Improving DWF Simulations: the Force Gradient Integrator and the Möbius Accelerated DWF Solver

Hantao Yin[*]

Department of Physics, Columbia University, New York, NY 10025, USA
E-mail: `yinnht@phys.columbia.edu`

Robert D. Mawhinney

Department of Physics, Columbia University, New York, NY 10025, USA
E-mail: `rdm@physics.columbia.edu`

We have implemented a variant of the force gradient integrator proposed by Kennedy et.al. and are using it in our production 2+1 flavor DWF simulations with pion masses of 180 MeV in $(4.5\text{fm})^3$ volumes. We find modest speed-ups ($\sim 20\%$) from using the force gradient integrator, compared to our previously used Omelyan integrator. On other ensembles, primarily finite temperature 2+1 flavor DWF QCD, we have extensively tuned the Hasenbusch preconditioning masses and achieved speed-ups of 2-3x. Here we have also switched to the force gradient integrator, but this change has not had any impact on the speed. We also report on an improved solver for DWF, which uses Möbius fermions, with a smaller fifth dimension than the original DWF fermions, as an intermediate step in the generation of solutions of the Dirac equation. This approach cuts the number of effective Dirac applications by approximately a factor of 2 when the conjugate gradient iteration count is large.

---

[*]Speaker.

1. Introduction

Ensemble generation constitutes a large portion of the computing time in a lattice calculation, and in many cases is the most time consuming part. Most lattice calculations use a variant of the Hybrid Monte Carlo (HMC) algorithm to generate lattice configurations for later measurements. As such, optimizing the HMC integration scheme is important for many lattice calculations.

We make use of the Hasenbusch mass splitting method and the recently devised force gradient integrator to optimize the molecular dynamics evolution in our lattice computations. In particular we propose a very efficient way to implement the force gradient integrator. The use of this new integrator in combination with the Hasenbusch mass splitting scheme has achieved a factor of 2 speed up in many of our practical calculations.

Domain wall fermions (DWF) are widely used for many lattice QCD calculations. To extend existing efforts on DWF physics, we present a method that employs the similarity between the kernels of Möbius fermion and standard DWF to help accelerate the solution of the DWF Dirac equations.

2. Force Gradient Integrator Implementation

As shown in [1] the force gradient PQPQP integrator (FGI) can be expressed as

$$e^{\frac{1}{6}\tau\zeta_S}e^{\frac{1}{2}\tau\zeta_T}e^{\frac{2}{3}\tau\zeta_S-\frac{1}{72}\tau^3\zeta_{\{S,\{S,T\}\}}}e^{\frac{1}{2}\tau\zeta_T}e^{\frac{1}{6}\tau\zeta_S} \tag{2.1}$$

The key step for this integrator involves the following update on the momentum field

$$p_i \leftarrow p_i - \frac{2}{3}\tau e_i(S) + \frac{1}{36}\tau^3 e^j(S)e_j e_i(S) \tag{2.2}$$

e_i can be seen as the generalization of the vector fields $\partial/\partial x$ in the SU(3) Lie group space of link variables. It follows the Leibniz rule and when acting on the gauge field it is effectively equivalent to left multiplication by the corresponding generator $e_i(U) = T_i U$.

To implement the second order derivative term $e^j(S)e_j e_i(S)$ in FGI, an approach using the Taylor expansion can be used. Since e_i and e_j are interchangeable in this extra term, the following approximation holds

$$\frac{2}{3}\tau e_i(S) - \frac{1}{36}\tau^3 e^j(S)e_j e_i(S) = \frac{2}{3}\tau\left(1 - \frac{1}{24}\tau^2 F^j e_j\right)e_i(S) = \frac{2}{3}\tau e^{-\frac{1}{24}\tau^2 F^j e_j}e_i(S) + \mathcal{O}(\tau^5) \tag{2.3}$$

Here $F^j = e^j(S)$ and it must be treated as independent of link variables in the above equation(e_i acting on F^j is defined to be 0 despite the fact that F^j depends on the gauge field). Note that since the FGI has a local error of $\mathcal{O}(\tau^5)$, the above approximation does not change the size of the error.

Effectively we obtain from the last expression

$$e^{-\frac{\tau^2}{24}F^j e_j}e_i(S) = e_i(S[U']) \tag{2.4}$$

with

$$U' = e^{-\frac{\tau^2}{24}F^j e_j}U = e^{-\frac{\tau^2}{24}F^j T_j}U \tag{2.5}$$

As a consequence we implement the force gradient step using the following two-step approach

1. (the force gradient step) Calculate $F^j = e^j(S[U])$. Then update the gauge field temporarily according to $U' = e^{-\frac{\tau^2}{24}F^j T_j}U$.

