifth dimensional features of the fermion fields are not changed by blocking.

5.3 The restriction operator and the interpolation operator

Although we have the coarse gauge fields and fine gauge fields, we need the interpolation operator to transform the fermion vectors from the coarse lattice to the fine lattice and the restriction operator to transform the fermion vectors from the fine lattice to the coarse lattice. The restriction operator will be denoted as R and the interpolation operator will be denoted as I. The operators will be discussed in this section.

Given that the lattice spacing differs by a factor of 2, the simplest restriction operator is:

$$R_s = \delta_{\mathbf{x},2\mathbf{x}} \tag{5.3}$$

where the coarse fermion vector is obtained from the fine fermion vector by preserving the sites with all coordinates even and discarding the sites that have any odd coordinates. The interpolation operator is a little more complex. Although the even sites of the fine lattice can be directly taken from the coarse lattice, the sites with odd coordinates can't. The sites with odd coordinates can only be calculated through interpolation. In one dimension, the odd site v_1 can be gotten by:

$$v_1 = \frac{1}{2}(U_0 v_0 + U_2 v_2) \tag{5.4}$$

where the U's are the fine gauge fields. Note that since this is a gauge theory, we use the gauge fields on the right hand side to interpolate the sites in a gauge covariant way. In the case of four dimensions, the process must be repeated four times until all the sites are interpolated. We will call this covariant interpolation operator as I_{cov}. Sometimes it's best that the restriction operator is the transpose of the interpolation operator. The restriction operator can be constructed similarly so that it's the transpose of the interpolation operator. We will call it R_{cov}.

The operators that are similar to eq. (5.4) are gauge covariant because they commute with the gauge transformation.

$$I_{cov} G_{2h} v_{2h} = G_h I_{cov} v_{2h} \tag{5.5}$$

where G is a gauge transformation matrix, v_{2h} is a coarse fermion vector. Throughout the chapter, we will use $2h$ as the subscript for coarse objects and h as the subscript for fine objects. Here, the coarse gauge transformation matrix G_{2h} is obtained through $R_s G_h$. The reason is that the coarse gauge fields are obtained through APE-like blocking and the gauge fixing matrices at the odd sites are cancelled out.

To test how good the interpolation operator and restriction operator are, we apply $I_{cov} R_s$ to the fine eigenvector ψ of $D^\dagger D$ with the lowest eigenvalue and then compare the vector $I_{cov}R_s\psi$ with the original eigenvector ψ. The perfect result would be that $I_{cov}R_s\psi = \psi$. In reality, this won't be the case because information is lost during the restriction. We use the normalized inner product to test how good the operators are. The normalized inner product between two vectors is defined as

$$\alpha = \frac{\langle v|u\rangle}{|v||u|} \tag{5.6}$$

In fact, in four dimension, because the interpolation is completed in four steps, the interpolation would be best at the first step and worst at the last step. To be specific, we will distinguish different sites according to the even-odd properties of the coordinates. If x, y, z, t are all even, the site will be called eeee. If one of the four coordinate is odd, the site will be called eeeo. If two of the four coordinates are odd, the site will be called eeoo. Likewise, there are eooo and oooo sites. Since the components on eeee sites are directly taken from the coarse lattice, the transformation is best on eeee sites. The components on oooo sites are interpolated last and it's expected that the transformation on oooo sites would be worst. In Table 5.2. The normalized inner product are calculated for $I_{cov}R_s\psi$ and ψ where ψ is a low-lying eigenvector. The norm of the components are also calculated. It can be seen that although the normalized inner product is good for all components, the norm of the components on the oooo sites is reduced greatly during the transformation. This will certainly cause problems if the operator is used in algorithms.

One of the reasons for the problem is that the gauge fields have large fluctuations and eq. (5.4) will reduce the norm during the interpolation. To solve the problem, one can make the gauge fields smooth by applying gauge transformation. In particular, one can work in Landau gauge to make the gauge fields smooth. Since the gauge fields are smooth, one can actually use the simple average instead of the covariant average:

$$v_1 = v_0 + v_2 \tag{5.7}$$

We will call this operator I and the transpose of the interpolation operator R. A table similar to Table 5.2 is included for I. Note that we also gauge transform the eigenvector so that the

	norm of components for $I_{cov}R_s\psi$	norm of components for ψ	normalized inner product
eeee	0.25	0.25	(1,0)
eeeo	0.22	0.25	(0.89,0)
eeoo	0.17	0.25	(0.85,0)
eooo	0.13	0.25	(0.85,0)
oooo	0.095	0.25	(0.86,0)
whole vector	0.70	1	(0.85,0)

Table 5.2: Comparison between $I_{cov}R_s\psi$ and ψ. ψ is a low-lying eigenvector and the table demonstrates that the interpolation operator I_{cov} reduces the norm of the components on the oooo sites.

comparison is consistent.

From Table 5.3, it can be seen that the interpolation operator works better when eq. (5.7) is used in Landau Gauge because the decrease in the norm is less severe. In addition, the normalized inner product goes up. Although the normalized inner products for the components on eooo and oooo sites are actually worse, the numbers only change slightly (from 0.85 to 0.83 and from 0.86 to 0.815). However, the norm of the components on oooo sites changes from 0.095 to 0.16, a 68% increase. Note that now the norm of the components on oooo sites is much closer to 0.25, the number for ψ, so the vector has a spatial structure that's more similar to the spatial structure of the low-lying eigenvector. Therefore, for the rest of the chapter, we will work in Landau Gauge and use I for the interpolation operator.

5.4 The low modes of the coarse lattice and the fine lattice

Since we would like to use the coarse lattice to solve the low-mode part of the fine lattice, it's important to make sure that the coarse lattice and the fine lattice have similar low-mode eigenvectors.

We first look at the lowest 1000 eigenvalues of $D^\dagger D$. In Fig 5.5, the eigenvalues of the coarse

	norm of components for $IR_s\psi$	norm of components for ψ	normalized inner product
eeee	0.25	0.25	(1,0)
eeeo	0.22	0.25	(0.91,0)
eeoo	0.20	0.25	(0.86,0)
eooo	0.18	0.25	(0.83,0)
oooo	0.16	0.25	(0.815,0)
whole vector	0.817	1	(0.87,0)

Table 5.3: Comparison between $IR_s\psi$ and ψ in Landau gauge. Comparing with Table 5.2, this shows that by working in Landau gauge with interpolation operator in eq. (5.7), the norm of the components on oooo sites is not decreased severely.

lattice are plotted against the eigenvalues of the fine lattice. The input fermion mass on the coarse lattice is 0.06507 and the fermion mass on the fine lattice is 0.019896. It's really interesting that the low eigenvalues of the coarse lattice and fine lattice are linearly related except for a few smallest eigenvalues. This is the benefit of using the renormalization group idea. Since they are on the renormalization group trajectory, the spectrum is linearly related. We don't have a clear understating why the factor is 3. This is mostly related to how the coarse gauge fields are obtained through APE-like smearing.

However, the smallest eigenvalues are a little different. To make the spectrum of the coarse and fine lattice in a perfect linear relation, we change the fermion mass on the coarse lattice to be 0.05. The eigenvalues are plotted in Fig 5.6. Now the eigenvalues of the coarse lattice are always about 3 times larger than the eigenvalues of the fine lattice. This is desirable because by multiplying a factor b, $bD_{2h}v_{2h}$ would be the same as D_hv_h if v_{2h} and v_h are only in the low-mode space and v_{2h} has a similar support in low-mode space as v_h does.

After we match the eigenvalues, we would like to match the eigenvectors. The first thing we can compare is $I\psi_{2h}$ and ψ_h where ψ_{2h} is the eigenvector of $D^\dagger D$ on the coarse lattice and

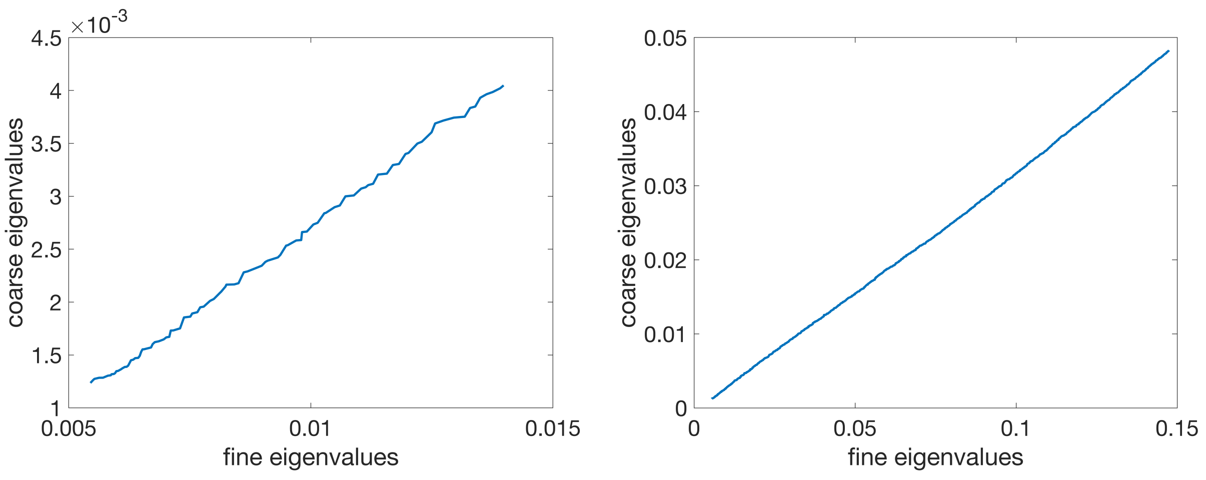


Figure 5.5: Coarse eigenvalues (fermion mass 0.06507 on the coarse lattice) and fine eigenvalues. Left: 100 eigenvalues; right: 1000 eigenvalues. The graphs show that the eigenvalues of the coarse and fine lattice are linearly related by a factor of 3 except for a few smallest eigenvalues.

.

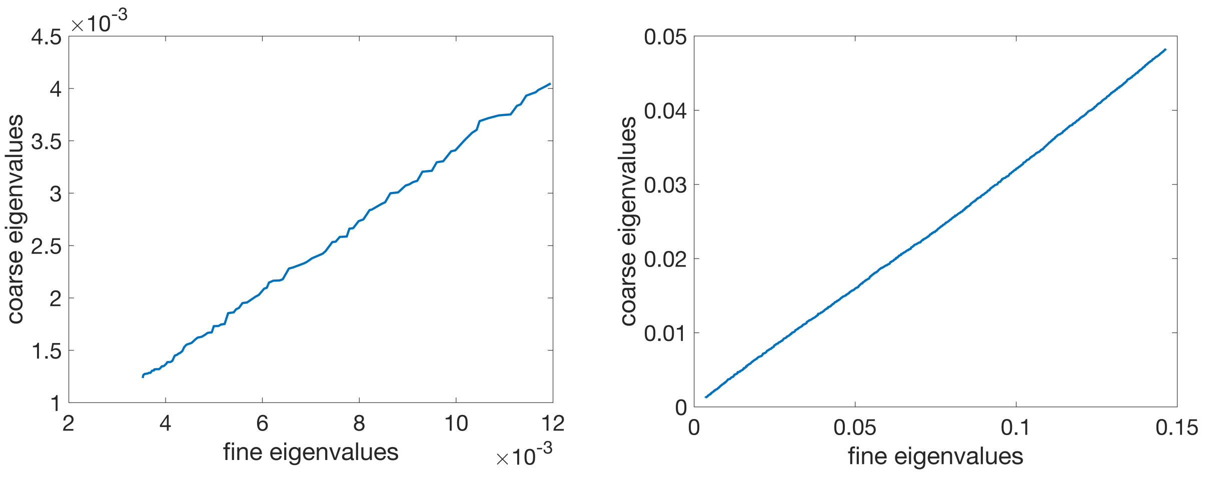


Figure 5.6: Coarse eigenvalues (fermion mass 0.05 on the coarse lattice) and fine eigenvalues. Left: 100 eigenvalues; right: 1000 eigenvalues. By tuning the coarse mass, all of the first 1000 eigenvalues of the coarse lattice are proportional to the first 1000 eigenvalues of the fine lattice.

.

