

# ADAPTIVE AGGREGATION BASED DOMAIN DECOMPOSITION MULTIGRID FOR THE LATTICE WILSON DIRAC OPERATOR \*

A. FROMMER<sup>†</sup>, K. KAHL<sup>†</sup>, S. KRIEG<sup>‡</sup>, B. LEDER<sup>†</sup>, AND M. ROTTMANN<sup>†</sup>

**Abstract.** In lattice QCD computations a substantial amount of work is spent in solving discretized versions of the Dirac equation. Conventional Krylov solvers show critical slowing down for large system sizes and physically interesting parameter regions. We present a domain decomposition adaptive algebraic multigrid method used as a preconditioner to solve the “clover improved” Wilson discretization of the Dirac equation. This approach combines and improves two approaches, namely domain decomposition and adaptive algebraic multigrid, that have been used separately in lattice QCD before. We show in extensive numerical tests conducted with a parallel production code implementation that considerable speed-up over conventional Krylov subspace methods, domain decomposition methods and other hierarchical approaches for realistic system sizes can be achieved.

**Key words.** multilevel, multigrid, lattice QCD, Wilson Dirac operator, domain decomposition, aggregation, adaptivity, parallel computing.

**AMS subject classifications.** 65F08, 65F10, 65Z05, 65Y05

**1. Introduction.** Lattice QCD simulations are among the world’s most demanding computational problems, and a significant part of today’s supercomputer resources is spent in these simulations [9, 31]. Our concern in this paper is three-fold: We want to make the mathematical modeling related with QCD and lattice QCD more popular in the scientific computing community and therefore spend some effort on explaining fundamentals. On top of that we develop a new and efficient adaptive algebraic multigrid method to solve systems with the discretized Dirac operator, and we show results for a large number of numerical experiments based on an advanced, production code quality implementation with up-to-date physical data.

The computational challenge in lattice QCD computations consists of repeatedly solving very large sparse linear systems

$$Dz = b, \quad (1.1)$$

where  $D = D(U, m)$  is a discretization, typically the Wilson discretization, of the Dirac operator on a four-dimensional space-time lattice. The Wilson Dirac operator depends on a gauge field  $U$  and a mass constant  $m$ . Recently, lattices with up to  $144 \times 64^3$  lattice points have been used, involving the solution of linear systems with 452,984,832 unknowns [1, 5, 6, 23, 26]. Usually these linear systems are solved by standard Krylov subspace methods. Their iteration count increases tremendously when approaching the physically relevant parameter values (i.e., physical mass constants and lattice spacing  $a \rightarrow 0$ ), a phenomenon referred to as “critical slowing down” in the physics literature. Thus it is of utmost importance to develop preconditioners which overcome these scaling issues. The most common preconditioners nowadays are odd-even preconditioning [20, 37], deflation [41], and domain decomposition [28, 40]. While these approaches yield significant speed-ups over the unpreconditioned versions, their scaling behavior is unchanged and critical slowing down still occurs.

---

\*This work was partially funded by Deutsche Forschungsgemeinschaft (DFG) Transregional Collaborative Research Centre 55 (SFB/TRR55)

<sup>†</sup>Department of Mathematics, Bergische Universität Wuppertal, 42097 Germany, {frommer, kkaehl, leder, rottmann}@math.uni-wuppertal.de.

<sup>‡</sup>Department of Physics, Bergische Universität Wuppertal, 42097 Germany and Jülich Supercomputing Centre, Forschungszentrum Jülich, 52428 Jülich, Germany, s.krieg@fz-juelich.de.

Multigrid methods have been considered in the lattice QCD community as well, motivated by their potential (e.g., for elliptic PDEs) of convergence independent of the lattice spacing. However, due to the random nature of the gauge fields involved, the treatment of the lattice Dirac equation by multigrid methods based solely on the underlying PDE, has been elusive for the last twenty years [8, 17, 34, 56]. With the advent of *adaptive algebraic* multigrid methods, effective preconditioners for QCD calculations could be constructed in recent years. The pioneering work from [3, 4, 14, 44] showed very promising results. There, an adaptive non-smoothed aggregation approach based on [15] has been proposed for the solution of the Wilson Dirac system. An implementation is publicly available as part of the QOPQDP package [43].

Within the physics community, another hierarchical technique, the recently proposed domain decomposition type solver named *inexact deflation* developed in [41] is widely used. A well-optimized code for this solver is publicly available [38]. Inexact deflation can be regarded as an adaptive method as well. It performs a setup phase which allows the construction of a smaller system, the *little Dirac* operator, which is then used as part of an efficient preconditioner. Although there is an intimate connection with the aggregation based multigrid approach from [15], inexact deflation seems to have been developed completely independently. As a consequence, the inexact deflation method does not resemble a typical multigrid method in the way its ingredients are arranged. In particular, it requires the little Dirac system to be solved to high accuracy in each iteration.

In this paper we present a multigrid method that combines aspects from [41], namely a domain decomposition smoother, and from non-smoothed aggregation as in [4, 44]. Our approach elaborates on the multigrid methods from [4, 44] in that we use a domain decomposition method as the smoother instead of the previously used Krylov subspace smoother. This allows for a natural and efficient parallelization, also on hybrid architectures. Moreover, we substantially improve the adaptive setup from [4, 44] and [41] in the sense that less time is required to compute the operator hierarchy needed for an efficient multigrid method. Our approach can also be regarded as turning the domain decomposition technique from [41] into a true multigrid method. The “little Dirac” system now needs to be solved only to low accuracy. This allows, in particular, to apply the method recursively and thus opens the way for a more efficient *multi-grid* method instead of just a two-grid method. With the inexact deflation approach this is not possible.

The paper is organized as follows. In section 2 we give an introduction into lattice QCD for the non-specialist and we introduce the domain decomposition Schwarz method in this context. In section 3 we first outline algebraic multigrid methods in general and then focus on aggregation based approaches. Thereby we address the peculiarities of lattice QCD and explain different possible adaptive strategies for the construction of the multigrid hierarchy. The inexact deflation method from [41] is discussed in section 4, where in particular we point out the differences to a multigrid method and describe the adaptive nature of its setup. In section 5 we finally formulate our multigrid approach, for which we present thorough numerical tests in section 6.

**2. Lattice Quantum Chromodynamics.** Quantum Chromodynamics (QCD) or the theory of strong interactions, is a quantum field theory in four-dimensional space-time and part of the standard model of elementary particle physics. It has a high predictive power, i.e., a small number of free parameters. Predictions that can be deduced from this theory are amongst others the masses of hadrons, composite particles bound by the strong interaction (e.g., nucleon, pion; cf. [22]). The masses

of hadrons and many other predictions have to be obtained non-perturbatively, i.e., via numerical simulations requiring the discretization and numerical evaluation of the theory. After a brief description of relevant aspects of continuum QCD we introduce its discretization on a hyper-cubic lattice and discuss the need of iterative (Krylov subspace) methods for the solution of the arising linear systems of equations. Due to their ill-conditioned nature, preconditioning is advised and the use of a domain decomposition method is discussed as a prerequisite for our multigrid construction.

**2.1. Continuum QCD.** A thorough description of QCD as a quantum field theory is beyond the scope of this paper. Instead we just introduce the reader to important concepts and the notation necessary to understand the lattice discretization.

The degrees of freedom of QCD are matter fields, called quarks, and gauge fields, called gluons. At the heart of the numerical methods for lattice QCD is a discretized version of the continuum Dirac equation

$$(\mathcal{D} + m)\psi = \eta \quad (2.1)$$

which describes the dynamics of the quarks and the interaction of quarks and gluons for a given gluon field background. Here,  $\psi = \psi(x)$  and  $\eta = \eta(x)$  represent matter fields. These depend on  $x$ , the points in space-time,  $x = (x_0, x_1, x_2, x_3)^1$ . The gluons are represented in the Dirac operator  $\mathcal{D}$  to be discussed below, and  $m$  represents a scalar mass parameter not depending on  $x$ . This mass parameter sets the mass of the quarks in the QCD theory.

In the continuum theory the Dirac operator  $\mathcal{D}$  can be written as

$$\mathcal{D} = \sum_{\mu=0}^3 \gamma_{\mu} \otimes (\partial_{\mu} + A_{\mu}) ,$$

where  $\partial_{\mu} = \partial/\partial x_{\mu}$  and  $A_{\mu}(x)$  is the gauge field. The anti-hermitian traceless matrices  $A_{\mu}(x)$  are elements of  $\mathfrak{su}(3)$ , the Lie algebra of the special unitary group  $SU(3)$ .

The quark fields  $\psi$  and  $\eta$  in (2.1) carry two indices that are suppressed, i.e.,  $\psi = \psi_{c\sigma}$ . These indices label internal degrees of freedom of the quarks; one is called color ( $c = 1, 2, 3$ ) and the other spin ( $\sigma = 0, 1, 2, 3$ ). At each point  $x$  in space-time, we can represent the spinor  $\psi(x)$ , i.e., the quark field  $\psi$  at a given point  $x$ , as a twelve component column vector

$$\psi(x) = (\psi_{10}(x), \psi_{20}(x), \psi_{30}(x), \psi_{11}(x), \dots, \psi_{33}(x))^T . \quad (2.2)$$

In case operations act unambiguously on the color but differently on the spin degrees of freedom we use the notation  $\psi_{\sigma}$  to denote those components of the quark field belonging to the fixed spin index  $\sigma$ . For a given point  $x$ ,  $\psi_{\sigma}(x)$  is thus represented by a three component column vector  $\psi_{\sigma}(x) = (\psi_{1\sigma}(x), \psi_{2\sigma}(x), \psi_{3\sigma}(x))^T$ . The value of the gauge field  $A_{\mu}$  at point  $x$  is in the matrix representation of  $\mathfrak{su}(3)$  and acts non-trivially on the color and trivially on the spin degrees of freedom, i.e.,  $(A_{\mu}\psi)(x) = (I_4 \otimes A_{\mu}(x))\psi(x)$ .

The  $\gamma$ -matrices  $\gamma_0, \gamma_1, \gamma_2, \gamma_3 \in \mathbb{C}^{4 \times 4}$  act non-trivially on the spin and trivially on the color degrees of freedom, i.e.  $(\gamma_{\mu}\psi)(x) = (\gamma_{\mu} \otimes I_3)\psi(x)$ . They are hermitian and

---

<sup>1</sup>Physical space-time is a four dimensional Minkowski space. We present the theory in Euclidean space-time since this version can be treated numerically. The two versions are equivalent, cf. [42].

unitary matrices which generate a Clifford algebra, satisfying

$$\gamma_\mu \gamma_\nu + \gamma_\nu \gamma_\mu = \begin{cases} 2 \cdot I_4 & \mu = \nu \\ 0 & \mu \neq \nu \end{cases} \quad \text{for } \mu, \nu = 0, 1, 2, 3. \quad (2.3)$$

Unlike the gauge fields  $A_\mu$ , the  $\gamma$ -matrices do not depend on  $x$ .

The covariant derivative  $\partial_\mu + A_\mu$  is a “minimal coupling extension” of the derivative  $\partial_\mu$ , ensuring that  $((\partial_\mu + A_\mu)\psi)(x)$  transforms in the same way as  $\psi(x)$  under local gauge transformations, i.e., a local change of the coordinate system in color space. As part of the covariant derivative the  $A_\mu$ ’s can be seen as connecting different (but infinitesimally close) space-time points. The combination of covariant derivatives and the  $\gamma$ -matrices ensures that  $\mathcal{D}\psi(x)$  transforms under the space-time transformations of special relativity in the same way as  $\psi(x)$ . Local gauge invariance and special relativity are fundamental principles of the standard model of elementary particle physics.

**2.2. Lattice QCD.** In order to compute predictions in QCD from first principles and non-perturbatively, the theory of QCD has to be discretized and simulated on a computer. The discretization error is then accounted for by extrapolation to the continuum limit based on simulations at different lattice spacings. One of the most expensive tasks in these computations is the solution of the discretized Dirac equation for a given right hand side. In this section we give a brief introduction into the principles of this discretization and discuss some properties of the arising linear operators. Since the discretization is typically formulated on an equispaced lattice, this treatment of QCD is also referred to as lattice QCD. For a more detailed introduction to QCD and lattice QCD we refer the interested reader to [19, 29, 42].

Consider a four-dimensional torus  $\mathcal{T}$ . On  $\mathcal{T}$  we have a periodic  $N_t \times N_s^3$  lattice  $\mathcal{L} \subset \mathcal{T}$  with lattice spacing  $a$  and  $n_{\mathcal{L}} = N_t \cdot N_s^3$  lattice points. In here  $N_s$  denotes the number of lattice points for each of the three space dimensions and  $N_t$  the number of lattice points in the time dimension. Hence, for any  $x, y \in \mathcal{L}$  there exists  $p \in \mathbb{Z}^4$  such that

$$y = x + a \cdot p, \quad \text{i.e., } y_\mu = x_\mu + a \cdot p_\mu \text{ for } \mu = 0, 1, 2, 3.$$

For shift operations on the lattice, we define shift vectors  $\hat{\mu} \in \mathbb{R}^4$  by

$$\hat{\mu}_\nu = \begin{cases} a & \mu = \nu \\ 0 & \text{else.} \end{cases}$$

A quark field  $\psi$  is defined at each point of the lattice, i.e., it is a function

$$\begin{aligned} \psi : \mathcal{L} &\rightarrow \mathbb{C}^{12} \\ x &\mapsto \psi(x) \end{aligned}$$

on the lattice  $\mathcal{L}$  which maps a point  $x \in \mathcal{L}$  to a spinor  $\psi(x)$ . As in continuum QCD, this spinor again has color and spin indices  $\psi_{c\sigma}$ ,  $c = 1, 2, 3$ ,  $\sigma = 0, 1, 2, 3$ . For future use we introduce the symbols  $\mathcal{C}$  and  $\mathcal{S}$  for the color and the spin space, i.e.,

$$\mathcal{C} = \{1, 2, 3\}, \quad \mathcal{S} = \{0, 1, 2, 3\}.$$

The gauge fields  $A_\mu(x)$  connecting infinitesimally close space-time points in continuum QCD have to be replaced by objects that connect points at *finite* distances.

