Lanczos-based Low-Rank Correction Method for Solving the Dyson Equation in Inhomogenous Dynamical Mean-Field Theory

Pierre Carrier[a,*], Jok M. Tang[a], Yousef Saad[a], James K. Freericks[b]

[a]*Department of Computer Science & Engineering and Minnesota Supercomputing Institute, University of Minnesota, Minneapolis, Minnesota, 55455*
[b]*Department of Physics, Georgetown University, Washington, D.C., 20057*

Abstract

Inhomogeneous dynamical mean-field theory has been employed to solve many interesting strongly interacting problems from transport in multilayered devices to the properties of ultracold atoms in a trap. The main computational step, especially for large systems, is the problem of calculating the inverse of a large sparse matrix to solve Dyson's equation and determine the local Green's function at each lattice site from the corresponding local self-energy. We present a new efficient algorithm, the Lanczos-based low-rank algorithm, for the calculation of the inverse of a large sparse matrix which yields this local (imaginary time) Green's function. The Lanczos-based low-rank algorithm is based on a domain decomposition viewpoint, but avoids explicit calculation of Schur complements and relies instead on low-rank matrix approximations derived from the Lanczos algorithm, for solving the Dyson equation. We report at least a 25-fold improvement of performance compared to explicit decomposition (such as sparse LU) of the matrix inverse. We also report that scaling relative to matrix sizes, of the low-rank correction method on the one hand and domain decomposition methods on the other, are comparable.

Keywords: Lanczos, Matrix inversion, Inhomogeneous Dynamical Mean-Field Theory, Green's functions
PACS: 31.15.xr, 31.15.aq, 71.10.Fd
2000 MSC: 15A09, 65Z05, 65F10, 65F50

1. Introduction

The Dyson equation is at the heart of many important physical problems, taking its roots in many-body formalisms for the Green's function [1, 2]. The real-space version of the Dyson equation is of particular importance in the framework of inhomogeneous dynamical mean-field theory (IDMFT) [3, 4, 5, 6, 7, 8]. In this framework, the solution of the Dyson equation, for a given lattice (e.g., square or cubic "optical" lattice), is being matched self-consistently to the solution of an impurity problem. A fixed-point iteration ensures that the two local Green's functions (for the lattice and for the impurity problem) are equal to each other. The self-energy, Σ_i, defined at each lattice site i, and introduced into the Dyson equation, is used for numerically matching the two problems. Details on the IDMFT algorithm can be found in [5, 4, 6, 7, 8, 9].

This paper focuses primarily on the first part of the IDMFT numerical method, the Dyson equation solver, which constitutes the most computationally intensive part of the IDMFT algorithm. Details on the second part, the impurity

---

*email: carrier@cs.umn.edu

solvers, applied to the Fermi-Bose Falicov-Kimball model [10, 11, 12, 13], can be found in Ref. [6, 8, 9], as it is used for our numerical simulations shown below.

The matrix form of the Dyson equation (along the imaginary time axis) used in any IDMFT implementation, is given by [2, 4]:

$$\mathbf{G}_{ij}(i\omega_k) = \left\{[\mathbf{G}_{ij}(i\omega_k; U = 0)]^{-1} - \Sigma_i(i\omega_k)\delta_{ij}\right\}^{-1} \equiv \left\{\mathbf{H}_{ij}(i\omega_k)\right\}^{-1}. \tag{1}$$