ψ_h is the eigenvector of $D^\dagger D$ on the fine lattice. We consider the inner product between $I\psi_{2h,i}$ and $\psi_{h,j}$. The results are plotted as color map in Fig 5.7 for the fermion mass 0.06507 on the

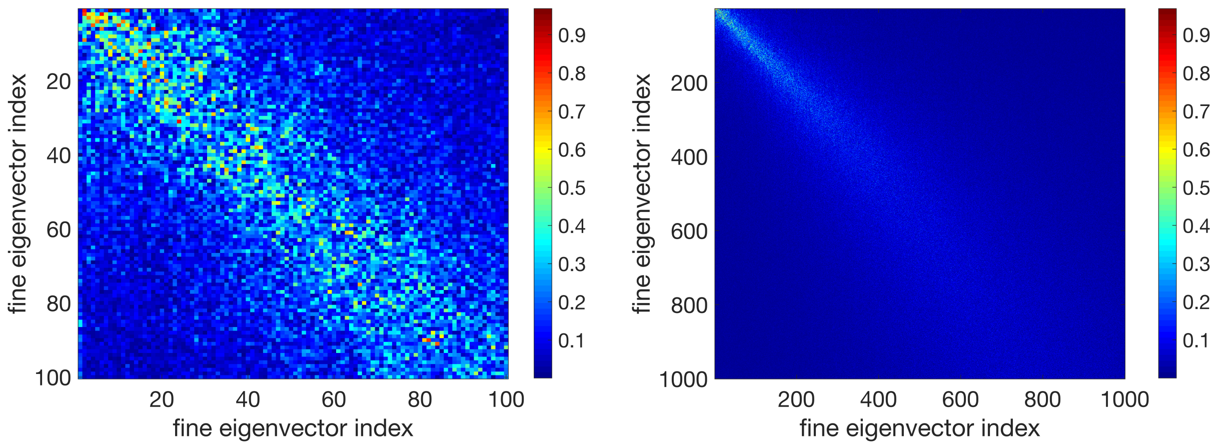


Figure 5.7: $|\langle I v_{2h}, v_h\rangle|$ for 100 and 1000 eigenvectors

.

coarse lattice and the results are similar for the fermion mass 0.05. Note that the norm of $I\psi_{2h,i}$ is around 3 so the normalized inner product is at most 0.3. From Fig 5.7, it can be seen that for the first 100 fine eigenvectors, each fine eigenvector corresponds to a small number (around 20) of coarse eigenvectors. However, for the eigenvectors with index 100-1000, there is only vague correspondence.

Although the one-to-one correspondence is weak, the property that we really care about is whether the inverse of the low modes is similar for the coarse and fine lattice. To do so, we compare two operators:

$$S_c = I\sum_i^N |\psi_{2h,i}\rangle\langle\psi_{2h,i}|\frac{1}{\lambda_{2h,i}}R \tag{5.8}$$

and

$$S_f = \sum_i^N |\psi_{h,i}\rangle\langle\psi_{h,i}|\frac{1}{\lambda_{h,i}} \tag{5.9}$$

These are simply the inversions of the low modes of $D^\dagger D$ by using the eigenvectors. To compare

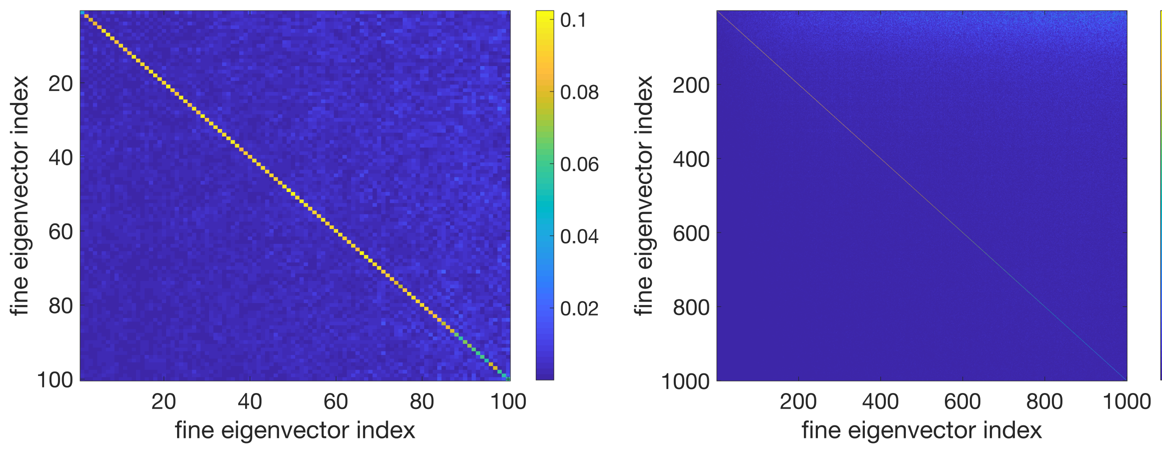


Figure 5.8: Results for eq. (5.9). Left: $N = 100$; right: $N = 1000$. The graphs are diagonally dominate which means that the coarse operator is a good approximation of the fine operator in the low-mode space.

them, we can multiply S_c and S_f^{-1} together and consider:

$$X = I \sum_i^N |\psi_{2h,i}\rangle\langle\psi_{2h,i}| \frac{1}{\lambda_{2h,i}} R \sum_j^N |\psi_{h,j}\rangle\langle\psi_{h,j}|\lambda_{h,j} \tag{5.10}$$

If S_c is exactly S_f, X would be the identity matrix. In reality, one can not examine the elements of X one bye one. However, one can test the behavior of X in the low-mode space by computing $\langle\psi_{h,m}|X|\psi_{h,n}\rangle$ for different n and m. Note $\langle\psi_{h,m}|X|\psi_{h,n}\rangle$ can be reduced as:

$$\sum_j^N \langle\psi_{h,n}|I|\psi_{2h,j}\rangle\langle\psi_{2h,j}|R|\psi_{h,m}\rangle \frac{\lambda_{h,m}}{\lambda_{2h,j}} \tag{5.11}$$

The results are plotted for in Fig 5.8 where the fermion mass is 0.05 on the coarse lattice. Although the diagonal elements are not large, the results are diagonally dominate. The diagonal elements are plotted in Fig 5.9. The dominance is weaker when n and m becomes large but this happens only after $n, m > 500$. This shows that the coarse operator is a good approximation of the fine operator in the low-mode space and we should be able to utilize the coarse operator to solve the low-mode space of the fine operator.

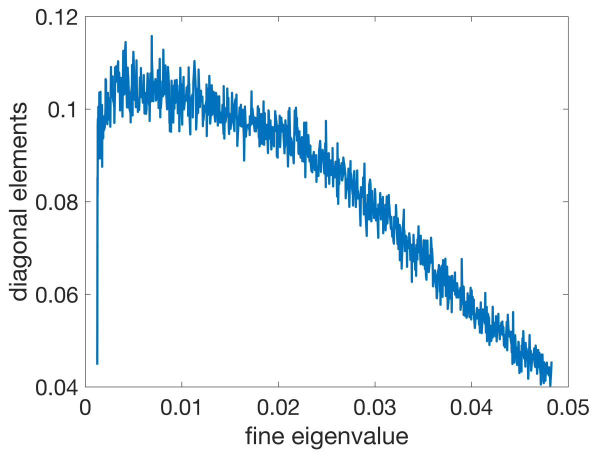


Figure 5.9: Diagonal elements for eq. (5.9) with $n = m$ and $N = 1000$

One can also notice that the diagonal elements change with n, m. The diagonal elements are around 0.1 for the first 500 eigenvectors (except for the lowest a few eigenvectors) but decrease afterwards. This means that a factor around 10 can be multiplied when S_c is used as an approximation for S_f in the low-mode space.

5.5 Restart algorithm

5.5.1 Introduction of the algorithm

Since we can approximate the low-mode part of the fine lattice with the low-mode part of the coarse lattice, the first idea is that we can use the coarse lattice in a restart scheme. The restart idea is briefly mentioned in section 2 and the complete algorithm is shown in Algorithm 3. In the first step, N_s iterations of the conjugate gradient are done on the fine lattice. This is used to solve the high modes since the coarse lattice is only helpful for solving the low-mode part. Then the residual is restricted to the coarse lattice and solved with the coarse operator. After the low-mode part is solved on the coarse lattice, it's interpolated back to the fine lattice as an approximation of the solution to the residual. In this step, sometimes a factor b is multiplied to make the norm correct. Because the interpolation operator is not perfect, usually some high modes will be introduced during the interpolation process. Therefore, the conjugate gradient is used again to solve the high-mode part. The process is repeated for N_o times. Lastly, the conjugate gradient is used to get the final solution.

Algorithm 3: The restart algorithm

```
Result: solution for $Ax = b$
Do conjugate gradient for $N_s$ iteration. The result is $x_0$ and $r_0 = b - Ax_0$
for $k = 0; k < N_o; k = k + 1$ do
    $r_{2h,k} = R_s r_k$
    $z_{2h,k} = A_{2h}^{-1} r_{2h,k}$
    $E_k = bIz_{2h,k}$
    Do conjugate gradient for $N_i$ iteration with $t_{k+1} = E_k + x_k$ as the starting vector. The
      result is $x_{k+1}$ and $r_{k+1} = b - Ax_{k+1}$
end
Do conjugate gradient until convergence with $x_{N_o}$ as the starting point.
```

The restart algorithm has been used in other circumstances. For example, in the Mobius accelerated domain wall fermion algorithm (MADWF), the same procedure is utilized except that an operator with smaller L_s is used instead of the coarse operator. MADWF is based on the overlap transformation and is very successful in terms of solving the low modes. So it's expected that the restart algorithm with the coarse operator would also work.

The success of the algorithm depends on a few things. First, the low modes of the coarse operator should be similar to the fine operator. This is confirmed by the last section. Secondly, during the interpolation, the high modes must be avoided as much as possible. Lastly, the norm needs to be adjusted accordingly with the factor b. We study the behavior of the restart algorithm and explain the second issue and the third issue in the next subsection.

norm of e	norm of E	normalized inner product between e and E
0.020	0.003	(0.61,0)

Table 5.4: Comparison between e and E

5.5.2 Analysis of the restart algorithm

The key result from the coarse lattice is E, which is an approximation of the true error $e = x' - x = A_h^{-1} r$. It's obtained through the process:

$$E = I A_{2h}^{-1} R r \tag{5.12}$$

We can compare E with e and see if they are the same. In the perfect situation, E is exactly e. Though this is unlikely, we need to understand the difference. In this chapter, r is the residual vector after 400 conjugate gradient iterations. We pick this residual vector because it's mainly composed of low modes and we are only interested in the quality of E in the low-mode space.

The comparison between e and E is shown in Table 5.4. The normalized inner product between e and E is (0.61,0). In addition, the norm of e and the norm of E are very different. The norm of e and the norm of E are different because the coarse operator has to be scaled to properly represent the low-mode space of the fine operator as explained in Fig 5.9. To understand why the normalized inner product is only a moderate value, we use the eigenvectors of A_h so that we can decompose e and E with the eigenvectors to understand why e and E are different.

In Fig 5.10, the normalized inner products between e, E and the eigenvectors are plotted. As we can see, both e and E have supports in the low modes, which is good. In Fig 5.11, the normalized inner products are plotted in the same graph for 30 and 70 eigenvectors. There are two observations. The first observation is that the normalized inner products between E and the eigenvectors are usually smaller than the normalized inner products between e and the eigenvectors. This means that E has more components that are not in the low-mode space. This explains why

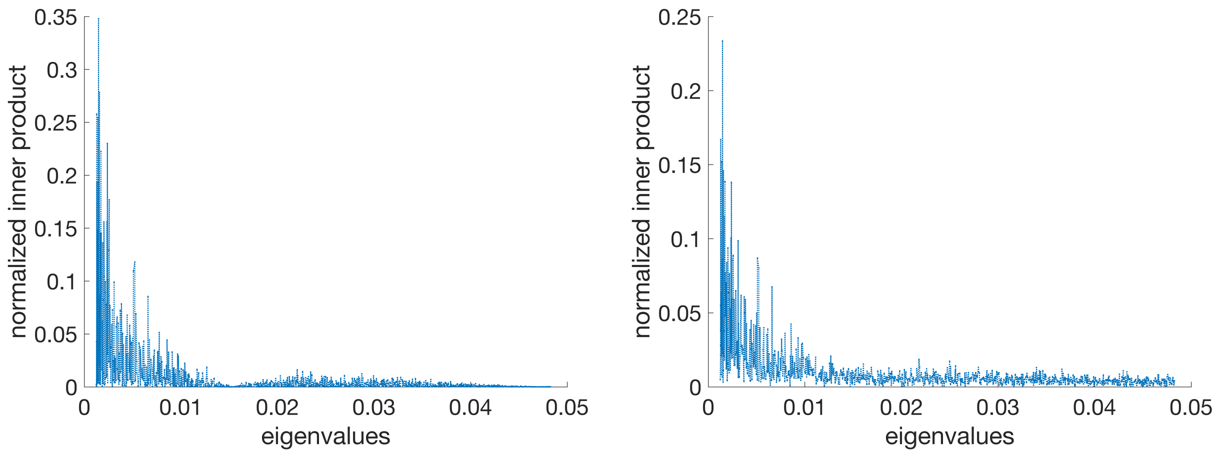


Figure 5.10: Left: normalized inner products between e and 1000 eigenvectors; Right: normalized inner products between E and 1000 eigenvectors. The fermion mass on the coarse lattice is 0.065.

the normalized inner product between e and E is around 0.6. To confirm this, we calculate:

$$\sqrt{\sum_{i=0}^{N} \langle v|\psi_i\rangle^2} \tag{5.13}$$

where $v = \frac{e}{|e|}$ or $v = \frac{E}{|E|}$ and ψ_i are the eigenvectors. In Fig 5.12, this is calculated. Note that while e is completely in the low-mode space, a large component of E is not in the low-mode space. This is clearly a problem and we will give several solutions for the issue.