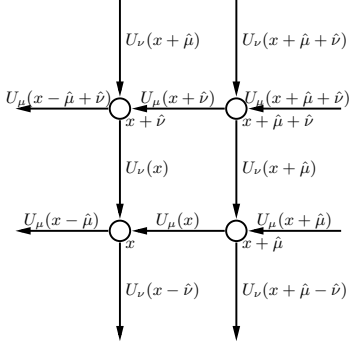


FIG. 2.1. Naming conventions on the lattice.

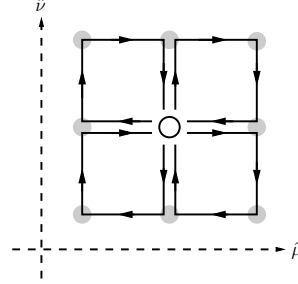


FIG. 2.2. The clover term.

To this purpose variables  $U_\mu(x)$  are introduced.  $U_\mu(x)$  connects  $x$  and  $x + \hat{\mu}$ , so that we regard  $U_\mu(x)$  as being associated with the *link* between  $x$  and  $x + \hat{\mu}$ . The link between  $x + \hat{\mu}$  and  $x$ , pointing in the opposite direction, is then given by  $U_\mu(x)^{-1}$ . The matrices  $U_\mu(x)$  are an approximation to the path-ordered exponential of the integral of  $A_\mu$  along the link. They satisfy

$$U_\mu(x) \in \text{SU}(3), \text{ in particular } U_\mu(x)^{-1} = U_\mu(x)^H.$$

Figure 2.1 illustrates the naming conventions on the lattice.  $U_\mu(x)$  is called a *gauge link* and the set of all gauge links  $\{U_\mu(x) : x \in \mathcal{L}, \mu = 0, 1, 2, 3\}$  is called *configuration*.

The covariant derivative of the continuum theory can be discretized in many ways. Here we restrict ourselves to the widely used Wilson discretization (cf. [57]), noting that the multigrid solver developed in this paper is in principle applicable to any discretization resulting in local couplings. We define forward covariant finite differences

$$(\Delta^\mu \psi_\sigma)(x) = \frac{1}{a} (U_\mu(x) \psi_\sigma(x + \hat{\mu}) - \psi_\sigma(x)) \doteq (\partial_\mu + A_\mu) \psi_\sigma(x)$$

and backward covariant finite differences

$$(\Delta_\mu \psi_\sigma)(x) = \frac{1}{a} (\psi_\sigma(x) - U_\mu^H(x - \hat{\mu}) \psi_\sigma(x - \hat{\mu})) .$$

Since  $(\Delta^\mu)^H = -\Delta_\mu$ , the centralized covariant finite differences  $(\Delta^\mu + \Delta_\mu)/2$  are anti-hermitian. The simplest discretization of the Dirac operator  $\mathcal{D}$  is then given by

$$D_N = \sum_{\mu=0}^3 \gamma_\mu \otimes (\Delta_\mu + \Delta^\mu) / 2.$$

This naive discretization generates unphysical eigenvectors, a standard phenomenon when discretizing first order derivatives using central finite differences, cf. [52], also known as the “species doubling effect” or “red-black instability”. The eigenspace for each eigenvalue of  $D_N$  has dimension 16, but only a one-dimensional subspace corresponds to an eigenfunction of the continuum operator. Wilson introduced the stabilization term  $a\Delta_\mu\Delta^\mu$ , a centralized second order covariant finite difference, to avoid this problem. The Wilson discretization of the Dirac operator is thus given by

$$D_W = \frac{m_0}{a} I + \frac{1}{2} \sum_{\mu=0}^3 \left( \gamma_\mu \otimes (\Delta_\mu + \Delta^\mu) - a I_4 \otimes \Delta_\mu \Delta^\mu \right), \quad (2.4)$$

where the mass parameter  $m_0$  sets the quark mass (for further details, see [42]).

The commutativity relations (2.3) of the  $\gamma$ -matrices imply a non-trivial symmetry of  $D_W$ . Defining  $\gamma_5 = \gamma_0\gamma_1\gamma_2\gamma_3$  we have  $\gamma_5\gamma_\mu = -\gamma_\mu\gamma_5$  for  $\mu = 0, 1, 2, 3$ , and since  $\gamma_\mu$  and  $\gamma_5$  are hermitian we see that  $\gamma_5\gamma_\mu$  is anti-hermitian. Thus the operator  $(\gamma_5\gamma_\mu) \otimes (\Delta_\mu + \Delta^\mu)$  is hermitian, being the product of two anti-hermitian operators. To describe the resulting  $\Gamma_5$ -symmetry of the Wilson Dirac operator, we define  $\Gamma_5 = I_{n_{\mathcal{L}}} \otimes \gamma_5 \otimes I_3$  and have  $(\Gamma_5 D_W)^H = \Gamma_5 D_W$ .

To reduce the order of the discretization error as a function of  $a$ , the Sheikholeslami-Wohlert or “clover” term (cf. [49] and Figure 2.2), depending on a parameter  $c_{sw}$ , is added to the lattice Wilson Dirac operator

$$D = D_W - \frac{c_{sw}}{32a} \sum_{\mu, \nu=0}^3 (\gamma_\mu \gamma_\nu) \otimes (Q_{\mu\nu} - Q_{\nu\mu}), \quad (2.5)$$

where  $(Q_{\mu\nu}\psi_\sigma)(x) = Q_{\mu\nu}(x)\psi_\sigma(x)$  with

$$\begin{aligned} Q_{\mu\nu}(x) = & U_\mu(x) U_\nu(x + \hat{\mu}) U_\mu(x + \hat{\nu})^H U_\nu(x)^H + \\ & U_\nu(x) U_\mu(x - \hat{\mu} + \hat{\nu})^H U_\nu(x - \hat{\mu})^H U_\mu(x - \hat{\mu}) + \\ & U_\mu(x - \hat{\mu})^H U_\nu(x - \hat{\mu} - \hat{\nu})^H U_\mu(x - \hat{\mu} - \hat{\nu}) U_\nu(x - \hat{\nu}) + \\ & U_\nu(x - \hat{\nu})^H U_\mu(x - \hat{\nu}) U_\nu(x - \hat{\nu} + \hat{\mu}) U_\mu(x)^H. \end{aligned}$$

The clover term is diagonal on the lattice  $\mathcal{L}$ . It removes  $\mathcal{O}(a)$ -discretization effects from the covariant finite difference discretization of the covariant derivative (for appropriately tuned  $c_{sw}$ ; see [49] and references therein). The resulting discretized Dirac operator  $D$  thus has discretization effects of order  $\mathcal{O}(a^2)$ . It is again  $\Gamma_5$ -symmetric, i.e., we have

$$(\Gamma_5 D)^H = \Gamma_5 D. \quad (2.6)$$

The  $\Gamma_5$ -symmetry induces a symmetry on the spectrum of  $D$ :

**LEMMA 2.1.** *Every right eigenvector  $\psi_\lambda$  to an eigenvalue  $\lambda$  of  $D$  corresponds to a left eigenvector  $\hat{\psi}_{\bar{\lambda}} = \Gamma_5 \psi_\lambda$  to the eigenvalue  $\bar{\lambda}$  of  $D$  and vice versa. In particular, the spectrum of  $D$  is symmetric with respect to the real axis.*

*Proof.* Due to  $D^H = \Gamma_5 D \Gamma_5$  we have

$$D\psi_\lambda = \lambda\psi_\lambda \Leftrightarrow \psi_\lambda^H D^H = \bar{\lambda}\psi_\lambda^H \Leftrightarrow (\Gamma_5 \psi_\lambda)^H D = \bar{\lambda}(\Gamma_5 \psi_\lambda)^H. \quad \square$$

Summarizing,  $D \in \mathbb{C}^{n \times n}$  is a sparse matrix which represents a nearest neighbor coupling on a periodic 4D lattice. The lattice has  $n_{\mathcal{L}} = N_t N_s^3$  sites, each holding 12 variables, so that  $n = 12n_{\mathcal{L}}$ .  $D$  has the symmetry property (2.6), depends on a mass parameter  $m_0$ , the Sheikholeslami-Wohlert constant  $c_{sw}$ , and a configuration  $\{U_\mu(x) : x \in \mathcal{L}, \mu = 0, 1, 2, 3\}$ . In practice  $m_0$  is negative, and for physically relevant mass parameters, the spectrum of  $D$  is contained in the right half plane, cf. Fig. 2.3 and Fig. 2.4.

While the continuum Dirac operator is normal, the Wilson Dirac operator is not, but it approaches normality when discretization effects become smaller. For small lattice spacing, large lattice sizes and physically relevant mass parameters we can thus expect that the whole field of values  $\mathcal{F}(D) = \{\psi^H D \psi : \psi^H \psi = 1\}$  of  $D$  is in the right half plane.

To explicitly formulate  $D$  in matrix terms we fix a representation for the  $\gamma$ -matrices as

$$\gamma_0 = \begin{pmatrix} & i \\ -i & \end{pmatrix}, \gamma_1 = \begin{pmatrix} & 1 \\ 1 & \end{pmatrix}, \gamma_2 = \begin{pmatrix} i & \\ -i & i \end{pmatrix}, \gamma_3 = \begin{pmatrix} & 1 \\ 1 & \end{pmatrix},$$

6

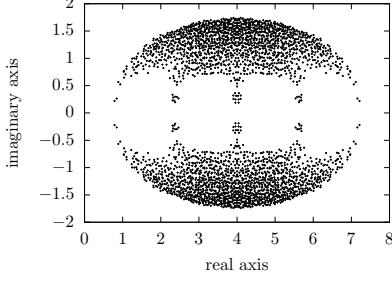


FIG. 2.3. Spectrum of a  $4^4$  Wilson Dirac operator;  $m_0 = 0$ ,  $c_{sw} = 0$ .

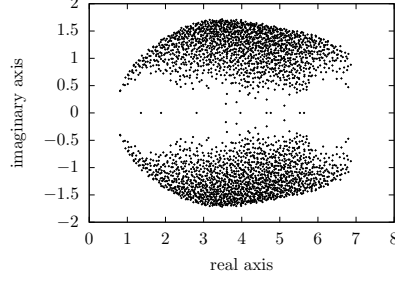


FIG. 2.4. Spectrum of a  $4^4$  "Clover improved" operator;  $m_0 = 0$ ,  $c_{sw} = 1$ .

resulting in

$$\gamma_5 = \text{diag}(1, 1, -1, -1).$$

Thus  $\gamma_5$  acts as the identity on spins 0 and 1 and as the negative identity on spins 2 and 3.  $D$  is then given via

$$\begin{aligned} (D\psi)(x) &= \frac{1}{a} \left( (m_0 + 4)I_{12} - \frac{c_{sw}}{32} \sum_{\mu, \nu=0}^3 (\gamma_\mu \gamma_\nu) \otimes (Q_{\mu\nu}(x) - Q_{\nu\mu}(x)) \right) \psi(x) \\ &\quad - \frac{1}{2a} \sum_{\mu=0}^3 ((I_4 - \gamma_\mu) \otimes U_\mu(x)) \psi(x + \hat{\mu}) \\ &\quad - \frac{1}{2a} \sum_{\mu=0}^3 ((I_4 + \gamma_\mu) \otimes U_\mu^H(x - \hat{\mu})) \psi(x - \hat{\mu}). \end{aligned}$$

**2.3. Domain Decomposition in Lattice QCD.** For ease of notation we from now on drop the lattice spacing  $a$ , so that the lattice  $\mathcal{L}$  is given as

$$\mathcal{L} = \{x = (x_0, x_1, x_2, x_3), 1 \leq x_0 \leq N_t, 1 \leq x_1, x_2, x_3 \leq N_s\}.$$

Let us also reserve the terminology *block decomposition* for a tensor type decomposition of  $\mathcal{L}$  into lattice-blocks. The precise definition is as follows.

DEFINITION 2.2. Assume that  $\{\mathcal{T}_1^0, \dots, \mathcal{T}_{\ell_0}^0\}$  is a partitioning of  $\{1, \dots, N_t\}$  into  $\ell_0$  blocks of consecutive time points,

$$\mathcal{T}_j^0 = \{t_{j-1} + 1, \dots, t_j\}, \quad j = 1, \dots, \ell_0, \quad 0 = t_0 < t_1 < \dots < t_{\ell_0} = N_t,$$

and similarly for the spatial dimensions with partitionings  $\{\mathcal{T}_1^\mu, \dots, \mathcal{T}_{\ell_\mu}^\mu\}$ ,  $\mu = 1, 2, 3$ .