The matrix $[\mathbf{G}_{ij}(i\omega_k; U = 0)]$ corresponds to the noninteracting Green's function, whose inverse includes a kinetic-energy term representing the nearest-neighbor hopping from site j to site i on a square (or cubic) lattice, plus the Matsubara frequency along the imaginary time axis, $i\omega_k$, on the diagonal $\mathbf{H}_{ii}$, and the local potential which involves the sum of the chemical potential minus the local trapping potential; that is, we have $[\mathbf{G}_{ij}(i\omega_k; U = 0)]^{-1} = t_{ij} + (i\omega_k + \mu - V_i)\delta_{ij}$. It is noninteracting in the sense that it excludes the many-body potential energy terms, represented instead by the self-energy, $\Sigma_i(i\omega_k)$. Note that the self-energy varies from site to site, and is therefore *inhomogeneous* [5]. The explicit values of $\Sigma_i(i\omega_k)$ on each lattice site i are deduced from an impurity solver, not described here [11, 14, 13, 9] for each lattice site. For convenience, we have defined $\mathbf{H}$ in eq. (1) as, $\mathbf{G} \equiv \{\mathbf{H}\}^{-1}$, to be used when describing the new algorithm below. The matrix $\mathbf{H}$ corresponds to a lattice model, and it is non-singular, of rank N. It is a complex-symmetric matrix, with complex-diagonal elements and real-valued off-diagonal elements, called hopping terms. A similar expression for the Dyson equation can also be deduced along the real frequency axis, $i\omega_k \rightarrow \omega + i\delta$ [4]. Note that the off-diagonal elements in the matrix $\mathbf{H}$ are extremely sparse because the hopping is restricted to nearest neighbors, so there are typically only Z nonzero terms, where Z is the coordination number of the lattice (4 for a square lattice and 6 for a simple cubic lattice). In IDMFT, the quantity of interest is essentially the diagonal of the inverse, $\mathbf{G}_{ii}$, which is then matched to the impurity Green's function [9] inside a self-consistency loop. Therefore, the main task of the numerical method developed below is to determine

$$diag(\mathbf{G}) \equiv diag\left((\mathbf{H})^{-1}\right)$$

from eq. (1).

2. The Lanczos-based Low-rank Correction Algorithm

The new algorithm that solves the diagonal of the inverse of the matrix in eq. (1) essentially combines two general approaches of solvers, one iterative and one direct. Those two approaches are: the Lanczos scheme [15] and the domain decomposition [16]. We first briefly recall the Lanczos algorithm and its important elements needed for the rest, then show an important example of domain decomposition, and describe finally the new Lanczos-based low-rank correction algorithm based on this example of domain decomposition.

2.1. Standard Lanczos

The standard Lanczos algorithm is a special case of the Arnoldi algorithm applied to real-symmetric or Hermitian matrices [17]. Let assume first that $\mathbf{H}$ is real-symmetric. The fundamental 3-terms recurrence of the Lanczos algorithm is given by [18]

$$\beta_{j+1}\mathbf{q}_{j+1} = \mathbf{H}\mathbf{q}_j - \alpha_j\mathbf{q}_j - \beta_j\mathbf{q}_{j-1}, \quad \text{for } j = 1, \ldots, N.$$

This recurrence requires only one "matrix-vector" operation per Lanczos iteration, corresponding to the following operation:

$$\mathbf{u}_j \leftarrow \mathbf{H}\mathbf{q}_j, \tag{2}$$

where $\mathbf{u}_j$ is a temporary variable holding the resulting matrix-vector product. In matrix form, the Lanczos recurrence gives rise to a tridiagonal matrix $T_j \equiv tridiag(\beta_j, \alpha_j, \beta_{j+1})$, at the jth Lanczos iteration. Once $j = N$, the rank of the matrix $\mathbf{H}$, we obtain the exact relation

$$\mathbf{H} = \mathbf{Q}_N\mathbf{T}_N\mathbf{Q}_N^T. \tag{3}$$

It would be possible in practice to determine $\mathbf{H}^{-1}$ from this above matrix decomposition (See for example Ref. [19]). However, for very large N this procedure is prohibitive, due to the necessity to use and store *all* N Lanczos vectors, in either a partial or full re-orthogonalization [18] implementation.
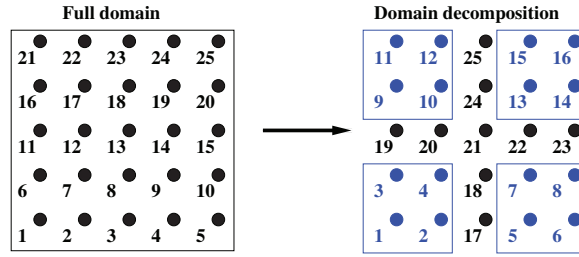
Figure 1: A 2-dimensional example of a domain, Ω, discretized with $n = 5^2$ equidistant grid points. The left part shows a conventional ordering of grid points; the right side shows a domain decomposition ordering, where grid points in the four subdomains are numbered first, followed by the interface points.