2. (the regular step) Calculate again $e_i(S[U'])$ using the updated gauge field. The corresponding force is added to the momentum field. Upon the completion of this step, restore the gauge field to U.

The above implementation of the force gradient integrator is different from [1], since the approximation in eq.(2.3), changes the $\mathcal{O}(\tau^5)$ error term. The resulting integrator is still symmetric and symplectic.

There is an additional benefit from using this implementation. The two steps described above are exactly the same except using different gauge fields U and U', and U' differs from U by only a small factor ($\sim \tau^2/24$). If a fermion action is involved, the solution from the first step can be used as an initial guess for the second step when solving the Dirac equations for the force terms. This "force gradient forecasting" reduces costs significantly. Unlike the chronological solver, it is independent of the direction of the molecular dynamics time and does not impact reversibility.

When testing the force gradient integrator on a $16^3 \times 32 \times 16$ lattice with 2+1 flavor dynamical DWF fermions (420MeV pion) using quotient and rational quotient actions, we were able to raise the top level step size from 1/4 as in the Omelyan to 1/3 for a unit trajectory length. However, due to the extra cost in the force gradient solve, the new integrator does not provide a significant speed up over a finely tuned Omelyan integrator in the case we tested. For larger 32^3 volumes, we have seen a modest benefit from the FGI and will test on 48^3 volumes soon, where the benefit may be larger.

3. Hasenbusch Mass Splitting

The Hasenbusch factorization of the DWF fermion determinant entering in our evolutions

$$\det\left(\frac{D^\dagger(m)D(m)}{D^\dagger(1)D(1)}\right) = \det\left(\frac{D^\dagger(m)D(m)}{D^\dagger(m+\mu)D(m+\mu)}\right)\det\left(\frac{D^\dagger(m+\mu)D(m+\mu)}{D^\dagger(1)D(1)}\right) \quad (3.1)$$

has been proposed in[2] to separate the contribution to the fermion force from the light eigenmodes and heavier ones. The inversion of $D^\dagger(m)D(m)$ is more expensive than the introduced operator $D^\dagger(m+\mu)D(m+\mu)$. Urbach *et al.*[3] proposed that by assigning a smaller force to the more expensive part (via tuning μ) and putting it on a coarser time scale by means of a multiple level integrator, an acceleration of the molecular dynamics evolution can be achieved. We have used this approach for a number of years, but there are overheads associated with multiple time scale integrators.

Here we propose an approach that uses the idea of mass splitting, but without different time scales. Instead of putting the part with larger force on a finer time scale, we split the force further via introducing additional intermediate masses. Namely we introduce

$$\det\left(\frac{D^\dagger(m)D(m)}{D^\dagger(1)D(1)}\right) = \prod_{i=1}^{k+1}\det\left(\frac{D^\dagger(m+\mu_{i-1})D(m+\mu_{i-1})}{D^\dagger(m+\mu_i)D(m+\mu_i)}\right) \quad (3.2)$$

with $0 = \mu_0 < \mu_1 < \cdots < \mu_{k+1} = 1$. One advantage of this scheme over [3] is that all intermediate masses $\mu_i(i = 1, 2, \cdots, k)$ can be tuned continuously, while any multiple time-step integrator scheme only admits a $1 : n$ ratio on the number of force term evaluations, with integer n being the number of evaluations of the nested integrator per evaluation of the upper level integrator.

The observed benefit of this scheme is that all Dirac equations related to the force terms of the quotient actions need not be solved very accurately, if enough intermediate masses are introduced and tuned. We were generally able to solve the force term Dirac equations only up to a residual of 10^{-6} instead of the previously used 10^{-8}, thus saving a considerable amount of time.

4. Möbius Accelerated Domain Wall Fermion (MADWF)

We start by considering the general Dirac equation,

$$D_{DW}(m)x = b, \tag{4.1}$$

with D_{DW} being the standard domain wall fermion Dirac operator. The goal is to use Möbius fermions to construct an approximate equation. Before solving equation (4.1), we solve the approximated equation and use the solution as an initial guess.

The primary tool to relate the standard domain wall fermion and the Möbius fermion is the domain wall - overlap transformation[5]. It transforms the Möbius Dirac operator to an equivalent 4D overlap operator. By applying $P^{-1}D_{DW}^{-1}(1)$ to both sides of equation (4.1)[1], we can transform the 5D Dirac matrix into a diagonal form in the s direction,

$$\begin{pmatrix} D_{OV} & & & \\ S_2 S_3 \cdots S_L (D_{OV} - 1)d & 1 & & \\ S_3 \cdots S_L (D_{OV} - 1)d & & 1 & \\ \cdots & & & \cdots \\ S_L (D_{OV} - 1)d & & & & 1 \end{pmatrix} y = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \cdots \\ c_L \end{pmatrix}, \tag{4.2}$$

with $y = (y_1, y_2, y_3, \cdots, y_L)^{\mathrm{T}} = P^{-1}x$, L is the size of the s direction and

$$c = (c_1, c_2, c_3, \cdots, c_L)^{\mathrm{T}} = P^{-1}D_{DW}(1)^{-1}b. \tag{4.3}$$

The details, including the definition of symbols $S_2, \cdots, S_L$ and d, can be found in [5].