A block decomposition of  $\mathcal{L}$  is a partitioning of  $\mathcal{L}$  into  $\ell = \ell_0 \ell_1 \ell_2 \ell_3$  lattice-blocks  $\mathcal{L}_i$ , where each lattice-block is of the form

$$\mathcal{L}_i = \mathcal{T}_{j_0(i)}^0 \times \mathcal{T}_{j_1(i)}^1 \times \mathcal{T}_{j_2(i)}^2 \times \mathcal{T}_{j_3(i)}^3.$$

Accordingly we define a block decomposition of all  $12n_{\mathcal{L}}$  variables in  $\mathcal{V} = \mathcal{L} \times \mathcal{C} \times \mathcal{S}$  into  $\ell$  blocks  $\mathcal{V}_i$  by grouping all spin and color components from the lattice-block  $\mathcal{L}_i$ ,

$$\mathcal{V}_i = \mathcal{L}_i \times \mathcal{C} \times \mathcal{S}. \quad (2.7)$$

Since the systems arising in lattice QCD calculations tend to have hundreds of millions of unknowns they require the use of parallel computers. For this reason and due to the fact that, as a rule, naive domain decomposition is already used to parallelize the matrix vector product  $Dz$  which is needed for Krylov subspace methods, it is natural to also use a domain decomposition approach as a preconditioner.

The method of choice here is a colored version of the multiplicative Schwarz method [48, 51]. Since the discretized Dirac operator has only nearest-neighbor couplings, only two colors are needed. For a block decomposition of the lattice and variable blocks  $\mathcal{V}_i$  according to (2.7), let the corresponding trivial embeddings, block systems and block solvers be denoted by

$$\mathcal{I}_{\mathcal{V}_i} : \mathcal{V}_i \rightarrow \mathcal{V}, \quad D_i = \mathcal{I}_{\mathcal{V}_i}^T D \mathcal{I}_{\mathcal{V}_i} \quad \text{and} \quad B_i = \mathcal{I}_{\mathcal{V}_i} D_i^{-1} \mathcal{I}_{\mathcal{V}_i}^T.$$

For red-black multiplicative Schwarz the lattice blocks are divided into two groups (red and black) such that no equation in  $D$  couples variables from different blocks of the same color. Given the residual  $r = b - Dx$ , the solutions  $e_i$  of the local systems

$$D_i e_i = \mathcal{I}_{\mathcal{V}_i}^T r, \tag{2.8}$$

yield the corrections for the iterate  $x$ . More precisely, with the shorthand  $B_{color} = \sum_{i \in color} B_i$  and

$$K = B_{black} (I - D B_{red}) + B_{red}$$

we can summarize one iteration ( $\nu = 1$ ) of red-black multiplicative Schwarz as (cf. [51])

$$z \leftarrow (I - KD)z + Kb. \tag{2.9}$$

Since the solution  $z^* = D^{-1}b$  satisfies  $z^* = (I - KD)z^* + Kb$ , the update for the error is  $e \leftarrow (I - KD)e$ , with  $I - KD$  the error propagation operator,

$$E_{SAP} = I - KD = (I - B_{black} D)(I - B_{red} D).$$

The red-black Schwarz method has been introduced to lattice QCD in [40] and has been used ever since in several lattice QCD implementations as a preconditioner (cf. [2, 28, 41]). In this context red-black Schwarz is also known as Schwarz Alternating Procedure (SAP). In what follows the application of  $\nu$  iterations of SAP to a vector  $b$  with initial guess  $z = 0$  is denoted by the linear operator

$$M_{SAP}^{(\nu)} b = \sum_{k=0}^{\nu-1} (I - KD)^k K b.$$

This representation follows by repeated application of (2.9). Note that  $E_{SAP} = I - M_{SAP} D$  with  $M_{SAP} = M_{SAP}^{(1)}$ .

Typically the solution of the local systems (2.8), required when computing  $B_i r$ , is approximated by a few iterations of a Krylov subspace method (e.g., GMRES). When incorporating such an approximate solver, the SAP method becomes a non-stationary iterative process. Hence it is necessary to use flexible Krylov subspace methods like FGMRES or GCR in case that SAP is used as a preconditioner (cf. [28, 40, 47]).

It turns out that SAP as a preconditioner is not able to remedy the unfavorable scaling behavior of Krylov subspace methods with respect to system size, quark mass

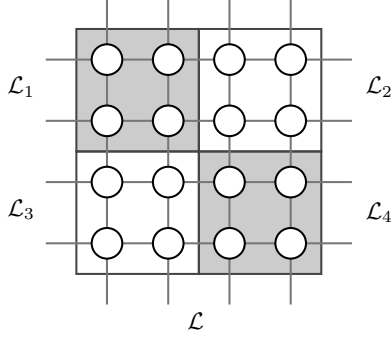


FIG. 2.5. Block decomposed lattice (reduced to 2D) with 2 colors

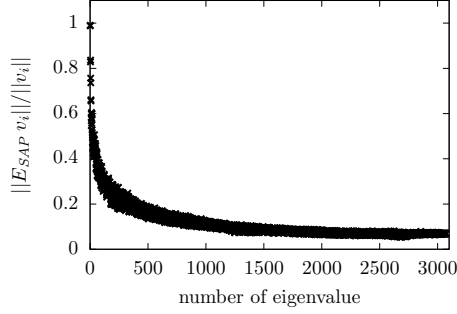


FIG. 2.6. Error component reduction on a  $4^4$  lattice with block size  $2^4$

and physical volume. When analyzing this behavior, one realizes that SAP reduces error components belonging to a large part of the spectrum very well, but a small part is almost not affected by SAP. We illustrate this in Figure 2.6 where the horizontal axis represents the eigenvectors  $v$  of  $D$  in ascending order of the absolute value of the corresponding eigenvalue; the vertical axis gives the ratio  $\|E_{SAP}v\|/\|v\|$ . The ratio is small for larger eigenvalues and becomes significantly larger for the small eigenvalues. This behavior is typical for a smoother in an algebraic multigrid method which motivated us to use SAP in this context.

**3. Algebraic Multigrid Methods.** Any multigrid method consists of two components, a smoother and a coarse grid correction [15, 32, 46, 54]. Typically, the smoother can be chosen as a simple iterative method. This can be a relaxation scheme like Jacobi or Gauss-Seidel or their block variants as well as Krylov subspace methods. Given the properties of SAP presented in the previous section we choose SAP as our smoothing scheme in the QCD context.

Let us reserve the term *near kernel* for the space spanned by the eigenvectors belonging to small (in modulus) eigenvalues of  $D$ . Since SAP is not able to sufficiently remove error components belonging to the near kernel (cf. Figure 2.6), the multigrid method treats these persistent error components separately in a smaller subspace with  $n_c$  variables. Thus, this subspace should approximate the near kernel. The typical algebraic multigrid setup is then as follows: We have to find an operator  $D_c$  which resembles  $D$  on that subspace both in the sense that it acts on the near kernel in a similar manner as  $D$  does, but also in terms of the connection structure and sparsity. The latter allows to work on  $D_c$  recursively using the same approach, thus going from two-grid to true multigrid. We also need a linear map  $R : \mathbb{C}^n \rightarrow \mathbb{C}^{n_c}$  to restrict information from the original space to the subspace and a linear map  $P : \mathbb{C}^{n_c} \rightarrow \mathbb{C}^n$  which transports information back to the original space. The coarse grid correction for a current iterate  $z$  on the original space is then obtained by restricting the residual  $r = b - Dz$  to the subspace, there solving

$$D_c e_c = Rr \quad (3.1)$$

and transporting the coarse error  $e_c$  back to the original space as a correction for  $z$ , resulting in the subspace correction

$$z \leftarrow z + PD_c^{-1}Rr, \quad r = b - Dz \quad (3.2)$$

with the corresponding error propagator

$$I - PD_c^{-1}RD.$$

Typically, the coarse grid system is obtained as the Petrov-Galerkin projection with respect to  $P$  and  $R$ , i.e.,

$$D_c = RDP.$$

The coarse grid correction  $I - P(RDP)^{-1}RD$  then is a projection onto  $\text{range}(RD)^\perp$  along  $\text{range}(P)$ . The action of the coarse grid correction is thus complementary to that of the smoother if  $\text{range}(P)$  approximately contains the near kernel and  $\text{range}(RD)^\perp$  approximately contains the remaining complementary eigenvectors (which are then efficiently reduced by the smoother). The latter condition is satisfied if  $\text{range}(R)$  approximately contains the *left* eigenvectors corresponding to the small eigenvalues. This can be seen by looking at exact eigenvectors: Since left and right eigenvectors are mutually orthogonal, if  $\text{range}(R) = \text{range}(RD)$  is spanned by left eigenvectors of  $D$ , then  $\text{range}(R)^\perp$  is spanned by the complementary right eigenvectors of  $D$ .

Once  $D_c$  is found a basic two-level algorithm consists of alternating the application of the smoother and the coarse grid correction. This procedure can be recursively extended to true multigrid by formulating a two-level algorithm of this kind for the solution of (3.1) until we obtain an operator which is small enough to solve (3.1) directly.

To be computationally beneficial, solving (3.1) has to be much cheaper than solving the original equation  $Dz = b$ . For this purpose  $D_c$  has to be very small or sparse. As the number of eigenvectors that are not sufficiently reduced by the SAP smoother grows with  $n$ , cf. [7], one should not aim at fixing  $n_c$  (like in deflation methods), but at finding sparse matrices  $R$  and  $P$  whose ranges approximate the left and right near kernel of  $D$  well, respectively.

**3.1. Aggregation-based Intergrid Transfer Operators.** Consider a block decomposition of the lattice  $\mathcal{L}$  with lattice-blocks  $\mathcal{L}_i$ . It has been observed in [41] that eigenvectors belonging to small eigenvalues of  $D$  tend to (approximately) coincide on a large number of lattice-blocks  $\mathcal{L}_i$ , a phenomenon which was termed “local coherence”. Local coherence means in particular that we can represent many eigenvectors with small eigenvalues from just a few by decomposing them into the parts belonging to each of the lattice-blocks. We refer to [41] for a detailed qualitative analysis of this observation. Local coherence is the philosophy behind the aggregation-based intergrid transfer operators introduced in a general setting in [11, 15] and applied to QCD problems in [4, 14, 44].

**DEFINITION 3.1.** *An aggregation  $\{\mathcal{A}_1, \dots, \mathcal{A}_s\}$  is a partitioning of the set  $\mathcal{V} = \mathcal{L} \times \mathcal{C} \times \mathcal{S}$  of all variables. It is termed a lattice-block based aggregation if each  $\mathcal{A}_i$  is of the form*

$$\mathcal{A}_i = \mathcal{L}_{j(i)} \times \mathcal{W}_i,$$

where  $\mathcal{L}_j$  are the lattice-blocks of a block decomposition of  $\mathcal{L}$  and  $\mathcal{W}_i \subseteq \mathcal{C} \times \mathcal{S}$ .

Aggregates for the lattice Wilson Dirac operator (2.5) will typically be realized as lattice-block based aggregates. Note that, however, the SAP smoother on the one hand and interpolation and restriction on the other hand do not have to be based on a *common* block decomposition of  $\mathcal{L}$ .

Starting from a set of *test vectors*  $\{v_1, \dots, v_N\}$  which represent the near kernel and a set of aggregates  $\{\mathcal{A}_1, \dots, \mathcal{A}_s\}$ , the interpolation  $P$  is obtained by decomposing the test vectors over the aggregates

$$(v_1 \mid \dots \mid v_N) = \begin{pmatrix} \vdots \\ \vdots \\ \vdots \end{pmatrix} \longrightarrow P = \begin{pmatrix} \boxed{\phantom{0}} & & & \\ & \boxed{\phantom{0}} & & \\ & & \ddots & \\ & & & \boxed{\phantom{0}} \end{pmatrix} \begin{matrix} \mathcal{A}_1 \\ \mathcal{A}_2 \\ \vdots \\ \mathcal{A}_s \end{matrix}. \quad (3.3)$$

Formally, each aggregate  $\mathcal{A}_i$  induces  $N$  variables  $(i-1)N+1, \dots, iN$  on the coarse system, and we define

$$Pe_{(i-1)N+j} = \mathcal{I}_{\mathcal{A}_i}^T v_j, \quad i = 1, \dots, s, j = 1, \dots, N. \quad (3.4)$$

Herein,  $\mathcal{I}_{\mathcal{A}_i}$  represents the trivial restriction operator for the aggregate  $\mathcal{A}_i$ , i.e.,  $\mathcal{I}_{\mathcal{A}_i}^T v_j$  leaves the components of  $v_j$  from  $\mathcal{A}_i$  unchanged while zeroing all others, and  $e_{(i-1)N+j}$  denotes the  $(i-1)N+j$ -th unit vector. For the sake of stability, the test vectors are orthonormalized locally, i.e., for each  $i$  we replace  $\mathcal{I}_{\mathcal{A}_i}^T v_j$  in (3.4) by the  $j$ -th basis vector of an orthonormal basis of  $\text{span}(\mathcal{I}_{\mathcal{A}_i}^T v_1, \dots, \mathcal{I}_{\mathcal{A}_i}^T v_N)$ . This does not alter the range of  $P$  nor does it change the coarse grid correction operator  $I - P(RDP)^{-1}RD$ , and it ensures  $P^H P = I$ .