As just stated, this Lanczos scheme applies only to real-symmetric (or Hermitian) matrices [18]. When $\mathbf{H}$ is complex-symmetric, as in IDMFT, the standard Lanczos scheme needs to be modified. Freund *et al.* [20] have shown that applying the Lanczos algorithm to complex-symmetric matrices can be done by replacing *everywhere* in the algorithm (including re-orthogonalization) the usual inner product by an indefinite inner product defined as a transposition only, excluding the conjugation, i.e.: $(\mathbf{q}, \mathbf{u})_{\mathbb{C}} = \mathbf{q}^T \mathbf{u}$, where $\mathbf{q}$ and $\mathbf{u}$ are any two complex vectors. The corresponding indefinite "norm" (inside quotation marks, because it does not satisfy fundamental properties of a norm) is: $\|q\|_{\mathbb{C}} = \sqrt{(\mathbf{q}, \mathbf{q})_{\mathbb{C}}}$, the principal square root. Note that the resulting indefinite "norm" can be a complex number, and thus result in a possible breakdown, or division by zero, in the algorithm (when $\|q\|_{\mathbb{C}}$ may become null for two non-null vectors). This potential breakdown is taken care of by allowing for a few extra iterations, making sure that the null space can be reached within a sufficient number of Lanczos steps, as further described below in the context of the Lanczos-based low-rank algorithm.

2.2. Standard Domain Decomposition

Domain decomposition is a very broad method [16]. We describe here only one application, used next in the Lanczos-based low-rank algorithm. Consider now two different site labelings of a square lattice depicted in Fig. 1. A straightforward labeling of the nearest-neighbor hopping matrix, as part of the local Green's function $\mathbf{G}_{ij}(U = 0)$ for lattice sites j to i of eq. (1), is shown on the left of Fig. 1, where lattice sites are labeled consecutively from bottom-left to top-right. This labeling corresponds to a unique domain. For hopping occuring only between nearest-neighbors, this straightforward numbering leads to a penta-diagonal matrix (similarly to a 5-point Laplacian operator), in two-dimensions. Of course, a multitude of different labelings can be constructed. Consider another important example, shown on the right of Fig. 1, where a decomposition into four subdomains of the nearest-neighbor lattice sites has been devised. The corresponding matrix $\mathbf{H}$, containing four subdomains, is no longer penta-diagonal, but has now the following form:

$$\mathbf{H} = \left(\begin{array}{cccc|c} \mathbf{B}_1 & 0 & 0 & 0 & \mathbf{E}_1 \\ 0 & \mathbf{B}_2 & 0 & 0 & \mathbf{E}_2 \\ 0 & 0 & \mathbf{B}_3 & 0 & \mathbf{E}_3 \\ 0 & 0 & 0 & \mathbf{B}_4 & \mathbf{E}_4 \\ \hline \mathbf{E}_1^T & \mathbf{E}_2^T & \mathbf{E}_3^T & \mathbf{E}_4^T & \mathbf{C} \end{array} \right).$$

The four blocks, $\mathbf{B}_i$, $i = 1, \ldots, 4$, can then be grouped into a block diagonal matrix $\mathbf{B}$, and similarly for the $\mathbf{E}$ (and $\mathbf{E}^T$):

$$\mathbf{H} = \left(\begin{array}{cc} \mathbf{B} & \mathbf{E} \\ \mathbf{E}^T & \mathbf{C} \end{array} \right). \tag{4}$$

The Schur complement

$$\mathbf{S} \equiv \mathbf{C} - \mathbf{E}^T \mathbf{B}^{-1} \mathbf{E} \tag{5}$$

is then defined such that the matrix in eq. (4) can be re-written as a product of lower and upper block-triangular matrices [17], as:

$$\mathbf{H} = \left(\begin{array}{cc} \mathbf{I} & \mathbf{0} \\ \mathbf{E}^T \mathbf{B}^{-1} & \mathbf{I} \end{array} \right) \left(\begin{array}{cc} \mathbf{B} & \mathbf{E} \\ \mathbf{0} & \mathbf{S} \end{array} \right),$$

where $\mathbf{I}$ is the identity matrix of proper dimension. The rank m of the Schur complement matrix $\mathbf{S}$ is equal to the number of interface points. In the example of Fig. 1 on the right, the interface points labeled from 17 to 25 give a rank $m = 9$ for $\mathbf{S}$, for an entire matrix $\mathbf{H}$ of rank $N = 25$.