The other issue can be found in Fig 5.11. The normalized inner products between E and the eigenvectors are large compared to the normalized inner products between e and the eigenvectors for a few smallest eigenvalues and small for large eigenvalues. This means that there is a mismatch between e and E in the low modes which can't be solved by simple scaling. This is further explained in Fig 5.13 by using a different coarse fermion mass to tune the inner products between E and the lowest 30 eigenvectors. For the first 30 eigenvectors, the match between e and E is best with coarse fermion mass 0.013. However, there is still mismatch for the eigenvectors with larger eigenvalues. This means that when the coarse operator is used, there is going to be some mismatch in the low modes. This again agrees with Fig 5.9. However, with an iterative method, the mismatch

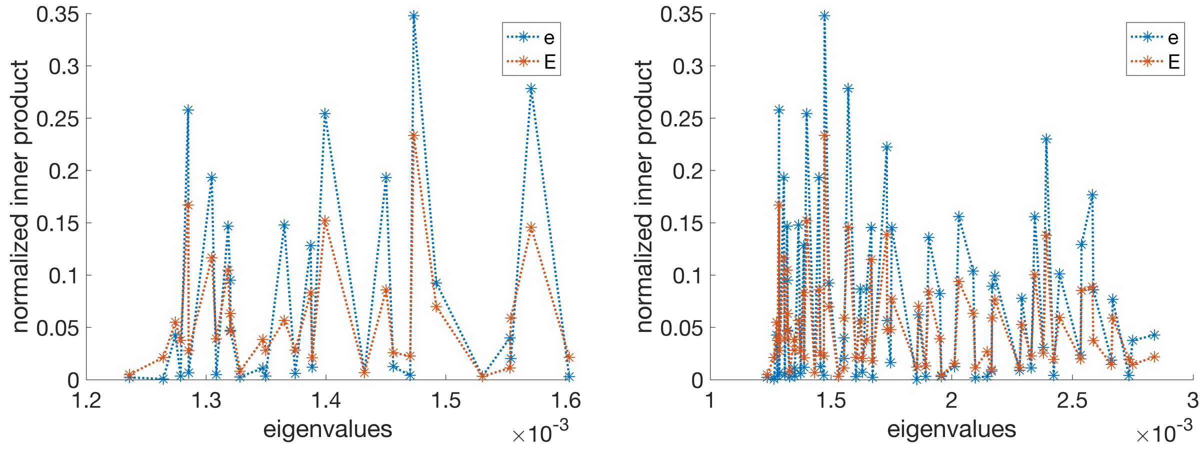


Figure 5.11: Normalized inner products between e, E and eigenvectors. Left: 30 eigenvectors; right 70 eigenvectors. The fermion mass on the coarse lattice is 0.065.

can be reduced.

5.5.3 Filtering the high modes with the overlap transformation and the polynomial filters

In the previous subsection, we realize that during the interpolation, some of the high modes are introduced. To make the algorithm successful, the high modes must be reduced. Here we develop two approaches, the overlap transformation and the polynomial filters. Note that when we mention high-mode filters, we refer to the filters on the fine lattice because the high modes are introduced to the fine lattice when the correction from the coarse lattice is added to the fine lattice.

The overlap transformation is the transformation that relates the five dimensional domain wall operator with the effective four dimensional operator. The four dimensional operator is called the overlap operator and hence the name "overlap transformation". The transformation was used in [38] for MADWF algorithm. The algorithm found that one can use an operator with a small L_s to solve the low modes of another operator with a large L_s by going through the overlap transformation. During the overlap transformation, the high modes in the fifth dimension are reduced. Therefore, the overlap transformation should be able to reduce the high modes introduced by the interpolation operator.

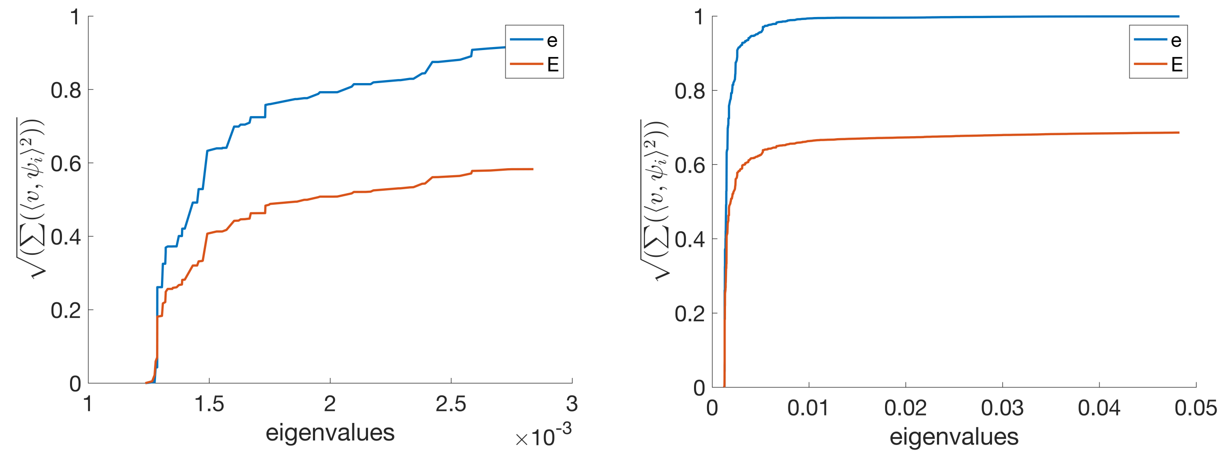


Figure 5.12: $\sqrt{(\sum(\langle v,\psi_i\rangle^2))}$ for e and E. Left: 70 eigenvectors; right: 1000 eigenvectors. The fermion mass on the coarse lattice is 0.065. The graphs show that results from the coarse lattice have large components in the high modes.

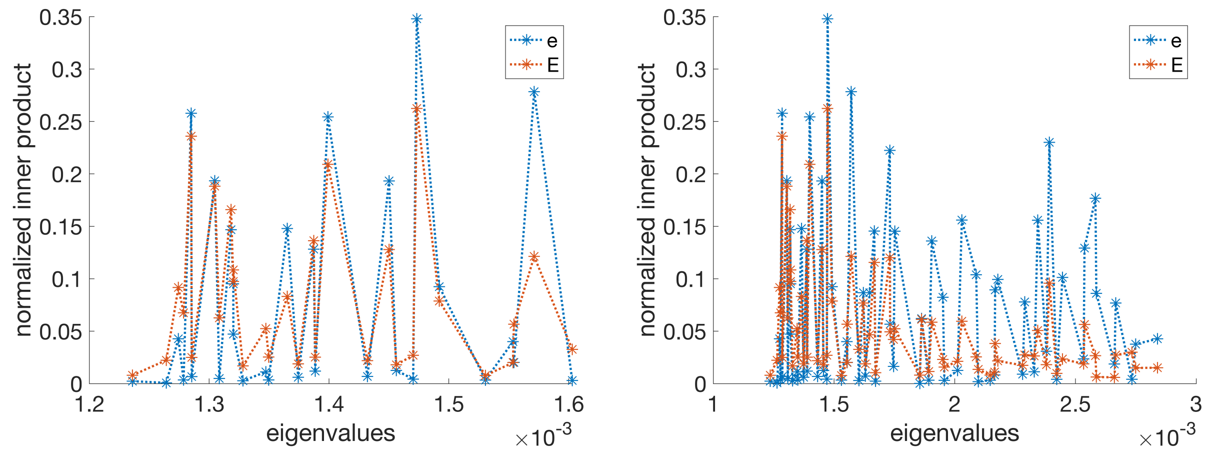


Figure 5.13: Normalized inner products between e, E and eigenvectors. Left: 30 eigenvectors; right 70 eigenvectors. The fermion mass on the coarse lattice is 0.013.

To find out how to utilize it with the coarse operator, it's best to start from the algorithm with MADWF in [38]. In MADWF, two operators, A_O and A_I, are related through the overlap transformation where A_O is the original operator and A_I is the operator with smaller L_s. They are related because they correspond to the same four dimensional overlap operator. In the discussion below, we will use the subscript 0 for the $s = 0$ component in the fifth direction. We start from the initial equation:

$$A_O x = b \tag{5.14}$$

Applying $P^{-1}V_O^{-1}$ to both sides, we have

$$P^{-1}V_O^{-1}A_O x = c = P^{-1}V_O^{-1}b \tag{5.15}$$

where P is the operator in [38] that is used to transform the equation to a lower triangular form and V_O is the Pauli-Villars term. After The multiplication, the equation on the $s = 0$ slice contains the equation with the overlap operator. In particular, the term on left hand side is $D_{OV}y_0$ where y_0 is the component of $y \equiv P^{-1}x$ on the $s = 0$ slice and D_{OV} is the four dimensional overlap operator. Taking c_0 as the component of c on the $s = 0$ slice, we have:

$$D_{OV}y_0 = c_0 \tag{5.16}$$

This is how A_O is related to the overlap operator. Then in MADWF, y_0 is gotten by solving a different equation involving A_I which is related to the same overlap operator. First, one construct the source c_I for A_I. c_I is constructed as $\{c_0, 0, 0..., 0\}$. Then the equation below is solved:

$$A_I P y_I = V_I P c_I \tag{5.17}$$

y_{I0} is taken as y_0. The other components of y are obtained by inserting $-y_{I0}$ to c. The result is denoted as l. With the following equation:

$$s = P^{-1}V_O^{-1}A_O Pl \tag{5.18}$$

the final solution is gotten by inserting y_{I0} to s.

To utilize the coarse lattice, we need to understand where we should insert the interpolation operator and the restriction operator. By observing the process, one can see that the vector that's passed into the inner operation is c_{I0} and the vector that's passed out from the inner operation is y_{I0}. Also, these two vectors relate A_I and A_O through the overlap transformation. Therefore, c_{I0} and y_{I0} should be modified for the multigrid approach. Specifically, we use $c'_I = Rc_I$ as the source for the coarse operator. After we get y_I in the coarse lattice, we use $y'_I = Iy_I$ and pass y'_{I0} to the fine lattice. This is the overlap transformation.

The other approach is to use a polynomial of A to filter the high modes. For a polynomial $p(A)$, when it's applied to an eigenvector ψ_λ with eigenvalue λ , the result is: $p(\lambda)\psi_\lambda$. Therefore, if $p(\lambda)$ is close to 0 for large λ and close to 1 for small λ, $p(A)$ will filter the high modes. There are several such polynomials, for example:

$$p_1(x) = (1 - x/300)^N \tag{5.19}$$

and

$$p_2(x) = \sum_0^N \frac{1}{N} T_n(1 - x/300) - \frac{1}{2N} T_0(1 - x/300) - \frac{1}{2N} T_N(1 - x/300) \tag{5.20}$$

where T_i are the Chebyshev polynomials. 300 in eq. (5.19) and eq. (5.20) are chosen to be larger than $\lambda_{max}/2$. In Fig 5.14, the numerical values are plotted for different orders of the polynomials. It can be seen that the Chebyshev polynomials are good for filtering low eigenvalues. Therefore, we will use the Chebyshev polynomials as the filter.

Without any filter, we find that $\sqrt{\sum_{i=0}^N \langle v|\psi_i\rangle^2}$ is around 0.65 for $E/|E|$ with 1000 eigenvectors. However, with the overlap transformation or the polynomial filter, the number is close to 0.9.