The restriction  $R$  is obtained in an analogous manner by using a set of test vectors  $\{\hat{v}_1, \dots, \hat{v}_N\}$  and the same aggregates to build  $R^H$ . Figure 3.1 illustrates a lattice-block based aggregation from a lattice point of view—again reduced to two dimensions—where in each aggregate  $\mathcal{A}$  we take  $\mathcal{W}_i$  as the whole set  $\mathcal{C} \times \mathcal{S}$ . Then the aggregates can be viewed as forming a new, coarse lattice and the sparsity and connection structure of  $D_c = RDP$  resembles the one of  $D$ , i.e., we have again a nearest neighbor coupling. Each lattice point of the coarse grid, i.e., each aggregate, holds  $N$  variables.

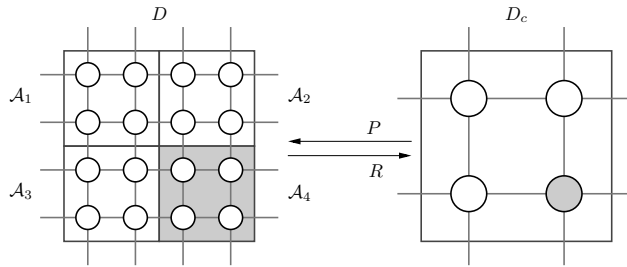


FIG. 3.1. Aggregation-based interpolation (geometrical point of view reduced to 2D)

**3.2. Petrov-Galerkin Approach in Lattice QCD.** The structure and the spectral properties of the Wilson Dirac operator  $D$  suggest to explicitly tie the restriction  $R$  to the interpolation  $P$ . The following construction of  $P$ —and thus  $R$ —is similar to constructions found in [4, 14, 41, 44] in the sense that the structure of all these interpolation operators is similar while the test vectors  $v_i$  upon which the interpolation is built—and therefore the action of the operators—are different.

Due to Lemma 2.1 it is natural to choose

$$R = (\Gamma_5 P)^H$$

in the aggregation based intergrid operators: if  $P$  is built from vectors  $v_1, \dots, v_N$  which approximate right eigenvectors with small eigenvalues of  $D$ , then  $R = (\Gamma_5 P)^H$  is built from vectors  $\hat{v}_i = \Gamma_5 v_i$  which approximate left eigenvectors with small eigenvalues.

As was pointed out in [4], it is furthermore possible to even obtain  $R = P^H$  by taking the special spin-structure of the Dirac operator into account when defining the aggregates. To be specific, we introduce the following definition.

**DEFINITION 3.2.** *The aggregation  $\{\mathcal{A}_i, i = 1, \dots, s\}$  is termed  $\Gamma_5$ -compatible if any given aggregate  $\mathcal{A}_i$  is composed exclusively of fine variables with spin 0 and 1 or of fine variables with spin 2 and 3.*

Assume that we have a  $\Gamma_5$ -compatible aggregation and consider the interpolation operator  $P$  from (3.3). Since  $\Gamma_5$  acts as the identity on spins 0 and 1 and as the negative identity on spins 2 and 3, when going from  $P$  to  $\Gamma_5 P$  each of the non-zero blocks in  $P$  belonging to a specific aggregate is either multiplied by  $+1$  or by  $-1$ . This gives

$$\Gamma_5 P = P \Gamma_5^c. \quad (3.5)$$

with  $\Gamma_5^c$  acting as the identity on the spin-0-1-aggregates and as the negative identity on the spin-2-3-aggregates.

**LEMMA 3.3.** *Let the aggregation be  $\Gamma_5$ -compatible and  $P$  the corresponding aggregation based prolongation as in (3.3) and  $R = (\Gamma_5 P)^H$ . Consider the two coarse grid operators*

$$D_c^{PG} = RDP, \quad \text{and} \quad D_c = P^H DP.$$

Then

- (i)  $D_c = \Gamma_5^c D_c^{PG}$ .
- (ii)  $I - P D_c^{-1} P^H D = I - P (D_c^{PG})^{-1} R D$ .
- (iii)  $D_c^{PG}$  is hermitian,  $D_c$  is  $\Gamma_5^c$ -symmetric.
- (iv) For the field of values  $\mathcal{F}(D) = \{\psi^H D \psi : \psi^H \psi = 1\}$ , we have  $\mathcal{F}(D_c) \subseteq \mathcal{F}(D)$ .

*Proof.* We first observe that just as  $\Gamma_5$  the matrix  $\Gamma_5^c$  is diagonal with diagonal entries  $+1$  or  $-1$ , so  $\Gamma_5^c = (\Gamma_5^c)^H = (\Gamma_5^c)^{-1}$ . Part (i) now follows from

$$D_c^{PG} = RDP = (\Gamma_5 P)^H DP = (P \Gamma_5^c)^H DP = \Gamma_5^c P^H DP = \Gamma_5^c D_c.$$

Consequently,

$$P (D_c^{PG})^{-1} R D = P D_c^{-1} \Gamma_5^c P^H \Gamma_5 D = P D_c^{-1} \Gamma_5^c \Gamma_5^c P^H D = P D_c^{-1} P^H D,$$

which gives (ii). For part (iii) we observe that

$$(D_c^{PG})^H = P^H D^H R^H = P^H D^H \Gamma_5 P = P^H \Gamma_5 D P = RDP = D_c^{PG}.$$

This shows that  $D_c^{PG}$  is hermitian, which is equivalent to  $D_c = \Gamma_5^c D_c^{PG}$  being  $\Gamma_5^c$ -symmetric. Finally, since  $P^H P = I$ , we have

$$\begin{aligned} \mathcal{F}(D_c) &= \{\psi_c^H D_c \psi_c : \psi_c^H \psi_c = 1\} = \{(P \psi_c)^H D (P \psi_c) : (P \psi_c)^H (P \psi_c) = 1\} \\ &\subseteq \{\psi^H D \psi : \psi^H \psi = 1\} = \mathcal{F}(D), \end{aligned}$$

which gives (iv).  $\square$

Lemma 3.3 has some remarkable consequences. Part (ii) shows that we end up with the same coarse grid correction, irrespectively of whether we pursue a Petrov-Galerkin approach (matrix  $D_c^{PG}$  with  $R = \Gamma_5 P$ ) or a Galerkin approach (matrix  $D_c$ , restriction is the adjoint of the prolongation). The Petrov-Galerkin matrix  $D_c^{PG}$  inherits the hermiticity of the matrix  $\Gamma_5 D$ , whereas the Galerkin matrix  $D_c$  inherits the  $\Gamma_5$ -symmetry (and thus the symmetry of the spectrum, see Lemma 2.1) of  $D$ . Moreover, if  $\mathcal{F}(D)$  lies in the right half plane, then so does  $\mathcal{F}(D_c)$  and thus the spectrum of  $D_c$ . It is known that the “symmetrized” Wilson Dirac operator  $\Gamma_5 D$  is close to maximally indefinite [30], i.e., the number of negative eigenvalues is about the same as the positive ones. This property is also inherited by  $\Gamma_5^c D_c = D_c^{PG}$ .

$\Gamma_5$ -symmetry implies an interesting connection between the eigensystem of  $\Gamma_5 D$  and the singular values and vectors of  $D$ . Indeed, if

$$\Gamma_5 D = V \Lambda V^H, \quad \Lambda \text{ diagonal}, \quad V^H V = I$$

denotes the eigendecomposition of the hermitian matrix  $\Gamma_5 D$ , then

$$D = (\Gamma_5 V \text{sign}(\Lambda)) |\Lambda| V^H = U \Sigma V^H \quad (3.6)$$

is the singular value decomposition of  $D$  with the unitary matrix  $U = \Gamma_5 V \text{sign}(\Lambda)$  and  $\Sigma = |\Lambda|$ .

The theory of algebraic multigrid methods for non-hermitian problems recently developed in [16] suggests to base interpolation and restriction on the right and left singular vectors corresponding to small singular values rather than on eigenvectors, so we could in principle use the relation (3.6). However, obtaining good approximations for the singular vectors belonging to small singular values is now much harder than obtaining good approximations to eigenvectors belonging to small eigenvalues, since the small singular values lie right in the middle of the spectrum of  $\Gamma_5 D$ , whereas the small eigenvalues of  $D$  lie at the “border” of its spectrum (and in the right half plane  $\mathbb{C}^+$  if  $\mathcal{F}(D) \subset \mathbb{C}^+$ ). Numerically we did not find that going after the singular values payed off with respect to the solver performance and it significantly increased the setup timing. These observations led us to the eigenvector based adaptive multigrid approach presented here; it also motivates that we consider  $D_c$  rather than  $D_c^{PG}$  as the “correct” coarse grid system to work with recursively in a true multigrid method.

In our computations, we take special  $\Gamma_5$ -compatible, lattice-block based aggregations.

**DEFINITION 3.4.** *Let  $\mathcal{L}_j, j = 1, \dots, s_L$  be a block decomposition of the lattice  $\mathcal{L}$ . Then the standard aggregation  $\{\mathcal{A}_{j,\sigma}, j = 1, \dots, s_L, \sigma = 0, 1\}$  is given by*

$$\mathcal{A}_{j,0} = \mathcal{L}_j \times \{0, 1\} \times \mathcal{C}, \quad \mathcal{A}_{j,1} = \mathcal{L}_j \times \{2, 3\} \times \mathcal{C}.$$

Aggregates of the standard aggregation always combine two spin degrees of freedom in a  $\Gamma_5$ -compatible manner and all three color degrees of freedom. For any given  $j$ , the two aggregates  $\mathcal{A}_{j,0}$  and  $\mathcal{A}_{j,1}$  are the two only aggregates associated with the lattice-block  $\mathcal{L}_j$ . The standard aggregates thus induce a coarse lattice  $\mathcal{L}_c$  with  $n_{\mathcal{L}_c}$  sites where each coarse lattice site corresponds to one lattice-block  $\mathcal{L}_j$  and holds  $2N$  variables with  $N$  the number of test vectors.  $N$  variables correspond to spin 0 and 1 (and aggregate  $\mathcal{A}_{j,0}$ ); another  $N$  variables to spin 2 and 3 (and aggregate  $\mathcal{A}_{j,1}$ ). Thus the overall system size of the coarse system is  $n_c = 2Nn_{\mathcal{L}_c}$ .

With standard aggregation, in addition to the properties listed in Lemma 3.3, the coarse system  $D_c = P^H D P$  also preserves the property that coarse lattice points can be arranged as a 4D periodic lattice such that the system represents a nearest neighbor coupling on this torus. Each coarse lattice point now carries  $2N$  variables.

We also note that applying  $R$  and  $P$  to a vector does not require any communication in a parallel implementation if whole aggregates are assigned to one process.

**3.3. Adaptivity in Aggregation-based AMG.** If no *a priori* information about the near kernel is available, the test vectors  $v_1, \dots, v_N$  to be used in an aggregation based multigrid method have to be obtained computationally during a *setup phase*. We now briefly review the setup concept of adaptive (smoothed) aggregation as described in [15]. We do so in the Galerkin context, i.e., we take  $R = P^H$ . The first fundamental idea of adaptivity in algebraic multigrid methods is to use the smoother to find error components not effectively reduced by the smoother, i.e., belonging to the near kernel. Starting with an initial random vector  $u$ , some iterations with the smoothing scheme on the homogeneous equations  $Du = 0$  yield a vector  $\tilde{v}$  rich in components that are not effectively reduced. The first set of test vectors then is the singleton  $\{v\}$ , and one constructs the corresponding aggregation based interpolation  $P$  from (3.3). This construction guarantees that  $v$  is in  $\text{range}(P)$  and thus is treated on the coarse grid. Once a first two- or multigrid method is constructed in this way, one can use it to generate an additional vector not effectively reduced by the current method by again iterating on the homogeneous system. This newly found vector is added to the set of test vectors upon which we build new interpolation and coarse grid operators. Continuing in this manner we ultimately end up with a multigrid method which converges rapidly, but possibly at a high computational cost for the setup if many vectors need to be generated and incorporated in the interpolation operator. To remedy this issue, already in [15], some sophisticated ideas to filter the best information out of the produced vectors, are proposed which have been partly used in the implementations of adaptive algebraic multigrid for QCD described in [4, 14, 44].

**3.4. Adaptivity in Bootstrap AMG.** It is possible to use the current multigrid method in an adaptive setup in more ways than just to test it for deficiencies by applying it to the homogeneous equation  $Du = 0$ . This is done in the *bootstrap* approach pursued in [12, 13] which we sketch now. Details will be discussed in connection with the inexact deflation method in sections 4.3 and 5.

The following observation is crucial: Given an eigenpair  $(v_c, \lambda_c)$  of the generalized eigenvalue problem on the coarse grid

$$D_c v_c = \lambda_c P^H P v_c,$$

we observe that  $(P v_c, \lambda_c)$  solves the constrained eigenvalue problem

$$\text{find } (v, \lambda) \text{ with } v \in \text{range}(P) \text{ s.t. } P^H (Dv - \lambda v) = 0$$

on the fine grid. This observation allows to use the coarse grid system as a source of information about the eigenvectors with small eigenvalues of the fine grid system. Computing eigenvectors with small  $\lambda_c$  on the coarse grid is cheaper than on the fine grid, and applying a few iterations of the smoother to the lifted vectors  $P v_c$  yields useful test vectors rich in components belonging to the near kernel of the fine grid system. As we will see, the setup process used in the “inexact deflation” approach from [41], explained in the next section, can also be interpreted as a bootstrap-type setup, where instead of using an exact solution to the coarse grid eigenproblem only approximations are calculated.