The main expression giving the inverse of $\mathbf{H}$ based on the domain decomposition of Fig. 1, can now be deduced from the above product of matrices. The resulting inverse takes the form:

$$
\begin{aligned}
\mathbf{H}^{-1} &= \begin{pmatrix} \mathbf{B}^{-1} & -(\mathbf{B}^{-1}\mathbf{E})\mathbf{S}^{-1} \\ \mathbf{0} & \mathbf{S}^{-1} \end{pmatrix} \begin{pmatrix} \mathbf{I} & \mathbf{0} \\ -(\mathbf{E}^T\mathbf{B}^{-1}) & \mathbf{I} \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} + \begin{pmatrix} (\mathbf{B}^{-1}\mathbf{E})\mathbf{S}^{-1}(\mathbf{E}^T\mathbf{B}^{-1}) & -(\mathbf{B}^{-1}\mathbf{E})\mathbf{S}^{-1} \\ -\mathbf{S}^{-1}(\mathbf{E}^T\mathbf{B}^{-1}) & \mathbf{S}^{-1} \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} + \begin{pmatrix} (\mathbf{B}^{-1}\mathbf{E}) \\ -\mathbf{I} \end{pmatrix} \mathbf{S}^{-1} \left((\mathbf{E}^T\mathbf{B}^{-1}) \quad -\mathbf{I} \right).
\end{aligned}
\tag{6}
$$

Standard domain decomposition methods for finding $\mathbf{H}^{-1}$ (or only its diagonal) are based on solving eq. (6), with the inverse of the Schur complement obtained from the triple matrix product in eq. (5). This expression involves several matrix multiplications plus an inversion of the full Schur complement matrix $\mathbf{S}$. Efficient methods which drop non-significant terms (when only the diagonal is needed) are essential for obtaining a rapid algorithm [21, 22, 23], at the price of higher complexity. Recalling that the dimension of the matrix $\mathbf{S}$ equals that of $\mathbf{C}$, i.e., the number of interface points m, then using more domains leads to more interface points, and consequently to a larger size Schur complement which is obviously more difficult to invert. We have gathered at this point all the necessary ingredients for developing the Lanczos-based low-rank algorithm.

2.3. Combination of Lanczos with Domain Decomposition

The Lanczos-based low-rank correction algorithm starts with an identical decomposition of the domain as shown on the right of Fig. 1, leading to eq. (6) for the inverse $\mathbf{H}^{-1}$. It however avoids any explicit calculation of the inverse of the Schur complement. The algorithm is obtained from eq. (6) by first grouping the inverse $\mathbf{H}^{-1}$ together with the blocks $\mathbf{B}^{-1}$ on the right-hand side of eq. (6), and consequently defining $\mathbf{X}$ as:

$$
\mathbf{X} \equiv \mathbf{H}^{-1} - \begin{pmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} = \begin{pmatrix} (\mathbf{B}^{-1}\mathbf{E}) \\ -\mathbf{I} \end{pmatrix} \mathbf{S}^{-1} \left((\mathbf{E}^T\mathbf{B}^{-1}) \quad -\mathbf{I} \right).
\tag{7}
$$