After the transformation the primary work is to solve the 4D equation

$$D_{OV}(m)y_1 = c_1 \tag{4.4}$$

with an overlap-like operator $D_{OV}(m)$. The rest of the work involves deriving the whole 5D solution from y_1, which can be computed with little effort, as will be seen below.

We first address the question of solving the 4D equation (4.4). We notice that $D_{OV}(m)$ is an approximation to the ideal overlap operator. In parallel we can also obtain a different finite approximation $D'_{OV}(m)$ from any 5D Möbius operator $D'_{DW}(m)$. For our purpose we replace $D_{OV}(m)$ in

---

[1] P is a simple matrix that shifts half of the spin component in the fermion vector by one in s direction, its detailed form can be found in [5].

equation (4.4), which comes from a standard domain wall fermion operator, by $D'_{OV}(m)$ that comes from a Möbius fermion operator. The solution y'_1 to the following equation

$$D'_{OV}(m)y'_1 = c_1 \tag{4.5}$$

can be directly used to approximate y_1, and thus reconstruct a 5D approximated solution to (4.1).

To derive the entire 5D solution y from y_1, we notice that for a true solution $y = (y_1, y_2, y_3, \cdots, y_L)^T$ we have

$$y_k = c_k - S_k S_{k+1} \cdots S_L (D_{OV} - 1) d y_1, \ k = 2, 3, \cdots, L. \tag{4.6}$$

This relation can also be used to construct the guessed solution once y'_1, the approximation to y_1, is obtained. Namely,

$$y'_k = c_k - S_k S_{k+1} \cdots S_L (D_{OV} - 1) d y'_1, \ k = 2, 3, \cdots, L, \tag{4.7}$$

where y'_k represents the approximation to y_k. This can be written in matrix form

$$\left(-D_{OV}y'_1, y'_2, y'_3, \cdots, y'_L\right)^T = P^{-1} D_{DW}(1)^{-1} D_{DW}(m) P \cdot \left(-y'_1, c_2, c_3, \cdots, c_L\right)^T. \tag{4.8}$$

Reconstructing the entire 5D guess in this way requires a Pauli-Villars DWF solve.

There remains the question of solving the 4D Möbius Dirac equation (4.5). To do this we reverse the above process. The detailed procedure can be found at [5]. By using the domain wall - overlap transformation on the 4D equation (4.5) we can transform it back to the 5D form

$$D'_{DW}(m)Py' = D'_{DW}(1)Pc' \tag{4.9}$$

with $c' = (c_1, 0, 0, \cdots, 0)$ and y'_1 being the 4D component of y' on the $s = 1$ hyperplane.

The algorithm then looks like

1. Construct c_1 from the original 5D source b, using equation (4.3).

2. Replace the 4D operator in equation (4.4) by a 4D operator from a Möbius fermion operator with appropriately chosen parameters.

3. Transform the 4D Möbius Dirac equation back to its 5D form (4.9) and solve the 5D equation, thus obtaining the 4D solution y'_1 to equation (4.5).

4. Use y'_1 as an approximation to the 4D domain wall Dirac equation (4.4), and reconstruct the 5D (approximated) solution using (4.8).

The above process requires two Pauli-Villars solves in standard DWF ((4.3) and (4.8)) and one Möbius solve on a lattice with possibly smaller size in s direction (4.9). The gain over a direct solve comes from the fact that the Möbius equation has a potentially smaller s dimension size. It's clear that the approximated solution y'_1 must be sufficiently close to the true solution y_1 to make this method beneficial. This leads to the problem of choosing the best parameters. The details of parameter tuning will be given later, in a separate paper.
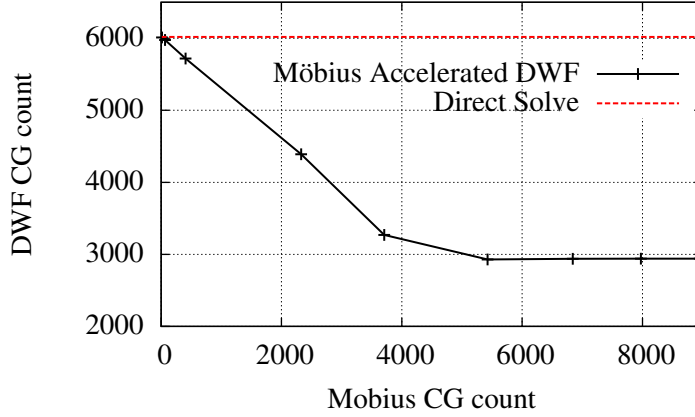
Figure 1: DWF CG count/Möbius CG count correlation, 160MeV, $32^3 \times 8$, with $L = 32$, $L' = 16$, $b = 1.5$, $c = 0.5$. A turning point is clearly presented in the curve, representing the best approximation can be obtained from solving the Möbius Dirac equation.