**4. Multigrid and Inexact Deflation.** A hierarchical approach for solving the Wilson Dirac equation (1.1), which lately received attention in the lattice QCD community, was proposed in [41]. It is a combination of what is called “inexact deflation” with an SAP preconditioned generalized conjugate residuals (GCR) method. The paper [41] does not relate its approach to the existing multigrid literature. The purpose of this section is to recast the formulations from [41] into established terminology from algebraic multigrid theory and to explain the limitations of the overall method from [41] which composes its multigrid ingredients in a non-optimal manner. We also explain how the setup employed in [41] to construct the “inexact deflation subspace” (i.e., the test vectors) can be viewed and used as an approximate bootstrap setup in the sense of section 3.4.

**4.1. Inexact Deflation.** The inexact deflation subspace constructed in [41] is the range of a linear operator  $P$  which resembles the definition of aggregation based interpolation from (3.3). As in the aggregation-based construction it uses a set of test vectors  $v_1, \dots, v_N$  which are “chopped” up over aggregates (called subdomains in [41]) to obtain the locally supported columns of  $P$ . These aggregates are not  $\Gamma_5$ -compatible, so the  $\Gamma_5$ -symmetry is not preserved on the coarse grid operator  $D_c$  which is obtained as  $D_c = P^H D P$ . Since the inexact deflation approach is not meant to be recursively extended to a true multilevel method, preserving important properties of the fine system on the coarse system is of lesser concern. However, within its two-level framework a (purely algebraic) deflating technique is applied when solving the coarse system.

Two projections  $\pi_L, \pi_R$  are defined in [41] as follows

$$\pi_L = I - D P D_c^{-1} P^H \quad \text{and} \quad \pi_R = I - P D_c^{-1} P^H D. \quad (4.1)$$

Clearly,  $\pi_R$  is the coarse grid correction introduced in section 3; cf. Lemma 3.3(i). In the context of inexact deflation these projections and the relation  $D\pi_R = \pi_L D$  are used to decompose the linear system of equations  $Dz = b$  as

$$D\pi_R z = \pi_L b, \quad D(I - \pi_R)z = (I - \pi_L)b.$$

The second equation can be simplified to  $(I - \pi_R)z = P D_c^{-1} P^H b$ . Thus the solution  $z$  can be computed as  $z = \pi_R z + (I - \pi_R)z = \chi + \chi'$ , where

$$\chi' = P D_c^{-1} P^H b$$

only requires the solution of the coarse grid system  $D_c$  and

$$D\chi = D\pi_R \chi = \pi_L b$$

is the “inexactly deflated” system which in [41] is solved by a right preconditioned Krylov subspace method. To be specific, the Krylov subspace is built for the operator

$$D\pi_R M_{SAP}^{(\nu)}$$

and the right hand side  $\pi_L b$ , and the Krylov subspace method is GCR (general conjugate residuals, cf. [47]), a minimum residual approach which automatically adapts itself to the fact that the preconditioner  $M_{SAP}^{(\nu)}$  is not stationary, see the discussion in section 2.3.

**4.2. Comparison of Multigrid and Inexact Deflation.** Although the ingredients of an aggregation based algebraic multigrid method as described in section 3 and of “inexact deflation” as described in the previous paragraph are the same, their composition makes the difference. In the multigrid context we combine the SAP smoothing iteration with the coarse grid correction such that it gives rise to the error propagator of a V-cycle with  $\nu$  post smoothing steps

$$E = (I - M_{SAP}^{(\nu)}D)(I - PD_c^{-1}P^H D).$$

Hence we obtain for one iteration of the V-cycle

$$z \leftarrow z + C^{(\nu)}r$$

where  $z$  denotes the current iterate and  $r$  the current residual  $b - Dz$ , and

$$C^{(\nu)} = M_{SAP}^{(\nu)} + PD_c^{-1}P^H - M_{SAP}^{(\nu)}DPD_c^{-1}P^H = M_{SAP}^{(\nu)}\pi_L + PD_c^{-1}P^H, \quad (4.2)$$

with the last equality following from the definition of the projectors in (4.1). Using the multigrid method as a right preconditioner in the context of a Krylov subspace method, the preconditioner is given by  $C^{(\nu)}$ , and the subspace is built for  $DC^{(\nu)}$ . We again should use a flexible Krylov subspace method such as flexible GMRES or GCR, since the smoother  $M_{SAP}$  is non-stationary and, moreover, we will solve the coarse system  $D_c$  only with low accuracy using some “inner iteration” in every step. The important point is that a rough approximation of the coarse grid correction in (4.2), i.e., the solution of systems with the matrix  $D_c$  at only low accuracy, will typically have only a negligible effect on the quality of the preconditioner, and it will certainly not hamper the convergence of the iterates towards the solution of the system since multiplications with the matrix  $D$  are done exactly. On the other hand, in the “inexact deflation” context the exact splitting of the solution  $z = \chi' + \chi$  with

$$\chi' = PD_c^{-1}P^H b, \quad D\pi_R\chi = \pi_L b$$

requires the same final accuracy for both  $\chi'$  and  $\chi$ . Therefore, when computing  $\chi'$ , the coarse grid system has to be solved with high accuracy. More importantly,  $D_c^{-1}$  also appears in  $\pi_R$  which is part of the “deflated” matrix  $D\pi_R$  in the system for  $\chi$ . In the inexact deflation context, this system is solved using SAP as a preconditioner. While we can allow for a flexible and possibly inexact evaluation of the preconditioner, the accuracy with which we evaluate the non-preconditioned matrix  $D\pi_R$  in every step will inevitably affect the accuracy attainable for  $\chi$ . As a consequence, in each iteration we have to solve the system with the matrix  $D_c$  arising in  $\pi_R$  with an accuracy comparable with the accuracy at which we want to obtain  $\chi$  (although the accuracy requirements could, in principle, be somewhat relaxed as the iteration proceeds due to results from [50, 55]).

The difference of the two approaches is now apparent. In the multigrid context we are allowed to solve the coarse system with low accuracy, in inexact deflation we are not. Since the coarse grid system is still a large system, the work to solve it accurately will by far dominate the computational cost in each iteration in inexact deflation. In the multigrid context we can solve at only low accuracy without noticeably affecting the quality of the preconditioner, thus substantially reducing the computational cost of each iteration. Moreover, such a low accuracy solution can be obtained even more efficiently by a recursive application of the two-grid approach, resulting in a true multigrid method. For a more detailed analysis of the connection between deflation methods (including inexact deflation) and multigrid approaches we refer to [33, 45, 53].

---

**Algorithm 4.1:** Inexact deflation setup – IDsetup( $n_{inv}, \nu$ ) as used in [41]

---

```

1 Let  $v_1, \dots, v_N \in \mathbb{C}^n$  be random test vectors
2 for  $\eta = 1$  to 3 do
3   for  $j = 1$  to  $N$  do
4      $v_j \leftarrow M_{SAP}^{(\eta)} v_j$ 
5 for  $\eta = 1$  to  $n_{inv}$  do
6   (re-)construct  $P$  and  $D_c$  from current  $v_1, \dots, v_N$ 
7   for  $j = 1$  to  $N$  do
8      $v_j \leftarrow (M_{SAP}^{(\nu)} \pi_L + P D_c^{-1} P^H) v_j$ 
9      $v_j \leftarrow \frac{v_j}{\|v_j\|}$ 

```

---

**4.3. Adaptivity in the Setup of Inexact Deflation.** To set up the inexact deflation method we need a way to obtain test vectors to construct the inexact deflation operators. Once these vectors are found the method is completely defined (see section 4.1). In analogy to the discussion of adaptive algebraic multigrid in sections 3.3 and 3.4, these test vectors should contain information about the eigenvectors belonging to small eigenvalues of the operator  $DM_{SAP}^{(\nu)}$ , the preconditioned system.

Though the setup proposed in [41] is similar in nature to the one described in section 3.3, it differs in one important way. Instead of working on the homogeneous equation  $D\psi = 0$  with a random initial guess to obtain the test vectors, it starts with a set of random test vectors  $\psi_j$  and approximately computes  $D^{-1}\psi_j$  using SAP. The (approximate) multiplication with  $D^{-1}$  will amplify the components of  $\psi$  belonging to the near kernel. These new vectors are now used to define  $P$  (and  $D_c$ ), yielding an inexact deflation method which can again be used to approximately compute  $D^{-1}\psi_j$  giving new vectors for  $P$ . The whole process is repeated several times; see Algorithm 4.1 for a detailed description where a total of  $n_{inv}$  of these cycles is performed. The update  $v_j \leftarrow (M_{SAP}^{(\nu)} \pi_L + P D_c^{-1} P^H) v_j$  in line 8 of the algorithm is equivalent to the application of the V-cycle iteration matrix  $C^{(\nu)}$  (cf. (4.2)). It can be interpreted as one step of an iteration to solve  $Dv = v_j$  with initial guess 0 and iteration matrix  $C^{(\nu)}$ .

This update of the test vectors can also be viewed in terms of the bootstrap AMG setup outlined in section 3.4. While the first part of the update,  $M_{SAP}^{(\nu)} \pi_L v_j$ , is the application of a coarse grid correction followed by smoothing, i.e., a test to gauge the effectiveness of the method (cf. section 3.3), the second part of the update,  $P D_c^{-1} P^H v_j$  is in  $\text{range}(P)$ . In contrast to the bootstrap methodology where an update in  $\text{range}(P)$  is obtained by interpolating eigenvectors with small eigenvalues of  $D_c$ , in the inexact deflation variant these “optimal” vectors are only approximated.

**5. DD- $\alpha$ AMG.** We now have all the ingredients available to describe our domain decomposition/aggregation based adaptive algebraic multigrid (DD- $\alpha$ AMG) method for the Wilson Dirac operator (1.1).

As its smoother we take  $M_{SAP}^{(\nu)}$ , i.e., we perform  $\nu$  iterations of red-black Schwarz as formulated in (2.9). Like  $\nu$ , the underlying block decomposition of the lattice  $\mathcal{L}$  is a parameter to the method which we will specify in the experiments.

The coarse system  $D_c$  is obtained as  $D_c = P^H D P$ , where  $P$  is an aggregation based prolongation obtained during the adaptive setup phase. The aggregates are

from a standard aggregation according to Definition 3.4, implying that it is in particular lattice-block based and  $\Gamma_5$ -compatible. Parameters of the aggregation are the underlying block decomposition of  $\mathcal{L}$  (which does not necessarily match the one underlying the SAP smoother) and the test vectors  $v_1, \dots, v_N$  upon which  $P$  is built. The coarse grid matrix  $D_c$  inherits all of the important properties of  $D$ , cf. Lemma 3.3.

We combine the smoothing iteration and the coarse grid correction into a standard  $V$ -cycle with no pre- and  $\nu$  steps of post smoothing so that the iteration matrix of one  $V$ -cycle is given by  $C^{(\nu)}$  from (4.2). Instead of using iterations with the  $V$ -cycle as a stand-alone solver, we run FGMRES, the flexible GMRES method (cf. [47]) with one  $V$ -cycle used as a (right) preconditioner.

It remains to specify how we perform the adaptive setup yielding the test vectors  $v_1, \dots, v_N$ . Extensive testing showed that a modification of the inexact deflation setup (Algorithm 4.1) is the most efficient. The modification is a change in the update of the vectors  $v_j$  in the second half. Instead of doing one iteration with  $C^{(\nu)}$  and initial guess 0 to approximately solve  $Dv = v_j$ , we use the currently available vector  $v_j$  as our initial guess, see Algorithm 5.1.

---

**Algorithm 5.1:** DD- $\alpha$ AMG-setup( $n_{inv}, \nu$ )

---

- 1 perform Algorithm 4.1 with line 8 replaced by
  - 2  $v_j \leftarrow v_j + C^{(\nu)}(v_j - Dv_j) \quad \{= C^{(\nu)}v_j + (I - C^{(\nu)}D)v_j\}$
- 

**6. Numerical Results.** We implemented the DD- $\alpha$ AMG method in the programming language C using the parallelization interface of MPI. The numerical tests focus mainly on a two-grid version of our code. As most of the work is spent on the coarse grid, a recursive extension of the two-grid method to more levels is attractive and we thus also show some results of a preliminary version of a true multigrid DD- $\alpha$ AMG implementation.

Our code is optimized to a certain extent, but certainly not to the extreme. As is customary in lattice QCD computations, we use a mixed precision approach where we perform the  $V$ -cycle of the preconditioner in single precision. Low level optimization (e.g., making use of the SSE-registers on Intel/AMD architectures) has not been considered, yet. All Krylov subspace methods (FGMRES, BiCGStab, GCR, CG) have been implemented in a common framework with the same degree of optimization to allow for a standardized comparison of computing times. This is particularly relevant when we compare timings with BiCGStab as well as with the multigrid method introduced in [4, 14, 44]. We also include a comparison with the inexact deflation approach, where an efficient implementation is publicly available.