The fundamental idea of this new algorithm is based on the fact that the right-hand side of eq. (7) has rank m, much smaller than N. Consequently, $\mathbf{X}$ in eq. (7), is also of rank $m \ll N$. To see this, recall that the rank of $\mathbf{S}$ corresponds to the number of interface points of the domain decomposition, related to $\mathbf{C}$ as defined in eq. (4). The matrix $\mathbf{X}$ has, by construction, the same rank as $\mathbf{S}$ (or $\mathbf{C}$). The diagonal of the inverse of $\mathbf{H}$ can thus be determined after $m \ll N$ Lanczos steps with the matrix $\mathbf{X}$:

$$
\mathbf{X} = \mathbf{Q}_m \mathbf{T}_m \mathbf{Q}_m^T,
\tag{8}
$$

i.e., instead of using the entire N steps on the original matrix $\mathbf{H}$, as in eq. (3). Beyond the mth Lanczos iteration the null space is reached and $q_{m+1}\beta_m e_m^* \equiv 0$ — so the expression in eq. (8) is *exact*. In this situation, we have the *exact* following identity, after combining eq. (8) with eq. (7):

$$
\mathbf{Q}_m \mathbf{T}_m \mathbf{Q}_m^T = \begin{pmatrix} (\mathbf{B}^{-1}\mathbf{E}) \\ -\mathbf{I} \end{pmatrix} \mathbf{S}^{-1} \left((\mathbf{E}^T\mathbf{B}^{-1}) \quad -\mathbf{I} \right).
\tag{9}
$$

Equation (9) is the fundamental identity of the Lanczos-based low-rank correction algorithm.

In any standard Lanczos implementation [18], the matrix-vector operation given in the expression (2) must now be replaced by a "matrix-vector" operation with $\mathbf{X}$, as:

$$
\mathbf{u}_j \leftarrow \mathbf{X}\mathbf{q}_j.
$$

This in fact translates as one system solve, one inversion of small block matrices, and an update based on the difference defined in eq. (7):

$$\text{Solve } \mathbf{H}\mathbf{v}_j \;=\; \mathbf{q}_j, \tag{10}$$

$$\text{Compute } \mathbf{w}_j \;=\; \begin{pmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} \mathbf{q}_j, \tag{11}$$

$$\text{Output: } \mathbf{u}_j \;\leftarrow\; \mathbf{v}_j - \mathbf{w}_j. \tag{12}$$

Equations (10)–(12) replace the usual matrix-vector operation of standard Lanczos implementations [18]. The initial problem of finding the inverse of a matrix has thus been replaced by one system solve per Lanczos step, for the matrix $\mathbf{H}$ in eq. (10). In the example of Fig. 1 on the right, only 9 out of 25 possible Lanczos steps are required on $\mathbf{X}$, and consequently, only 9 system solves based on eq. (10) are necessary. Computing the block diagonals in eq. (11) is relatively straightforward, since each block has relatively small size.

The diagonal of the inverse is finally obtained from the expression:

$$diag\left(\mathbf{H}^{-1}\right) = diag\begin{pmatrix} \mathbf{B}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} + diag\left(\mathbf{Q}_m\mathbf{T}_m\mathbf{Q}_m^T\right), \tag{13}$$

deduced from eq. (7) and eq. (9) for the diagonal terms only. The diagonal of $\mathbf{B}^{-1}$ is a constant matrix and needs to be calculated only once at the very beginning of the Lanczos steps. Standard domain decomposition given by eq. (6) and using eq. (5) requires two matrix-matrix multiplications for obtaining the inverse of the Schur complement, while in the Lanczos scheme, the $\mathbf{T}_m$ matrix is simply a tridiagonal matrix $tridiag(\beta_m, \alpha_m, \beta_{m+1})$, which implies that the product of the three matrices $diag[\mathbf{Q}_m\mathbf{T}_m\mathbf{Q}_m^T]$ in eq. (13) is much simpler, and corresponds to the following algebraic expression obtained by updating only three Lanczos vectors:

$$diag\left(\mathbf{Q}_m\mathbf{T}_m\mathbf{Q}_m^T\right) = \beta_i(\mathbf{q}_{i-1} \odot \mathbf{q}_i) + \alpha_i(\mathbf{q}_i \odot \mathbf{q}_i) + \beta_{i+1}(\mathbf{q}_i \odot \mathbf{q}_{i+1}).$$

The symbol $\odot$ stands for term-by-term vector product.