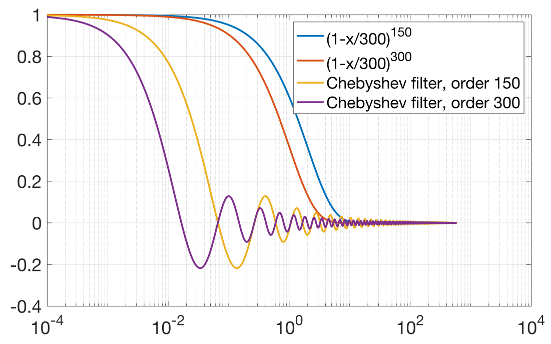


Figure 5.14: Numerical values for different polynomial filters

Therefore, a large amount of the high modes are filtered out. In addition, the normalized inner product between e and E is increased to (0.83, 0) for the overlap transformation and (0.89, 0) for the Chebyshev filter, while the normalized inner product between e and E with no filter is (0.61,0). This shows that the filters are indeed helpful.

5.5.4 Results for the restart algorithm

As discussed in previous sections, there are two major issues with the restart algorithm: 1) the interpolation process will introduce high modes which need to be filtered out through the polynomial filter or the overlap transformation; 2) the norm of the vectors after being multiplied by the coarse and fine operator are different and an overall factor is needed.

In Fig 5.15. The residual is plotted when there are 400 iterations in each restart and the coarse fermion mass is 0.05. In the coarse level, 1000 eigenvectors are used to solve the low modes. The b's are the factors multiplied to adjust the norm. It can be noticed that the best result is for $b = 9$, this agrees with Fig 5.9 where it's shown that in the low-mode space, the norm should be adjusted by multiplying 9 so that $IA_{2h}^{-1}R$ and A_h^{-1} have similar effects.

In Fig 5.16 and Fig 5.17, the overlap transformation and the Chebyshev polynomials are used as filters. Also, the norm are adjusted in each restart so that the norm of E is the same as the norm of e. In both graphs, the restart algorithm has a better convergence rate than the conjugate gradient algorithm. However, there are a few defects. First, the cost for the overlap transformation

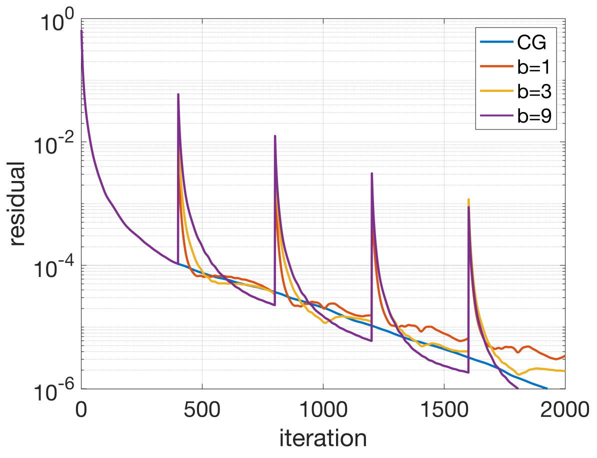


Figure 5.15: Residual for the restart algorithm. The fermion mass on the coarse lattice is 0.05. There are 400 iterations after each restart. 1000 coarse eigenvectors are used to solve the low-mode part of the coarse lattice. $b = 9$ is optimal which agrees with Fig 5.9

.

and the Chebyshev polynomials is huge. For overlap transformation, the total effective overhead for each restart is around 70 fine $D^\dagger D$ operations. The total number of restarts is 4 so the cost is 280 operations. This makes the advantage of the algorithm small. For Chebyshev polynomials, the effective overhead is around 170 fine $D^\dagger D$ operations for each restart. Although the convergence rate is good, the cost is huge with 14 restarts. Therefore it's not optimal.

Another observation is that the residual goes up after every restart. The reason is that during the restart, the low modes get corrected but some high modes are introduced even in the presence of the filter. Although the components introduced by the high modes are small, the residual is increased by a large amount because the high modes have high eigenvalues. To solve this issue, we introduce the renormalization based preconditioned Conjugate gradient in the next two sections.

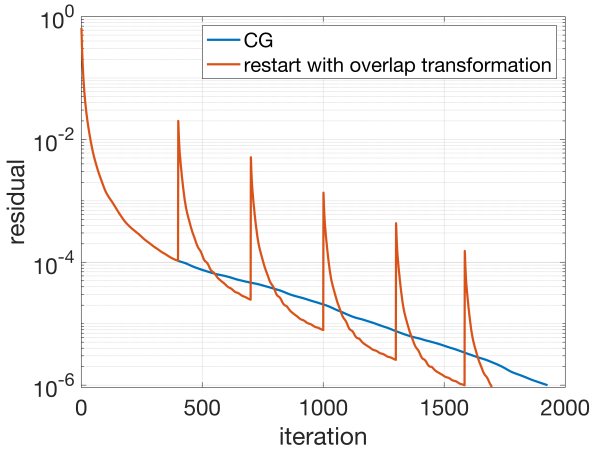


Figure 5.16: The restart algorithm with overlap transformation. The fermion mass on the coarse lattice is 0.065. There are 300 iterations after each restart and the norm is adjusted by a fixed factor.Although the convergence rate is increased, the cost of each restart is around 70 operations of $D^\dagger D$.

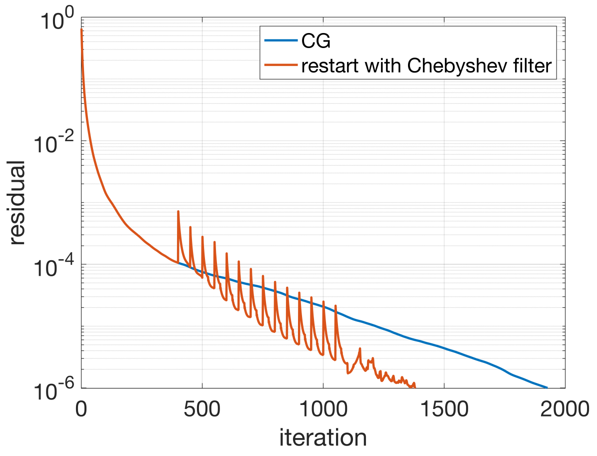


Figure 5.17: The restart algorithm with Chebyshev filters (order 151). The fermion mass on the coarse lattice is 0.05. There are 50 iterations in each restart and the norm is adjusted by a different factors in each restart. Although the convergence rate is increased, the cost of each restart is around 170 operations of $D^\dagger D$.

5.6 Renormalization group based preconditioned conjugate gradient: direct application of the coarse lattice in the preconditioner

In the previous section, we saw that the coarse lattice can solve the low modes. However, with the restart algorithm, the high modes are re-introduced in each restart so the restart is not very effective. To alleviate the problem, we use the coarse lattice in the preconditioned conjugate gradient algorithm and utilize the coarse lattice to reduce the condition number. In this section, we will use the coarse inverse in the preconditioner directly. In next section, we will apply the high-mode solver as a smoother in the preconditioner.

The preconditioned conjugate gradient algorithm is introduced in the previous chapter so we will focus on the preconditioner. Since the coarse inversion is needed in every iteration, we calculate the coarse eigenvectors and use the coarse eigenvectors to do the inversion in the low-mode space. The basic preconditioner that uses the coarse operator to solve the low modes is:

$$M^{-1} = I(\sum_{i}^{N} \frac{|\psi_{2h,i} >< \psi_{2h,i}|}{\lambda_{2h,i}})R \tag{5.21}$$

where $\psi_{2h,i}$ are the eigenvector of the coarse operator. Note that since the preconditioner must be symmetric, here we use $R = I^T$. However, this preconditioner doesn't work because it projects the vector into a subspace of the fine operator and it's not positive definite. Therefore, we have to add an identity matrix and introduce the parameter b to adjust the norm:

$$M^{-1} = 1 + bI(\sum_{i}^{N} |\psi_{2h,i} >< \psi_{2h,i}| \frac{1}{\lambda_{2h,i}})R \tag{5.22}$$

If $b = 0$, we are not using the preconditioner. If $b >> 1$, the weight of the preconditioner is very large.

Although the approach improves the convergence rate, the improvement is not great because the interpolation operator introduces high modes. We find that it's best to incorporate high-mode filter. The final preconditioner is in the form:

$$M^{-1} = 1 + bPI(\sum_i^N |\psi_{2h,i} >< \psi_{2h,i}| \frac{1}{\lambda_{2h,i}})RP \tag{5.23}$$

where P is the high-mode filter.

First of all, we emphasize the importance of the high-mode filter by comparing the results when there is not a filter and when there is an exact filter. In the case of the exact filter, we use the eigenvectors of the fine operator and we have the filter:

$$P = \sum_i^N |\psi_{h,i} >< \psi_{h,i}| \tag{5.24}$$

The results are in Fig 5.18. The left graph is for the residuals when there is no filter and the right graph is the case when there is the exact filter. It can be seen that although around 400 iterations are saved when there is no filter, the convergence rate is much faster when there is the exact filter. In fact, the convergence rate of the right graph in Fig 5.18 approaches the convergence rate when fine eigenvectors are used instead of the coarse eigenvectors. This means that if a good filter is used, the coarse eigenvectors are almost perfect as the replacement of the fine eigenvectors. Lastly, note that the best convergence rate is achieved when $b = 9$ or $b = 27$. This agrees with Fig 5.9.

Two kinds of filters are in the previous section, the overlap transformation and the Chebyshev polynomials. Although they are promising as filters, they are too expensive.

To filter the high modes effectively, we need insights to the properties of the high modes. Considering that QCD is approximately free in high energy, the eigenvectors with high eigenvalues, or the high modes should be approximately plain wave and should be filtered by Fourier transformation or geometric smoothing. These two approaches are tried. With geometric smoothing, after the interpolation, we add the neighbors of a site with a weight to the site. Therefore, the fermion vector is smoothed and the high-frequency modes are removed. With Fourier transformation, we Fourier transform the fermion vector to momentum space and only preserve the low momentum part of the vector. Then the truncated vector is transformed back to the position space. Through the

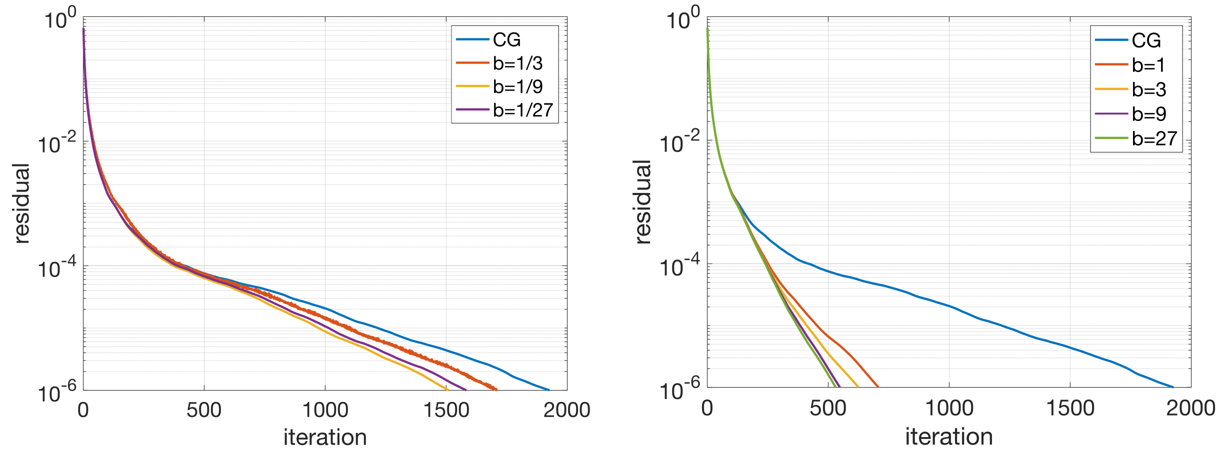


Figure 5.18: Left: the residuals when there is no filter; Right: the residuals when the exact filter is used. Comparing the two graphs, we see that the high mode filter is important to improve the performance of the preconditioner. $b = 9$ and $b = 27$ are optimal for the right graph which agrees with our understanding about the low-mode correspondence between the coarse operator and the fine operator.