A commonly used technique in lattice QCD computations is odd-even preconditioning. A lattice site  $x$  is called even if  $x_1 + x_2 + x_3 + x_4$  is even, else it is called odd. Due to the nearest neighbor coupling, the Wilson Dirac operator has the form

$$D = \begin{pmatrix} D_{ee} & D_{eo} \\ D_{oe} & D_{oo} \end{pmatrix},$$

if we order all even sites first. Herein,  $D_{ee}$  and  $D_{oo}$  are block diagonal with  $12 \times 12$  diagonal blocks. Instead of solving a system with  $D$  we can solve the corresponding system for the odd lattice sites given by the Schur complement  $D_S = D_{oo} - D_{oe}D_{ee}^{-1}D_{eo}$  and then retrieve the solution at the even lattice sites, cf. [44]. The inverse  $D_{ee}^{-1}$

is pre-computed once for all, and the operator  $D_S$  is applied in factorized form. A matrix-vector multiplication with  $D_S$  thus requires the same work as one with  $D$  while the condition of  $D_S$  improves over that of  $D$ . Typically, this results in a gain of 2 – 3 in the number of iterations and execution time. Within BiCGStab we use odd-even preconditioning for  $D$ . In all multigrid approaches we use odd-even preconditioned restarted GMRES with a restart length of 30 when we solve the coarse system involving  $D_c$ . We implemented all odd-even preconditioned operators similarly in spirit to what was proposed for the Wilson Dirac operator in [36].

	parameter		default
setup	number of iterations	$n_{inv}$	6
	number of test vectors	$N$	20
	size of lattice-blocks for aggregates		$4^4$
	coarse system relative residual tolerance (stopping criterion for the coarse system) <sup>(*)</sup>		$5 \cdot 10^{-2}$
solver	restart length of FGMRES	$n_{kv}$	25
	relative residual tolerance (stopping criterion)	$tol$	$10^{-10}$
smoother	number of post smoothing steps <sup>(*)</sup>	$\nu$	2
	size of lattice-blocks in SAP <sup>(*)</sup>		$4^4$
	number of minimal residual (MR) iterations to solve the local systems (2.8) in SAP <sup>(*)</sup>		4

TABLE 6.1  
Parameters for the DD- $\alpha$ AMG two-level method. (\*) : same in solver and setup

Table 6.1 summarizes the default parameters used for DD- $\alpha$ AMG in our experiments. Besides those discussed in section 5, the table also gives the stopping criterion used for the solves with the coarse system  $D_c$  (the initial residual is to be decreased by a factor of 20) and the stopping criterion for the entire FGMRES iteration (residual to be decreased by a factor of  $10^{10}$ ). In each SAP iteration we have to (approximately) solve the local systems (2.8). Instead of requiring a certain decrease in the residual we here fix the number of iterative steps (to 4). The iterative method we use here is the odd-even preconditioned minimal residual method MR, i.e., restarted GMRES with a restart length of 1, where each iterative step is particularly cheap.

For the various configurations and respective matrices we found that this default set of parameters yields a well performing solver, with only little room for further tuning. The size of the lattice-blocks ( $4^4$ ) fits well with all lattice sizes occurring in practice, where  $N_t$  and  $N_s$  are multiples of 4. The number of setup iterations,  $n_{inv}$ , is the only one of these parameters which should be tuned. It will depend on how many systems we have to solve, i.e., how many right hand sides we have to treat. When  $n_{inv}$  is increased, the setup becomes more costly, while, at the same time, the solver becomes faster. Thus the time spent in the setup has to be balanced with the number of right hand sides, and we will discuss this in some detail in section 6.2. The default  $n_{inv} = 6$  given in Table 6.1 should be regarded as a good compromise.

The configurations we used are listed in Table 6.2. In principle the pion mass  $m_\pi$  and the lattice spacing (not listed) determine the condition of the respective matrix, e.g., the smaller  $m_\pi$ , the more ill-conditioned the respective matrix is. The physical pion mass is  $m_{\pi_{phys}} = 135$  MeV which is taken on by the configurations 4 and 5. The conditioning of the matrices is indicated by the iteration count of CGNR, the CG method applied to the normal equations  $D^H D\psi = D^H b$  (without odd-even

id	lattice size $N_t \times N_s^3$	pion mass $m_\pi$ [MeV]	CGNR iterations	shift $m_0$	clover term $c_{sw}$	provided by
1	$48 \times 16^3$	250	7,055	-0.095300	1.00000	BMW-c [23, 24]
2	$48 \times 24^3$	250	11,664	-0.095300	1.00000	BMW-c [23, 24]
3	$48 \times 32^3$	250	15,872	-0.095300	1.00000	BMW-c [23, 24]
4	$48 \times 48^3$	135	53,932	-0.099330	1.00000	BMW-c [23, 24]
5	$64 \times 64^3$	135	84,207	-0.052940	1.00000	BMW-c [23, 24]
6	$128 \times 64^3$	270	45,804	-0.342623	1.75150	CLS [18, 26]

TABLE 6.2

*Configurations used together with their parameters. For details about their generation we refer to the references. Pion masses rounded to steps of 5 MeV.*

preconditioning), in which we required the norm of the residual  $r = b - D\psi$  to decrease by a factor of  $10^{10}$ .

We ran DD- $\alpha$ AMG on the various configurations, analyzed the behavior of the setup routine and performed different scaling tests. All results have been computed on the Juropa machine at Jülich Supercomputing Centre, a cluster with 2,208 compute nodes, each with two Intel Xeon X5570 (Nehalem-EP) quad-core processors. Unless stated otherwise the `icc`-compiler with the optimization flags `-O3 -ipo -axSSE4.2 -m64` was used.

**6.1. Comparison with BiCGStab.** First we compare a mixed precision<sup>2</sup>, odd-even preconditioned implementation of BiCGStab with the DD- $\alpha$ AMG method using the standard parameter set for a  $64^4$  configuration at physical pion mass which represents an ill-conditioned linear system with  $n = 201,326,592$ .

	BiCGStab	DD- $\alpha$ AMG	speed-up factor	coarse grid
setup time		22.9s		
solve iter	13,450	21		3,716 <sup>(*)</sup>
solve time	91.2s	3.15s	29.0	2.43s
total time	91.2s	26.1s	3.50	

TABLE 6.3

*BiCGStab vs. DD- $\alpha$ AMG with default parameters (Table 6.1) on configuration 5 (Table 6.2), 8,192 cores, (\*) : coarse grid iterations summed up over all iterations on the fine grid.*

The results reported in Table 6.3 show that we obtain a speed-up factor of 3.5 over BiCGStab with respect to the total timing. Excluding the setup time, we gain a factor of 29. The right most column shows that in this ill-conditioned case about 77% of the solve time of DD- $\alpha$ AMG goes into computations on the coarse grid.

**6.2. Setup Evaluation.** Lattice QCD computations are dominated by two major tasks: generating configurations within the Hybrid Monte-Carlo (HMC) algorithm [35] and evaluating these configurations, i.e., calculating observables. Both tasks require solutions of the lattice Dirac equation.

The HMC generates a sequence of stochastically independent configurations. The configuration is changed in every step, and the Wilson Dirac equation has to be solved

<sup>2</sup>The mixed precision implementation uses double precision flexible GMRES(25) preconditioned by 50 steps of single precision, odd-even preconditioned BiCGStab

only once per configuration. Thus HMC requires a new setup—or at least an update—for the interpolation and coarse grid operator in every step. Therefore the costs of setup/update and solve have to be well-balanced.

The calculation of observables typically requires several solves for a single configuration. Therefore one would be willing to invest more time into the setup in order to obtain a better solver.

number of setup steps $n_{inv}$	average setup timing	average iteration count	lowest iteration count	highest iteration count	average solver timing	average total timing
1	2.08	149	144	154	6.42	8.50
2	3.06	59.5	58	61	3.42	6.48
3	4.69	34.5	33	36	2.37	7.06
4	7.39	27.2	27	28	1.95	9.34
5	10.8	24.1	24	25	1.82	12.6
6	14.1	23.0	23	23	1.89	16.0
8	19.5	22.0	22	22	2.02	21.5
10	24.3	22.5	22	23	2.31	26.6

TABLE 6.4

*Evaluation of DD- $\alpha$ AMG-setup( $n_{inv}, 2$ ) cf. Algorithm 5.1,  $48^4$  lattice, ill-conditioned configuration (Table 6.2: id 4), 2,592 cores, averaged over 20 runs.*

Table 6.4 illustrates how the ratio between setup and solve can be balanced depending on the amount of right hand sides. In this particular case 2 steps in the setup might be the best choice if only a single solution of the system is needed (minimal time for setup + 1 solve). For many right hand sides, where the time spent in the solver dominates, 5 steps in the setup might be the best choice. Doing up to 7 steps can lower the iteration count of the solver even further, but the better the test vectors approximate the near kernel, the more ill-conditioned the coarse system becomes, i.e., lowering the iteration count of the solver means increasing the iteration count on the coarse system.

The numbers shown have been averaged over 20 runs, because the measurements vary due to the choice of random initial test vectors. The fourth and the fifth column of Table 6.4 show that the fluctuations in the iteration count of the solver are modest. For  $n_{inv} \geq 4$  the fluctuations almost vanish completely.

Table 6.5 gives the iteration count of BiCGStab and DD- $\alpha$ AMG for a set of 6 stochastically independent configurations from a single HMC simulation. The BiCGStab iteration count shows a clear dependence on the gauge fields just as DD- $\alpha$ AMG for small values of  $n_{inv}$ . For  $n_{inv} \geq 4$  the iteration count varies only marginally.

**6.3. Scaling Tests.** We now study the scaling behavior of the solver as a function of the mass parameter and the system size. While the former determines the condition number of the Wilson Dirac operator, the latter has an effect on the density of the eigenvalues. In particular, increasing the volume leads to a higher density of small eigenvalues [7]. In a weak parallel scaling test we also analyze the performance as a function of the number of processors used.

**Mass Scaling.** For this study we used a  $48^4$  lattice configuration. We ran the setup once for the mass parameter  $m_0 = -0.09933$  in the Wilson Dirac operator (2.4). This represents the most ill-conditioned system where the pion mass with 135 MeV is physical. We then used the interpolation operator obtained for this system for a

BiCGStab iteration counts						
	conf 1	conf 2	conf 3	conf 4	conf 5	conf 6
	7,950	8,350	9,550	8,600	8,100	9,950
DD- $\alpha$ AMG iteration counts						
$n_{inv}$	conf 1	conf 2	conf 3	conf 4	conf 5	conf 6
1	161	208	175	183	181	272
2	62	75	67	67	64	85
3	34	37	36	37	35	39
4	27	28	28	29	27	29
5	24	25	25	25	24	25
6	23	23	23	24	23	23

TABLE 6.5

Configuration dependence study of BiCGStab and DD- $\alpha$ AMG with DD- $\alpha$ AMG-setup( $n_{inv}, 2$ ) for 6 different, ill-conditioned configurations on  $48^4$  lattices, (Table 6.2: id 4), 2,592 cores.

variety of other mass parameters, where we then ran the DD- $\alpha$ AMG solver without any further setup.

$m_0$	BiCGStab		DD- $\alpha$ AMG		coarse system	
	iteration count	solver timing	iteration count	solver timing	$\emptyset$ iteration count	timing (% solve time)
-0.04933	400	2.60s	17	0.59s	11.2	0.13s (22.0)
-0.06933	600	4.10s	19	0.72s	15.4	0.20s (27.8)
-0.08933	1,550	9.82s	20	0.92s	28.6	0.37s (40.2)
-0.09133	1,700	10.6s	21	1.04s	33.4	0.47s (45.2)
-0.09333	2,250	13.7s	21	1.13s	39.7	0.55s (48.7)
-0.09533	2,850	17.4s	22	1.28s	46.9	0.68s (53.1)
-0.09733	3,750	23.7s	23	1.48s	56.5	0.84s (56.8)
-0.09933	6,250	42.0s	24	1.89s	79.3	1.22s (64.5)

TABLE 6.6

Mass scaling of DD- $\alpha$ AMG for  $n_{inv} = 5$ ,  $48^4$  lattice (Table 6.2: id 4), 2,592 cores.

In Table 6.6 we compare BiCGStab and DD- $\alpha$ AMG with respect to the timing for one right hand side and the scaling with the mass parameter  $m_0$ . For the smallest  $m_0$ , DD- $\alpha$ AMG is 22.2 times faster than BiCGStab and even for the largest value of  $m_0$  there remains a factor of 3.9. We also see that the two methods scale in a completely different manner. The BiCGStab solve for the smallest  $m_0$  is 18.5 times more expensive than the solve for the largest one. On the other hand the DD- $\alpha$ AMG timings just increase by a factor of 3.2, the iteration count even only by a factor of 1.4. The coarse grid iteration count, however, increases by a factor of 8.0.

**System Size Scaling.** In Table 6.7 we report tests on the scaling with the system size for constant mass parameter and (physical) lattice spacing. We again compare DD- $\alpha$ AMG with BiCGStab. The iteration count of BiCGStab for  $N_t \times N_s^3$  lattices appears to scale with  $N_s$  and thus almost doubles from  $N_s = 16$  to  $N_s = 32$  whereas for DD- $\alpha$ AMG we observe an almost constant iteration count and time.

**Weak Scaling.** For a weak scaling test we ran 100 iterations of DD- $\alpha$ AMG with  $n_{inv} = 5$  in the setup on lattices ranging from size  $16^4$  on a single node (8 cores/node)

lattice size $N_t \times N_s^3$	BiCGStab		DD- $\alpha$ AMG		
	iteration count	solver timing	setup timing	iteration count	solver timing
$48 \times 16^3$	1,550	7.03s	6.59s	20	0.89s
$48 \times 24^3$	2,150	10.7s	7.29s	20	0.83s
$48 \times 32^3$	2,600	13.1s	7.15s	21	0.92s

TABLE 6.7

Lattice size scaling of DD- $\alpha$ AMG,  $n_{inv} = 6$  setup iterations, lattices generated with the same mass parameter and lattice spacing (Table 6.2: id 1, 2 and 3), local lattice size  $4 \times 8^3$ .