Note that, as mentioned before in the Lanczos Section, the algorithm of Freund *et al.* [20] applied to complex-symmetric matrices potentially give null eigenvalues in the matrix $\mathbf{T}_m$, once the m Lanczos steps are done. This fact translates as the null space being visited *before* the termination of the m Lanczos iterations. It implies that once m Lanczos iterations are performed, the Lanczos vector basis does not yet form a complete basis of the low-rank matrix $\mathbf{X}$, and therefore eq. (8) still contains an error term that is not yet *exactly* null. We use α_j, $j = 1, \ldots, m$, the diagonal terms of $\mathbf{T}_m$, as a stopping criteria; as soon as α_j becomes null then the null space is reached. In practice, a few extra iterations beyond m are allowed. In the IDMFT implementation, we find that no more than 5 or 10 extra iterations are required, for m of the order of 100 for instance. Other stopping criteria, such as $\|\mathbf{q}\| < \epsilon$, using the $\mathcal{L}_2$ norm on the Lanczos vectors, or by eventually computing all eigenvalues of $\mathbf{T}_m$ when m is relatively small, can be used for determining when the null space is reached beyond the m Lanczos steps, and then stop.

Note finally that the Lanczos-based low-rank algorithm is well-suited to recursive calls, as for any domain decomposition implementation. Various levels of recursion of the domain in the Lanczos-based low-rank correction method are done by alternating the number of domains, (d_x, d_y), respectively along the x and y directions, using for instance the series $(d_x, d_y) = (2,1), (1,2), (2,1), \ldots$ We find that this approach is the optimal recursion, reducing both the number of interface points and the number of necessary recursions.

3. Numerical Simulations

We evaluate next the performance of the new algorithm, applied to IDMFT for a two-dimensional Fermi-Bose Falicov-Kimball model Hamiltonian [12, 13, 6, 7]. In IDMFT, multiple inversions of eq. (1) need to be performed, i.e., for each Matsubara frequency included in the calculation, $i\omega_k$ for $k = 0$ up to k_{max}, and at each self-consistency iteration of the IDMFT loop (where the impurity and the Dyson solutions are matched, as briefly described in the Introduction). More details on the parallel IDMFT algorithm and its performance can be found in Refs. [6, 7, 9]. Note that a "probing" method for inverting eq. (1), that takes advantage of diagonal dominancy for some matrices with large

Matsubara frequencies $i\omega_k$, has been implemented and is also discussed in Refs. [24, 9]. However, as the temperature parameter T of the simulation diminishes, i.e., the ultimate goal in "ultracold" simulations [6, 7, 8], the matrices tend to become indefinite. The matrices $\mathbf{H}$ are also mostly indefinite in the real-time Green's function problem. Therefore, we focus in this short paper on the computer performance of the Lanczos-based low-rank algorithm for the inversion of general matrices, including indefinite matrices.

The original version of the IDMFT code [6] used LAPACK routines, where a decomposition into $\mathbf{H} = \mathbf{LU}$ was then completed with a full inversion, $\mathbf{G} \equiv \mathbf{H}^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}$, for finally extracting the diagonal. These complex-valued LAPACK routines, however, make use of *dense* matrix techniques. For the sake of comparison, we compare instead the performance of the Lanczos-based low-rank implementation to that of a preconditioned-GMRES [17] solver that finds the solutions of the following systems of equations, using sparse techniques:

$$\mathbf{H}x^i = e^i, \quad \text{for} \ \ i = 1, N, \tag{14}$$

where the e^i's are Euclidian vectors. Each vector x^i, solution of the above system, corresponds to the columns of $\mathbf{G} = \mathbf{H}^{-1}$; the diagonal of the inverse is then extracted from the x^i_i components. We note that the performance of this rather expensive algorithm, is still better than using the $\mathbf{U}^{-1}\mathbf{L}^{-1}$ decomposition from LAPACK; sparse solvers for eq. (14) thus give a higher bound of performance to the dense $\mathbf{U}^{-1}\mathbf{L}^{-1}$ decompositions previously used [6].