5. Defect Correction and Production Use

The Möbius approximation to the standard DWF Dirac equation can only achieve a certain accuracy (Figure (1)), as the two operators are not exactly the same. A simple defect-correction method can be used to achieve a higher accuracy in constructing the guessed solution. In this approach, one takes the guessed solution and calculates the residue, then constructs a second guess with the residue as the new source vector using the Möbius approach. The second guess can be simply added to the original guess to improve it. This method can be repeated if necessary.

	Direct CG solve	Möbius Accelerated DWF			
Operation	-	Möbius equation(*3)	Pauli-Villars(*6)	others	total
Op. Count	11290*32=3.6e5	4.6e3*12	1.0e2*32	-	2.0e5
time(s)	2672	285	25	125	1138

Table 1: Cost comparison of MADWF CG solver with a regular (zero started) CG solver. $L = 32$, $L' = 12$. Data obtained from a 512 node partition on BG/P, both solve to 10^{-10}. Note that there are 3 Defect-correction steps, each includes 2 Pauli-Villars unit mass inversions. The factor 32 and 12 in the "Op. Count" row are due to different 5th dimension sizes.

Table (1) compares this method and the direct solve approach on a $32^3 \times 64$, $L = 32$ lattice. The corresponding Möbius approximation has 5th dimension size $L' = 12$. Since multiple 5th dimension sizes are used we count the number of 4D Wilson Dirac operator applications (denoted by "Op. count" in the table). The table shows that MADWF reduces the number of 4D Wilson Dirac operator applications from 3.6e5 to 2.0e5, by a factor of 1.8. The MADWF solver uses 42% of wall clock time, when compared with a direct solve. Note that while some gain is due to code optimization, most is due to the fact that MADWF requires many fewer 4D Wilson Dirac operations.

We also note that the Möbius parameters need only be tuned once for a specific ensemble. Changing configuration or source vector has little effect on the quality of the Möbius approxima-

tion. Once tuned, the same set of Möbius parameters can be used to approximate the DWF Dirac equation for the entire ensemble.

6. Conclusion

We have presented a method to implement the force gradient integrator. By testing the force gradient integrator on an existing lattice we have shown that for small lattices the extra cost is high enough so any benefit is small. However, due to its better scaling behavior it's possible that the force gradient integrator performs better than other second order integrators on larger lattices or with longer trajectories.

We have explored the Hasenbusch mass splitting scheme and applied this method to our practical calculation. We have found that extra quotient actions introduced in this way allows the Dirac equations associated with the force term evaluations to be solved less accurately, reducing the cost of HMC considerably.

We have shown that the similarity between the 4D Möbius fermion Dirac operator and the Shamir domain wall fermion operator can be used to accelerate solving the DWF Dirac equation. We have achieved a factor of 2 speed up on a specific large lattices ($32^3 \times 64$, $L_s = 32$) that we are using in practical calculations.

7. Acknowledgment

We thank A. Kennedy and M. Clark for notes regarding the force gradient integrator and a prototype of a force gradient C++ implementation. We also thank our colleagues in the RBC and UKQCD collaborations for discussions and tests on the program. This work is done using the QCDOC supercomputers of the RBRC group, NYBlue at BNL and the BG/P of the ALCF at Argonne. We use the MDWF package for simulating the Möbius fermions.

References

[1] A. D. Kennedy, M. A. Clark and P. J. Silva, *Force Gradient Integrators*, arXiv:0910.2950v1[hep-lat].

[2] M. Hasenbusch, *Speeding up the hybrid Monte Carlo algorithm for dynamical fermions*, Physics Letters B 519 (2001) 177182.

[3] C. Urbach, K. Jansen, A. Shindler, U. Wenger, *HMC algorithm with multiple time scale integration and mass preconditioning*, Comput. Phys. Commun., 174(2006) 87-98.

[4] R.C. Brower, H. Neff and K. Orginos, *Möbius Fermions: Improved Domain Wall Chiral Fermions*, Nucl. Phys. B(Proc. Suppl.) 140(2005) 686-688.

[5] R.C. Brower, H. Neff and K. Orginos, *Möbius Fermions*, Nucl. Phys. B(Proc. Suppl.) 153(2006) 191-198.