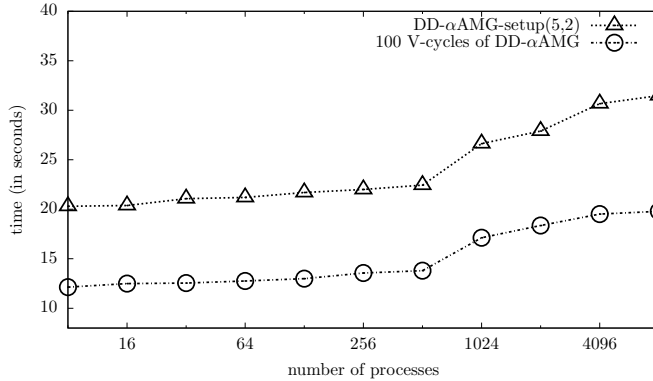


FIG. 6.1. Weak scaling test of DD- $\alpha$ AMG. The lattice size is increased with the number of processes, keeping the local lattice size per process fixed to  $16 \cdot 8^3$ .

to  $128^2 \cdot 64^2$  on 1,024 nodes with  $16 \cdot 8^3$  local lattice size on each core, cf. Figure 6.1.

For the scaling study we fixed the number of iterations on the coarse grid to be exactly 50 steps of odd-even preconditioned GMRES so that we always have the same number of 100 `MPI_Allreduce` operations. In Figure 6.1 we see the usual  $\log(p)$  dependence,  $p$  the number of processes, caused by global communication, together with an exceptional increase when going from 512 to 1,024 processes. Additional measurements show that this is due to the fact that the `MPI_Allreduce` operations take substantially longer for 1,024 processors, a machine-specific feature of Juropa. Apart from this, our method scales well up to 8,192 processes.

**6.4. Comparison with the Inexact Deflation Method.** The inexact deflation code of [41] is publicly available [38]. We now compare its performance with DD- $\alpha$ AMG.<sup>3</sup>

We have chosen the parameters of both methods equally except for the number of post-smoothing steps  $\nu$ . For the inexact deflation method  $\nu = 5$  and for DD- $\alpha$ AMG  $\nu = 2$  turned out to provide the fastest solver, respectively. We used the `gcc` compiler with the `-O3` flag and hand coded low-level `SSE` optimization for the inexact deflation method and the `icc` compiler with the optimization flags `-O3 -ipo -axSSE4.2 -m64` for the DD- $\alpha$ AMG method. These compiler options provide the optimal choices for

<sup>3</sup>Based on the preprint [27] of the present paper, the inexact deflation method has been upgraded in the spirit of DD- $\alpha$ AMG (cf. [39]). The new version is termed “with inaccurate projection”. We here compare with the older, “exact projection” version.

the respective codes. Since our focus is on algorithmic improvements we did not work on customized SSE optimization for DD- $\alpha$ AMG, which should, in principle, give additional speed-up. The following results were produced on the same  $48^4$  lattice as in sections 6.2 and 6.3 and on a  $128 \times 64^3$  lattice (Table 6.2, id 6).

setup steps $n_{inv}$	Inexact deflation			DD- $\alpha$ AMG		
	setup timing	iteration count (coarse)	solver timing	setup timing	iteration count (coarse)	solver timing
1	1.01s	233 (82)	10.1s	2.08s	149 (24)	6.42s
2	1.87s	155 (145)	10.2s	3.06s	59 (46)	3.42s
3	2.69s	108 (224)	9.96s	4.69s	35 (63)	2.37s
4	3.43s	84 (301)	9.25s	7.39s	27 (68)	1.95s
5	6.14s	70 (320)	7.50s	10.8s	24 (75)	1.82s
6	5.68s	63 (282)	5.21s	14.1s	23 (84)	1.89s
8	7.71s	54 (267)	4.12s	19.5s	22 (99)	2.02s
10	10.1s	49 (265)	3.62s	24.3s	22 (116)	2.31s

TABLE 6.8

Comparison of DD- $\alpha$ AMG and inexact deflation, coarse system solver tolerance  $10^{-12}$  and  $\nu = 5$  in inexact deflation, ill-conditioned system on a  $48^4$  lattice (Table 6.2: id 4), 2,592 cores.

Table 6.8 compares inexact deflation and DD- $\alpha$ AMG for a whole range for  $n_{inv}$ . We see that  $n_{inv} = 5$  provides the fastest DD- $\alpha$ AMG solver which is two times faster than the fastest inexact deflation solver which requires  $n_{inv} = 10$ . For the calculation of observables where the same system has to be solved for several right-hand-sides, this factor of two directly carries over to the total computation time since the setup cost then is negligible. When looking at combined times for setup and solve for one right hand side,  $n_{inv} = 2$  is best for DD- $\alpha$ AMG, where it takes 6.48s. The best choice for inexact deflation is  $n_{inv} = 6$  requiring 10.89s.

We also see that except for very small values for  $n_{inv}$ , the number of iterations required in DD- $\alpha$ AMG is less than half of that in inexact deflation. The numbers in parenthesis denote the average number of coarse solver iterations in each iteration of the respective method. For DD- $\alpha$ AMG the number of iterations on the coarse grid increases with the work spent in the setup. Hence, the lowest DD- $\alpha$ AMG-iteration count does not necessarily provide the fastest solver in the two grid setting. In inexact deflation the number of iterations on the coarse grid is not that clearly tied to  $n_{inv}$ . Since in inexact deflation the coarse system must be solved very accurately, the number of iterations needed to solve the coarse grid system is higher than in DD- $\alpha$ AMG. It is only moderately (a factor of 2 to 4) higher, though, because the code from [38] uses an additional adaptively computed preconditioner for the GCR iterations on the coarse system, whereas we use the less efficient odd-even preconditioning in DD- $\alpha$ AMG.

For the same number of test vectors, DD- $\alpha$ AMG produces a coarse system which is twice as large (and contains four times as many non zeros in the coarse grid operator) as that of inexact deflation with the benefit of preserving the  $\Gamma_5$  structure on the coarse grid. The DD- $\alpha$ AMG coarse grid system seems to be more ill-conditioned—an indication that the important aspects of the fine grid system are represented on the coarse grid—and the resulting coarse grid correction clearly lowers the total iteration count more efficiently and thus speeds up the whole method.

An ill-conditioned coarse grid system offers a potential for substantial improve-

ments when passing from two-grid to multigrid. In Figure 6.2 we therefore report results of a study with a yet experimental version of our multigrid code. It compares the solver times for odd-even preconditioned BiCGStab with DD- $\alpha$ AMG using 2, 3 and 4 levels for different choices of the mass parameter  $m_0$ . The gain for going from 2 to 3 levels is very noticeable; and for small values of  $m_0$ , corresponding to the physically interesting regimes, we observe an improvement of a factor of 2.5 to 3. Using more levels is a feature which is not available in the inexact deflation code.

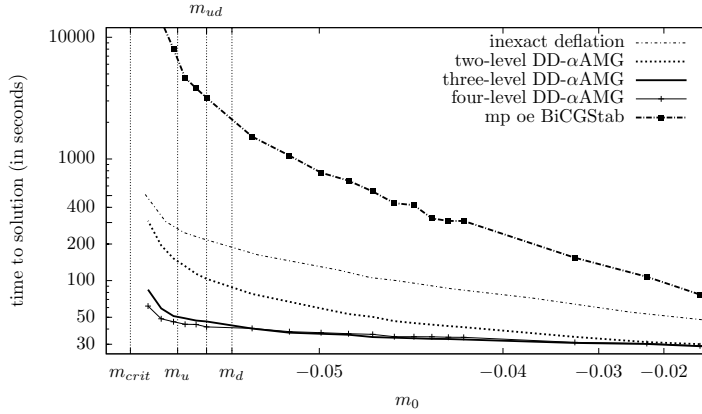


FIG. 6.2. Mass scaling of 2, 3 and 4 level DD- $\alpha$ AMG,  $64^4$  lattice (Table 6.2: id 5), restart length  $n_{kv} = 10$ , 128 cores. Here,  $m_{ud}$  is the average light quark mass (at and above which most of the current simulations are performed), assuming a common mass for the up and the down quark. Some recent simulations, e.g., [10, 21, 25], already distinguish the mass of the up and the down quark, and this will become more important in the near future. Then the regime close to  $m_u$  becomes relevant.

	Inexact deflation	DD- $\alpha$ AMG	speed-up factor
smooth iter	5	2	
setup iter	5	3	
setup time	10.9s	7.85s	1.39
solve iter	31	45	
solve time	8.63s	5.81s	1.49
total time	19.5s	13.6s	1.43

TABLE 6.9

Comparison of DD- $\alpha$ AMG with inexact deflation on an ill-conditioned system on a  $128 \times 64^3$  lattice (Table 6.2, id 6), same parameters as in Table 6.8, 8,192 cores.

Finalizing our discussion, we compare in Table 6.9 inexact deflation and DD- $\alpha$ AMG for another configuration typical for many recent lattice QCD computations. Configuration 6 differs from the other configurations in Table 6.2 in the way it was generated, resulting in quite different discretization effects. Again we took the default parameter set, but now with relatively cheap setup phases. This results in a gain factor of more than 1.4 for setup and solve in DD- $\alpha$ AMG against inexact deflation. Still 55% of the execution time is spent in coarse system solves in DD- $\alpha$ AMG.

**6.5. Comparison with GCR-Smoothing.** The general applicability of algebraic multigrid ideas to lattice QCD systems was first considered in [4, 14, 44] where the resulting method is simply called “AMG”, a terminology that we keep for the

following discussion. The method has been implemented as part of the QOPQDP library, see [43], which is publicly available. As we explained in sections 3.2, 3.3 and 3.4 AMG motivated many of the choices in DD- $\alpha$ AMG, particularly preservation of  $\Gamma_5$ -symmetry and aggregation based interpolation.

There are two major differences to the method presented here. One is the choice of the smoothing iteration. While DD- $\alpha$ AMG uses some steps of SAP, which can be regarded as a “block smoothing”, AMG uses “point smoothing”, i.e., some steps of standard (odd-even preconditioned) GCR. The other important difference is the setup. Different variants are considered in [3, 4, 14, 44], and the QOPQDP code proceeds by computing the test vectors as quite precise approximations to eigenvectors. They are obtained one at a time by applying a sufficient number of BiCGStab iterations, at the same time keeping the current vector orthogonal to all previous ones.

Table 6.10 reports a comparison of DD- $\alpha$ AMG with AMG for two of our configurations. We compared different choices of parameters with our standard parameter settings for DD- $\alpha$ AMG. We stopped the iterations when the initial residual was decreases by a factor of  $10^{-5}$  (instead of  $10^{-10}$ ), the reason for this being that in QOPQDP configurations are represented in single precision, only. For the default choice of parameters in AMG we see that the setup is substantially more costly (factors between 2 and 4 in time), while the number of iterations for each system solve is slightly less for DD- $\alpha$ AMG. We can make the effort in the AMG setup comparable to that of DD- $\alpha$ AMG by reducing the limit on the maximum number of BiCGStab iterations to be performed on each test vector (*msi*), but then the number of iterations for each solve increases in AMG and solve times become always larger than with DD- $\alpha$ AMG. The domain decomposition smoother involves less global communication than GCR, which turns out to have a substantial influence on the solve times for a higher number of cores. For example, on 8,192 cores, the solve times are 2 to 3 times smaller than in AMG.

id 5, 128 cores			id 6, 256 cores			
	AMG-d	AMG-20	DD- $\alpha$ AMG	AMG-d	AMG-10	DD- $\alpha$ AMG
setup time	2424s	826s	896s	2464s	607s	656s
solve iter	14	22	10	13	21	11
solve time	45.4s	66.0s	57.1s	36.5s	50.4s	37.3s
id 5, 8192 cores			id 6, 8192 cores			
	AMG-d	AMG-40	DD- $\alpha$ AMG	AMG-d	AMG-20	DD- $\alpha$ AMG
setup time	52.3s	24.6s	27.7s	89.9s	29.1s	32.3s
solve iter	14	16	10	13	16	11
solve time	4.75s	5.51s	1.82s	3.49s	3.43s	1.86s

TABLE 6.10

Comparison of DD- $\alpha$ AMG and AMG. AMG-d uses default parameter settings, AMG-k sets *msi* = *k* so that setup time is comparable to DD- $\alpha$ AMG. SSE optimization switched off in AMG.

**Acknowledgments.** We thank two anonymous referees for several valuable suggestions. We also thank the Budapest-Marseille-Wuppertal collaboration for providing configurations and compute time on Juropa. We would also like to acknowledge James Brannick (Pennsylvania State University) for his advice regarding the development of the multigrid method, Kalman Szabó (Bergische Universität Wuppertal) for his support with implementations, Wolfgang Söldner (University of Regensburg) for

his help with the I/O-interfaces, and Norbert Eicker (Bergische Universität Wuppertal and JSC) for sharing his expertise on Juropa.