The performances of the Lanczos-based low-rank method shown next were obtained from calculations on a Cray XT5 supercomputer (the machine "Kraken") located at the NICS, as part of the Teragrid. The dimension of the matrices were set to $N = n^2 = 101^2$, where n represents the number of lattice sites in both the x and y directions. 63 Matsubara frequencies, and their corresponding matrices, are calculated (on separate processors) for each of the self-consistency steps. The accuracy of the final solution is one part in 10^8 for all calculations. A preconditioned-GMRES sparse solver, based on eq. (14), takes at most 717.64 CPU seconds (~ 12 minutes) for finding *one* of the 63 inverses of $\mathbf{H}(i\omega_k)$, among all the possible inversions occuring in the entire self-consistency iteration. (It corresponds in fact to ω_0, the smallest Matsubara frequency.) This implies that if for example 20 self-consistency IDMFT iterations are necessary for achieving self-consistency, then the total time for the IDMFT loop is about $20 \times 718 = 14,360$ seconds, or ~4 hours, when a method based on eq. (14) is used. Using the Lanczos-based low-rank correction method, the exact same IDMFT calculation takes at most 28.66 CPU seconds for finding the same inverses. Thus, the 20 iterations in the IDMFT loop discussed above take a mere $20 \times 29 = 580$ seconds, i.e., less than 10 minutes. This corresponds to a dramatic increase in performance (irrespective of the type of matrices, indefinite or diagonally dominant), i.e., at least a 25-fold increase in computational speed (the increase versus the dense LAPACK approach is much larger).

We report in addition that the scaling with the dimension N of the matrices, between domain decomposition with an explicit calculation of the Schur complement based on eq. (6), and the Lanczos-based low-rank correction, are equivalent [9]. They however have different prefactors (slightly higher in the Lanczos-based algorithm) due to different levels of optimization in the programs. This implies that the performance of the Lanczos-based low-rank method is equivalent to that of calculating explicitly the Schur complement inverses, a method much more complex in its implementation. This property will become important when considering in the near future 3-dimensional implementations of this algorithm.

4. Conclusions and Perspectives

The Lanczos-based low-rank correction method elegantly combines two important and usually distinct numerical methods for finding the diagonal of the inverse of complex symmetric sparse matrices: the Lanczos iterative scheme, and the direct domain decomposition method. We have shown how this new algorithm avoids the explicit computation of the Schur complement. The algorithm transforms the inversion problem into a much smaller set of system solves, using a low-rank version of the original matrix to be inverted, and where the right-hand side vectors of the system solves are Lanczos vectors. We used a 2-dimensional implementation of IDMFT applied to the Fermi-Bose Falicov-Kimball model Hamiltonian and showed that the performance of this new algorithm corresponds to a 25-fold improvement, compared to other sparse methods. The algorithm is currently being implemented inside a three-dimensional IDMFT Dyson solver applied to the Hubbard model [25], and combined with a continuous time quantum Monte-Carlo impurity solver to solve for system sizes N on the order of a few million for direct comparison to experiment [26].

5. Aknowledgements

[1] L. P. Kadanoff, G. Baym, Quantum Statistical Mechanics, W. A. Benjamin, Inc., New York, USA, 1962.

[2] E. N. Economou, Green's Functions in Quantum Physics, Springer, Berlin, Germany, 2006.

[3] M. Potthoff, W. Nolting, Metallic surface of a Mott insulator-Mott insulating surface of a metal, Phys. Rev. B 60 (1999) 7834–7849.

[4] J. K. Freericks, Transport in multilayered nanostructures: The dynamical mean-field theory, Imperial College Press, London, U.K., 2006.

[5] M.-T. Tran, Inhomogeneous phases in the Falicov-Kimball model: Dynamical mean-field approximation, Phys. Rev. B 73 (2006) 205110–1–9.

[6] J. K. Freericks, Optimizing pre-formed molecules in mixtures of ultracold ^{40}K and ^{87}Rb on an optical lattice: a challenge grant and capabilities application project, Proceedings of the DOD High Performance Computing Modernization Program Users Group Conference 2009, San Diego, CA, June 15–18, 2009, edited by D. E. Post, IEEE Computer Society, Los Alamitos, CA, 2009, (2009) 205–212.