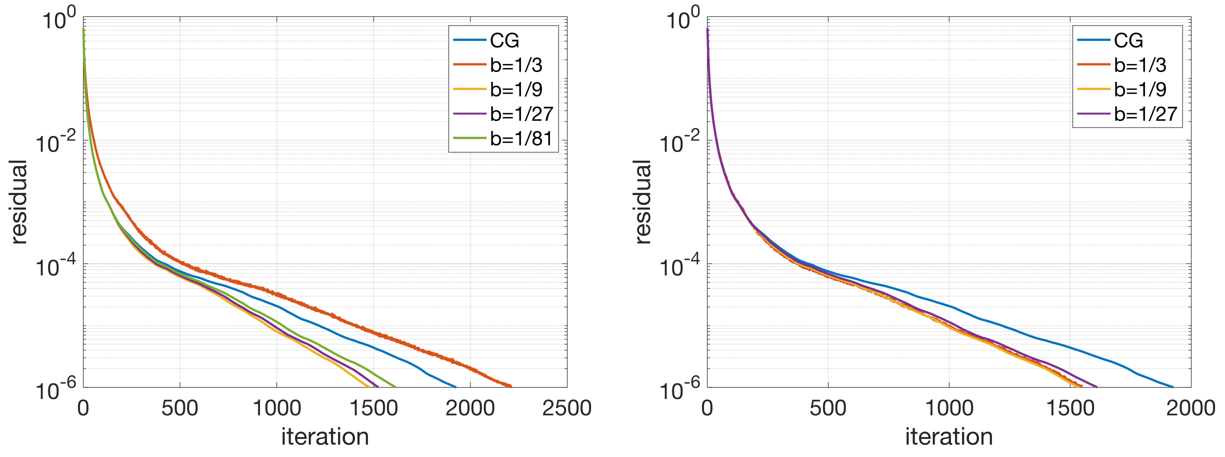
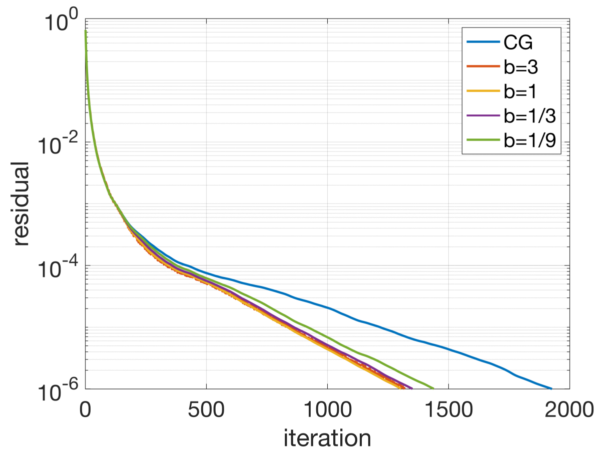


Figure 5.19: Left: results for geometric filter where the weight of the neighbors is 0.08; Right: results for Fourier filter where we keep the sites in momentum space if $|k_1| \leq \pi/4, |k_2| \leq \pi/4, |k_3| \leq \pi/4, |k_4| \leq \pi/4$. There is small improvements but the results show that the geometric filter and Fourier filter are not effective enough to filter our the high modes.

truncation, the high-frequency modes are removed. The results from these two filters are plotted in Fig 5.19. Although there is slight improvement compared to the case when there is no filter, the results show that geometric smoothing and Fourier transformation are not sensitive enough to filter the high modes effectively.

The filter that turns out to be relatively effective is the filter with the idea of domain decomposition. The idea was first demonstrated clearly in [32]. The idea is based on the fact that there is a scale associated with the gluon field through the glueball mass. When the distance between two points is longer than the scale, there is little interaction. For eigenvectors, it means that once we have a few exact low-mode eigenvectors, it's possible to construct some inexact low-mode eigenvectors by combining the local pieces of the exact low-mode eigenvectors. Specifically, one can split the lattice into N_b blocks. One can get N_b basis vectors out of one eigenvector. Each basis vector equals the component of the eigenvector in one block and is 0 everywhere else. Therefore, if one has N_e exact eigenvectors, one has $N_e \times N_b$ basis vectors, then one can construct inexact eigenvectors by combining the basis vectors. In fact, this technique is very useful in generating inexact eigenvectors [39]. In practice, people can generate 1000 exact eigenvectors and then use the 1000 exact eigenvectors to generate 1000 inexact eigenvectors with relatively small efforts.

In our case, we don't want to use exact eigenvectors since it's expensive to generate exact eigenvectors. Instead, we use the near-null vectors. One vector is called a near-null vector if $Av \approx 0$ (The detailed requirement of the precision depends on the low eigenvalues.). One can generate the near-null vectors by solving $Ax = 0$ using the conjugate gradient to some precision. In our case, we generate the near-null vectors by solving it to precision 0.01 where the lowest eigenvalue of A is 0.00123598 and 0.01 is the eigenvalue of the 219th eigenvector. This means that in our case, $|Av| < 0.01$. The near-null vectors are similar to low-mode eigenvectors in the sense that after being multiplied by the matrix, the results are small. It's reasonable to assume that the near-null vectors are in the low-mode space of the eigenvectors and they can be used to generate basis vectors as discussed above. Then each near-null vector is used as the filter for a general vector u according to the following procedure:



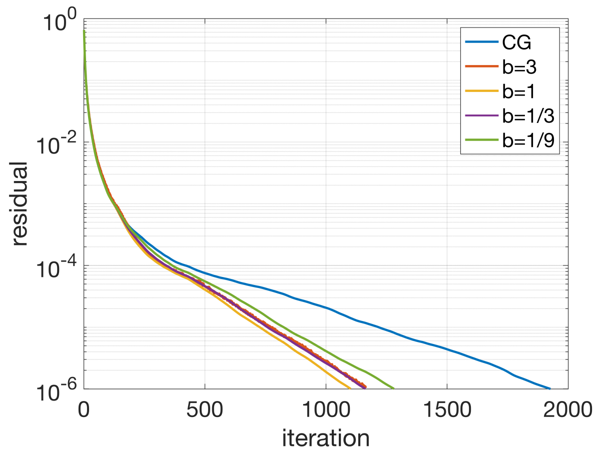


Figure 5.20: Results when the near-null vectors are used as the filter. Left: 5 near-null vectors are used; Right: 10 near-null vectors are used. The results are for fine mass 0.019896. The near-null vectors are effective in filtering out the high modes and the optimal parameter is $b = 1$.

1) Construct N_b vectors u_{bi} from the vector u using the same process that N_b basis vectors v_{bi} are generated from one near-null vector;

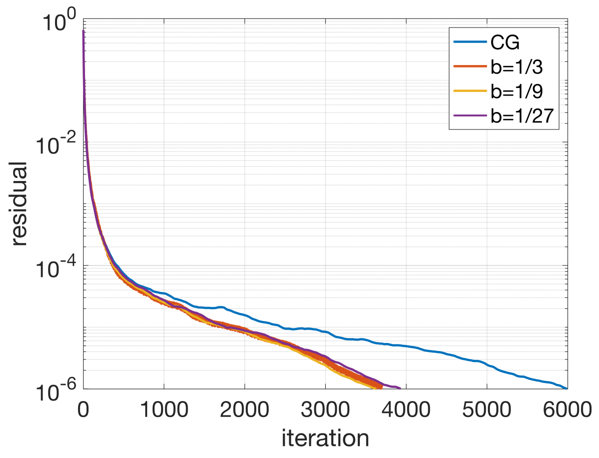
2) Calculate the inner products α_i between u_{bi} and v_{bi} for each i;

3) Use the inner products as the coefficients for the basis vectors and construct the new vector $Pu = \sum \alpha_i v_{bi}$.

In the case that there are several near-null vectors. The results from each near-null vector are added as the final results.

It should also be noted that when there are multiple near null vectors, the basis vectors must be orthogonal to each other and the norm of each basis vector should be scaled to 1.

We find that with 5 or 10 near-null vectors, the results are good compared to the case when there is no filter. In addition, the filter is particularly important when the mass is light. In Fig 5.20 and Fig 5.21, the results are plotted for fine fermion mass 0.019896 and 0.000787. In both cases, there is obvious improvement in the convergence rate. In particular, for Fig 5.20, the iteration number is changed from 1927 to 1300 in the best case, a 33% decrease. For Fig 5.21, the iteration number is changed from 5900 to 3300, a 44% decrease.



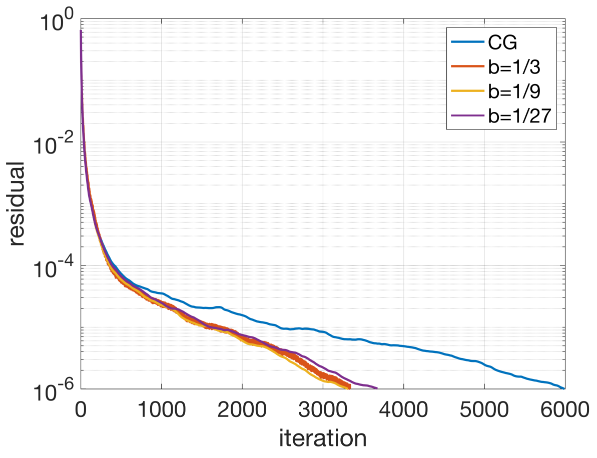


Figure 5.21: Left: Results when the near-null vectors are used as the filter. Left: 5 near-null vectors are used; Right: 10 near-null vectors are used. The results are for fine mass 0.000787 which is why the iteration number for CG is much larger than the other cases in this chapter. The near-null vectors are effective in filtering out the high modes for light mass. The optimal parameter is $b = 1/9$

At this point, we would like to point out that the filter with the near-null vectors and the polynomial filter are built upon the knowledge of the eigenvectors and eigenvalues of the operator. The geometric smoother and the filter based on the Fourier transformation don't need any information about the operator. Until now, we find that only the filters based on the eigenvectors and eigenvalues of the operator are effective.

Because the filter with the near-null vectors requires the knowledge of the near-null space, there is overhead in the calculation of the near-null vectors. For the heavy mass, the iteration number for each near-null vector is around 450. For the light mass, the iteration number for each near-null vector is around 1600. Therefore, if only one equation is to be solved, the overhead is not justified. However, the overhead is justified if 4 or more equations are needed to be solved. During the measurements, usually tens of equations are needed to be solved and the number of iterations saved could be more than one third of the total iteration number. In addition, in our current results, the equation is solve to the 10^{-6}. If the equation is solve to residual 10^{-8}, more iterations will be saved.

5.7 Renormalization group based preconditioned conjugate gradient algorithm: using high-mode solver in the preconditioner

In the previous section, it's found that the high-mode filter is important to improve the algorithm. However, the polynomial filters that are used in section 4 are very expensive. To solve the issue, we apply the method in [40] where the filter also functions as a high-mode solver in the preconditioner. In this method, since both high modes and low modes are solved in the preconditioner, the convergence rate will be much better and cost of the filter could be justified. The preconditioner is consisted of 5 steps:

1) $u = P(r, u)$ (pre-smoothing)

2) $d = R(A_h u - r)$

3) $v = A_{2h}^{-1} d$

4) $u = u - bIv$

5) $u = P(r, u)$ (post-smoothing)

In step 1, one uses the zero vector as the starting point for u. In step 4, we include a factor b to adjust the norm. The 5 steps are taken as a preconditioner where it takes r as the input and u as the output:

$$u = M^{-1} r \tag{5.25}$$

The process is found to be symmetric and positive definite for certain P and can be used in the preconditioned conjugate gradient algorithm. P is usually called a smoother because it's used to solve the high modes and smooth the vector. There is both pre-smoothing and post-smoothing because the preconditioner has to be symmetric. Initially, both pre-smoothing and post-smoothing mainly function as high-mode solver. When r is mainly composed of low modes, the pre-smoothing is not very useful but the post-smoothing will filter the high modes brought by the coarse correction.

In this section, we will use two kinds of smoothers, the Jacobi iteration and polynomial solvers.

There are other kinds of methods to be explored that are very attractive, for example, the multi-splitting algorithm in the last chapter. Those solvers require further techniques and adaption to be used and are not yet applied.

5.7.1 Jacobi iteration as the smoother

We have explained the Jacobi iteration in the previous chapter. Here we explore the properties of the Jacobi iteration in more details. The Jacobi iteration is:

$$x_{k+1} = D^{-1}(b - (A - D)x_k) \tag{5.26}$$

We would like to point out that the error $e_k = x_k - x$ decreases according to:

$$e_{k+1} = J^{k+1} e_0 \tag{5.27}$$

where J is defined as:

$$J = 1 - D^{-1}A \tag{5.28}$$

This shows that as long as the maximum eigenvalue of J is smaller than 1, e will decrease monotonically. With appropriate modification of the Jacobi iteration, J can decrease the high modes yet leave the low modes of e unchanged. Therefore, it can serve as the smoother in the preconditioner.

The algorithm will converge when the matrix A is strictly diagonally dominated, which means:

$$a_{ii} > \sum_{j \neq i} |a_{i,j}| \tag{5.29}$$

This is not the case for our matrix $D^{\dagger}D$. To solve the issue, instead of splitting the matrix A as $A = D + (A - D)$, where D is the diagonal part, the matrix is split as: $A = (D + \lambda) + (A - D - \lambda)$. The iteration is now:

$$x_{k+1} = (D + \lambda)^{-1}(b - (A - D - \lambda)x_k) = x - (D + \lambda)^{-1}Ax_k + (D + \lambda)^{-1}b \tag{5.30}$$

The error decreases according to:

$$e_{k+1} = J^{k+1} e_0 \tag{5.31}$$

where J is:

$$J = (D + \lambda)^{-1}(A - D - \lambda) \tag{5.32}$$

This is found to converge for our operator with proper λ.