## REFERENCES

- [1] C. ALEXANDROU, M. BRINET, J. CARBONELL, M. CONSTANTINOU, P. A. HARRAUD, P. GUICHON, K. JANSEN, T. KORZEC, AND M. PAPINUTTO, *Nucleon electromagnetic form factors in twisted mass lattice QCD*, Phys. Rev., D83:094502 (2011).
- [2] S. AOKI, K.-I. ISHIKAWA, N. ISHIZUKA, T. IZUBUCHI, D. KADOH, K. KANAYA, Y. KURAMASHI, Y. NAMEKAWA, M. OKAWA, Y. TANIGUCHI, A. UKAWA, N. UKITA, AND T. YOSHIE, *2+1 flavor lattice QCD toward the physical point*, Phys. Rev., D79:034503 (2009).
- [3] R. BABICH, J. BRANNICK, R. C. BROWER, M. A. CLARK, S. D. COHEN, J. C. OSBORN, AND C. REBBI, *The role of multigrid algorithms for LQCD*, PoS, LATTICE2009:031 (2009).
- [4] R. BABICH, J. BRANNICK, R. C. BROWER, M. A. CLARK, T. A. MANTEUFFEL, S. F. MCCORMICK, J. C. OSBORN, AND C. REBBI, *Adaptive multigrid algorithm for the lattice Wilson-Dirac operator*, Phys. Rev. Lett., 105:201602 (2010).
- [5] T. BAE, Y.-C. JANG, C. JUNG, H.-J. KIM, J. KIM, K. KIM, W. LEE, S. R. SHARPE, AND B. YOON, *Kaon B-parameter from improved staggered fermions in  $n_f = 2+1$  QCD*, Phys. Rev. Lett., 109:041601 (2012).
- [6] G. S. BALI, P. C. BRUNS, S. COLLINS, M. DEKA, B. GLÄSSLE, M. GÖCKELER, L. GREIL, T. R. HEMMERT, R. HORSLEY, J. NAJJAR, Y. NAKAMURA, A. NOBILE, D. PLEITER, P. E. L. RAKOW, A. SCHÄFER, R. SCHIEL, G. SCHIERHOLZ, A. STERNBECK, AND J. ZANOTTI, *Nucleon mass and sigma term from lattice QCD with two light fermion flavors*, Nucl. Phys., B866 (2013), pp. 1–25.
- [7] T. BANKS AND A. CASHER, *Chiral symmetry breaking in confining theories*, Nucl.Phys., B169 (1980), p. 103.
- [8] R. BEN-AV, M. HARMATZ, S. SOLOMON, AND P. G. LAUWERS, *Fermion simulations using parallel transported multigrid*, Phys. Lett., B253 (1991), pp. 185–192.
- [9] T. BERGRATH, M. RAMALHO, R. KENWAY, ET AL., *PRACE scientific annual report 2012*, tech. report, PRACE, 2012. [http://www.prace-ri.eu/IMG/pdf/PRACE\\_Scientific\\_Annual\\_Report\\_2012.pdf](http://www.prace-ri.eu/IMG/pdf/PRACE_Scientific_Annual_Report_2012.pdf), p. 32.
- [10] S. BORSANYI, S. DÜRR, Z. FODOR, J. FRISON, C. HOELBLING, S. D. KATZ, S. KRIEG, T. KURTH, L. LELLOUCH, T. LIPPERT, A. PORTELLI, A. RAMOS, A. SASTRE, AND K. K. SZABO, *Isospin splittings in the light baryon octet from lattice QCD and QED*, (2013).
- [11] D. BRAESS, *Towards algebraic multigrid for elliptic problems of second order*, Computing, 55 (1995), pp. 379–393.
- [12] A. BRANDT, *Multiscale scientific computation: Review 2001*, in Multiscale and Multiresolution Methods, T. J. Barth, T. Chan, and R. Haimes, eds., vol. 20 of Lecture Notes in Computational Science and Engineering, Springer Berlin Heidelberg, 2002, pp. 3–95.
- [13] A. BRANDT, J. BRANNICK, K. KAHL, AND I. LIVSHITS, *Bootstrap AMG*, SIAM J. Sci. Comput., 33 (2011), pp. 612–632.
- [14] J. BRANNICK, R. C. BROWER, M. A. CLARK, J. C. OSBORN, AND C. REBBI, *Adaptive multigrid algorithm for lattice QCD*, Phys. Rev. Lett., 100:041601 (2007).
- [15] M. BREZINA, R. FALGOUT, S. MACLACHLAN, T. MANTEUFFEL, S. MCCORMICK, AND J. RUGE, *Adaptive smoothed aggregation ( $\alpha$ SA) multigrid*, SIAM Review, 47 (2005), pp. 317–346.
- [16] M. BREZINA, T. MANTEUFFEL, S. MCCORMICK, J. RUGE, AND G. SANDERS, *Towards adaptive smoothed aggregation ( $\alpha$ SA) for nonsymmetric systems*, SIAM J. Sci. Comput., 32 (2010), pp. 14–39.
- [17] R. BROWER, E. MYERS, C. REBBI, AND K. MORIARTY, *The multigrid method for fermion calculations in quantum chromodynamics*, Tech. Report Print-87-0335, IAS, Princeton, 1987.
- [18] CLS, *Coordinated lattice simulation*. <https://twiki.cern.ch/twiki/bin/view/CLS/>.
- [19] T. DEGRAND AND C. E. DETAR, *Lattice Methods for Quantum Chromodynamics*, World Scientific, 2006.
- [20] T. A. DEGRAND AND P. ROSSI, *Conditioning techniques for dynamical fermions*, Comput. Phys. Commun., 60 (1990), pp. 211–214.
- [21] A. DUNCAN, E. EICHEN, AND H. THACKER, *Electromagnetic splittings and light quark masses in lattice QCD*, Phys. Rev. Lett., 76 (1996), pp. 3894–3897.
- [22] S. DÜRR, Z. FODOR, J. FRISON, C. HOELBLING, R. HOFFMANN, S. D. KATZ, S. KRIEG, T. KURTH, L. LELLOUCH, T. LIPPERT, K. SZABO, AND G. VULVERT, *Ab initio determination of light hadron masses*, Science, 322 (2008), pp. 1224–1227.
- [23] S. DÜRR, Z. FODOR, C. HOELBLING, S. D. KATZ, S. KRIEG, ET AL., *Lattice QCD at the physical*

- point: Simulation and analysis details*, JHEP, 08(2011)148 (2011).
- [24] S. DÜRR, Z. FODOR, C. HOELBLING, S. D. KATZ, S. KRIEG, T. KURTH, L. LELLOUCH, T. LIPP-  
PERT, K. K. SZABO, AND G. VULVERT, *Lattice QCD at the physical point: Light quark  
masses*, Phys. Lett. B701, (2011), pp. 265–268.
  - [25] J. FINKENRATH, F. KNECHTLI, AND B. LEDER, *One flavor mass reweighting in lattice QCD*,  
Nucl. Phys., B877 (2013), pp. 441–456.
  - [26] P. FRITZSCH, F. KNECHTLI, B. LEDER, M. MARINKOVIC, S. SCHAEFER, R. SOMMER, AND F. VI-  
ROTTA, *The strange quark mass and Lambda parameter of two flavor QCD*, Nucl. Phys.,  
B865 (2012), pp. 397–429.
  - [27] A. FROMMER, K. KAHL, S. KRIEG, B. LEDER, AND M. ROTTMANN, *Adaptive aggregation based  
domain decomposition multigrid for the lattice Wilson Dirac operator*, arXiv:1303.1377,  
(2013).
  - [28] A. FROMMER, A. NOBILE, AND P. ZINGLER, *Deflation and flexible SAP-preconditioning of  
GMRES in lattice QCD simulation*, tech. report, 2012. arXiv:1204.5463 [hep-lat].
  - [29] C. GATtringER AND C. B. LANG, *Quantum Chromodynamics on the Lattice*, vol. 788 of Lect.  
Notes Phys., Springer, 1st ed., 2009.
  - [30] I. GOHBERG, P. LANCASTER, AND L. RODMAN, *Indefinite Linear Algebra and Applications*,  
Birkhäuser, Basel, 2005.
  - [31] M. GUEST, G. ALOISIO, R. KENWAY, ET AL., *The scientific case for  
HPC in Europe 2012 - 2020*, tech. report, PRACE, October 2012.  
<http://www.prace-ri.eu/PRACE-The-Scientific-Case-for-HPC>, p. 75.
  - [32] W. HACKBUSCH, *Multi-Grid Methods and Applications*, vol. 4 of Springer Series in Computa-  
tional Mathematics, Springer, 1st ed., 2003.
  - [33] K. KAHL AND H. RITTICH, *Analysis of the deflated conjugate gradient method based on sym-  
metric multigrid theory*. Preprint BUW-IMACM 12/19, 2012.
  - [34] T. KALKREUTER, *Multigrid methods for propagators in lattice gauge theories*, J. Comput. Appl.  
Math., 63 (1995), pp. 57–68.
  - [35] A. D. KENNEDY, *Algorithms for dynamical fermions*, arXiv:hep-lat/0607038, (2006).
  - [36] S. KRIEG AND T. LIPPert, *Tuning lattice QCD to petascale on Blue Gene/P*, NIC Symposium  
2010, (2010), pp. 155–164.
  - [37] T. LIPPert, *Parallel SSOR preconditioning for lattice QCD*, Parallel Computing, 25 (1999),  
pp. 1357–1370.
  - [38] M. LÜSCHER, *DD-HMC algorithm for two-flavour lattice QCD*.  
<http://luscher.web.cern.ch/luscher/DD-HMC>, used version: DD-HMC-1.2.2, September  
2008.
  - [39] M. LÜSCHER, *openQCD simulation program for lattice QCD with open boundary conditions*.  
<http://luscher.web.cern.ch/luscher/openQCD/>, used version: openQCD-1.2, May 2013.
  - [40] ———, *Solution of the Dirac equation in lattice QCD using a domain decomposition method*,  
Comput. Phys. Commun. 156, (2004), pp. 209–220.
  - [41] ———, *Local coherence and deflation of the low quark modes in lattice QCD*, JHEP,  
07(2007)081 (2007).
  - [42] I. MONTVAY AND G. MÜNSTER, *Quantum Fields on a Lattice*, Cambridge Monographs on  
Mathematical Physics, Cambridge University Press, 1994.
  - [43] J. C. OSBORN, *Multigrid solver for clover fermions, implementation within QOPQDP*.  
<http://usqcd.jlab.org/usqcd-docs/qopqdp/>, used version: QOPQDP 0.19.0, April 2013.
  - [44] J. C. OSBORN, R. BABICH, J. BRANNICK, R. C. BROWER, M. A. CLARK, S. D. COHEN, AND  
C. REBBI, *Multigrid solver for clover fermions*, PoS, LATTICE2010:037 (2010).
  - [45] H. RITTICH, *Deflation in multigrid methods*, master’s thesis, Bergische Universität Wuppertal,  
2011.
  - [46] J. RUGE AND K. STÜBEN, *Algebraic multigrid*, in Multigrid Methods, S. F. McCormick, ed.,  
vol. 3 of Frontiers in Applied Mathematics, SIAM, Philadelphia, 1987, pp. 73–130.
  - [47] Y. SAAD, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, PA, USA, 2nd ed.,  
2003.
  - [48] H. SCHWARZ, *Gesammelte mathematische Abhandlungen*, Vierteljahrsschrift Naturforsch. Ges.  
Zürich, (1870), pp. 272–286.
  - [49] B. SHEIKHOLESAMI AND R. WOHLERT, *Improved continuum limit lattice action for QCD with  
Wilson fermions*, Nucl. Phys., B259 (1985), pp. 572–597.
  - [50] V. SIMONCINI AND D. B. SZYLD, *Theory of inexact Krylov subspace methods and applications  
to scientific computing*, SIAM J. Sci. Comput., 25 (2003), pp. 454–477.
  - [51] B. F. SMITH, P. E. BJØRSTAD, AND W. D. GROPP, *Domain Decomposition: Parallel Multilevel  
Methods for Elliptic Partial Differential Equations*, Cambridge University Press, New York,  
1996.

- [52] G. SMITH, *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, Oxford Applied Mathematics and Computing Science Series, Clarendon Press, 1985.
- [53] J. M. TANG, S. P. MACLACHLAN, R. NABBEN, AND C. VUIK, *A comparison of two-level preconditioners based on multigrid and deflation*, SIAM J. Matrix Anal. Appl., 31 (2010), pp. 1715–1739.
- [54] U. TROTTEBERG, C. W. OOSTERLEE, AND A. SCHÜLLER, *Multigrid. With Guest Contributions by A. Brandt, P. Oswald, K. Stüben*, Academic Press, Orlando, FL, 2001.
- [55] J. VAN DEN ESHOF AND G. L. SLEIJPEN, *Inexact Krylov subspace methods for linear systems*, SIAM J. Matrix Anal. Appl., 26 (2004), pp. 125–153.
- [56] J. C. VINK, *Multigrid inversion of staggered and Wilson fermion operators with  $SU(2)$  gauge fields in two-dimensions*, Phys. Lett., B272 (1991), pp. 81–85.
- [57] K. G. WILSON, *Quarks and strings on a lattice*, in New Phenomena in Subnuclear Physics. Part A. Proceedings of the First Half of the 1975 International School of Subnuclear Physics, Erice, Sicily, July 11 - August 1, 1975, A. Zichichi, ed., vol. 321 of CLNS Reports, New York, 1977, Plenum Press.