[7] J. K. Freericks, M. M. Maska, A. Hu, T. M. Hanna, C. J. Williams, P. S. Julienne, R. Lemanski, Improving the efficiency of ultracold dipolar molecule formation by first loading onto an optical lattice, Phys. Rev. A 81 (2010) 011605(R)–1–4.

[8] J. K. Freericks, H. R. Krishnamurthy, P. Carrier, Y. Saad, Efficiently generalizing ultracold atomic simulations via inhomogeneous dynamical mean-field theory from two to three dimensions, Proceedings of the High Performance Computing Modernization Program Users Group Conference 2010, Chicago, IL, June 14–17, 2010, edited by D. E. Post,IEEE Computer Society, Los Alamitos, CA, 2010, (2010) to appear.

[9] P. Carrier, J. M. Tang, Y. Saad, J. K. Freericks, Lanczos-based low-rank correction methods for evaluating diagonals of Green's functions, in preparation, for J. Comput. Phys.

[10] L. M. Falicov, J. C. Kimball, Simple model for semiconductor-metal transitions: SmB_6 and transistion-metal oxides, Phys. Rev. Lett. 22 (1969) 997–999.

[11] U. Brandt, C. Mielsch, Thermodynamics and correlation functions of the Falicov-Kimball model in large dimensions, Z. Phys. B 75 (1989) 365–370.

[12] C. Ates, K. Ziegler, Quantum phases in mixtures of fermionic atoms, Phys. Rev. A 71 (2005) 063610–1–7.

[13] M. Iskin, J. K. Freericks, Dynamical mean-field theory for light-fermion-heavy-boson mixtures on optical lattices, Phys. Rev. A 80 (2009) 053623–1–10.

[14] V. Zlatić, J. K. Freericks, R. Lemański, G. Czycholl, Exact solution of the multicomponent Falicov-Kimball model in infinite dimensions, Philo. Mag. B 81 (2001) 1443–1467.

[15] H. D. Simon, The Lanczos algorithm with partial reorthogonalization, Mathematics of Computation 42 (1984) 115–142.

[16] B. Smith, P. Bjorstad, W. Gropp, Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations, Cambridge University Press, Cambridge, UK, 1996.

[17] Y. Saad, Iterative Methods for Sparse Linear Systems, Society for Industrial and Applied Mathematics, Philadelphia, USA, 2003.

[18] B. Parlett, The Symmetric Eigenvalue Problem, Society for Industrial and Applied Mathematics, Englewood Cliffs, 1998.

[19] R. Sidje, Y. Saad, Rational approximation to the Fermi - Dirac function with applications in density functional theory, MSI Report umsi-2008-279 (2008) 1–26.

[20] R. W. Freund, Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices, SIAM J. Sci. Stat. Comput. 13 (1) (1992) 425–448.

[21] S. Cauley, J. Jain, C.-K. Koh, V. Balakrishnan, A scalable distributed method for quantum-scale device simulation, Journal of Applied Physics 101 (2007) 123715.

[22] L. Lin, C. Yang, J. C. Meza, J. Lu, L. Ying, E. Weinan, Selinv – an algorithm for selected inversion of a sparse symmetric matrix, ACM Trans. Math. Software to appear (2010) 1–20.

[23] J. Tang, Y. Saad, Domain-decomposition-type methods for computing the diagonal of a matrix inverse, MSI Reports umsi-2010-114 (2010) 1–26.

[24] J. Tang, Y. Saad, A new method for computing the diagonal of a matrix inverse, MSI report umsi-2010-42 (2010) 1–15.

[25] J. Hubbard, Electron correlations in narrow energy bands, Proc. R. Soc. London 276 (1963) 238–257.

[26] R. Jördens, L. Tarruell, D. Greif, T. Uehlinger, N. Strohmaier, H. Moritz, T. Esslinger, L. De Leo, C. Kollath, A. Georges, V. Scarola, L. Pollet, E. Burovski, E. Kozik, M. Troyer, Quantitative determination of temperature in the approach to magnetic order of ultracold fermions in an optical lattice, Phys. Rev. Lett. 104 (2010) 180401–1–4.