In addition, we can also use the weighted Jacobi iteration to split A according to $A = \frac{1}{\omega}D + (A - \frac{1}{\omega}D)$. The iteration is:

$$x_{k+1} = ((1-\omega) + \omega J)x_k + \omega D^{-1} b \tag{5.33}$$

The error decreases according to:

$$e_{k+1} = (1 - \omega D^{-1} A)^{k+1} e_0 \tag{5.34}$$

The Jacobi iteration requires the knowledge of the diagonal elements. The diagonal elements of $D^{\dagger}D$ for the Mobius domain wall operator are calculated in the appendix.

Since the Jacobi iteration is a solver, we can use it directly to solve the equation. The residuals are plotted in Fig 5.22. It can be seen that with the Jacobi iteration, the residual decreases fast initially but the convergence rate slows down afterwards. This is because it solves the high modes relatively fast but it solves the low modes very slowly. Although the Jacobi iteration is slow compared to the conjugate gradient algorithm, the conjugate gradient algorithm can't be used as a smoother because it distorts the low modes when the high modes are solved, defeating the purpose of using the coarse lattice to solve the low modes.

In Fig 5.23, we show the results where we use the Jacobi iteration in the preconditioner. We see that when $b = 1/27$, the coarse lattice improves the algorithm compared to $b = 0$. However, the improvement is marginal. In addition, although the number of iterations for the preconditioned conjugate gradient is much smaller compared to conjugate gradient, the total number of operations

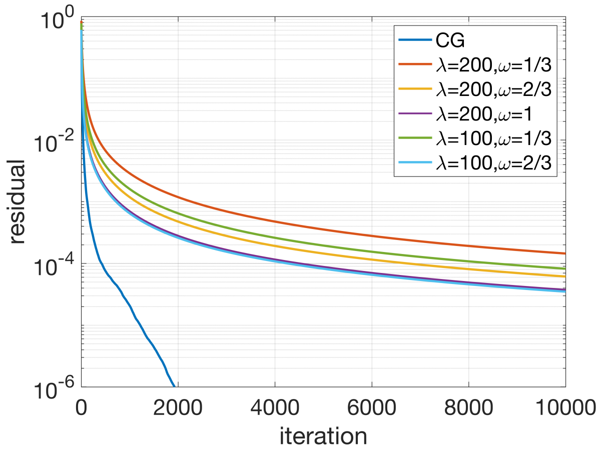


Figure 5.22: Jacobi method for different ω and λ

for $D^{\dagger}D$ is more than 11000 in the preconditioned conjugate gradient compared to 1950 in the conjugate gradient. The main reason that the algorithm is not effective is that the Jacobi iteration is not a very good high-mode solver.

5.7.2 Polynomial solvers as the smoother

The previous results show that the Jacobi iteration is not a good high-mode solver. To improve the results, we use polynomial solvers to solve the high modes.

The idea arises from the derivation of a solver called Chebyshev iteration [41]. There are many versions of the Chebyshev iterations, two of which are:

$$x_{k+1} = x_k - \alpha_{k+1}(Ax_k - b) \tag{5.35}$$

and:

$$x_{k+1} = x_k - \alpha_{k+1}(Ax_k - b) - \beta_{k+1}(x_k - x_{k-1}) \tag{5.36}$$

With the iteration in eq. (5.35), the error decreases according to:

$$e_k = \Pi_1^N (1 - \alpha_k A) e_0 \tag{5.37}$$

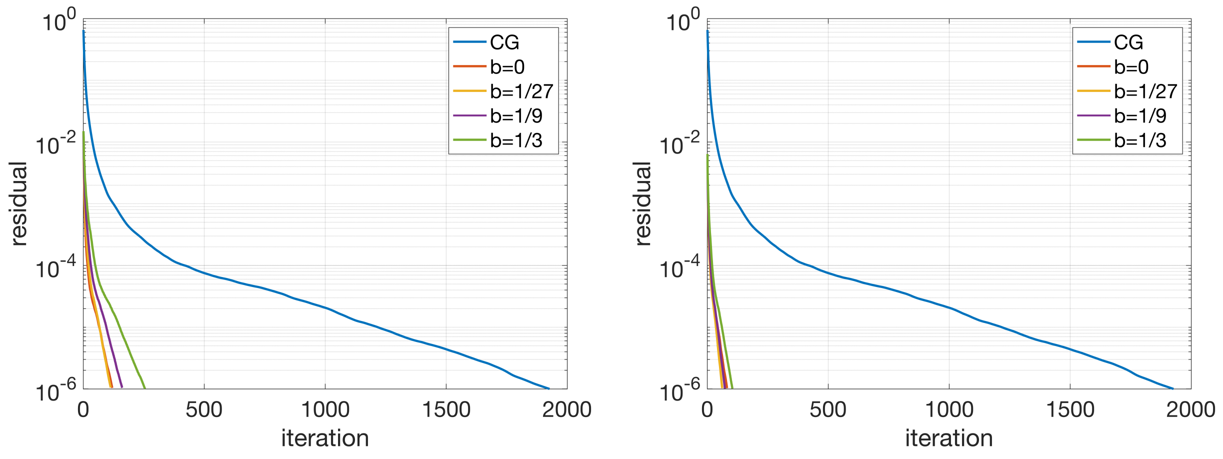


Figure 5.23: Jacobi iteration as the smoother. Left: 50 Jacobi iterations; right: 100 Jacobi iterations. $\lambda = 200$. By using the Jacobi iteration in the preconditioner, the convergence rate is improved greatly. The optimal parameter is $b = 1/27$ which means that coarse lattice is useful. However, the number of operations of $D^\dagger D$ is larger than CG because the operations are needed during the Jacobi iteration.

With the iteration in eq. (5.36), the error decreases according to:

$$e_k = O_k e_0 \tag{5.38}$$

where O_k satisfies:

$$O_k(A) = (1 - \beta_k - \alpha_k A)O_{k-1} + \beta_k O_{k-2} \tag{5.39}$$

The Chebyshev iteration is obtained when careful choices about α and β are made so that the polynomials that are applied to the error are the Chebyshev polynomials. The Chebyshev polynomials will decrease the error in the range of the eigenvalues uniformly so it's not helpful in our case. However, the idea is helpful in the sense that we can use the iteration in eq. (5.35) or eq. (5.36) so that the polynomial O_k will be a high mode filter. In particular, we can choose α and β

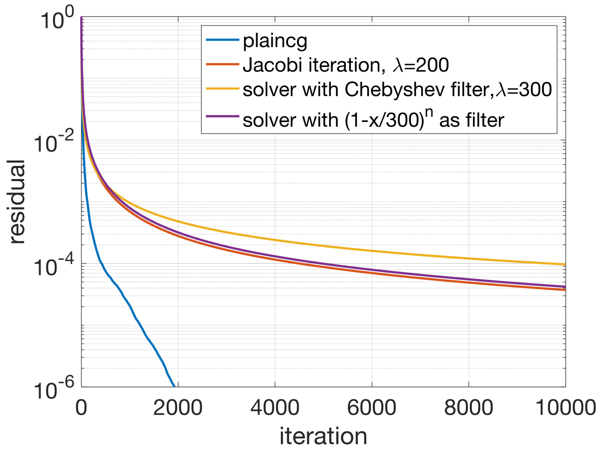


Figure 5.24: Jacobi iteration and the polynomial solvers.

so that O_k will be the filters described in section 3:

$$O_N = (1 - \frac{A}{\lambda})^N \tag{5.40}$$

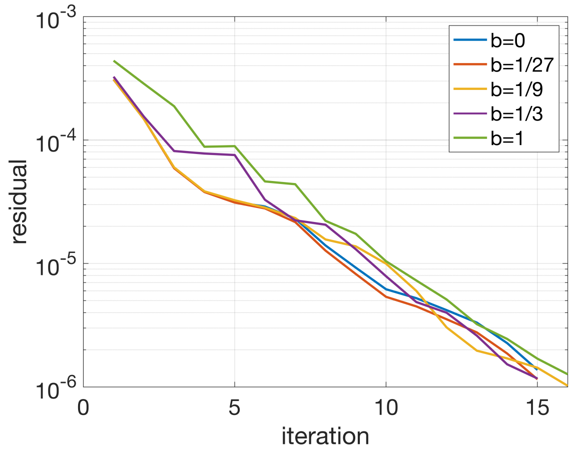
and

$$O_N = \sum_0^N \frac{1}{N} T_n - \frac{1}{2N} T_0 - \frac{1}{2N} T_N \tag{5.41}$$

where T_n are Chebyshev polynomials and the argument of T_n should be $(1 - \frac{A}{\lambda})$. λ should be larger than $\frac{\lambda_{max}}{2}$. With $\lambda = \lambda_{max}$, the minimum point is at λ_{max}. However, with λ slightly larger than $\frac{\lambda_{max}}{2}$, the range of high modes that are filtered is largest as shown in Fig 5.14. The detailed choices of α and β are calculated in the appendix.

Similar to the Jacobi iteration, the polynomial solvers can solve the equations. The results are plotted in Fig 5.24. The solvers are fast initially when they work on the high modes and become slow with the low modes. Although the polynomial solver with the Chebyshev polynomials converges slowly, it should be noted that Chebyshev polynomials removes the high modes faster as shown in Fig 5.14.

In Fig 5.25, we show the results where we use the polynomial solvers in the preconditioner. It can be seen that by utilizing the Chebyshev polynomials, the algorithm is four times faster than the



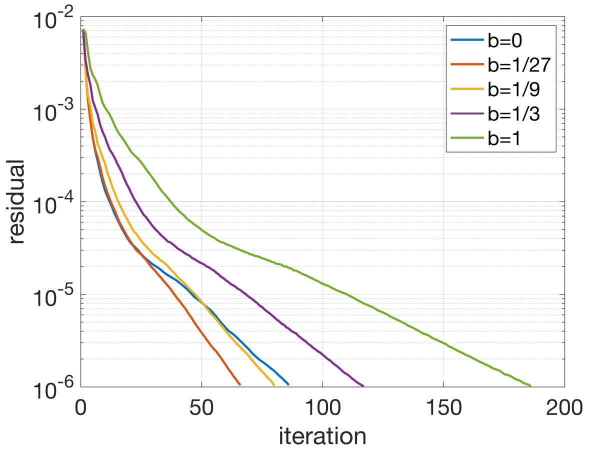


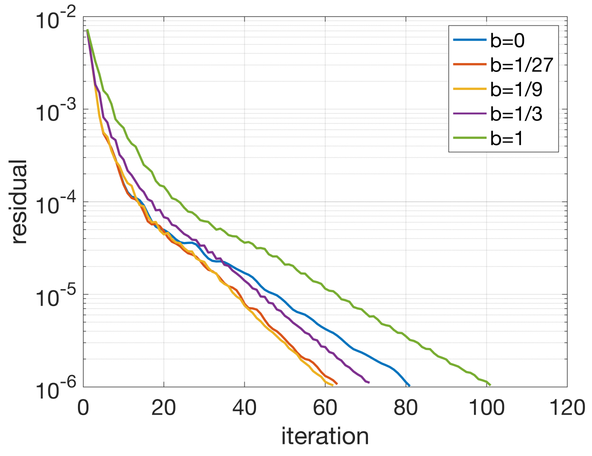
Figure 5.25: Left: Chebyshev polynomials are applied on the error during the inner solver; right: $(1 - \frac{A}{\lambda})^k$ is applied to the error during the inner solver, $\lambda = 300$. There are 100 iterations in the inner solver. By using the solver which utilizes the Chebyshev polynomials, the convergence rate is four times faster compared to the case where Jacobi iteration is used.

algorithm where the Jacobi iteration is used in the preconditioner. However, the total number of operations of $D^\dagger D$ is 3000 which means that it's still not good enough.

In Fig 5.26, we use the near-null vectors to further remove the high modes. With 10 near-null vectors, the convergence rate is much faster compared to $b = 0$ or the conjugate gradient. Although the total number of the operations of $D^\dagger D$ is similar to the conjugate gradient, it should be noticed that the convergence rate is much better. Additionally, the equations are currently solved to 10^{-6}. If we solve the equation to a higher precision, the algorithm would be beneficial. Lastly, since the number of preconditioned conjugate gradient iteration is small, the overhead brought by the interpolation and restriction will also be small. This is the advantage of the preconditioner compared to the previous section.

5.8 Summary and outlook

In this chapter, several algorithms are developed based on the idea of the renormalization group. Since we can block the fine gauge fields to coarse gauge fields according to the renormalization



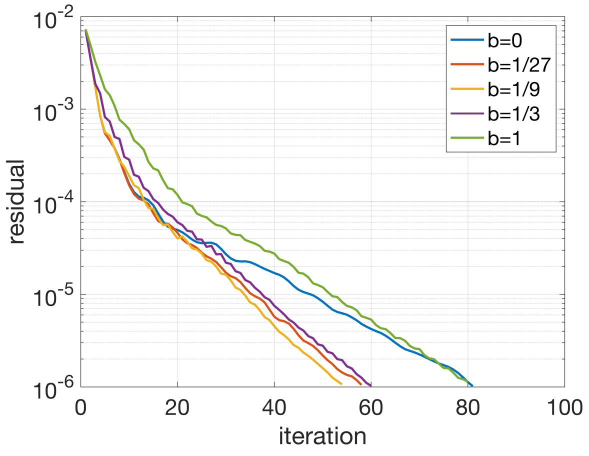


Figure 5.26: Left: 5 near-null vectors; right: 10 near-null vectors. $\lambda = 300$. 20 inner iterations. Chebyshev polynomials are used on the error. With the near-null vectors, the convergence rate is improved with the coarse lattice and it's optimal when $b = 1/9$.

group, the low-mode space of the coarse operator corresponds to the low-mode space of the fine operator. Following the idea, we can use the coarse lattice to solve the low modes of the fine lattice.

By using the coarse lattice in the preconditioner and utilizing the near-null vectors as the filter, we are able to achieve better convergence rate compared to the conjugate gradient algorithm. The iteration number as well as the operations of $D^\dagger D$ is reduced by 33% to 44% when the coarse operator is used directly in the preconditioner. By using the polynomial solver in the preconditioner, the convergence rate is increased significantly and the overhead from restriction and interpolation is small. Although the number of the operations of $D^\dagger D$ is close to the number of the conjugate gradient iterations, the convergence rate is much faster and the algorithm will be beneficial when the precision requirement is higher than 10^{-6}.

However, we believe that we haven't fully harvested the power of the coarse lattice. For further development, it's critical to understand the source of the high modes introduced by the interpolation operator. One possible source is that the coarse operator and the fine operator have different structures in the fifth dimension. The residual mass of the coarse lattice (0.0085) is almost twice the residual mass of the fine lattice (0.0045). This implies that the low modes of the coarse and

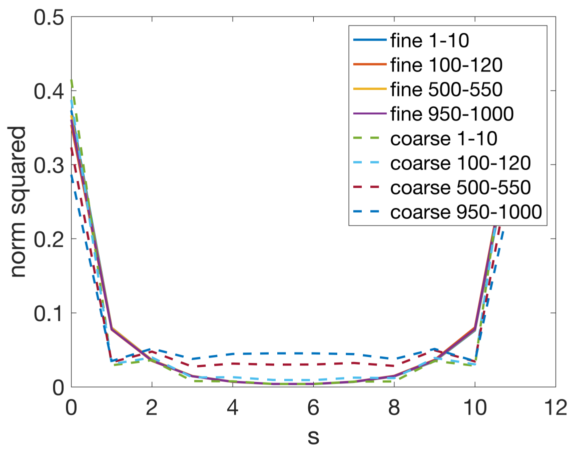


Figure 5.27: Average eigenvector supports on the fifth dimension. The numbers in the legend indicate the eigenvector index. The results show that the eigenvectors on the coarse lattice have different structures compared to the eigenvectors on the fine lattice.

fine lattice have different structures in the fifth dimension. In Fig 5.27, the average norm squared on different fifth-dimension slices are plotted. It verifies that the low modes have different fifth-dimension dependence for the coarse lattice and the fine lattice. Since the low modes are different, when the coarse lattice is used to solve the low modes of the fine lattice, there will be mismatch in the results. Up till now, we have treated the problem as a four dimensional problem and we haven't touched the fifth dimension. We believe it'll be important to correct the mismatch in the fifth dimension. In addition, using more efficient high-mode solvers, possibly some solvers that are similar to the multisplitting algorithm, may also be beneficial.

Conclusion

In this work, we developed different approaches to study the topological properties of the lattice gauge fields, calculated m_η and $m_{\eta'}$ and constructed the multisplitting algorithm as well as the renormalization group based preconditioned conjugate gradient algorithm. Now we summarize our results and discuss how the work could be beneficial for future research.

We utilized different methods to probe the topological properties of the lattice gauge fields. We calculated the topological charge from the smoothed gauge fields and probed the topological properties using the close quark loops. What's unique is that we studied the low-lying eigenvectors of the Shamir domain wall operators on the gauge field without smoothing and we were able to calculate topological properties from the eigenvectors. However, it's difficult to obtain clear individual topological modes because of the residual mass caused by the finite fifth dimension. For future work, it'll be desirable to consider low-lying eigenvectors of other fermion operators that don't suffer from the residual mass. Once individual topological modes are obtained, they could be used in multiple circumstances including the study of the topological evolution during the gauge field generation. Specifically, it can help identifying the long lived topological modes during the evolution.

m_η and $m_{\eta'}$ were calculated by considering both the fermion correlators and the topological charge density correlators. The mixing angle was also obtained. The errors of η and η' were decreased to percentage level. In particular, by including the easily calculated topological charge density correlators, the error for η' was decreased by 18%. Our current work hasn't included the systematic effect caused by the lattice spacing and the lattice volume. By including different ensembles, the results will be more robust.

The multisplitting preconditioned conjugate gradient algorithm was developed. The algorithm uses the multisplitting algorithm as the preconditioner and it can decrease the communication and use the local computational power more effectively. This algorithm has already been implemented by [35] and the algorithm is very helpful for the gauge evolution.

We constructed several versions of the renormalization group based preconditioned conjugate gradient algorithm. [35] constructed an algorithm that's able to block fine gauge fields to coarse gauge fields so that the they lie on the renormalization group trajectory. We found that the two lattices share the same low-mode space and we were able to use the coarse operator in the preconditioned conjugate gradient algorithm. By using the near-null vectors as the filter, the iteration number is deceased by 33% to 44% compared to the conjugate gradient algorithm. When we use the polynomial solver in the preconditioner, the convergence rate is optimized and we have relatively small overhead from the restriction and interpolation. Nevertheless, we believe we haven't fully utilized the low modes of the coarse lattice. We found that many high modes are introduced during the interpolation. For future development, it'll be important to understand how the high modes are introduced to avoid the issue. One possible direction is to match the structure of the coarse operator and the fine operator in the fifth dimension. Additionally, one could also utilize approaches like the multisplitting algorithm as better solvers in the preconditioner.

References

[1] H. J. Rothe, *Lattice gauge theories: an introduction*. World Scientific Publishing Company, 2005, vol. 74.

[2] Y Iwasaki and T Yoshie, "Renormalization group improved action for su (3) lattice gauge theory and the string tension," *Physics Letters B*, vol. 143, no. 4-6, pp. 449–452, 1984.

[3] K. G. Wilson, "Quarks and strings on a lattice," in *New phenomena in subnuclear physics*, Springer, 1977, pp. 69–142.

[4] H. B. Nielsen and M. Ninomiya, "No-go theorum for regularizing chiral fermions," Science Research Council, Tech. Rep., 1981.

[5] Y. Shamir, "Chiral fermions from lattice boundaries," *Nuclear Physics B*, vol. 406, no. 1-2, pp. 90–106, 1993.

[6] R Arthur, T Blum, P. Boyle, N. Christ, N Garron, R. Hudspith, T Izubuchi, C Jung, C Kelly, A. Lytle, *et al.*, "Domain wall qcd with near-physical pions," *Physical Review D*, vol. 87, no. 9, p. 094 514, 2013.

[7] R. C. Brower, H. Neff, and K. Orginos, "Möbius fermions: Improved domain wall chiral fermions," *Nuclear Physics B-Proceedings Supplements*, vol. 140, pp. 686–688, 2005.

[8] G. McGlynn, "Pos (lattice 2015) 019 algorithmic improvements for weak coupling simulations of domain wall fermions," 2015.

[9] P. Boyle, N. Christ, N Garron, C Jung, A Jüttner, C Kelly, R. Mawhinney, G McGlynn, D. Murphy, S Ohta, *et al.*, "Low energy constants of s u (2) partially quenched chiral perturbation theory from n f= 2+ 1 domain wall qcd," *Physical Review D*, vol. 93, no. 5, p. 054 502, 2016.

[10] R. Mawhinney and D. Murphy, "Nlo and nnlo low energy constants for $su(2)$ chiral perturbation theory," *arXiv preprint arXiv:1511.04419*, 2015.

[11] A. Kennedy, "Hybrid monte carlo," *Nuclear Physics B-Proceedings Supplements*, vol. 4, pp. 576–579, 1988.

[12] M. Lüscher, "Construction of a selfadjoint, strictly positive transfer matrix for euclidean lattice gauge theories," *Communications in Mathematical Physics*, vol. 54, no. 3, pp. 283–292, 1977.

[13] A. A. Belavin, A. M. Polyakov, A. S. Schwartz, and Y. S. Tyupkin, “Pseudoparticle solutions of the yang-mills equations,” *Physics Letters B*, vol. 59, no. 1, pp. 85–87, 1975.

[14] G. Hooft, “Symmetry breaking through bell-jackiw anomalies,” *Physical Review Letters*, vol. 37, pp. 8–11, 1976.

[15] S. Coleman, “The uses of instantons,” in *The whys of subnuclear physics*, Springer, 1979, pp. 805–941.

[16] P. De Forcrand, M. G. Perez, and I.-O. Stamatescu, “Topology of the su (2) vacuum: A lattice study using improved cooling,” *Nuclear Physics B*, vol. 499, no. 1-2, pp. 409–449, 1997.

[17] M. Lüscher, “Properties and uses of the wilson flow in lattice qcd,” *Journal of High Energy Physics*, vol. 2010, no. 8, pp. 1–18, 2010.

[18] G. Liu, “Quark eigenmodes and lattice qcd.,” 2003.

[19] T. Kalkreuter and H. Simma, “An accelerated conjugate gradient algorithm to compute low-lying eigenvalues—a study for the dirac operator in su (2) lattice qcd,” *Computer Physics Communications*, vol. 93, no. 1, pp. 33–47, 1996.

[20] D. Calvetti, L. Reichel, and D. C. Sorensen, “An implicitly restarted lanczos method for large symmetric eigenvalue problems,” *Electronic Transactions on Numerical Analysis*, vol. 2, no. 1, p. 21, 1994.

[21] S. Weinberg, “The u (1) problem,” *Physical Review D*, vol. 11, no. 12, p. 3583, 1975.

[22] J. Beringer *et al.*, “Particle data group,” *Phys. Rev. D*, vol. 86, no. 010001, 2012.

[23] R. G. Miller, “The jackknife-a review,” *Biometrika*, vol. 61, no. 1, pp. 1–15, 1974.

[24] A. Ranganathan, “The levenberg-marquardt algorithm,” *Tutoral on LM algorithm*, vol. 11, no. 1, pp. 101–110, 2004.

[25] N. Christ, C Dawson, T Izubuchi, C Jung, Q Liu, R. Mawhinney, C. Sachrajda, A Soni, R Zhou, *et al.*, “η and η mesons from lattice qcd,” *Physical review letters*, vol. 105, no. 24, p. 241 601, 2010.

[26] E. V. Shuryak and J. Verbaarschot, “Screening of the topological charge in a correlated instanton vacuum,” *Physical Review D*, vol. 52, no. 1, p. 295, 1995.

[27] H Fukaya, S Aoki, G Cossu, S Hashimoto, T Kaneko, J Noaki, J. Collaboration, *et al.*, “η meson mass from topological charge density correlator in qcd,” *Physical Review D*, vol. 92, no. 11, p. 111 501, 2015.

[28] L. Lellouch, R. Sommer, A. Vladikas, B. Svetitsky, *et al.*, *Modern perspectives in lattice QCD: Quantum field theory and high performance computing*, August 2009. Oxford University Press, 2011, vol. 93.

[29] J. R. Shewchuk *et al.*, *An introduction to the conjugate gradient method without the agonizing pain*, 1994.

[30] D. P. O'leary and R. E. White, "Multi-splittings of matrices and parallel solution of linear systems," *SIAM Journal on algebraic discrete methods*, vol. 6, no. 4, pp. 630–640, 1985.

[31] M. A. Clark, R. Babich, K. Barros, R. C. Brower, and C. Rebbi, "Solving lattice qcd systems of equations using mixed precision solvers on gpus," *Computer Physics Communications*, vol. 181, no. 9, pp. 1517–1528, 2010.

[32] M. Lüscher, "Solution of the dirac equation in lattice qcd using a domain decomposition method," *Computer physics communications*, vol. 156, no. 3, pp. 209–220, 2004.

[33] Y. Osaki and K.-I. Ishikawa, "Domain decomposition method on gpu cluster," *arXiv preprint arXiv:1011.3318*, 2010.

[34] R. Babich, G Shi, M. Clark, R. Brower, B Joó, and S Gottlieb, "Scaling lattice qcd beyond 100 gpus," in *SC'11: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, IEEE, 2011, pp. 1–11.

[35] J. Tu, "Lattice qcd simulations towards strong and weak coupling limits," Ph.D. dissertation, Columbia University, 2020.

[36] J. Brannick, R. C. Brower, M. Clark, J. C. Osborn, and C. Rebbi, "Adaptive multigrid algorithm for lattice qcd," *Physical review letters*, vol. 100, no. 4, p. 041 601, 2008.

[37] A. Frommer, K. Kahl, S. Krieg, B. Leder, and M. Rottmann, "Adaptive aggregation-based domain decomposition multigrid for the lattice wilson–dirac operator," *SIAM journal on scientific computing*, vol. 36, no. 4, A1581–A1608, 2014.

[38] H. Yin and R. D. Mawhinney, "Improving dwf simulations: The force gradient integrator and the m\" obius accelerated dwf solver," *arXiv preprint arXiv:1111.5059*, 2011.

[39] M. Clark, C. Jung, and C. Lehner, "Multi-grid lanczos," in *EPJ Web of Conferences*, EDP Sciences, vol. 175, 2018, p. 14 023.

[40] O. Tatebe, "The multigrid preconditioned conjugate gradient method," in *NASA conference publication*, NASA, 1993, pp. 621–621.

[41] M. H. Gutknecht and S. Röllin, "The chebyshev iteration revisited," *Parallel Computing*, vol. 28, no. 2, pp. 263–283, 2002.

[42] R. C. Brower, H. Neff, and K. Orginos, “The möbius domain wall fermion algorithm,” *Computer Physics Communications*, vol. 220, pp. 1–19, 2017.

Appendix A: The Lanczos algorithm

The Lanczos algorithm is used repeatedly to calculate the eigenvectors of a Hermitian matrix. Here we briefly introduce the algorithm as well as one of the variation: the implicitly restarted Lanczos algorithm.

For the derivation of the Lanczos algorithm, it's worthwhile to start from the power iteration. In the power iteration, a vector v is repeatedly multiplied by the $n \times n$ matrix A. Since the component with the largest eigenvalue gets amplified, $A^n v$ will approach the eigenvector with the largest eigenvalue. However, the method is not efficient because it only pays attention to the last results. The Lanczos algorithm preserve the intermediate information by utilizing the Krylov space. Through careful design, the algorithm is able to give a $n \times m$ matrix V with orthogonal columns and tridiagonal real symmetric matrix $T = V^\dagger AV$. By calculating the eigenvector y of the matrix T, the eigenvector of the matrix A is Vy. The algorithm is demonstrated below.

Once the matrix T is obtained, it could be factorized by QR decomposition and other methods.

Although the Lanczos algorithm is very effective, it's numerically unstable and can't be used in practice. To solve the instability, one of the most effective methods is the implicitly restarted Lanczos algorithm [20]. The key idea is that the Lanczos algorithm starts from the vector v_1. If v_1 is mainly composed with the desired eigenvectors, the algorithm will converge quickly. In the implicitly restarted Lanczos algorithm, the Lanczos iteration is first applied for $m + k$ times. Then by proper manipulation of the matrix V and T, the starting vector v_1 as well as the results for the first m iteration of the Lanczos iteration are updated implicitly. Afterwards, another k steps of Lanczos iteration followed by the restart is done. The process is repeated until the eigenvectors are converged. This is the algorithm that's used in this research.

Algorithm 4: The Lanczos algorithm

```
Result: A $n \times m$ matrix $V$ with orthogonal columns and tridiagonal real symmetric matrix
        $T = V^{\dagger}AV$
Let $v_1$ be a random vector with length $n$ and norm 1
Initialize $w'_1 = Av_1$, $\alpha_1 = \langle w'_1 v_1 \rangle$, $w_1 = w'_1 - \alpha_1 v_1$
for $j$=2,...$m$ do
    $\beta_j = ||w_{j-1}||$
    if $\beta_j \neq 0$, $v_j = w_{j-1}/\beta_j$; else let $v_j$ be a random vector with norm 1 and orthogonal to
     $v_1, ...v_{j-1}$
    $w'_j = Av_j$
    $\alpha_j = \langle w'_j v_j \rangle$
    $w_j = w'_j - \alpha_j v_j - \beta_j v_{j-1}$
end
$v_1, ...v_m$ are the columns of the matrix $V.T$ has diagonal elements $\alpha_1, ...\alpha_m$. For
 off-diagonal elements, $T_{j,j+1} = T_{j+1,j} = \beta_{j+1}$.
```

Appendix B: The diagonal elements of $D^\dagger D$ for the Mobius domain wall operator

The diagonal elements of $D^\dagger D$ for the Mobius domain wall operator are needed for the Jacobi algorithm in chapter 5. Here we show the calculation of the diagonal elements.

We follow the notation of [42]. For the Mobius domain wall operator, we need the components:

$$D_+^s = b_5(s)D^{Wilson}(M_5) + 1 \tag{B.1}$$

and

$$D_-^s = c_5(s)D^{Wilson}(M_5) - 1 \tag{B.2}$$

where

$$D_{Wilson}(M_5) = M + 4 - \frac{1}{2}D_{hop} \tag{B.3}$$

and

$$D_{hop} = (1-\gamma_\mu)U_\mu(x)\delta_{x+\mu,y} + (1+\gamma_\mu)U_\mu^\dagger(y)\delta_{x-\mu,y} \tag{B.4}$$

And the Mobius domain wall operator is:

$$D_{DW}^5 = \begin{pmatrix} D_+^1 & D_-^1 P_- & 0 & \dots & -mD_-^1 P_+ \\ D_-^2 P_+ & D_+^2 & D_-^2 P_- & \dots & 0 \\ 0 & D_-^3 P_+ & D_+^3 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ -mD_-^{L_s} P_- & 0 & 0 & \dots & D_+^{L_s} \end{pmatrix} \tag{B.5}$$

In the case of the Shamir domain wall operator, $D_- = 1$ because $c_s = 0$ and $b_s = 1$.

We concentrate on the Mobius operator where b and c are constant for different s. We first consider $D_{Wilson}(M_5)^\dagger D_{Wilson}(M_5)$. This is calculated as:

$$(M + 4 - \frac{1}{2}D^\dagger_{hop})(M + 4 - \frac{1}{2}D_{hop}) \tag{B.6}$$

D_{hop} has diagonal elements 0, so to calculate the diagonal elements, we only need to consider:

$$(M + 4)^2 + \frac{1}{4}D^\dagger_{hop}D_{hop} \tag{B.7}$$

For the diagonal elements of $D^\dagger_{hop}D_{hop}$, the site must hop forward then hop back. So for the color part, the matrix is $U^\dagger_\mu(x)U_\mu(x) = 1$. For the spin part, we have:

$$\frac{1}{4}(1 - \gamma_\mu)^2 + (1 + \gamma_\mu)^2 = 1 \tag{B.8}$$

because $\gamma^\dagger_\mu = \gamma_\mu$ and $\gamma^2_\mu = 1$. Since there are four directions, we have diagonal elements:

$$\text{diag}(D_{Wilson}(M_5)^\dagger D_{Wilson}(M_5)) = (M + 4)^2 + 4 \tag{B.9}$$

$D^\dagger_+D_+$ and $D^\dagger_-D_-$ could be calculated similarly.

For D^5_{DW}, we first look at $s \neq 1, L_s$. We need to consider

$$D^\dagger_+D_+ + P_+D^\dagger_-D_-P_+ + P_-D^\dagger_-D_-P_- = D^\dagger_+D_+ + D^\dagger_-D_- \tag{B.10}$$

so the diagonal elements are:

$$(b(4 - M) + 1)^2 + 4b^2 + (c(4 - M) - 1)^2 + 4c^2 \tag{B.11}$$

For $s = 1$, we have:

$$D^\dagger_+D_+ + P_+D^\dagger_-D_-P_+ + m^2P_-D^\dagger_-D_-P_- \tag{B.12}$$

So for spin 0, 1 components, the diagonal elements are:

$$(b(4-M)+1)^2+4b^2+(c(4-M)-1)^2+4c^2 \tag{B.13}$$

For spin 2, 3 components, the diagonal elements are:

$$(b(4-M)+1)^2+4b^2+m^2((c(4-M)-1)^2+4c^2) \tag{B.14}$$

For $s = L_s$, spin 0, 1 components, we have:

$$(b(4-M)+1)^2+4b^2+m^2((c(4-M)-1)^2+4c^2) \tag{B.15}$$

For spin 2, 3 components, we have:

$$(b(4-M)+1)^2+4b^2+(c(4-M)-1)^2+4c^2 \tag{B.16}$$

This concludes the calculation of the diagonal elements of $D^\dagger D$ for the Mobius domain wall operator.

Appendix C: The high-mode polynomial solvers

In chapter 5, we need solvers that solve the high-mode part of the equation $Ax = b$ first. Here we show two such solvers with the idea that the solver should apply a high-mode polynomial filter to the error.

The solvers are motivated by the Chebyshev iteration [41]. There are multiple versions of the Chebyshev iteration. Two of which are:

$$x_{k+1} = x_k - \alpha_{k+1}(Ax_k - b) \tag{C.1}$$

and

$$x_{k+1} = x_k - \alpha_{k+1}(Ax_k - b) - \beta_{k+1}(x_k - x_{k-1}) \tag{C.2}$$

With iteration (C.1), the error $e_k = x_k - x$ decrease according to:

$$e_k = \Pi_1^N (1 - \alpha_k A) e_0 \tag{C.3}$$

With iteration (C.2), the error decreases according to:

$$e_k = O_k e_0 \tag{C.4}$$

where O_k satisfies:

$$O_k(A) = (1 - \beta_k - \alpha_k A) O_{k-1} + \beta_k O_{k-2} \tag{C.5}$$

With proper choices of α and β, the operator that applies to the error could be the Chebyshev polynomial. In our case, we would like the operator that applies to the error to be a high-mode filter. In this appendix, we consider two high-mode filters that appeared in chapter 5 and we'll calculate α and β so that the operator that applies to the error will be these operators:

$$O_N = (1 - \frac{A}{\lambda})^N \tag{C.6}$$

and

$$O_N = \sum_0^N \frac{1}{N} T_n - \frac{1}{2N} T_0 - \frac{1}{2N} T_N \tag{C.7}$$

where T_n are Chebyshev polynomials and the argument of T_n should be $(1 - \frac{A}{\lambda})$. λ should be larger than $\frac{\lambda_{max}}{2}$.

To apply (C.6) to the error, we could simply use iteration (C.1) and use

$$\alpha = \frac{1}{\lambda} \tag{C.8}$$

To apply (C.7) to the error, we have to find a recursion relation for the (C.7). For Chebyshev polynomials, we have $T_{n+1} = 2AT_n - T_{n-1}$. Therefore, we have the recursion relation:

$$2N(1 - \frac{A}{\lambda})O_N - (N-1)O_{N-1} = O_{N+1} \tag{C.9}$$

This is equivalent to:

$$O_{N+1} = (\frac{2N}{N+1} - \frac{2NA}{(N+1)\lambda})O_N - \frac{N-1}{N+1} O_{N-1} \tag{C.10}$$

Compared to (C.5), we find:

$$\alpha_{k+1} = \frac{2N}{N+1} \tag{C.11}$$

and

$$\beta_{k+1} = -\frac{N-1}{N+1} \tag{C.12}$$

The first two iterations can't be obtained in a similar way. The operators O_1 and O_2 are:

$$O_1 = 1 - \frac{A}{2\lambda} \tag{C.13}$$

and

$$O_2 = (1 - \frac{A}{\lambda})(1 - \frac{A}{2\lambda}) \tag{C.14}$$

O_1 and O_2 could be construed with iteration (C.1).

As a conclusion, we could use iteration (C.1) or (C.2) with the calculated coefficients in (C.8) or (C.11) to (C.12) and the solver will solve the high-mode part of the equation first.